EDM’09

Educational Data Mining 2009

2nd International Conference On Educational Data Mining, Cordoba, Spain, July 1-3, 2009

Tiffany Barnes,

Michel Desmarais,

Cristóbal Romero,

Sebastián Ventura (Eds.)

ISBN: 978-84-613-2308-1

Preface

The Second International Conference on Educational Data Mining (EDM2009) was held at the University of Cordóba, Spain, on July 1–3, 2009. It follows the first edition of the conference held in Montreal in 2008, and a series of workshops within the AAAI, AIED, EC-TEL, ICALT, ITS, and UM conferences. EDM2010 will be held in Pittsburg, US.

EDM brings together researchers from computer science, education, psychology, psychometrics, and statistics to analyze large data sets to answer educational research questions. The increase in instrumented educational software and databases of student test scores, has created large repositories of data reflecting how students learn. The EDM conference focuses on computational approaches for using those data to address important educational questions. The broad collection of research disciplines ensures cross fertilization of ideas, with the central questions of educational research serving as a unifying focus.

We received a total of 54 submissions from 24 countries. Submissions were reviewed by three reviewers and 20 of them were accepted as full papers (37.03% acceptance rate). 13 other submissions were accepted as poster or as student papers. All papers will appear both on the web, at www.educationaldatamining.org, as well as in the printed proceedings. The conference also included invited talks by Professor Arthur C. Graesser from University of Memphis and by Professor Bamshad Mobasher from DePaul University.

We would like to thank the Universidad de Córdoba, Escuela Universitaria Politécnica, Junta de Andalucía y Ministerio de Ciencia e Innovación for their generous sponsorship of EDM2009. We would like to thank the program committee members, local committee, web chair, the reviewers and the invited speakers for their enthusiastic help in putting this conference together.

Tiffany Barnes
Michel C. Desmarais
Cristóbal Romero
Sebastián Ventura

Conference Organization

Conference Organisation

Conference Chairs:	Cristóbal Romero Morales, University of Cordoba, Spain
	Sebastián Ventura, University of Cordoba, Spain
Program Chairs:	Tiffany Barnes, University of North Carolina Charlotte, USA
	Michel Desmarais, Ecole Polytechnique de Montreal, Canada
Web Chair:	Arnon Hershkovitz, Tel Aviv University, Israel

Program Committee

Esma Aïmeur, University of Montreal, Canada
Ryan Baker, Carnegie Mellon University, USA
Joseph E. Beck, Worcester Polytechnic Institute, USA
Christophe Choquet, Université du Maine, France
Rebecca Crowley, University of Pittsburgh School of Medicine, USA
Aude Dufresne, University of Montreal, Canada
Neil Heffernan, Worcester Polytechnic Institute, USA
Roland Hubscher, Bentley College, USA
Brian Junker, Carnegie Mellon University, USA
Judy Kay, University of Sydney, Australia
Kenneth Koedinger, Carnegie Mellon University, USA
Agathe Merceron, University of Applied Sciences (TFH) Berlin, Germany
Brent Martin, Canterbury University, New Zealand
Gordon McCalla, University of Saskatchewan, Canada
Bruce McLaren, Deutsches Forschungszentrum für Künstliche Intelligenz, Germany
Tanja Mitrovic, Canterbury University, New Zealand
Jack Mostow, Carnegie Mellon University, USA
Roger Nkambou, Université du Québec à Montréal (UQAM), Canada
Mykola Pechenizkiy, Eindhoven University of Technology, Netherlands
Steven Tanimoto, University of Washington, USA
Kalina Yacef, University of Sydney, Australia
Osmar Zaiane, University of Alberta, Canada

Steering Committee

Esma Aïmeur, University of Montreal, Canada
Ryan Baker, Carnegie Mellon University, USA
Tiffany Barnes, University of North Carolina at Charlotte, USA
Joseph E. Beck, Worcester Polytechnic Institute, USA
Neil Heffernan, Worcester Polytechnic Institute, USA
Brian Junker, Carnegie Mellon University, USA
Kalina Yacef, University of Sydney, Australia

Local Organization

Jose L. Ávila, University of Cordoba, Spain
Pedro G. Espejo, University of Cordoba, Spain
Eva L. Gibaja, University of Cordoba, Spain
Pedro Gutierrez, University of Cordoba, Spain
Juan C. Fernandez, University of Cordoba, Spain
María Luque, University of Cordoba, Spain
David Martín, University of Cordoba, Spain
Juan Luis Olmo, University of Cordoba, Spain
Mykola Pechenizkiy, Eindhoven University of Technology, The Netherlands
José R. Romero, University of Cordoba, Spain
Amelia Zafra, University of Cordoba, Spain

External Reviewers

Mihaela Cocea
Philippe Fournier-Viger
Eva L. Gibaja
Arnon Hershkovitz
Sebastien Iksal
Collin Lynch
Tara Madhyastha
Noboru Matsuda
Manolis Mavrikis
Philip Pavlik
José Raúl Romero
Oliver Scheuer
Ben Shih
John Stamper
Leigh Ann Sudol
Amelia Zafra

Table of Contents

Regular papers

Posters and student papers

Indexes

A Comparison of Student Skill Knowledge Estimates

Elizabeth Ayers[1], Rebecca Nugent[1], and Nema Dean[2]
{eayers, rnugent}@stat.cmu.edu, {nema}@stats.gla.ac.uk
[1]Department of Statistics, Carnegie Mellon University
[2]Department of Statistics, University of Glasgow

A fundamental goal of educational research is identifying students' current stage of skill mastery (complete/partial/none). In recent years a number of cognitive diagnosis models have become a popular means of estimating student skill knowledge. However, these models become difficult to estimate as the number of students, items, and skills grows. There exist alternatives such as sum-scores and the capability matrix. While initial theoretical work on sum-scores has been done, the behavior of sum-scores and the capability matrix is not well understood with respect to each other or to estimates from cognitive diagnosis models. In this paper we compare the performance of the three estimates of student skill knowledge under a variety of clustering methods using simulated data with varying levels of missing values.

1 Introduction

A fundamental goal of educational research is identifying students' current stage of skill mastery (complete/partial/none). In addition, finding groups of students with similar skill set profiles is important to provide feedback for classroom instruction. In recent years a number of cognitive diagnosis models [3,8] have become a popular means of estimating student skill knowledge. However, these models become difficult and time-consuming to estimate as the number of students, items, and skills increases [8]. Two alternative estimates, sum-scores [3,6] and the capability matrix [1], can be used to estimate student skill knowledge in (near to) real time. Estimates are subsequently clustered to identify similar skill set profiles.

While initial theoretical work on sum-scores has been done [3], the behavior and performance of sum-scores and the capability matrix is not well understood in comparison with each other or with estimates from cognitive diagnosis models. The performance of the methods when missing values occur is also of interest. Moreover, which clustering method to employ is an open question. In this work we take a step back and compare the performance of three estimates of student skill knowledge under a variety of clustering methods. In Section 2, we describe the three different estimates of student skill knowledge. In Section 3, we give a brief introduction to the clustering methods used. In Section 4, we show results from a simulation study incorporating varying amounts of missing data. Finally, in Section 5, we offer conclusions and thoughts on future work.

2 Estimates of Student Skill Knowledge

While there may be several possible methods to estimate student skill knowledge, this paper will consider one traditional Bayesian estimation procedure and two simpler statistics. First, we introduce notation that will be common among the methods. We begin by assembling the skill dependencies of each item into a Q-matrix [2,12]. The Q-matrix, also referred to as a transfer model or skill coding, is a $J \times K$ matrix where $q_{jk} = 1$ if item j requires skill k and 0 if it does not, J is the total number of items, and K is the total number of skills. The Q-matrix is usually an expert-elicited assignment matrix. This paper assumes the Q-matrix is known and correct.

There are (at least) two ways in which Q-matrices can differ. First, each item could require only a single skill or multiple skills. A Q-matrix can then be comprised of all single skill items, single and multiple skill items, or all multiple skill items. Second, the Q-matrix may have a balanced or unbalanced design. In a balanced design, all single skill items occur the same number of times, and each combination of skills occurs the same number of times. For example, if K = 3 and J = 30 one possible balanced design would be: five single skill items for each skill, four double skill items for each pair of skills, and three triple skill items. A design could be unbalanced in two ways. Either all skills or combinations of skills are present but do not occur the same number of times or there are missing skills or combinations of skills.

$$Q = \begin{bmatrix} q_{1,1} & q_{1,2} & \dots & q_{1,K} \\ \vdots & \ddots & & \vdots \\ q_{J,1} & q_{J,2} & \dots & q_{J,K} \end{bmatrix}, \quad Y = \begin{bmatrix} y_{1,1} & y_{1,2} & \dots & y_{1,J} \\ \vdots & \ddots & & \vdots \\ y_{N,1} & y_{N,2} & \dots & y_{N,J} \end{bmatrix}$$

We then assemble student responses in a $N \times J$ response matrix Y where y_{ij} indicates both if student i attempted item j and whether or not they answered item j correctly and N is the total number of students. If student i did not answer item j then $y_{ij} = NA$. The indicator $I_{y_{ij} \neq NA} = 0$ expresses this missing value. If student i attempted item j ($I_{y_{ij} \neq NA} = 1$), then $y_{ij} = 1$ if they answered correctly, or 0 if they answered incorrectly.

2.1 DINA Model Estimates

The first method of estimating student skill knowledge uses a common conjunctive cognitive diagnosis model. The deterministic inputs, noisy "and" gate model (DINA; [8]) models student responses as

$$P(Y_{ij} = 1 \mid \eta_{ij}, s_j, g_j) = (1 - s_j)^{\eta_{ij}} g_j^{1-\eta_{ij}} \tag{1}$$

where $\alpha_{ik} = I_{\{\text{Student } i \text{ has skill } k\}}$ indicates if student i possesses skill k, $\eta_{ij} = \prod_{k=1}^{K} \alpha_{ik}^{q_{jk}}$ indicates if student i has all skills needed for item j, $s_j = P(Y_{ij} = 0 \mid \eta_{ij} = 1)$ is the slip parameter, and $g_j = P(Y_{ij} = 1 \mid \eta_{ij} = 0)$ is the guess parameter. If a student is missing any of the required skills, the probability that they will answer an item correctly drops due to the conjunctive assumption.

We estimate the student skill knowledge parameters of the DINA model, the α_{ik}, using Markov Chain Monte Carlo methods with the program WinBUGS (Bayesian Inference Using Gibbs Sampling, [9]). In the model, the α_{ik} are 0/1 indicating whether or not student i has mastered skill k. Our estimates will be $\hat{\alpha}_{ik} \in [0, 1]$. We can think of the $\hat{\alpha}_{ik}$ as the probability that student i has mastered skill k.

2.2 Sum-scores

The second estimate we consider is the sum-score method of [3,6]. Here $W_i = (W_{i1}, W_{i2}, ..., W_{iK})$ is a vector of sum-scores where the k^{th} component is defined as

$$W_{ik} = \sum_{j=1}^{J} y_{ij} q_{jk}, \tag{2}$$

where y_{ij} and q_{jk} are the corresponding entries from the response matrix Y and Q-matrix. Thus, the components of W_i are simply the number of items student i answered correctly for each skill k. When an item requires more than one skill it will contribute to more than one component of W_i. The range of W_{ik} may be different for each k if the skills are required by a different number of problems.

2.3 Capability Matrix

Finally, we consider the *capability matrix* defined in [1]. The capability matrix B is an $N \times K$ matrix where B_{ik} is the proportion of correctly answered items involving skill k that student i attempted. Thus,

$$B_{ik} = \frac{\sum_{j=1}^{J} I_{y_{ij} \neq NA} \cdot y_{ij} \cdot q_{jk}}{\sum_{j=1}^{J} I_{y_{ij} \neq NA} \cdot q_{jk}}, \tag{3}$$

where y_{ij} and q_{jk} are the corresponding entries from the response matrix Y and Q-matrix. The capability matrix expands on sum-scores by accounting for the number of items requiring skill k that student i answered. In this manner the statistic scales for the number of items in which the skill appears as well as for missing data. If a student has not seen all of the items requiring a particular skill, we still derive an estimate based on the available information. If student i completes no items involving skill k, then $B_{ik} = NA$. In this case, we impute an uninformative value (e.g., 0.5, mean, median) to map students to the hypercube. Exploring the performance of these imputation choices is ongoing. For this paper we assume that the data are complete or that missing B-values are appropriately imputed.

We can note that both the DINA model estimates and the B-matrix values map students into a K-dimensional hypercube (for each dimension, zero indicates total lack of skill mastery, one is complete skill mastery, and values in between are less certain). The 2^K corners of the hypercube correspond to natural skill set profiles $C_i = \{C_{i1}, C_{i2}, ..., C_{iK}\}, C_{ik} \in \{0, 1\}$.

Additionally, we can note theoretical connections between the sum-scores and B-matrix values. If there are no missing response values y_{ij}, then

$$W_{ik} = J_k B_{ik}, \tag{4}$$

where J_k is the number of items that require skill k. When all students have answered all questions and there is a balanced Q-matrix design (i.e., $J_1 = J_2 = ... = J_K$), the two estimates will mapto the same (scaled) feature space. In this case, we expect the two estimates to perform similarly. However, when there is either missing data or an unbalanced

Q-matrix design, the space to which the estimates map will be different. In this case, we cannot guarantee that performance will be similar.

3 Clustering Methods

To identify groups of students with similar skill set profiles, we cluster the student skill knowledge estimates. In this paper we will compare the performance of three common clustering methods: hierarchical agglomerative clustering, K-means, and model-based clustering. In the sections below we briefly introduce each of these methods.

3.1 Hierarchical Agglomerative Clustering

Hierarchical agglomerative clustering (HAC; [10]) links groups in order of closeness to form a tree structure from which a clustering solution can be extracted. Euclidean distance is most commonly used to measure the distance between groups. The method also requires the user to specify how to measure the distance between groups. We will use "complete" linkage where the distance between any two groups is defined as the largest distance between two observations, one from each group. In HAC, all observations begin as their own group. The two closest groups are merged and all inter-group distances are recalculated. We continue merging groups and recalculating distances until a single group with all observations is formed. Once the tree structure is formed, we can extract the desired number of clusters G by cutting the tree at a height corresponding to G branches.

3.2 K-means

K-Means [5] is a popular iterative descent algorithm for data $X = \{\underline{x}_1, \underline{x}_2 ..., \underline{x}_n\}$, $\underline{x}_i \in \Re^K$. It uses squared Euclidean distance as a dissimilarity measure and tries to minimize within-cluster distance and maximize between-cluster distance. For a given number of clusters G, K-Means searches for cluster centers m_g and assignments A that minimize the criterion

$$\min_{A} \sum_{g=1}^{G} \sum_{A(i)=g} ||\underline{x}_i - m_g||^2.$$

The algorithm alternates between optimizing the cluster centers for the current assignment (by the current cluster means) and optimizing the cluster assignment for a given set of cluster centers (by assigning to the closest current center) until convergence (i.e. cluster assignments do not change). It tends to find compact, spherical clusters and requires *a priori* both the number of clusters G and a starting set of cluster centers. The final cluster assignment can be sensitive to the choice of centers; a common method for initializing K-Means is to randomly choose G observations.

3.3 Model-based Clustering

Model-based clustering [4, 11] is a parametric statistical approach that assumes: the data $X = \{\underline{x}_1, \underline{x}_2, ..., \underline{x}_n\}$, $\underline{x}_i \in \Re^K$ are an independently and identically distributed sample from an unknown population density $p(\underline{x})$; each population group g is represented by a

Table 1: Clustering the DINA Model Estimates of Student Skill Knowledge

N	J	K	Q-matrix design	DINA	HAC	K-means	MBC	MBC 2^K
250	30	3	Single, bal	1.000	1.000	0.8739	0.9966	1.000
				(0.0054)	*(0.0054)*	*(0.0736)*	*(0.0895)*	*(0.0349)*
250	30	3	Both, bal	0.9793	0.9781	0.8367	0.8915	0.9632
				(0.0179)	*(0.0200)*	*(0.1192)*	*(0.0882)*	*(0.1087)*
250	30	3	Both,unbal, all	0.9657	0.9657	0.7789	0.9129	0.9350
				(0.0285)	*(0.2920)*	*(0.0941)*	*(0.0505)*	*(0.0758)*
250	30	3	Both,unbal,miss	0.9240	0.9131	0.7696	0.8811	0.9132
				(0.0395)	*(0.0427)*	*(0.0858)*	*(0.0696)*	*(0.0428)*
250	30	3	Mult, bal	0.4677	0.5127	0.5012	0.5282	0.4979
				(0.0292)	*(0.0443)*	*(0.0578)*	*(0.0690)*	*(0.0411)*
250	30	3	Mult, unbal, all	0.4629	0.4874	0.4948	0.5130	0.4790
				(0.0430)	*(0.0536)*	*(0.0816)*	*(0.0736)*	*(0.0495)*
250	30	3	Mult, unbal, miss	0.3239	0.4070	0.3835	0.4266	0.4090
				(0.0380)	*(0.0596)*	*(0.0521)*	*(0.0837)*	*(0.0630)*
500	68	5	Both, bal	0.9463	0.9428	0.7132	0.8348	0.9243
				(0.0184)	*(0.0188)*	*(0.0428)*	*(0.1123)*	*(0.0488)*
500	68	5	Both, unbal, miss	0.8724	0.8729	0.6665	0.8213	0.8624
				(0.0247)	*(0.0219)*	*(0.0466)*	*(0.0960)*	*(0.0226)*
300	40	7	Single	0.9041	0.8891	0.7674	0.3050	0.8881
				(0.0262)	*(0.0286)*	*(0.0409)*	*(0.1203)*	*(0.0282)*

(often Gaussian) density $p_g(\underline{x})$; and $p(\underline{x})$ is a weighted mixture of these density components, i.e. $p(\underline{x}) = \sum_{g=1}^{G} \pi_g \cdot p_g(\underline{x}; \theta_g)$ where $\sum \pi_g = 1$, $0 < \pi_g \leq 1$ for $g = 1, 2, ..., G$, and $\theta_g = (\mu_g, \Sigma_g)$ for Gaussian components. The method chooses the number of components G by maximizing the Bayesian Information Criterion (BIC) and estimates the means and variances (μ_g, Σ_g) via maximum likelihood. While it may assume Gaussian components, its flexibility on their shape, volume, and orientation allows student groups of varying shapes and sizes. When multiple students map to the same location, model-based clustering is known to overfit the data by using spikes with near singular covariance in these locations [4]. To alleviate this concern, we jitter the student skill estimates by a small amount (0.01). The effect on our results is minimal.

4 Simulation Study

To compare the skill knowledge estimates and clustering methods described above we did a simulation study using generated data from the DINA model (Equation 1). The Q-matrix design is varied to include balanced and unbalanced combinations of single and multiple skill items. Then, for a fixed Q-matrix design, we simulate 20 different student populations. Skill difficulties are always set to equal medium difficulty; inter-skill correlations are set to zero. These choices evenly spread students among the 2^K natural skill set profiles $[0, 1]^K$. For each student population, we generate true skill set profiles C_i. We then draw slip and guess parameters from a random uniform distribution ($s_j \sim$ Unif(0,0.30); $g_j \sim$

Table 2: Clustering the Sum-scores Estimates of Student Skill Knowledge

N	J	K	Q-matrix design	HAC	K-means	MBC	MBC 2^K
250	30	3	Single, bal	0.9910 *(0.0110)*	0.8549 *(0.0960)*	0.9191 *(0.2899)*	0.9957 *(0.0071)*
250	30	3	Both, bal	0.7644 *(0.1095)*	0.8156 *(0.1110)*	0.9321 *(0.1181)*	0.9442 *(0.0515)*
250	30	3	Both,unbal, all	0.6398 *(0.0889)*	0.7707 *(0.0951)*	0.6970 *(0.2138)*	0.8494 *(0.0713)*
250	30	3	Both,unbal,miss	0.6482 *(0.0511)*	0.6728 *(0.0650)*	0.7066 *(0.2064)*	0.7661 *(0.1095)*
250	30	3	Mult, bal	0.3950 *(0.0339)*	0.4720 *(0.0648)*	0.4383 *(0.0675)*	0.4375 *(0.0517)*
250	30	3	Mult, unbal, all	0.3862 *(0.0533)*	0.4606 *(0.0670)*	0.4380 *(0.0696)*	0.4481 *(0.0428)*
250	30	3	Mult, unbal, miss	0.2689 *(0.0273)*	0.2827 *(0.0848)*	0.3314 *(0.0352)*	0.3099 *(0.0347)*
500	68	5	Both, bal	0.4006 *(0.0560)*	0.5859 *(0.0442)*	0.5893 *(0.1223)*	0.6523 *(0.0432)*
500	68	5	Both, unbal, miss	0.4104 *(0.0373)*	0.54412 *(0.0366)*	0.6010 *(0.0537)*	0.6265 *(0.0397)*
300	40	7	Single	0.7348 *(0.0526)*	0.6474 *(0.0456)*	0.0973 *(0.0362)*	0.7080 *(0.0453)*

Unif(0,0.15)). Given profiles and slip/guess parameters, we generate the student response matrix Y.

As we know the true underlying skill set profiles C_i, we can calculate their agreement with the clustering partitions using the Adjusted Rand Index (ARI; [7]), a common measure of agreement between two partitions. The expected value of the ARI is zero and the maximum value is one, with larger values indicating better agreement.

Tables 1, 2, and 3 show the clustering results for the DINA model estimates, sum-scores, and the capability matrix, respectively. In each table, N is the number of students, J is the number of items, and K is the number of skills. The Q-matrix design describes the Q-matrix used when generating the student responses (see Section 2 for more details). Here *single* indicates that there were only single skill items, *both* indicates that there were both single and multiple skill items, and *mult* indicates that there were only multiple skill items. Also, *bal* indicates that the Q-matrix had a balanced design. An unbalanced design is denoted by *unbal* and *all* or *miss* shows whether all combinations were present or if some were missing. For the DINA model estimates (Table 1), we rounded the $\hat{\alpha}_{ik}$ to 0/1 to find the closest skill set profile. For the remaining methods in Table 1 and for all methods in Tables 2 and 3 we cluster the unrounded $\hat{\alpha}_{ik}$. When using HAC and K-means, we set the number of clusters equal to 2^K as suggested by [3]. For MBC we search over an appropriate range; MBC 2^K indicates that we set the number of clusters to 2^K. For each set of 20 simulations,

Table 3: Clustering the Capability Matrix Estimates of Student Skill Knowledge

N	J	K	Q-matrix design	HAC	K-means	MBC	MBC 2^K
250	30	3	Single, bal	0.9910	0.8190	0.9957	0.9957
				(0.0104)	*(0.0835)*	*(0.0071)*	*(0.0071)*
250	30	3	Both, bal	0.7644	0.7947	0.9353	0.9411
				(0.1095)	*(0.1056)*	*(0.1583)*	*(0.0300)*
250	30	3	Both,unbal, all	0.7273	0.8082	0.6252	0.8281
				(0.0867)	*(0.1227)*	*(0.1719)*	*(0.1543)*
250	30	3	Both,unbal,miss	0.6698	0.7390	0.4563	0.6693
				(0.0813)	*(0.0778)*	*(0.1267)*	*(0.1628)*
250	30	3	Mult, bal	0.4045	0.4530	0.4586	0.4499
				(0.0347)	*(0.0508)*	*(0.0624)*	*(0.0382)*
250	30	3	Mult, unbal, all	0.3899	0.4585	0.4518	0.4580
				(0.0509)	*(0.0550)*	*(0.0822)*	*(0.0589)*
250	30	3	Mult, unbal, miss	0.2700	0.3638	0.2803	0.2840
				(0.0291)	*(0.0737)*	*(0.0620)*	*(0.0457)*
500	68	5	Both, bal	0.4096	0.5711	0.5951	0.6647
				(0.0504)	*(0.0543)*	*(0.1284)*	*(0.0928)*
500	68	5	Both, unbal, miss	0.4327	0.5435	0.5560	0.6291
				(0.0405)	*(0.0350)*	*(0.2027)*	*(0.1050)*
300	40	7	Single	0.7399	0.6437	0.0906	0.7109
				(0.0545)	*(0.0402)*	*(0.0168)*	*(0.0409)*

we report the median ARI and the standard deviation.

First, we examine performance differences across Q-matrix designs. The first Q-matrix has only three skills; each skill occurs in 10 single skill items. The ARI for all three methods of estimation and all clustering methods is 1 in nearly all cases. Across the methods, K-means has the lowest ARI. This is not surprising as we randomly select $2^K = 8$ observations as the starting centers. A more informed set of starting centers (i.e., the natural skill set profiles) may lead to better performance. For the $K = 3$ examples, the ARI is higher when there are only single skill items compared to when there are both single and multiple skill items and only multiple skill items. The lone exception is MBC with sum-scores (*Single, bal* = 0.9191, *Both, bal* = 0.9321). The standard deviation in this case (0.2899) is rather large and indicates a wide range of ARI values for these 20 simulated datasets.

We now take a closer look at Q-matrices with at least some multiple skill items. We can note that the performance of all three clustering methods is better (as indicated by a higher ARI) when there are both single and multiple skill items in the Q-matrix, compared to only multiple skill items (also true across all three methods of estimation). In addition, when the Q-matrix has a balanced design, as opposed to an unbalanced design, the recovery of the true skill set profiles is better. In general, the performance of the three estimates of the student skill knowledge is similar across the clustering methods. This similar performance is particularly interesting since using sum-scores and the capability matrix yield large com-

Table 4: Clustering the DINA Model Estimates of Student Skill Knowledge for $N = 250, J = 30, K = 3$ with Missing Response Data

Q-matrix design	% missing	DINA	HAC	K-means	MBC	MBC 2^K
Both, bal	0	0.9793	0.9781	0.8367	0.8915	0.9632
Both, bal	10	0.4584	0.4690	0.4750	0.4725	0.4754
Both, bal	20	0.4326	0.4550	0.4581	0.4544	0.4567
Both, bal	30	0.4006	0.4340	0.4276	0.4267	0.4306
Both, bal	40	0.3513	0.3825	0.3850	0.3655	0.3681
Both, unbal, miss	0	0.9240	0.9131	0.7696	0.8811	0.9132
Both, unbal, miss	10	0.9084	0.9057	0.7516	0.8274	0.8009
Both, unbal, miss	20	0.8775	0.8651	0.7294	0.7560	0.7578
Both, unbal, miss	30	0.8193	0.8160	0.7256	0.7052	0.6948
Both, unbal, miss	40	0.7694	0.7746	0.7181	0.6515	0.6114

Table 5: Clustering the Sum-Score Estimates of Student Skill Knowledge for $N = 250, J = 30, K = 3$ with Missing Response Data

Q-matrix design	% missing	HAC	K-means	MBC	MBC 2^K
Both, bal	0	0.7644	0.8156	0.9321	0.9442
Both, bal	10	0.6255	0.7671	0.8280	0.8489
Both, bal	20	0.5000	0.6717	0.4854	0.7526
Both, bal	30	0.4191	0.5855	0.4131	0.5309
Both, bal	40	0.3168	0.5072	0.2951	0.3867
Both, unbal, miss	0	0.6482	0.6728	0.7066	0.7661
Both, unbal, miss	10	0.5744	0.6091	0.3608	0.6563
Both, unbal, miss	20	0.4834	0.5556	0.3264	0.5414
Both, unbal, miss	30	0.3686	0.4876	0.2725	0.3961
Both, unbal, miss	40	0.3266	0.4203	0.2514	0.2624

putational savings when compared to estimating the DINA model using WinBUGS (up to 700 times faster; [1]). Moreover, in this simulation study the data are generated from the DINA model; we would expect the Bayesian estimation to perform well in this best-case scenario. For sum-scores and the capability matrix to perform as well as, and better than in some cases, the DINA model is noteworthy.

The above results are for student populations with complete response data. In practice, missing responses (unanswered questions) will be ubiquitous. We chose two Q-matrix designs with $N = 250$, $J = 30$, and $K = 3$ (*Both, bal* and *Both, unbal, miss*) and removed 0, 10, 20, 30, and 40% of the student responses completely at random for each of the 20 student populations. Results can be seen in Tables 4, 5, and 6. Note that the 0% missing corresponds to the previously shown results. Again, we report the median ARI. The standard deviations are not shown due to space limitations. They ranged from 0.03 to 0.16 and were generally ordered as DINA model (lowest), capability matrix, and sum-score (highest).

Table 6: Clustering the Capability Matrix Estimates of Student Skill Knowledge for $N = 250, J = 30, K = 3$ with Missing Response Data

Q-matrix design	% missing	HAC	K-means	MBC	MBC 2^K
Both, bal	0	0.7644	0.7947	0.9353	0.9411
Both, bal	10	0.6682	0.7894	0.6633	0.8786
Both, bal	20	0.6028	0.7491	0.5350	0.7655
Both, bal	30	0.6022	0.7141	0.5021	0.5505
Both, bal	40	0.4842	0.6103	0.3948	0.4086
Both, unbal, miss	0	0.6698	0.7390	0.4563	0.6693
Both, unbal, miss	10	0.6032	0.6980	0.4766	0.5473
Both, unbal, miss	20	0.5761	0.6629	0.4687	0.4654
Both, unbal, miss	30	0.5351	0.6251	0.4764	0.4775
Both, unbal, miss	40	0.5108	0.5658	0.4144	0.4335

In general, as the amount of missing data increases, the ARI decreases across all three estimation methods and all methods of clustering. However, some methods show more substantial decreases than others. When using the capability matrix, K-means shows relatively stable performance for both Q-matrix designs. For the *Both, unbal, miss* design, HAC and MBC also show stable performances. When using sum-scores, the performance drops more noticeably across all clustering methods which may reflect that the capability matrix scales for the number of questions answered while sum-scores do not. In the *Both, bal* case, the performance of the capability matrix estimates is generally better than both the DINA model estimates and the sum-scores (particularly true for K-means). For HAC, sum-scores and the capability matrix perform similarly (both better than the DINA model estimates). For the *Both, unbal, miss* case, the performance of the DINA model estimates is better than both sum-scores and the capability matrix estimates. When using the capability matrix estimates, K-means clustering performs best; its ARI values are only slightly lower than those of the DINA model.

5 Conclusions

Simulated examples show that recovery of the true skill set profiles is best when only single skill items occur. For Q-matrices with multiple skill items, recovery is improved if there are also single skill items present. These results hold across all three clustering methods and all three estimates of student skill knowledge. In addition, we note that the more computationally attractive capability matrix and the sum-score estimates perform similarly to the Bayesian estimation of the DINA model.

However, when there are missing responses, the performance of the estimation procedures changes. In general, the ARI values decrease as the percent of missingness increases (across all estimation and clustering methods). When the Q-matrix has a *Both, bal* design, the capability matrix estimates perform better than both the DINA model and sum-score estimates. In the *Both, unbal, miss* design, the DINA model estimates perform better than sum-scores and the capability matrix estimates.

These results can be used to guide the design of exams and tutor problems. For better estimation of student skill knowledge, single skill items should be included for each skill. In addition, students should be encouraged to finish all items. Whether or not it is by design, when students use online tutors, for example, they often do not complete all the items. In this case, it is particularly important for single skill items to be included. In the presence of missing responses, however, care should be taken when choosing an estimation method and a clustering method. The best choice is not obvious.

While there are benefits of using the capability matrix and/or sum-scores, we note that if an item requires multiple skills and a student answers incorrectly, all skills required by the item will receive a penalty, even if the student has mastered one (or more) of the skills. In future work, we will explore the behavior of alternative estimates that better account for multiple skill items. Possible methods could use empirical performance on single skill items or weight by the number of skills required by the incorrectly answered item.

References

[1] Ayers, E, Nugent, R, Dean, N. “Skill Set Profile Clustering Based on Student Capability Vectors Computed from Online Tutoring Data”. *Educational Data Mining 2008: 1st International Conference on Educational Data Mining, Proceedings* (refereed). R.S.J.d. Baker, T. Barnes, and J.E. Beck (Eds), Montreal, Quebec, Canada, June 20-21, 2008. p.210-217.

[2] Barnes, T.M. (2003). *The Q-matrix Method of Fault-tolerant Teaching in Knowledge Assessment and Data Mining*. Ph.D. Dissertation, Department of Computer Science, North Carolina State University.

[3] Chiu, C. (2008). *Cluster Analysis for Cognitive Diagnosis: Theory and Applications*. Ph.D. Dissertation, Educational Psychology, University of Illinois at Urbana Champaign.

[4] Fraley, C. and Raftery, A. Mclust: Software for model-based cluster analysis. *Journal of Classification*, 1999, 16, 297-306.

[5] Hartigan, J. and Wong, M.A. A k-means clustering algorithm. *Applied Statistics*, 1979, 28, 100-108.

[6] Henson, J., Templin, R., and Douglas, J. Using efficient model based sum-scores for conducting skill diagnoses. *Journal of Education Measurement*, 2007, 44, 361-376.

[7] Hubert, L. and Arabie, P. Comparing partitions. *Journal of Classification*, 1985, 2, 193-218.

[8] Junker, B.W. and Sijtsma K. Cognitive Assessment Models with Few Assumptions and Connections with Nonparametric Item Response Theory. *Applied Psych Measurement*, 2001, 25, 258-272.

[9] Lunn, D.J., Thomas, A., Best, N., and Spiegelhalter, D. WinBUGS – a Bayesian modelling framework: concepts, structure, and extensibility. *Statistics and Computing*, 2000, 10, 325–337.

[10] Mardia, K.V., Kent, J.T., and Bibby, J.M. *Multivariate Analysis*. Academic Press, 1979.

[11] McLachlan, G.J., and Basford, K.E. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, 1988.

[12] Tatsuoka, K.K. (1983). Rule Space: An Approach for Dealing with Misconceptions Based on Item Response Theory. *Journal of Educational Measurement*. 1983, Vol. 20, No. 4, 345-354.

Differences Between Intelligent Tutor Lessons, and the Choice to Go Off-Task

Ryan S.J.d. Baker
rsbaker@cmu.edu
Human Computer Interaction Institute, Pittsburgh Science of Learning Center,
Carnegie Mellon University

Abstract. Recent research has suggested that differences between intelligent tutor lessons predict a large amount of the variance in the prevalence of gaming the system [4]. Within this paper, we investigate whether such differences also predict how much students choose to go off-task, and if so, which differences predict how much off-task behavior will occur. We utilize an enumeration of the differences between intelligent tutor lessons, the Cognitive Tutor Lesson Variation Space 1.1 (CTLVS1.1), to identify 79 differences between tutor lessons, within 20 lessons from an intelligent tutoring system for Algebra. We utilize a machine-learned detector of off-task behavior to predict 58 students' off-task behavior within that tutor, in each lesson. Surprisingly, the best model predicting off-task behavior from lesson features contains only one feature: lessons that involve equation-solving. We discuss possible explanations for this finding, and further studies that could shed light on this relationship.

1 Introduction

What underlies students' choices, while they use educational software? In particular, why do students choose to game the system or go off-task, while using educational software? Much of the research on these questions has focused on the role that stable or semi-stable student individual differences play in driving these types of behaviors [2, 3, 8, 9]. Take, for example, the case of gaming the system ("attempting to succeed in an interactive learning environment by exploiting properties of the system rather than by learning the material" [cf. 5]). Several studies have been published that attempt to explain gaming behavior in terms of stable or semi-stable individual differences between students, such as a student's attitude towards mathematics or goal orientation [2, 8, 9]. These studies have generally found statistically significant relationships. However, the relationships found in these studies only explain 5-9% of the variance in gaming behavior ($r^2 = 0.05$ to 0.09) [2,8], a relatively low degree of explanatory power.

By contrast, [7] found that the differences between intelligent tutor lessons predict a large proportion of the variance in gaming behavior. In an analysis of 58 students' behavior within 20 lessons in an intelligent tutor for algebra (corresponding to the majority of a year's curriculum), a combination of features of tutor lessons was found to predict 56% of the variance in gaming behavior ($r^2 = 0.56$). In particular, lessons that incorporated interest-increasing text into problem scenarios had significantly less gaming; lessons with various types of ambiguity had more gaming; lessons with ineffective hints had more gaming; and lessons based on equation-solving had less gaming. These results suggest that it may be possible to bypass the intrusiveness and high development costs of interactive responses to gaming [cf. 1, 4, 22] simply by altering these features of lessons, designing lessons with less extraneous ambiguity and more attempts to increase student interest.

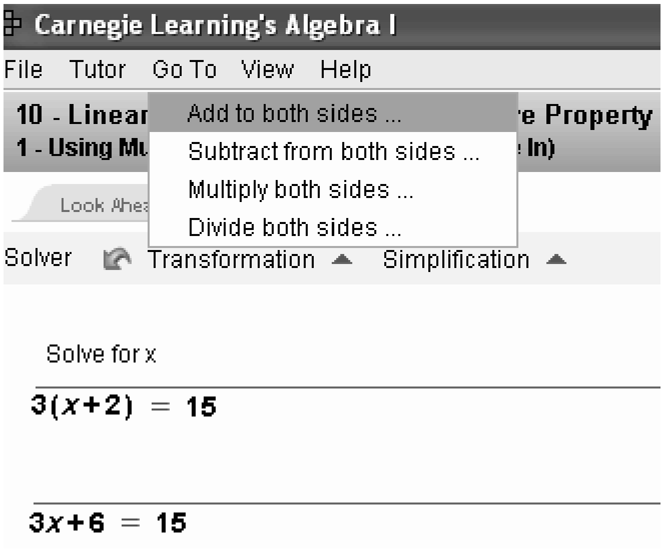
The discovery that gaming the system can be well predicted by small-scale differences in educational software design raises the question of whether other prominent learner behaviors are similarly associated with small-scale features of software design. In this paper, we investigate whether small-scale differences in software design can predict variance in off-task behavior. Off-task behavior shares many characteristics with gaming behavior. Both behaviors have been found to be associated with poorer learning in intelligent tutoring systems, although gaming the system's impact on learning is both larger and more immediate [6, 11]. Additionally, the two behaviors have each been found to be weakly associated with some of the same student individual differences [3], in particular negative attitudes towards computers and mathematics.

In this study, we apply a previously validated detector of off-task behavior [3] to data obtained from the PSLC DataShop [15], representing an entire school year of use of Cognitive Tutor Algebra, a widely used intelligent tutoring system. During the school year, students worked through a variety of lessons on different topics. These lessons had moderate variation in subject matter and considerable variation in design, making it possible to observe which differences in subject matter and/or design are associated with differences in how much off-task behavior occurs. We apply an existing taxonomy of the differences between tutor lessons [7] to these lessons, and investigate which lesson features are most strongly associated with off-task behavior.

2 Data and Models Applied

Data was obtained from the PSLC DataShop [15] (dataset: Algebra I 2005-2006 Hampton Only), for 58 students' use of Cognitive Tutor Algebra during an entire school year. The data set was composed of approximately 437,000 student transactions (entering an answer or requesting help) in the tutor software. All of the students were enrolled in algebra classes in one high school in the Pittsburgh suburbs. The school used Cognitive Tutors two days a week, as part of its regular mathematics curriculum. None of the classes were composed predominantly of gifted or special needs students. The students were in the 9th and 10th grades (approximately 14-16 years old).

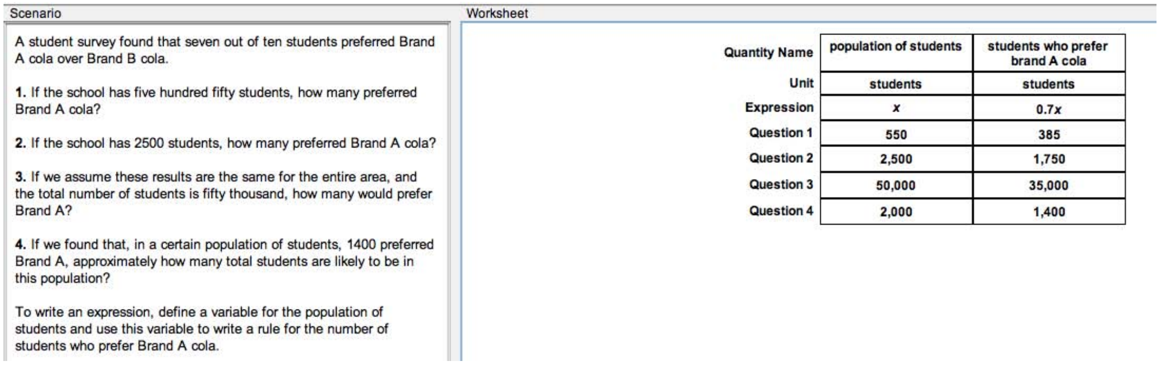
The Cognitive Tutor Algebra curriculum involves 32 lessons, covering a complete selection of topics in algebra, including formulating expressions for word problems, equation solving, and algebraic function graphing. Three lessons from Cognitive Tutor Algebra are shown in Figure 1. Data from 8 lessons was eliminated from consideration, as taxonomy codings were not available for those lessons (these lessons were not coded in [7], due to having limited data from those lessons available for that paper's analyses of interest). On average, each student completed 10.7 tutor lessons (among the set of 24 lessons considered), for a total of 619 student/lesson pairs.



Solve for x

$3(x+2) = 15$

$3x+6 = 15$



A student survey found that seven out of ten students preferred Brand A cola over Brand B cola.

1. If the school has five hundred fifty students, how many preferred Brand A cola?

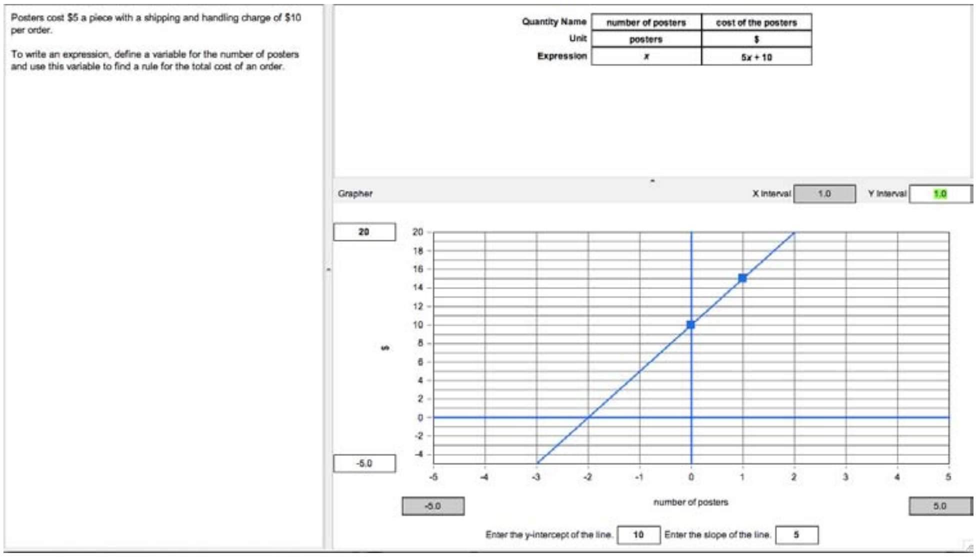
2. If the school has 2500 students, how many preferred Brand A cola?

3. If we assume these results are the same for the entire area, and the total number of students is fifty thousand, how many would prefer Brand A?

4. If we found that, in a certain population of students, 1400 preferred Brand A, approximately how many total students are likely to be in this population?

To write an expression, define a variable for the population of students and use this variable to write a rule for the number of students who prefer Brand A cola.

Quantity Name	population of students	students who prefer brand A cola
Unit	students	students
Expression	x	$0.7x$
Question 1	550	385
Question 2	2,500	1,750
Question 3	50,000	35,000
Question 4	2,000	1,400



Posters cost $5 a piece with a shipping and handling charge of $10 per order.

To write an expression, define a variable for the number of posters and use this variable to find a rule for the total cost of an order.

Quantity Name	number of posters	cost of the posters
Unit	posters	$
Expression	x	$5x + 10$

Enter the y-intercept of the line. 10 Enter the slope of the line. 5

Figure 1. Three lessons from Cognitive Tutor Algebra. Top: The Equation-Solver. Middle: Story Problem with Worksheet. Bottom: Function Graphing.

To determine how often each student was off-task, in each lesson, each student's actions were labeled using Baker's [3] detector of off-task behavior. The detector was developed using data from 429 students' classroom use of three lessons from an intelligent tutor on middle school mathematics. Applying this detector makes it tractable to study off-task behavior across a wide variety of tutor lessons. By contrast, other well-known methods are intractable – for instance, conducting quantitative field observations on a similar number of tutor lesssons and students would involve sending out two or more research assistants to classrooms for an entire year.

The detector, under cross-validation, achieved a correlation of 0.55 to field observations of off-task behavior – hence, it can be considered reasonably reliable for these purposes. The detector is also able to distinguish off-task behavior from on-task conversation, by looking at the student actions that occur immediately before and after a seemingly idle pause. We show the model that predicts off-task behavior within the detector in Table 1. The detector makes a prediction as to whether each action is off-task, and then aggregates across actions to indicate what proportion of student actions was off-task (or, alternatively, what proportion of student time was off-task). Full details on this detector are available in [3]. Two features (F3 and F6) involved features that were not available for this data set (string and generally-known). However, F3 and F6 together accounted for only 4.4% of the cross-validated correlation accounted for by this model [3] – hence, this model can still be expected to be accurate even in the absence of these features.

Table 1. The model of off-task behavior (OT) used in this paper, from [3]. In all cases, param1 is multiplied by param2, and then multiplied by value. Then the six features are added together. If the sum is greater than 0.5, the action is considered to be off-task. Features that were not applicable to the current data set are indicated in gray. "Pknowretro", a feature found in many behavior detectors, refers to the probability the student knew the skill if the action was the first opportunity to practice the current skill on the current problem step, and is -1 otherwise.

	param 1	param 2	value	Interpretation
F1	timelast3SD	timelast5SD	-0.08	OT: Very fast actions immediately before or after very slow actions
F2	timeSD	timeSD	0.013	OT: Extremely fast actions or extremely slow actions
F3	string	pknowretro	-0.36	OT: Less likely on well-known string-input steps OT: More likely when inputting a string after error
F4	notfirstattempt	recent8help	-0.38	Not OT: Asking for a lot of help
F5	notright	pknowretro	-0.16	OT: Two errors or help-requests in a row Not OT: Errors or help requests on skills the student has already mastered
F6	pctwrong	generally-known	0.04	OT: Indicated by many errors on skills students generally know prior to starting this lesson

Table 2. The 79 features of the Cognitive Tutor Lesson Variation Space (CTLVS1.1) used in study. Features captured using data mining methods (as opposed to hand-coding) marked with *.

Difficulty, Complexity of Material, and Time-Consumingness	
1*. Avg. % error	2. Lesson consists solely of review of material encountered in previous lessons
3*. Avg. probability that student will learn a skill at each opportunity to practice skill [cf. 12]	4*. Avg. initial probability that student will know a skill when starting tutor [cf. 12]
5. Avg. # of "distractor" values per problem	6. % of problems where "distractor" values given
7. Max number of mathematical operators needed to give correct answer on any step in lesson	8. Maximum number of mathematical operators mentioned in hint on any step in lesson
9. Intermediate calculations must be done outside of software (mentally or on paper) for some problem steps (ever occurs)	10. % of hints that discuss intermediate calculations that must be done outside of software
11*. Total number of skills in lesson	12*. Avg. time per problem step
13. % of problem statements that incorporate multiple representations (ex: diagram and text)	14. % of problem statements that use same numeric value for two constructs
15. Avg. number of distinct/separable questions or problem-solving tasks per problem	16. Maximum number of distinct/separable questions or problem-solving tasks in any problem
17. Avg. # of numbers manipulated per step	18*. Avg. # of times each skill repeated per problem
19*. Number of problems in lesson	20*. Avg. time spent in lesson
21. Avg. number of problem steps per problem	22. Minimum number of answers or interface actions required to complete problem
Quality of Help Features	
23*. Avg. amount that reading on-demand hints improves performance on future opportunities to use skill [cf. 10]	24*. Avg. Flesch-Kincaid Grade Reading Level [16] of hints
25. % of hints using inductive support, going from example to abstract concept/principle	26. % of hints that explicitly explain concepts or principles underlying current problem-solving step
27. % of hints that explicitly refer to abstract principles	28. On average, # of hints must student request before concrete features of problems are discussed
29. Avg. number of hint messages per hint sequence that orient student to math sub-goal	30. % of hints that explicitly refer to scenario content (instead of solely math constructs)
31. % of hint sequences that use terminology specific to this software	32. % of hint messages which refer solely to interface features
33. % hint messages that teacher can't understand	34. % of hint messages with complex noun phrases
35. % of skills where the only hint message explicitly tells student what to do	
Usability	
36. First problem step in first problem of lesson is either clearly indicated, or follows established convention (such as top-left cell in worksheet)	37. % of steps where student must change a value in a cell that was previously treated as correct (example: self-detection of errors)
38. After student completes step, system indicates where in interface next action should occur	39. % of steps where it is necessary to request hint to figure out what to do next
40. Not immediately apparent what icons in toolbar mean	41. Screen cluttered with interface widgets; difficult to determine where to enter answers
42. Problem-solving task is not immediately clear	43. Format of answer changes between problem steps without clear indication
44. If student has skipped step, and asks for hint, hints refer to skipped step without explicitly highlighting in interface (ever seen)	45. If student has skipped step, and asks for hint, skipped step is explicitly highlighted in interface (ever seen)
Relevance and Interestingness	
46. % of problems which appear to use real data	47. % of problem statements with story content
48. % of problem statements with scenarios relevant to potential student careers	49. % of problem statements with scenarios relevant to students' current daily life
50. % of problem statements which involve fantasy (example: being a rock star)	51. % of problem statements which involve concrete details unfamiliar students (example: dog sleds)
52. % of problem statements which involve concrete people/places/things	53. % of problem statements with text not directly related to problem-solving task
54. Avg. number of person proper names in problem statements	

Aspects of "buggy" messages notifying student why action was incorrect	
55. % of buggy messages that indicate concept student demonstrated misconception in	56. % of buggy messages that indicate how student's action was result of procedural error
57. % of buggy messages that refer solely to interface action	58. Buggy messages given by icon, which can be hovered over to receive buggy message
Design Choices Which Make It Easier to Game the System	
59. % of multiple-choice steps	60. Avg. number of choices in multiple-choice
61. % of hint sequences with final hint that explicitly tells student what the answer is, but not what/how to enter it in the tutor software	62. Hint gives directional feedback (example: "try a larger number") (ever seen)
63. Avg. number of feasible answers for each problem step	
Meta-Cognition and Complex Conceptual Thinking (or features that make them easy to avoid)	
64. Student is prompted to give self-explanations	65. Hints ever give explicit metacognitive advice
66. % of problem statements that use common word to indicate mathematical operation to use (example: "increase")	67. % of problem statements that indicate math operation with uncommon terminology ("pounds below normal" for subtraction)
68. % of problem statements that explicitly tell student which math operation to use ("add")	
Software Bugs/Implementation Flaws (generally rare)	
69. % of problems where grammatical error is found in problem statement	70. Reference in problem statement to interface component that does not exist (ever occurs)
71. Student can advance to new problem despite still visible errors	72. Hint recommends student do something which is incorrect or non-optimal (ever occurs)
73. % of problem steps where hints are unavailable	
Miscellaneous	
74. Hint requests that student perform some action	75*. Avg. length of text in popup widgets
76. Value of answer is very large (over four significant digits) (ever seen)	77. % of problem statements which include question or imperative
78. Student selects action from menu, tutor software performs action (as opposed to typing in answers, or direct manipulation)	79. Lesson is an equation-solver lesson

Each tutor lesson's attributes was represented using the Cognitive Tutor Lesson Variation Space version 1.1 (CTLVS1.1) [7], an enumeration of how Cognitive Tutor lessons can differ from one another. The CTLVS1.1 was developed by a diverse design team, including cognitive psychologists, educational designers, a mathematics teacher, and EDM researchers. The CTLVS1.1, shown in Table 2, consists of 79 features for how cognitive tutors differ from each other. The CTLVS1.1 was labeled with reference to the 24 lessons studied in this paper by a combination of educational data mining and hand-coding by the educational designer and mathematics teacher.

3 Analysis Methods and Results

The goal of our analyses was to determine how well each difference in lesson features predicts how much students will go off-task in a specific lesson. To this end, we combined the labels of the CTLVS1.1 features for each of the 22 lessons in Cognitive Tutor Algebra, and the assessments of how often each of the 58 students in the data set were off-task in each of the 22 lessons.

Our first step in conducting the analysis was to determine if the 79 features of the CTLVS1.1 grouped into a smaller set of factors. We empirically grouped the 79 features of the CTLVS1.1 into 6 factors, using the implementation of Principal Component Analysis (PCA) given in SPSS. These same 6 factors were previously successful in discovering a factor that was statistically significantly associated with gaming the system [7].

We analyzed whether the correlation between any of these 6 factors and the frequency of off-task behavior was significant. However, none of the factors was statistically significantly associated with off-task behavior – the closest factor to significance had F(1,21)= 0.37, p=0.55.

Taking the 79 features individually, only two were found to be statistically significantly associated with the choice to go off-task. Using an (overly conservative) Bonferroni adjustment [20] to control for the number of statistical tests conducted, only one feature was still found to be statistically significant. This feature was whether the lesson was an equation-solver lesson (as opposed to other types of lessons, such as story problems). An equation-solver lesson is shown at the top of Figure 1. Students were statistically significantly less likely to go off-task within equation-solver lessons, $r^2 = 0.55$, F(1, 21)=27.29, p<0.001, Bonferroni adjusted p<0.001.

To put this relationship into better context, we can look at the proportion of time students spent off-task in equation-solver lessons as compared to other lessons. On average, students spent 4.4% of their time off-task within the equation-solver lessons, much lower than is generally seen in intelligent tutor classrooms [5,6] or, for that matter, in traditional classrooms [cf.17, 18]. By contrast, students spent 14.1% of their time off-task within the other lessons, a proportion of time-on-task which is much more in line with previous observations. The difference in time spent per type of lesson is, as would be expected, statistically significant, t(22)=4.48, p<0.001.

The other feature found to be statistically significantly associated with off-task behavior, prior to the Bonferroni adjustment, was the proportion of hints that are solely bottom-out hints (more bottom-out-only-hints, less off-task behavior). However, a model including both of these two features was not statistically significantly better than the model that only considered whether the lesson was an equation-solver lesson, F(1, 21)=0.73, p=0.40.

4 Discussion and Conclusions

The results found here suggest that differences between lessons explain a large proportion of the variance in how much off-task behavior occurs, just as with gaming the system. However, the nature of the models found is quite different. Whereas the model that best explains how much gaming occurs was a complex set of fine-grained features [7], the model that best explains off-task behavior consists of a single, very coarse-grained difference. This leaves us with a problem of interpretation. Why were students off-task so much less within these equation-solver lessons?

One hypothesis is that there is some combination of features distinct to equation-solver lessons that produce less off-task behavior, but only when the full combination is encountered. For example, it is possible that the combination of features found in the equation-solver lessons (such as less complex hints, in combination with direct interaction with the equations, in problems that are generally shorter), combine to produce a state of very positive continued engagement (e.g. flow [13]) that precludes off-task behavior. It may be that this positive engagement is promoted by a specific combination of features only found in these lessons, explaining why off-task behavior was not associated with any of the finer-grained features in the CTLVS1.1, once the coarser feature of whether the lesson used the equation-solver was included. Relatedly, it might be that the task of equation-solving is somehow more engaging, in and of itself, than other mathematical problem-solving tasks, leading students to engage in a lower degree of off-task behavior.

A second hypothesis is that teacher behavior causes the lower off-task behavior within the equation-solver lessons. A conversation with a colleague with school teaching experience indicated that teachers in the United States are often particularly worried about students' performance on equation-solving on state standardized exams (personal communication, L.A. Sudol). This concern may lead teachers to monitor a student more closely, if the student is working through an equation-solver lesson. This hypothesis could be tested through observing teachers' behavior with quantitative field observations [cf. 5], as students use either equation-solver lessons or other lessons. It is worth noting that this hypothesis may also help explain the lower incidence of gaming the system in equation-solving lessons [e.g. 7].

Determining which of these hypotheses best explains the lower incidence of off-task behavior in equation-solver lessons has the potential to help us understand this behavior better. In turn, this knowledge has the potential to aid us in developing learning software that students engage with to a greater degree. In doing so, it is essential to avoid decreasing off-task behavior in ways that could increase the prevalence of other behaviors associated with poorer learning, such as gaming the system. It is also essential to avoid reducing off-task behavior in ways that would make instruction generally less effective – a potential danger in many visions of educational games in the classroom.

More broadly, we believe that the methods used in this paper point to new opportunities for the field of educational data mining. The creation of taxonomies such as the CTLVS1.1 will enable an increasing number of data mining analyses about how differences in educational software concretely influence student behavior. In turn, these analyses can inform a deeper scientific understanding of the interactions between students and educational software.

Acknowledgements

The author would like to thank Leigh Ann Sudol, Kenneth R. Koedinger, Vincent Aleven, and Albert Corbett for very helpful comments and suggestions. This work was funded by NSF grant REC-043779 to "IERI: Learning-Oriented Dialogs in Cognitive Tutors: Toward a Scalable Solution to Performance Orientation", and by the Pittsburgh Science of Learning Center, National Science Foundation award SBE-0354420.

References

[1] Arroyo, I., Ferguson, K., Johns, J., Dragon, T., Meheranian, H., Fisher, D., Barto, A., Mahadevan, S., Woolf. B.P. Repairing Disengagement with Non-Invasive Interventions. *Proceedings of the 13th International Conference on Artificial Intelligence in Education (AIED-2007)*, p. 195-202.

[2] Arroyo, I., & Woolf, B. Inferring learning and attitudes from a Bayesian network of log file data. *Proceedings of the 12th International Conference on Artificial Intelligence in Education,* 2005, p. 33-40.

[3] Baker, R.S.J.d. Modeling and understanding students' off-task behavior in intelligent tutoring systems, *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2007, 1059-1068.

[4] Baker, R.S.J.d., Corbett, A.T., Koedinger, K.R., Evenson, S.E., Roll, I., Wagner, A.Z., Naim, M., Raspat, J., Baker, D.J., Beck, J. Adapting to When Students Game an Intelligent Tutoring System. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 2006, p. 392-401.

[5] Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System". *Proc. ACM CHI 2004*, p. 383-390.

[6] Baker, R.S.J.d., Corbett, A.T., Roll, I., Wagner, A.Z., Koedinger, K.R. The Relationship Between Gaming the System and Learning in Cognitive Tutor Classrooms. Manuscript under review.

[7] Baker, R.S.J.d., de Carvalho, A.M.J.B., Raspat, J., Aleven, V., Corbett, A.T., Koedinger, K.R. Educational Software Features that Encourage and Discourage "Gaming the System". *Proceedings of the International Conference on Artificial Intelligence in Education,* in press.

[8] Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A., Koedinger, K. Why Students Engage in "Gaming the System" Behavior in Interactive Learning Environments. *Journal of Interactive Learning Research*, 2008, 19(2), p. 185-224.

[9] Beal, C.R., Qu, L., Lee, H. Mathematics motivation and achievement as predictors of high school students' guessing and help-seeking with instructional software. *Journal of Computer Assisted Learning,* 2008, 24 (6), p. 507-514.

[10] Beck, J.E. Using Learning Decomposition to Analyze Student Fluency Development. *Proceedings of the Educational Data Mining Workshop, at the 8th International Conference on Intelligent Tutoring Systems*, 2006, p. 21-28

[11] Cocea, M., Hershkovitz, A., Baker, R. The Impact of Off-Task and Gaming on Learning: Immediate or Aggregate? *Proceedings of the International Conference on Artificial Intelligence in Education,* in press.

[12] Corbett, A.T., Anderson, J.R. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction,* 1995, 4, p. 253-278.

[13] Csikszentmihalyi, M. *Flow: The Psychology of Optimal Experience,* 1990. New York: Harper and Row

[14] Inkpen, K.M., Ho-Ching, W., Kuederle, O., Scott, S.D., Shoemaker, G.B.D. This is fun! We're all best friends and we're all playing: supporting children's synchronous collaboration. *Proceedings of the 1999 Conference on Computer Support for Collaborative Learning.*

[15] Koedinger, K., Cunningham, K., Skogsholm A., Leber, B. An open repository and analysis tools for fine-grained, longitudinal learner data. *Proceedings of the First International Conference on Educational Data Mining,* 2008*,* p. 157-166.

[16] Kincaid, J. P., Fishburne, R. P., Rogers, R. L., Chissom, B. S. Derivation of new readability formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy enlisted personnel. Research Branch Report 8-75, Naval Technical Training, 1975.

[17] Lee, S.W., Kelly, K.E., Nyre, J.E. Preliminary Report on the Relation of Students' On-Task Behavior With Completion of School Work. *Psychological Reports,* 1999, 84, p. 267-272.

[18] Lloyd, J.W., Loper, A.B. Measurement and Evaluation of Task-Related Learning Behavior: Attention to Task and Metacognition. *School Psychology Review,* 1986, 15 (3), p. 336-345.

[19] Rodrigo, M.M.T., Baker, R.S.J.d., Lagud, M.C.V., Lim, S.A.L., Macapanpan, A.F., Pascua, S.A.M.S., Santillano, J.Q., Sevilla, L.R.S., Sugay, J.O., Tep, S., Viehland, N.J.B. Affect and Usage Choices in Simulation Problem Solving Environments. *Proceedings of Artificial Intelligence in Education 2007*, p. 145-152.

[20] Rosenthal, R., Rosnow, R.L. *Essentials of Behavioral Research: Methods and Data Analysis: 3rd Edition*, 1991. Boston, MA: McGraw-Hill.

[21] Skinner, E.A., Kindermann, T.A., Furrer, C. A motivational perspective on engagement and disaffection: Conceptualization and assessment of children's behavioral and emotional participation in academic activities in the classroom. *Educational and Psychological Measurement,* in press.

[22] Walonoski, J.A., & Heffernan, N.T. Prevention of off-task gaming behavior in intelligent tutoring systems. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems,* 2006, p. 722-724.

A User-Driven and Data-Driven Approach for Supporting Teachers in Reflection and Adaptation of Adaptive Tutorials

Dror Ben-Naim, Michael Bain, Nadine Marcus
(drorb, mike, nadinem)@cse.unsw.edu.au
School of Computer Science and Engineering
University of New South Wales, Sydney, Australia

Abstract: It has been recognized that in order to drive Intelligent Tutoring Systems (ITSs) into mainstream use by the teaching community, it is essential to support teachers through the entire ITS process: Design, Development, Deployment, Reflection and Adaptation. Although research has been done on supporting teachers through design to deployment of ITSs, there is surprisingly little discussion about support for teachers' Reflection - the ability to draw conclusions from ITS usage, and Adaptation - adapting the content to better meet the needs of students. We describe our work on developing analysis tools and methodologies that support reflection and adaptation by teachers. The work was done in the context of helping teachers understand student's behavior in Adaptive Tutorials by post-analysis of the system's data-logs. We used a hybrid solution – part of the data-mining effort is teacher driven and part is automated. We tested our approach by comparing the results of expert analysis of two Adaptive Tutorials with and without an automated Refinement Suggestion Tool, and found it to be a useful teacher's aid. By using this tool, teachers act as 'action researchers', confirming or disproving their hypotheses about the best way to use ITS technology.

1 Introduction

Intelligent Tutoring Systems (ITSs) can dramatically increase learners' comprehension by adapting the learning activity to the learners' needs, based on an intelligent assessment of their level of knowledge. This is the "Dream of ITS" (cf. "The Dream of AI") – that one day a system will be "smart" enough to teach better than human teachers. Whether this dream is to become a reality is arguable, even as ITS technologies are being intensively researched by the scientific community. In recent years, it has been recognized that whether or not the dream is realized, we must make ITSs as widely available as traditional web based educational systems. However, this is not a straightforward task, partially due to the sheer amount of content existing in traditional web based systems, compared with the relatively small amount of specialized content existing in ITSs[4], and also due to the complex nature of ITS's and their relative inaccessibility to teachers. In order to address this issue, teachers require better support through the entire ITS process: Design, Development, Deployment, Reflection and Adaptation.

To-date, research on supporting teachers in the ITS process has been focused on aiding teachers to author intelligent content, mainly through the advent of ITS authoring tools[10], but it is now clear that the ITS design paradigm needs to be updated. A new design paradigm offers teachers a different place in the ITS process; while the core authoring is in the hands of well-prepared design teams, teachers can extend the system and fine tune it to meet their specific needs[4].

This shift in the teacher's role is also acknowledged in the work of Diana Laurillard who proposed the Conversational Framework for the effective use of educational technology[6]. The Conversational Framework (CF) can be considered both a learning theory and a practical framework for designing educational environments. It models the interaction between teachers and learners as a stepwise "conversation" across four dimensions: discussion, adaptation, interaction and reflection. In [7], Laurillard describes the role of the teacher as an "action researcher", "collaborating to produce their own development of knowledge about teaching with technology". However, she also argues that support for reflection and adaptation is severely lacking with regards to eLearning content. This is because teachers rarely have the ability to reflect on (analyze and conclude) and adapt (change or edit) software based instructional material. The argument is even stronger for intelligent content offered by specialized systems such as ITSs.

This paper presents work that aims to support teachers through the process of the reflection and adaptation of Adaptive Tutorials (AT's) running on the Adaptive eLearning Platform (AeLP)[2]. An important challenge we faced in analyzing the Adaptive Tutorials in the AeLP was how to develop data-mining tools for the purpose of aiding teachers, without becoming too domain-specific or overwhelming them with a large number of association rules or classifiers which are difficult to understand. In particular, we aim to ensure the tool is easy to use and do not want to cognitively overload the teachers[14]. Moreover, students' interaction in the AeLP can vary dramatically between different AT's. Our contribution is through developing a refinement and adaptation strategy that can scale across different domains. We achieve this though a hybrid approach – user-driven and data-driven. The user-driven approach manifests itself in the development of an interactive analysis and discovery tool called the Adaptive Tutorial Analyzer (ATA). Teachers use the ATA for the purpose of analyzing students' performance in Adaptive Tutorials. The data-driven approach manifests itself in the development of a Refinement Suggestion Panel that draws teachers' attentions to patterns in the data that requires their attention. In this paper we show how both of these strategies complement each other.

2 Related work

Analyzing student behavior in an ITS is a complex problem, and the task of making sense of the data in ITS's logs is within the domain of educational data-mining[13]. Generally speaking, educational data mining is a data-driven field motivated to augment human-programmed knowledge, e.g. to ease the modeling of the correct way a problem should be solved ([8]), or to accurately predict a student's performance based on analysis of previous years' logs ([9]). However, some researchers previously highlighted the fact that patterns found in educational systems' data-sets are only useful if interpreted in the pedagogical context of the educational activity. In the work of [5] the researchers used an iterative process of discovery and interpretation with the goal of making sense of patterns discovered by data-mining algorithms they used.

We followed similar reasoning: patterns in the data-logs of Adaptive Tutorials are senseless without a teacher's pedagogical and domain specific insights. However, unlike [5] who rely solemnly on analysis of click-streams, the AeLP logs the entire system's internal state per each student's 'check' event (student pressing the 'check' button). As such, the data-logs are extremely multidimensional, up-to hundreds of attribute-values per student action. Furthermore, the system's snapshot depends on the specifics of the Virtual Apparatus (VA) that was used for the Adaptive Tutorial (see [2] for a description of how Adaptive Tutorials are constructed from Virtual Apparatuses), and as such we need tools that are domain independent but that can be utilized for the purpose of domain specific inquiry.

Another comprehensive study on analyzing ITS's data-logs was carried by [11] where data-mining algorithms were used in order to analyze the logs of a Constraint-Based ITS called SQL-Tutor. The researchers used a variety of tools such as WEKA and SQL in order to carry out multiple analysis tasks that resulted in some refinement suggestion to their system. One difference in our work is that the AeLP is a platform on which 10 different adaptive tutorials, each equivalent to SQL-Tutor in its scope and depth, are currently running. Our approach is thus to enable teachers to conduct analysis tasks, rather than specialist data-mining researchers. Furthermore, while the AeLP does use constructs analogues to Constraints (called trap-states), for the authoring of adaptive activities, it also uses solution traces, that are closer to Model Tracing based ITS's. This suggests that a richer knowledge representation is required for automated analysis.

Work on employing mining and visualization in order to analyze students' trails in a web-based educational system is also discussed in [12]. The data-set is again a navigation pattern or a "click-stream" and the researchers' approach was to interpret the student's navigation as a graph – considering each hypertext page as a node and transition between pages as edges. The tool is meant to be used as an aid for teachers to better understand student navigation. While similar to our concept to the AT-Analyzer, our efforts differ again in that the trails, or traces we are concerned with are not simply HTML pages requested, but traces through an entire solution state-space within an Adaptive Tutorial (see [3] for detailed explanation).

3 The Adaptive eLearning Platform

The Adaptive eLearning Platform (AeLP) is a web-based implementation of Virtual Apparatus Framework for eLearning content development[2]. The AeLP is used for authoring Adaptive Tutorials, deploying them to students or into LMSs, monitoring student progress and analyzing student behavior. The AeLP has been fielded since 2006 at the University of New South Wales, where Adaptive Tutorials developed using the AeLP have been incorporated into the syllabi of 10 major courses (ranging between 50 to 600 students per semester), and are accessed by over 2000 students per semester.

From a pedagogical point of view, AT's are similar in nature to teaching laboratory activities and are analogous to the concept of Tutorial Simulations as described in [6]. AT's exhibit three levels of adaptivity: students experience adaptive feedback with remediation targeted to their intrinsic misconceptions, while their activities are also sequenced adaptively based on performance. The third level of adaptivity is content adaptation through analysis and reflection. Teachers are provided with analysis tools that enable reflection and adaptation of their content. By analyzing students' behavior, teachers can refine and adapt their content, to better meet the needs of their students, e.g.: changing questions, adding new adaptive feedback or changing the sequence of activities. The work described in this paper concerns development of tools and processes to better facilitate this level of adaptation.

4 User-Driven and Data-Driven Analysis Strategy

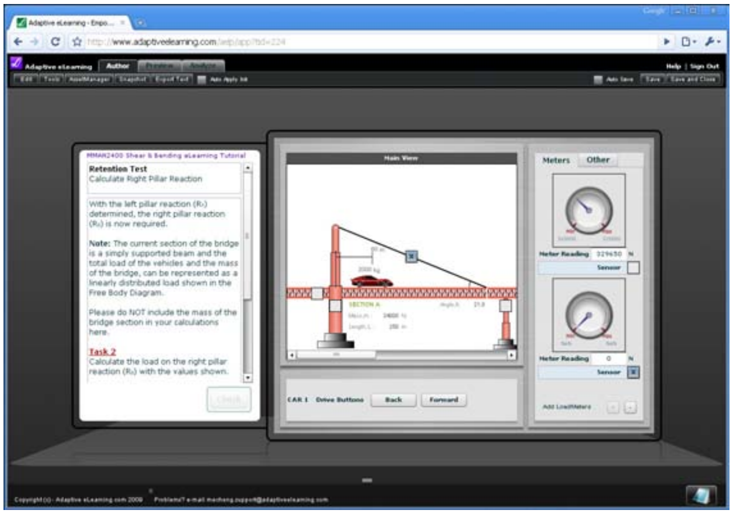
We presented our work on the AT-Analyzer in [3]. The analysis of adaptive tutorials is always performed with the purpose of refining and improving them for the next time they run. Teachers perform analysis on past AT-Sessions (instances of running an AT on a group of students), while the changes are saved to the next AT session. In that sense we support the Conversational Framework notion of teachers acting as "action researchers", interested in affirming or disproving their hypotheses regarding their content and its effect on learners[7]. Based on their analysis, teachers then need to be able to revise and change - to adapt - their content.

4.1 The Interaction-Snapshot Data Log

For each student interaction event, the AeLP stores a student-identifiable, time-stamped snapshot of the entire system's inspectable state-space. This state-space contains generic AeLP properties (e.g. *session.attemptNumber*, or *inputPanel.selectedChoice*) and the entire internal state the VA is in (e.g. *VA.propertyA* and *VA.propertyB*). The combined set of attribute-values is the student's Interaction-Snapshot-Vector. In addition to the interaction snapshot, the data also contains a trap-state ID. This ID is a unique identifier of the trap-state that was fired when processing the student's interaction. This trap-state can either be "correct" thus allowing the student to progress in their activity, or it could be an error-state, which contains some feedback to be shown to the student. In this way, the log database contains not only what the students were doing, but also the system's decision over their interactions.

4.2 An Example Adaptive Tutorial

As an example, consider an Adaptive Tutorial that was developed for a 1st year course in Solid Mechanics: the Bridge Inspection Simulator [Figure 1]. This AT features a bridge simulation, in which students can "drive" a car on a 3 section bridge. Students can position the car in different locations on the bridge sections, and take load and shear stress measurements on the bridge's poles and cables using virtual sensors. Here is an illustrative example question in this Adaptive Tutorial: "A second car C2 of mass m2 is positioned on section C (right hand side cantilever) of the bridge at x=250m. Position your car C1 of mass m1 on section A (left hand side cantilever) such that the tension on both sections' cables is the same. Enter the tension in Newtons in the input panel." The correct trap-state is defined as: *car1.x = 60 AND userInput = 60.* The teacher then defines an error trap-state that targets a familiar misconception. For example if a student positions the car at *car1.x = 50*, the teacher knows that they answered under the false assumption that *m1 == m2*, which is incorrect. A trap-state called *sameMassError* will target the condition *car1.x = 50* and will feature a hint feedback that will tell the students to look carefully for the masses in the question. The other trap-state for this system will be just the empty *defaultWrong* trap-state, which means that any student who did not enter 50 or 60 will be given some generic default feedback (e.g. "Wrong, Try Again."). We will return to this example in the following sections.



MMAN2400 Shear & Bending eLearning Tutorial

Retention Test
Calculate Right Pillar Reaction

With the left pillar reaction (R_L) determined, the right pillar reaction (R_R) is now required.

Note: The current section of the bridge is a simply supported beam and the total load of the vehicles and the mass of the bridge, can be represented as a linearly distributed load shown in the Free Body Diagram.

Please do NOT include the mass of the bridge section in your calculations here.

Task 2
Calculate the load on the right pillar reaction (R_R) with the values shown.

Figure 1: An AT in Mechanical Engineering: students can "drive" a car through different sections of a bridge and use sensors to inspect the load on different elements.

4.3 User-Driven Analysis Support – The Interactive Solution Trace Graph

In the Virtual Apparatus Framework, one can think of the process a student takes in order to solve a task as a trace through the problem's state-space. The idea behind the Solution Trace Graph [Figure 2] is to visualize the time-based vector of interaction-snapshots as a graph transition where each column is a solution attempt and each edge represents a transition between solution trap-states. Working with the Solution Trace Graph, teachers drill down on interaction data in order to gain insights regarding students' behavior, of which some of the most important are:

Finding adaptive feedback that was ineffective: if a high proportion of interactions entering a trap-state ended up landing back in the same trap-state on the next attempt column, it might imply that the feedback was not helping the students. We call this condition a trap-state's self-loop. For example, if 50% of students who landed in *sameMassError* landed again in *sameMassError* in their next attempt, the teacher might conclude that his feedback did not help the students to understand their mistake, and might change it, to be more specific.

Specializing an overly general trap-state: let's assume that 50% of students answered the aforementioned question correctly in the first attempt, 20% landed in *sameMassError*; the teacher will be interested to inspect what happened with the remaining 30% of students who landed in the *defaultWrong* trap-state. Using the STG, the teacher will inspect the interaction-snapshots leading into *defaultWrong*, and might notice a pattern, say, that 70% of those entered 100m as the answer. When researching why such a mistake was so prominent, the teacher might notice that students' mistake was that they used a + instead of a – in their calculation. The teacher will then simply add a new trap-state: *plusMinusConfusionError*, targeting this misconception.

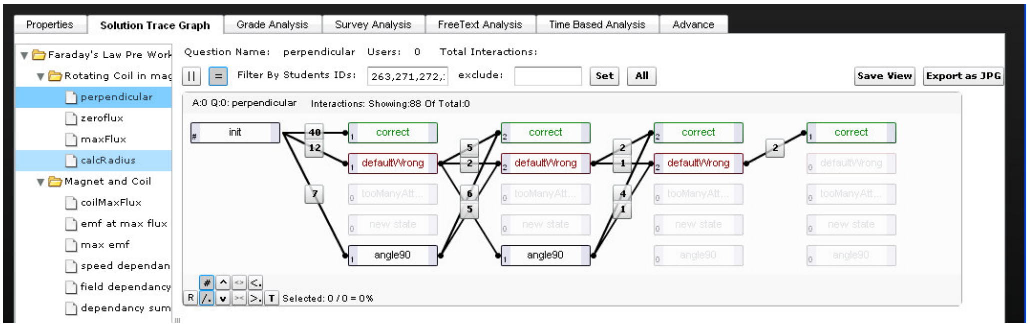

Question Name: perpendicular Users: 0 Total Interactions:
Filter By Students IDs: 263,271,272,: exclude:
A:0 Q:0: perpendicular Interactions: Showing:88 Of Total:0
Selected: 0 / 0 = 0%

Figure 2 – A Solution Trace Graph is used to visually analyze students' solution-traces through the problem's state-space. In this example it is easy to see that 40 out of 59 students attempting this question answered correctly on the first attempt and that 6 out of the 7 who landed in the *angle90* trap-state proceed to *correct* after given the adapted feedback.

4.4 Data Driven Analysis Support - The Refinement Suggestion Panel

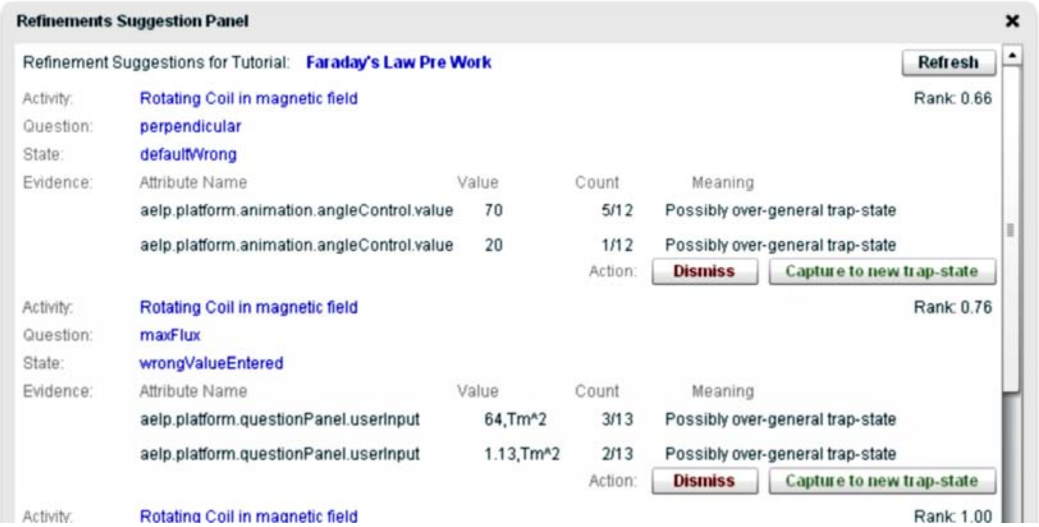
Based on our experience with the AT-Analyzer and the STG, it became clear that some aspects of the reflection work could be automated. Subsequently, a Refinement Suggestion Panel (RSP) was designed [Figure 3]. It offers teachers a list of the most relevant issues that might need their attention. Such suggestions, for example, highlight the fact that a question is too easy, or too hard, that a particular adaptive feedback seems to be ineffective, that a new trap-state should be defined, and more.

The issues discovered are ordered by calculated relevance, and teachers can chose to dismiss an issue or act on it. Relevancy is measured by functions specified for particular aspects of the type of suggestion.

Finding adaptive feedback that was ineffective: automating the detection of this type of issue is relatively simple: an algorithm that exhausts all edge transitions in the entire AT's STG, and sorts the results on self-loop ratios across solution attempts, was developed. The RSP then presents the teacher with a list sorted in descending order. The top case is the trap-state with the highest self-loop ratio (weighted relative to edge count size, so that a 40 out of 50 ratio will appear before an 8 out of 10).

Specifying an Overly General Trap-State: in order to detect this type of issue, we need an automated way to search for association rules in each trap-state's interaction-snapshot data. In the example above, we are interested that the RSP will show the teacher the fact that 70% out of the 30% who landed in *defaultWrong*, entered 100m. Remembering that snapshots can contain tens or hundreds of attributes (defined by the VAs API, e.g. *car1.mass*, *sectionA.mass* etc.), the immediately apparent problem is how to get rid of all the non-interesting rules, containing attributes that are meaningless from an educational point of view. In other words, how do we target the *car.x* attribute?

One way to solve this is to look for what extra information is available to us. If we look at the question-attributes-set - the set of all attributes that are used in a question's existing trap-states, we will then see that the teacher targeted *car.x* and *userInput*. This information gives us a clue about what attributes are useful and we will only perform an association rule and feature selection search on this limited set of attributes. For the example shown in [Figure 3], we identified an overly general trap-state - *defaultWrong* (the antecedent of *defaultWrong* is the negation of all other custom trap-states, in other words- if no other trap-state fired, *defaultWrong* is fired). The association rule that was found has the antecedent containing the condition: ….*angleControl.value == 70* with coverage = 12/59 students and confidence = 0.42 (5/12). By capturing this new rule as new trap-state, the teacher will refine the overly general *defaultWrong*.


Refinements Suggestion Panel
×
Refresh
Dismiss
Capture to new trap-state
Dismiss
Capture to new trap-state

Refinement Suggestions for Tutorial: Faraday's Law Pre Work

Activity: Rotating Coil in magnetic field
Rank: 0.66
Question: perpendicular
State: defaultWrong

Evidence: Attribute Name	Value	Count	Meaning
aelp.platform.animation.angleControl.value	70	5/12	Possibly over-general trap-state
aelp.platform.animation.angleControl.value	20	1/12	Possibly over-general trap-state

Action:

Activity: Rotating Coil in magnetic field
Rank: 0.76
Question: maxFlux
State: wrongValueEntered

Evidence: Attribute Name	Value	Count	Meaning
aelp.platform.questionPanel.userInput	64,Tm^2	3/13	Possibly over-general trap-state
aelp.platform.questionPanel.userInput	1.13,Tm^2	2/13	Possibly over-general trap-state

Action:

Activity: Rotating Coil in magnetic field
Rank: 1.00

Figure 3 – The Refinement Suggestion Panel draws teachers' attention to possible issues that might need their attention. Teachers can act on those suggestions by capturing them as new trap-states.

Ranking the relevancy of this type of refinement suggestions is based on coverage and confidence, (thresholds of these parameters are user defined). For each coverage-level cohort we sort results by confidence before adding it to the RSP.

An obvious down-side of this approach is that patterns including attributes that are not in the question-attribute-set cannot be found in this way, and brought to the attention of the teacher. For example, it is possible that some of the students who made the *sameMassError* also put a "virtual load sensor" on the wrong "docking station", and thus were reading an erroneous value for their calculation. In this case, the interaction-snapshots of these students will contain the attribute value *sensor1.dockStation=1*. This association rule between the two attributes values *sensor1.dockStation=1 -> userInput=100m* is extremely important from a pedagogical point of view, but cannot be found by the RSP.

However, using the STG, the teacher can initiate an "all-in" rule search, that might find this association rule. Furthermore, the teacher can choose to look for associations and features on any subset of snapshot attributes, e.g. adding to the two sensors' "dock stations" attributes to the searched attribute set. If the teacher then decides to add a new trap-state using a new attribute, this new attribute now belongs to the question-attributes-set, and subsequently *will* be used by the RSP's data-driven analysis.

In this way, the user-driven and data-driven analyses complement each other, leveraging expert knowledge with data-mining efficiency, yielding a powerful yet teacher-friendly analysis tool.

5 Results and Analysis

For the purpose of a preliminary study regarding its usefulness, we used the RSP to generate suggestions for two Adaptive Tutorials: the Bridge Inspection Simulation and the Faraday's Law Tutorial in Physics (also described in [2]). The former was already analyzed by the teacher, using the ATA and the STG, while the bridge activity was not. In the case of the Faraday's law AT, we compared the refinement suggestions given by the RSP to the teacher's analysis in order to see if we were able to replicate their refinement actions. In the case of the bridge tutorial the teacher worked with the RSP in their analysis and investigated its usefulness.

The Faraday's law AT was run on a group of 59 students in the second half of 2007 and resulted in 982 interactions in the database, each containing a snapshot of 14 to 18 attribute-values representing the state of the system and VA per a user 'check' event. We ran the RSP on the Faraday AT's data log with a limit of 2 suggestions per question and we got a total of 28 suggestions. The top 3 ranked suggestions matched the same exact three improvements that were found by the teacher. For example: in a question that asked students to rotate a magnetic coil situated in constant magnetic field to the angle that will result in maximum magnetic flux through it (*correct* trap-state is *VA.angleControl.value = 0*), the RSP suggested for refinement the *defaultWrong* trap-state. It appeared that 5 students out of the 12 landing on the *defaultWrong* trap-state had the *VA.angleControl.value = 70*, which is in fact the question's *initState* [Figure 3]. In other words - those 5 students did not attempt to solve the question at all, and just pressed 'check', possibly attempting to game the system[1]. The teacher added a trap-state targeting the following conditions: *VA.angleControl.value == 70 AND session.timeOnQuestion < 15 seconds*, and attached feedback that politely asked the students to actually attempt solving the question by manipulating the VA's control. Out of the remaining new 25 suggestions, the teacher chose to use 5 and dismissed the rest. The teacher's impressions were very positive and they found the tool both easy to use and understandable.

The Bridge AT was used by 220 students during the second half of 2008 and generated a total of 7014 interaction entries in the database. A typical interaction entry included a system snapshot of around 18 attribute-value pairs. This time we limited the RSP to show only the top 10 suggestions, and 5 of them were accepted by the teacher as valid. Again, most suggestions targeted the *defaultWrong* trap-state – the state that most needed further specifications. All 5 states accepted by the teacher were found to be dealing with students not properly attempting a question, or selecting 'check' prematurely. This analysis has two conclusions: it supports the development approach of continual-refinement where an AT is developed initially to only remediate to the most obvious misconceptions and misbehaviors, and using the analysis tool the teacher gradually refine its rule base. The second conclusion is that the RSP algorithm described above does not work well when the tutorial questions are parameterized: in the Bridge AT, each student was given a different set of initial parameters (bridge length and height, masses of cars etc) and thus the *correct* trap-states are defined as functional dependencies between attributes-values and not constants. We discuss this further in the next section.

While further work still remains, we found that overall the teachers response was positive and the RSP is an important step forward in helping teachers understand how the AT's are being used by their students.

6 Future work

Based on our analysis, it appears that further work needs to be done on dealing with functional dependency between attributes. Consider the following simple example question: "Calculate the force the car is applying on the bridge (in Newtons), and enter it in the input panel." But this time, assume the Virtual Apparatus is set to randomize the car mass for each student. We now need to define the *correct* trap-state as: *userInput == car.mass*9.81*. In this case, the snapshot attributes will contain functional relationships between attributes, and a simple attribute association rule or feature selection test will not be able to identify any patterns. Possible future work is to allow that when functional relationship between attributes appears in a question's condition set, we provide the data-mining algorithm with a test that encodes that functional dependency. For example, we can define a new variable, *V=inputPanel.userInput / car.mass* and do feature selection on *V*. The RSP can then discover high probability for *V=1* which occurs when students forgot to multiply by the gravitational constant. This is an important feature, because it lets us incorporate relational logic in the rule association search in a manner that is easy for the teacher to understand.

7 Conclusion:

We have presented a hybrid analysis strategy for Adaptive Tutorials. A key aspect of our work is the fact the Adaptive Tutorials are constructed using the Virtual Apparatus Framework, thereby enabling rich content with a high degree of interactivity to be authored. This, however, presents challenges for the analysis of student activity in such complex environments. Towards this objective we implemented a user-driven analysis tool – the Solution Trace Graph, and complemented it with a data-driven analysis tool – the Refinement Suggestion Panel. We showed in this paper that one way in which the two approaches complement each other is that when a teacher adds a new attribute into a question's condition-set the RSP includes this attribute in its automated rule-finding algorithm. Based on a preliminary study and analysis, we found that the combined strategy was successful in leveraging the experts' domain knowledge to direct the data-mining process, improving effectiveness and efficiency.

By building such evaluation tools and techniques into ITS technology we allow teachers to understand and reflect on students' behavior, and subsequently adapt activities to better match student knowledge levels and address misconceptions. In that sense, and in accordance with the Conversational Framework, the teacher is acting as an active educational researcher, confirming or disproving their hypotheses about the best way to use ITS technology in pursuit of their pedagogical goals.

8 References

1. Baker, R.S., Corbett, A.T., Koedinger, K.R, *Detecting Student Misuse of Intelligent Tutoring Systems.* Proceedings of the 7th International Conference on Intelligent Tutoring Systems, 2004: p. 531-540.

2. Ben-Naim, D., N. Marcus, and M. Bain, *Virtual Apparatus Framework Approach to Constructing Adaptive Tutorials*, in *The 2007 International Conference on E-Learning, E-Business, Enterprise Information Systems, and E-Government*, A.B. Hamid R. Arabnia, Editor. 2007, CSREA Press: Las Vegas, Nevada, USW. p. 3-10.

3. Ben-Naim, D., N. Marcus, and M. Bain, *Visualization and Analysis of Student Interaction in an Adaptive Exploratory Learning Environment*, in *The 1st Int. Workshop in Intelligent Support for Exploratory Environments in the European Conference on Technology Enhanced Learning*. 2008, CEUR-WS: Maastricht, The Netherlands.

4. Brusilovsky, P., Knapp, J. and Gamper, J., *Supporting teachers as content authors in intelligent educational systems.* Int. J. Knowledge and Learning, 2006. **2**(3/4): p. 191-215.

5. Hübscher, R. and S. Puntambekar. *Integrating Knowledge Gained From Data Mining With Pedagogical Knowledge*. in *Educational Data Mining 2008: 1st International Conference on Educational Data Mining*. 2008. Montreal, Quebec, Canada.

6. Laurillard, D., *Rethinking University Teaching: A Conversational Framework for the Effective Use of Learning Technologies*. 2002: Routledge. 268.

7. Laurillard, D., *The teacher as action researcher: using technology to capture pedagogic form.* Studies in Higher Education, 2008. **33**(2): p. 139 - 154.

8. McLaren, B.M., et al. *Bootstrapping Novice Data: Semi-Automated Tutor Authoring Using Student Log Files*. in *Proceedings of the Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes. 7nth International Conference on Intelligent Tutoring Systems (ITS-2004)*. 2004.

9. Merceron, A. and K. Yacef. *Interestingness Measures for Association Rules in Educational Data*. in *1st International Conference on Educational Dat Mining (EDM08)*. 2008. Montreal, Canada.

10. Murray, T., *Authoring intelligent tutoring systems: an analysis of the state of the art.* International Journal of Artificial Intelligence in Education, 1999. **10**: p. 98-129.

11. Nilakant, K. and A. Mitrovic. *Application of data mining in constraint-based intelligent tutoring systems*. in *artificial intelligence in education, AIED*. 2005.

12. Romero, C., et al. *Mining and Visualizing Visited Trails in Web-Based Educational Systems*. in *Educational Data Mining 2008: 1st International Conference on Educational Data Mining*. 2008. Montreal, Quebec, Canada.

13. Romero, C. and S. Ventura, *Educational data mining: A survey from 1995 to 2005.* Expert Systems with Applications, 2008. **33**(1): p. 135-146.

14. Sweller, J., J. Van Merriengoer, and F. Paas, *Cognitive architecture and instructional design.* Educational Psychology Review, 1998. **20**: p. 251-296.

Detecting Symptoms of Low Performance Using Production Rules

Javier Bravo and Alvaro Ortigosa
{javier.bravo, alvaro.ortigosa}@uam.es
Computer Science Department, Universidad Autónoma de Madrid

Abstract. E-Learning systems offer students innovative and attractive ways of learning through augmentation or substitution of traditional lectures and exercises with online learning material. Such material can be accessed at any time from anywhere using different devices, and can be personalized according to the individual student's needs, goals and knowledge. However, authoring and evaluation of this material remains a complex a task. While many researchers focus on the authoring support, not much has been done to facilitate the evaluation of e-Learning applications, which requires processing of the vast quantity of data generated by students. We address this problem by proposing an approach for detecting potential symptoms of low performance in e-Learning courses. It supports two main steps: generating the production rules of C4.5 algorithm and filtering the most representative rules, which could indicate low performance of students. In addition, the approach has been evaluated on the log files of student activity with two versions of a Web-based quiz system.

1. Introduction

Modern information technologies have found their ways into the classrooms: new applications (learning management systems, virtual labs, etc.), new devices (PDAs, Smartphones), new protocols (SMS, Bluetooth) are used by teachers and students in real-life education [4]. These technologies have facilitated the wider adoption of online e-Learning systems in the last decade. Among other benefits of these systems, we can obtain more interactivity, reach learning experience, flexibility of access, etc. However, design and evaluation of e-Leaning systems yet remain complex tasks that require both time and expertise that many classroom teachers do not have. In most of cases instructors or course designers need to design his/her e-Learning course. Development of better technologies for authoring and evaluation of e-Learning systems is a very important research and practical problem, which becomes even more challenging, when the technology uses classroom instructors as its target audience.

In this paper we try to address this problem by proposing an approach to evaluation of e-Learning systems based on the analysis of log data. Evaluation of e-Learning systems is a complex and time-consuming task. One of the challenges for an instructor evaluating an e-Learning system is lack of evidence from students, since he/she has access only to their interactions with the learning material and cannot observe their behaviors, or receive the feedback from them in a timely manner. Due to the fact that student's behaviors are hidden inside their interactions, instructors should analyze them in order to assess the performance of students and evaluate the learning software. Furthermore, e-Learning systems generate vast quantity of data (log files). These data consist of records of students' actions within the system. Since the log files are often very big, traditional data analysis tools and techniques are not always useful. Therefore, an alternative technology needs to be applied.

We considered that data mining methodology is more adequate for this task, because it is a technology that blends traditional data analysis methods with sophisticated algorithms for processing large volumes of data in order to discover meaningful information [11]. Our work is focused on evaluating an e-Learning system by exploring its usage logs in order to find patterns that could indicate low performance. It is worth to mentioning that these patterns are called *symptoms*. An example of a symptom is a significantly higher number of failures for a given exercise than the average number of failures for this exercise. It is difficult for instructors to detect symptoms of this kind without the help of data mining methods. At the same time, it can be highly beneficial for an instructor to be aware that a certain student has problems with particular set of exercises, and hence be able to intervene. How do such symptoms occur? What factors can trigger them? The present work tries to answer these questions by providing a method for detecting symptoms of low student performance in the course.

Naturally, not every symptom observed in the real data, is an indicator of variations in performance in the course. It is useful to identify the relevant symptoms causing drops in performance; it is also useful to distinguish the most important of them. For these purposes, we have developed a method helping to filter and rank relevant symptoms. The method is based on production rules of C4.5 algorithm.

This paper is organized as follows. The next section presents a brief overview of related work. Section 3 describes the data set we have used in this study. The proposed approach, as well as the results of the evaluation are detailed in Section 4. Finally, the last section concludes the paper with discussion and plans for future work.

2. State of the Art

Over the last years data mining has become a very popular technology in many areas of information sciences including e-Learning. Romero and Ventura provide a comprehensive overview of data mining applications for education [8]. Large pool of student activity has been collected by many research facilities. A good example is the open data repository *DataShop* that stores over 110 datasets containing nearly 18 million student actions [5].

Many researchers recognize the potential of data mining methods to help diagnose problems in e-Learning applications. For example, Ueno proposes a method for detecting irregular learning processes using student response time [12]. The method uses Bayesian predictive model and parametric statistical tests to identify potential outliers. Our approach it is somewhat similar to Ueno's work; however, it is based on the use of decision trees and the analysis of students' answers to learning problems. While Ueno's system aims to help students, the goal of our project is to help course designers to design and evaluate e-Learning courses.

Another relevant work has been done by Meceron and Yacef [6]. They propose to use the *cosine* and *lift* as two alternative measures of interestingness for association rules instead of *confidence* and *support*. These alternative measures are very successful in filtering uninteresting *association rules*. The selection of the interesting or relevant rules is one of the foci of our work, but in our case we filter production rules instead of association rules.

3. Data Description

The input data for this project have been provided by the School of Information Sciences at University of Pittsburgh (Pittsburgh, USA). The data were collected through several semesters of students' interaction with *QuizGuide* and *QuizPACK* systems within the framework of introductory programming course. Only the data in 2007 were selected for this work, since the students of different years are not comparable because the contents of the course are different by each year. Consequently, records corresponding to 55 students of seven different groups were selected. They took an adaptive course of "Introduction to C programming" generating **52734** interactions with the system.

3.1 QuizGuide and QuizPACK: providers of the data

QuizGuide is a Web-based service that provides personalized access to self-assessment quizzes for C programming language [9]. It does not serve the quizzes itself; instead it stays as a wrapper between the quiz provider and the student's browser and augments the quizzes with adaptive navigation cues. To guide students to the right learning content QuizGuide exploits adaptive link annotation technique. Quizzes and questions in QuizGuide are annotated with adaptive icons. Every icon delivers to a student two kinds of information: his/her individual progress with the corresponding content, the relevance of the content to the student's current learning goal. The goal information is calculated based on the course schedule and the relation of the quizzes to different topics in the course. The progress information is calculated based on the individual student's history of correct and incorrect attempts for the corresponding question, as well as other questions covering the same domain concepts.

The quizzes are generated and evaluated by *QuizPACK* system [10]. Every QuizPACK question consists of a simple C program that students need to evaluate and answer what will be the output of the program or what will be the final value of the target variable. The important feature of QuizPACK is question generation. When the question is delivered to a student, one of the numeric values in the question text is dynamically instantiated with a random value. Consequently the students can try the same question again and again with different correct answers.

3.2 Log Data Description

When a student answers a question in QuizGuide a new log entry is added to the database. These entries consist of several fields. For our analysis we augmented the log entries with information about domain concepts. Table 1 shows three instances of the enhanced data log we analyzed (52734 cases).

Table 1. Log entries of students' interactions with QuizGuide

userId	groupId	result	activity	concept	conceptParent	session	success
uid_199	*gid_190*	*0.00*	*2dimensional_array1*	*printf*	*output*	*21BC5*	*no*
uid_199	*gid_190*	*1.00*	*2dimensional_array1*	*printf*	*output*	*21BC5*	*yes*
uid_359	*gid_72*	*1.00*	*2dimensional_array1*	*printf*	*output*	*A0B33*	*yes*

The header of Table 1 contains the following attributes:

- **userId**: id of the student in the system.
- **groupId**: id of the group to which the student belongs. Six groups belong to different colleges and one group to "world group". This last group is created for the students who did not belong to these colleges and follow the e-Learning course by free access.
- **result**: outcome of the student's attempt. Two values are possible: 0 (incorrect answer) and 1 (correct answer).
- **activity**: name of the quiz or activity.
- **concept**: name of the concept covered by the activity.
- **conceptParent**: name of the concept parent.
- **session**: id of the web session (a session combines students activity between login and logout).
- **success**: verbal representation of the field "result" (result = 1 corresponds to success = "yes"; result = 0 corresponds to success = "no").
- The system also stores the **time stamp** when the student started the activity.

It is important to point out that a student can perform the activity more than once. This situation can be observed in the first and second row of Table 1. For instance, the first row shows that the student *uid_199*, belongs to the group *gid_190*, obtained *0* (success = *no*) in the activity *2dimensional_array1*. This table also provides information between the activities and concepts, e.g., this activity is related to the concept *printf* and the general concept of this activity is *output*. In addition, this row provides information about the session id, *21BC5*. However, the second row exhibits that the same student solved the exercise successfully in another attempt. Thereby, interactions and cases are the same concepts, but students are not equivalent than cases since they can repeat the activities many times.

4. Analysis of the Data

The goal of this work is to analyze the log data and to find significant patterns of behavior, which can provide support for improving the teaching material. In this context, decision trees were selected as the data mining technique, since they produced good results in previous works [2]. In particular, the current work uses the **C4.5** algorithm [7].

The next step on the analysis process was to define which attributes from the log files would be analyzed and which one used as the class variable. Considering the intention of finding activities that can present difficulties for a subset of the students, two attributes were selected: *activity* (name of the activity) and *groupId* (group id). The idea was to detect activities that were significantly more difficult for students of a given group, compared to the students of the other groups. These attributes were considered enough for the task at hand, without adding excessive complexity to the problem. The class variable is *success* representing the outcome (is given as *yes* or *not* depending on whether the student solved the activity successfully or not) of students' attempts to answer QuizPACK question. Consequently, two classes compose the classification model, The space of the problem is 644* combinations, which together with the data size indicate that the data analysis is complex enough to apply the decision trees method.

The distribution of training data is shown in Table 2. The first row shows the values of the *success* attribute (class variable) and the number of cases covered by them. In this case, proportions of the classes *yes* and *no* are rather similar (46% versus 54%). This feature is suitable for building decision trees, which benefit from the balanced classes. The next seven rows provide the values of *groupId* and the number of cases for each of these values. It is clear that the group gid_67 has the lowest number of cases (70), and group gid_190 holds most of them (38382). Even though the data sets of groups with fewer cases could be removed, we decided to keep them, since every set of cases is valuable. Moreover, as it is more substantial for this research the model might be able to represent every set of data than provides better predictions. Finally, the last row of the table 2 offers the names of activities available (46 activities) in the adaptive course "Introduction to C programming". For example, the first three activities (*2dimensional_array1, 2dimensional_array2 and 2dimensional_array3*) range over exercises of coding arrays in C language.

Table 2. Distribution of training data

Properties	**Values**	**Number of cases**
success	yes	24010
	no	28724
groupId	gid_67	70
	gid_72	1629
	gid_190	38382

* The space of the problem is defined by the whole combinations of the attributes. In this case, *activity* variable has 46 different values, *groupId* has seven values, and two classes are possible. As a result, the space of this problem is 46*7*2 = 644. It is worth to mentioning that the space is a measure of the problem complexity.

	gid_373	3879
	gid_394	280
	gid_441	6536
	gid_442	1958
activity	2dimensional_array1,2dimensional_array2,2dimensional_array3, arithmetic_expression1,arithmetic_expression2, character_array1,character_array2,character_array3, character_processing1,character_processing2,character_processing3, conditional_operator1,complex_conditional1,complex_conditional2, function1,function2, function3,printing1,printing2, variable1,variable2,variable3,constant1,constant2, globvar_simplefunc1,globvar_simplefunc2, increment_decrement1,compound_assignment1, logical_expression1,logical_operator1,if_else1,if_else2, do1,do2,for1,for2,while1,while2, nested_loop1, switch1 pointer1,pointer2,pointer3,array1,array2,array3	

The third step in the analysis was to use the algorithm **C4.5** in order to obtain the decision tree. The pruning in the tree was deactivated, that is to say, the confidence factor (CF) is set to 100%. The reason of setting this CF is because overfitting does not represent a problem, since the resulting decision tree is not used to make predictions. Actually, the better model fits the training data, the better these data can be described by the model, which is the goal of this work. Another relevant parameter of the **C4.5** algorithm consists of grouping attribute values. This option supported two concerns: insufficiency of data and information-gain ratio criterion. The first concern is that useful patterns of the data are not detected due to insufficiency of data; therefore grouping values can get enough data for detecting these patterns. The second concern is related to the performance of information-gain ratio is lower when the attribute has many values [7, Chapter 5]. This option produces better results when the data contain discrete attributes with many values. In the case of this study, the attribute *groupId* contains 7 different values and the attribute *activity* has 47 values. As there are discrete attributes with many values in the data, this work exploits grouping attribute values option. As a result, applying the algorithm **C4.5** a decision tree with size of 168 is obtained. It is worth to bring out that the tree is composed by 113 leaves (54 leaves with class *no* and 59 with class *yes*) and 55 decision nodes.

Subsequently, the decision tree was analyzed. As the objective of this research is to find difficulties leaves with value *no* are more interesting than others. The initial attempt was to use the **key node** method [1], but this method requires analyzing every path from the leaf to the top of the tree in order to find the relevant decision nodes. Therefore, analyzing 54 paths requires to analyze each decision node of each path. One of the main problems is that some paths could be irrelevant or redundant, and **key node** method does not ensure avoid redundant paths. For this reason, the next attempt was to utilize an alternative method the *production rules*[ψ] of **C4.5**. These rules are based on two

[ψ] Quinlan defines a production rule as *left part --> right part* [7, p. 8-12]. The left-hand side contains the conditions and the right-hand side is referred to the class. If the case satisfies all the conditions (conjunction of attribute-based tests), it is classified by the value on the right part. For example, the rule *Rule 32: activity = variable2 --> class no [69.5%]* indicates if a case contains the activity *variable2* this case will be classified to class *no* with almost 70% of accuracy.

assumptions: the rules are easier to understand and the rules avoid redundant data. Every node in the decision tree has a context established by the tests' outcomes of the previous node. In this sense, a rule is easier to understand since the context is on the left side [7, Chapter 7]. The other assumption is related to the fact that the trees sometimes produce similar sub-trees. In that regard, the process of building the rules removes irrelevant conditions and avoids redundant rules. Applying the *C4.5rules* program to the data generated 20 rules. As in the case of decision trees only the rules in which the right side is *no* are selected (11 selected rules).

The last step was to filter the rules in order to select the relevant rules. Following this idea two approaches are possible: from the point of used cases (i.e. the cases in which all the conditions of the rule are satisfied) and from the point of accuracy (i.e. the percentage of correctly classified cases). Taking into account data description, we have chosen the former approach. *C4.5rules* also provides a ranking of non redundant rules based on the error rate (table 3 contains this information). It is important to note that three redundant rules corresponding to *no* class are not including in this ranking. For this reason, Table 3 presents eight rules with *no* class (they are highlighted in the table). The column **Rule** shows the id rule, the next column indicates the number of conditions in the rule, the column **Error** is an estimation of number of cases classified incorrectly, the column **Used** provides the used cases, the column **Wrong** exhibits the number of cases classified incorrectly, and the last column indicates the value of the class.

Table 3.Ranking of rules

Rule	Size	Error (%)	Used	Wrong	Class
21	2	0.0	74	0	yes
20	2	19.2	159	30	yes
13	2	22.5	60	13	yes
9	2	25.6	236	60	yes
25	1	32.5	3657	1189	yes
16	2	35.5	280	99	yes
40	1	38.9	16061	6526	yes
17	2	46.4	738	342	yes
3	2	47.3	278	131	yes
19	2	17.6	**71**	12	**no**
23	2	18.2	**2608**	475	**no**
8	2	20.9	**242**	50	**no**
33	1	29.8	**1771**	494	**no**
7	2	32.1	**344**	110	**no**
6	2	32.4	**2508**	813	**no**
12	2	34.2	**20419**	7522	**no**
14	2	39.7	**253**	100	**no**

As the filter criterion is based on used cases, the selected rules were ordered by this column. Another selection is needed because some rules are not enough representatives. This new sub-set of rules is obtained by using a lower band of used cases. Therefore, a rule is not enough representative if it is below this limit. Experiments with the data disclosed that the adequate lower band is calculated as 10% of sum of used cases of the rules. In this case, the sum of used cases is equal to 26445; thereby the lower band is

2645. Only three rules satisfied this threshold (rules 6, 23 and 12), hence they were selected. Table 4 demonstrates these rules.

Table 4.Representative rules

Rule 6:	`groupid in {gid_190, gid_373};`
	`activity in {character_array2, printing2, while2}`
		->	class no [67.6%]

Rule 23:	`groupid in {gid_190, gid_441, gid_373, gid_394};`
	`activity in {2dimensional_array2, do2, array1, array3}`
		->	class no [81.8%]

Rule 12:	`groupid = gid_190;`
	`activity in {2dimensional_array1, 2dimensional_array2, 2dimensional_array3, arithmetic_expression1, arithmetic_expression2, character_array1, character_array2, character_array3, character_processing1, character_processing2, character_processing3, conditional_operator1, do2, function1, logical_expression1, nested_loop1, pointer2, pointer3, printing2, variable2, array1, array2, array3, do1, for1, for2, function3, if_else1, if_else2, logical_operator1, pointer1, while2, globvar_simplefunc2}`
		->	class no [65.8%]

Rule 6 indicates that most of the students from the groups 190 and 373 showed difficulties in the activities character_array2, printing2 and while2. This rule might indicate a symptom of low performance on these quizzes for these groups of students. The next rule demonstrates problems that the students from the groups 190, 441, 373 and 394 had when they where trying activities 2dimensional_array2, do2, array1 and array3. Hence, further attention should be paid to these groups and activities. Several questions can be asked. Are the exercises adequate? Were students presented with the necessary knowledge? Lastly, the rule 12 shows that students from group 190 experienced problems in the most of activities. This rule could indicate that the students in the group 190 did not have enough background knowledge to take the course “Introduction to C programming”.

The previous analysis showed the whole process of achieving the representative symptoms, i.e. selected production rules. Therefore, the **proposed method** is summarized in the following lines:

- *Generate the production rules by using C4.5 algorithm.*
- *Select the rules of the ranking table in which the right part indicates a “failure” in activities. In our case, a failure is indicated by value no of the class success.*
- *Filter limit* $= \left. \sum_{i=first-selected-rule}^{last-selected-rule} used - cases_i \middle/ 10 \right.$
- *Select the rules which cover a number of used cases greater than the filter limit.*
    - *After this selection is possible to obtain only one rule. In this case, it is advisable to select also the next closer rule to the filter limit.*

It is important to note that the information displayed in these last rules could be useful to instructors or course designers for enhancing their e-Learning courses. Thus, it would be a valuable help to show this information to them, since they could evaluate their courses better by finding the last symptoms.

5. Conclusions and Future Work

This work has presented full analysis of real data of an e-Learning course. In this particular case 52734 instances were analyzed by using *decision trees* and *production rules*. The objective of the analysis was to find symptoms that could indicate presence of low performance in courses provided by *QuizGuide* and *QuizPACK* systems, i.e., to find particular activities in which a given category of students showed difficulties providing feedback to the instructors.

The analysis demonstrated that three rules indicate the presence of symptoms of drops in performance for several profiles of students. In fact, one of the rules exhibits that the group 190 showed difficulties in the course, since they failed the majority of the activities in the course. This information could be relevant for the instructor or designer of the e-Learning course, because he/she can improve it by adding new activities, and/or modifying existing activities or course structure. Furthermore, the other two rules showed that students who belong to groups 373, 441 and 394 presented problems in solving particular activities. This fact also indicates drops in the performance of the system for these groups of students. This information is also useful for the instructor or course designer, since he/she can modify these activities in order to improve the learning process for these particular students.

It is worth noting that this analysis can be performed with other type of data, even for data of different contexts. However, experiments with other types of data showed that the filter limit of this work is less accurate than others like third quartile of used cases.

Decision trees and production rules present some weaknesses. They are strongly related to the distribution of the data and proportion of classes. Therefore, small variations in the data could cause different conclusions. Consequently, the future work includes supporting these techniques with other Data Mining methods such as *association rules*. Other future line includes supporting the method described in this work in **ASquare** [3]. The goal of this tool is to provide a high abstraction level interface. Thereby, **ASquare** supplies more intelligible results to human beings; that is to say, a person without knowledge of Data Mining can understand the results.

Finally, it is important to test and validate the proposed analysis with other types of data. In this sense, future work also includes to apply this analysis for data of different educational systems in order to improve the detection method.

Acknowledgements

This work has been funded by Spanish Ministry of Science and Education through the HADA project TIN2007-64718. We would like to thank the University of Pittsburgh for providing us the interaction data of their students. The authors would also like to thank to Leila Shafti who contributed in this work with her helpful comments.

References

[1] Bravo, J., Ortigosa, A., and Vialardi, C. A Problem-Oriented Method for Supporting AEH Authors through Data Mining. *Proceedings of International Workshop on Applying Data Mining in e-Learning (ADML'07) held at the Second European Conference on Technology Enhanced Learning (EC-TEL 2007), 2007,* p. 53-62.

[2] Bravo, J., Vialardi, C., and Ortigosa, A. Using Decision Trees for Discovering Problems on Adaptive Courses. *Proceedings of E-Learn 2008: World Conference on E-Learning in Corporate, Government, Healthcare & Higher Education*, 2008, p. 268-277.

[3] Bravo, J., Vialardi, C., and Ortigosa, A. ASquare: A Powerful Evaluation Tool for Adaptive Hypermedia Course System. *Proceedings of Hypertext Conference,* 2008, p. 219-220.

[4] Chen, F., Myers, B., and Yaron, D. Using Handheld Devices for Tests in Classes. *Tech. Rep. CMU-CS-00-152, Carnegie Mellon University School of Computer Science, and Tech. Rep. CMU-HCII-00-101, Human Computer Interaction Institute. Human Computer Interaction Institute*, 2000.

[5] Koedinger, K.R., Cunningham, K., Skogsholm, A., and Leber, B. An open repository and analysis tools for fine-grained, longitudinal learner data. *Proceedings of Educational Data Mining 2008: 1St International Conference on Educational Data Mining,* 2008, p. 157-166.

[6] Merceron, A. and Yacef, K. Interestingness Measures for Association Rules in Educational Data. *Proceedings of Educational Data Mining 2008: 1St International Conference on Educational Data Mining,* 2008, p. 57-66.

[7] Quinlan, J.R. C4.5: Programs for Machine Learning. *Morgan Kaufmann Publishers*, California, USA, 1993.

[8] Romero, C. and Ventura, S. Educational Data Mining: a Survey from 1995 to 2005. *Expert Systems with Applications*, 2007, 33(1), p. 135-146.

[9] Brusilovsky, P., Sosnovsky, S., and Shcherbinina, O. QuizGuide: Increasing the Educational Value of Individualized Self-Assessment Quizzes with Adaptive Navigation Support. In J. Nall and R. Robson (Eds.), *Proceedings of E-Learn,* 2004, p. 1806-1813.

[10] Brusilovsky, P. and Sosnovsky, S. Individualized exercises for self-assessment of programming knowledge: An evaluation of QuizPACK. *Journal on Educational Resources in Computing (JERIC)*, 2005, 5(3), p. 6.1-6.22.

[11] Tan, P., Steinbach, M., and Kumar, V. Introduction to Data Mining. *Pearson-Addison Wesley Publishers*, Boston, USA, 2006.

[12] Ueno, M. Online outlier detection of learners' irregular learning processes. In Romero, C. and Ventura, S. (Eds.) *Data Mining in E-Learning*, 2006, p. 261-278.

Predicting Students Drop Out: A Case Study

Gerben W. Dekker[1], Mykola Pechenizkiy[2] and Jan M. Vleeshouwers[1]
g.w.dekker@student.tue.nl, {m.pechenizkiy, j.m.vleeshouwers}@tue.nl
[1]Department of Electrical Engineering, Eindhoven University of Technology, the Netherlands
[2]Department of Computer Science, Eindhoven University of Technology, the Netherlands

Abstract. The monitoring and support of university freshmen is considered very important at many educational institutions. In this paper we describe the results of the educational data mining case study aimed at predicting the Electrical Engineering (EE) students drop out after the first semester of their studies or even before they enter the study program as well as identifying success-factors specific to the EE program. Our experimental results show that rather simple and intuitive classifiers (decision trees) give a useful result with accuracies between 75 and 80%. Besides, we demonstrate the usefulness of cost-sensitive learning and thorough analysis of misclassifications, and show a few ways of further prediction improvement without having to collect additional data about the students.

1 Introduction

The monitoring and support of the first year students is a topic that is considered very important at many educational institutions. At some of the faculties yearly student enrollment for a bachelor program can be lower than desired, and when coupled with a high drop out rate of freshmen the need in effective approaches for predicting student drop out as well as identifying the factors affecting it speaks for itself.

At the Electrical Engineering (EE) department of Eindhoven University of Technology (TU/e), the drop out rate of freshmen is about 40%. Apart from the department's aim to enforce an upper bound to the drop-out rate, there are other reasons to want to identify successful and unsuccessful students in an early stage. In the Netherlands, there is the legal obligation that universities have to provide students with the necessary support to evaluate their study choice. In general, students who choose to pursue their study career at another institution, should do this at an early stage. For EE students there is a very concrete reason to evaluate before the end of the first semester: the EE program of the nearby Fontys University of Applied Science accepts TU/e drop outs in their curriculum until the beginning of January, without any time losses involved. Besides, there is always a subset of students which the department considers a "risk group", i.e. students who may be successful but who need extra attention or specific individual care in order to succeed. Detecting this risk group in an early stage is essential for keeping these students from dropping out. It enables the department to direct its resources to the students who need it most.

Current approach at EE department. To support students in making this decision, every enrolled student receives a study advice in December. This advice tells the student whether or not he or she is encouraged to proceed his study career at the faculty. It is based upon the grades and other results of the student so far and upon information obtained from 1st-semester-teachers and student-mentors, examined and interpreted by the department's student counselor. The final semester examinations are not taken into account, because they are in January; postponing the advice until after the results are known would preclude students from switching to Fontys. The advices seem to be quite accurate in practice: students who are assessed as potentially successful are in general the same students that are successful after a year. Moreover, the students who are not encouraged to proceed their current study program, generally do not continue into the second year.

The objectives. Despite the success, the assessment remains unsatisfactory because of its rather subjective character. Therefore, a more robust and objective founding of the process may lead to advices which are more consistently followed up by students. Besides, a closer analysis is likely to lead to an improved selection process.

First of all, the department is interested in which of the currently available student data are the strongest predictors of success, and in the performance of this predictor. Obviously, the lower the predictor's quality, the more the department is curious to know what information makes the current assessment work. If the predictor quality is high, the department's interests are directed towards: (1) using the predictor as a back-up of the current assessment process; (2) identifying success-factors specific to the EE program; (3) identifying what data might result in a further increase of the predictor quality, and as a consequence, collect these data; (4) considering a more differentiated view on the risk group; (5) modifying the assessment process time-line, resulting in an earlier prediction, ideally even before entering the study. Furthermore, if strong predictors for academic success can be found, these will also be used to gain understanding of success and risk factors regarding the curriculum. Awareness of these factors by teachers, education personnel and management will help to select appropriate measures to support the risk group, eventually resulting in a decrease of the drop-out rate.

In this paper we present the results of the educational data mining case study aimed to address these identified issues. First, we discuss related work on addressing the problem of student dropout (Section 2). Then, we consider the settings of our EDM case study and present the analysis of classification results (Section 3). In Section 4 we present the further evaluation of one of the models. We conclude this paper with a summary of the results and discussions of further work in Section 5.

2 Background and Related Work

The topic of explanation and prediction of academic performance is widely researched. In the earlier studies, the model of Tinto [12] was the predominant theoretical framework for considering factors in academic success. Tinto considers the process of student attrition as a socio-psychological interplay between the characteristics of the student entering university and the experience at the institute. This interaction between the student's past and the academic environment leads to a degree of integration of the student into this new environment. According to this model, a higher degree of integration is directly related to a higher commitment to the educational institute and to the goal of study completion. Later studies tried to operationalize this model identifying the factors like peer group interactions, interactions with faculty, faculty concern for student development and teaching, academic and intellectual development, and institutional and goal commitments that affect the student's integration [10]. These factors proved to have a predictive capacity across different institutions, and showed therefore to be a potential tool in identifying students who might drop out. Other studies tried to identify the significant factors in a more detailed way. Many studies included a wide range of potential predictors, including personality factors, intelligence and aptitude tests, academic achievement, previous college achievements, and demographic data and some of these factors seemed to be stronger than others, however there is no consistent agreement among different studies ([1], [3], [5], [13]). One of the recent European studies [3] has confirmed that sex (only in technical schools), age at enrollment, score on pre-university examination, type of pre-university education, type of financial support, father's level of education and whether or not living at the university town may all have an impact on the drop out. All studies show that academic success is dependent on many factors, where grades and achievements, personality and expectations, as well as sociological background all play a role.

The use of data-mining techniques in this field, known as educational data mining (EDM), is relatively new. The methodology is not yet transparent and it is not clear which data mining algorithms are preferable in this context. Clustering as means of data exploration and classification for building predictors have been tried in [4]. Association analysis has become also a popular approach in EDM [7], while one of the recent EDM case studies indicates that it is easy to underestimate the required efforts and overestimate the usefulness of this technology for small datasets [6]. The results of the case study presented in [2] indicate that Bayesian networks and neural networks are consistently outperformed by decision tree algorithms on relatively small educational datasets. However, the related work is still too scarce and in general it is hard to conclude from the recent studies (e.g. [2], [4], [8], [11]) which approach should be favored or even to measure whether learnt models outperform more traditional ways of predicting academic success.

3 Prediction of student drop out

In this case study we consider data collected over the period 2000 – 2009 that contains information about all the students being involved in the EE program. We selected a target dataset of 648 students who were in their first year phase at the department and came either from VWO (which is pre-university secondary education) or from polytechnical education (finishing at least a year of education at a polytechnical school grants access to university too). The latter group is a minority of about 10% of the considered students in the dataset.[1]

In order to get labels for the supervised learning of predicting models the students are classified in the following way: if a student was able to get his *propedeuse* (in the

[1] The further discussion of background knowledge and different issues related to the data preprocessing, data cleaning and transformation processes goes beyond the scope of this paper. An interested reader can find this information in the online technical report at http://www.win.tue.nl/~mpechen/research/edu.html.

Netherlands, a diploma which a student acquires after having successfully completed the first year at a university) in three years, he is classified as successful, and otherwise as unsuccessful.

We considered three datasets: a dataset with pre-university data only containing 495 instances (242 instances classified as unsuccessful, 253 instances classified as successful), each described with 13 attributes (Appendix A), a dataset with university grades only containing 516 instances (253 instances classified as unsuccessful, 263 instances classified as successful), each described with 74 attributes (for each of the 37 available courses we have two attributes saying how many attempts were taken, and what the highest grade was), and dataset with both sets of attributes containing also 516 students (missing values for pre-university data were replaced with zeros).

In our experimental study we used several popular Weka [14] classifiers (with their default settings unless specified otherwise). We compared the two decision tree algorithms CART (*SimpleCart*) and C4.5 (*J48*), a Bayesian classifier (*BayesNet*), a logistic model (*SimpleLogistic*), a rule-based learner (*JRip*) and the Random Forest (*RandomForest*). We also considered the *OneR* classifier as a baseline and as an indicator of the predictive power of particular attributes.

These classifiers are run on the dataset containing the pre-university data. We used 10-fold cross validation for estimating generalization performance. The statistical significance of differences in performance of *OneR* and other learners is tested with the two-sided paired t-tester in Weka's Experimenter, using a significance level of 5%.

3.1 Classification with pre-university or university data only

The classification accuracies for the dataset containing only the pre-university related data are shown in Table 1. The OneRule classifier reached the accuracy of 68% taking the *VWO Science mean* as a predictor. None of the other classification algorithms was able to learn a model which would outperform it (statistically) significantly.

Attribute ranking (with respect to the class attribute) according to the information gain criterion showed that the *VWO Science mean*, *VWO main* and *VWO Math mean* were by far the best attributes in information gain (information gains 0.16, 0.13, 0.12 respectively), with the next "closest" attribute *VWO Year* lagging behind (0.05). Furthermore, these three attributes are highly correlated and therefore it is logical to expect it would be hard to learn a more complex and yet generalizable classifier with a relatively small dataset. Learning a classifier with feature selection also does not improve the results a lot. Learning a *J48* tree using only the three mentioned attributes gives an average accuracy of 71%.

Table 1. Classification accuracy on pre-university dataset

Classifiers	*OneR*	*CART*	*J48 -M 2*	*J48 -M 10*	*BayesNet*	*Logit*	*JRip*	*RF*
Accuracy	0.68	0.68	0.70	0.69	0.71	0.69	0.70	0.65

The same classification techniques were applied to the dataset with the university grades (Table 2). The OneRule algorithm results in the classifier which checks the grade for Linear Algebra (*LinAlgAB*), and decides positive if this grade is bigger than 5.5 (that is exactly the minimum for passing a course). Again we can see that more sophisticated classification techniques do not improve accuracy very much. However, it is worth noticing that the CART classifier is statistically significantly better than the base line with a classification accuracy that is 4.8% higher on average.

Table 2. Classification accuracy on university grades dataset

Classifiers	*OneR*	*CART*	*J48 -M 2*	*J48 -M 10*	*BayesNet*	*Logit*	*JRip*	*RF*
Accuracy	0.76	0.81 o	0.79	0.79	0.75	0.79	0.78	0.80

o – statistically significant improvement

The CART classifier learnt a compact tree with five leaves and uses *LinAlgAB* as root of the tree, and *CalcA*, *Calc1* and *Project nAttempts* as further discriminators. It is worth noticing that the grades of the Networks course are not used at all, while some of its attributes have higher information gains. Correlation analysis however does show that correlation between Linear Algebra and Networks attributes is rather strong, but weak between Linear Algebra and Calculus attributes.

3.2 Classification with complete data

Classification accuracies for the dataset containing both pre-university and university related data are shown in Table 3 (column indexes correspond to those in Tables 1 and 2).

Table 3. Accuracy and rates of total dataset

Classifiers	*OneR*	*CART*	*J48 -M 2*	*J48 -M 10*	*BayesNet*	*Logit*	*JRip*	*RF*
Accuracy	0.75	0.79	0.80 ○	0.80	0.75	0.79	0.77	0.79
True positives	0.64	0.79 ○	0.80 ○	0.75 ○	0.72 ○	0.79 ○	0.73 ○	0.82 ○
False negatives	0.36	0.21 ○	0.20 ○	0.25 ○	0.28 ○	0.21 ○	0.27 ○	0.18 ○
True negatives	0.86	0.80 ●	0.80 ●	0.84	0.79	0.80 ●	0.82	0.77 ●
False positives	0.14	0.20 ●	0.20 ●	0.16	0.21 ●	0.20 ●	0.18	0.23 ●

○, ● – statistically significant improvement or degradation

It can be seen that these accuracies are comparable with those achieved on the dataset with university related data only. Apparently, the pre-university data does not add much independent information that can improve classification accuracy. However, we can see that the trees learnt with *J48* are now statistically significantly better than the base line model. The other tree-based classifiers also achieve reasonable accuracy, while the *Bayes Net* and *JRip* algorithms slightly fall behind.

To get a better insight on the performance of classifiers, the scoring of the algorithms is shown in more detail now. A remarkable fact is that the base line model has a higher false negative rate than all other models. This is an interesting finding, because according to the student counselor it is better to give an erroneous positive advice to a student who should actually be classified as negative, than to give a erroneous negative advice to a student who should be classified as positive. Cost-sensitive learning can be used to balance classification accuracies or boost the accuracy for a particular type of prediction.

3.3 Boosting accuracy with cost-sensitive learning

In order to “advice” a classification algorithm to prefer one type of misclassification to another a cost matrix (that has a direct mapping to the confusion matrix) is commonly used as an input to a meta classifier:

	classified as negative	classified as positive
actual negative	$C(-,-)$	$C(-,+)$
actual positive	$C(+,-)$	$C(+,+)$

By choosing the weights $C(i, j)$ in a certain way we can achieve a more balanced classification in case of severe class imbalances (using the diagonal entries), or a more cost-effective classification (using the off-diagonal entries).

Since cost matrices are equivalent under scaling, and we only want to increase the cost of false negatives over false positives, it suffices to build a matrix with only one free coefficient and structure [[0 1] [C 0]], with C > 1.

Since our experiments favored tree-based learners we used *J48*, *J48graft* and *CART* as base classifiers in Weka’s *CostSensitiveClassifier*. To prevent the tree from growing too big, we used the *CfsSubsetEval* feature subset selection algorithm that tries to select the most predictive attributes with low intercorrelation. The *J48* and *J48graft* classifiers were forced to have at least 10 instances for each node in order to prevent overfitting and unnecessarily complex models. Combining these *CART*, *J48* and *J48graft* with the two ways of using the cost matrix in cost-sensitive approach (*data weighing* and *model cost*), six experiments were conducted using F measure for defining the precision-recall tradeoff (we used $\beta = 1.5$). For each combination, the settings giving the highest F measure is presented in Table 4. The tree learnt with the “plain” *J48* is presented in the first data column.

The results indicate that it is necessary to sacrifice some of the achieved accuracy to be able to shape the misclassification. Only model 5 achieves a high accuracy and a high F measure, all other models lose in accuracy if F is increased. During the experiment, it became clear that there is not much room for enhancement: if recall increased to values higher than 85%, the overall accuracy results were unacceptable. The only exception is model 7 (notice the size of this tree being much larger comparing to other models and also seem to be too detailed to be meaningful for decision making).

In some cases, small trade-offs could be made changing C. Compare for instance model 5 with model 6: a three percent point drop in accuracy gives a three percent rise in recall.

The created decision trees are remarkably similar: in every tree the *LinAlgAB* attribute is dominant, with *CalcA* as first node in most of the cases. When *NetwB* is chosen as the first node, the recall is lower, although the difference is too small to draw decisive conclusions.

Table 4. Accuracy results with cost-sensitive learning

	1	2	3	4	5	6	7	8
Type	J48	J48	J48	CART	CART	CART	J48graft	J48graft
Learner option	-	Data weighting	Model cost	Data weighting	Model cost	Model cost	Data weighting	Model cost
$C(+,-)$	-	2	3	2	3	4	4	3.2
Confusion matrix	212 41 651 98	175 78 49 214	206 47 62 201	169 84 50 213	201 52 57 206	181 72 51 212	160 93 31 232	161 92 56 207
Accuracy	0.79	0.75	0.79	0.74	0.79	0.76	0.76	0.71
Precision	0.83	0.73	0.81	0.72	0.80	0.75	0.71	0.69
Recall	0.75	0.81	0.76	0.81	0.78	0.81	0.88	0.79
F_β	0.77	0.79	0.78	0.78	0.79	0.79	0.82	0.76
nLeaves	5	11	5	10	7	7	21	8
TreeDepth	3	6	3	5	4	4	8	5
Root node	LinAlgAB <= 5	LinAlgAB <= 5	LinAlgAB <= 5	LinAlgAB < 5.5	LinAlgAB <= 5.5	LinAlgAB <= 5.5	LinAlgAB <= 5	LinAlgAB <= 5
First node	NetwB <= 5.7	CalcA <= 5	NetwB <= 5.7	VWO-Science-mean	CalcA < 5.15	CalcA < 5.15	CalcA <= 5	CalcA <= 5
Second node	CompB-nAttempts	CompB-nAttempts	CompB-nAttempts	LinAlgA, CalcA	VWO-Science-mean	VWO-Science-mean	VWO-Science-mean	LinAlgB, NetwA2

4 Further evaluation of the obtained results

As the final step, we examined one of the models (model 7 from Table 4) in more detail to see if we can gain better understanding of the classifier errors. The student counselor compared all the wrongly classified instances of model 7 with his own given advices to check for interesting patterns. One of the first assessed things was the question whether the learned model is incorrect or the classification criterion is chosen incorrect. To examine this, two methods were used. Firstly, the false negative and false positive sets have been checked manually by the student counselor. His conclusions were that about 25% of the false negatives should be true negatives instead. This finding might indicate a wrong classification measure. Concerning the false positive set a conclusion is less obvious: about 45% of this set was classified as positive by the student counselor as well as by the tree, but did not meet the classification criterion. A substantial subset of these students have chosen not to continue their bachelor program in Electrical Engineering although all indications for a successful continuation were present. Qualifying these students as *false* positive does not seem to be appropriate. So from this evaluation based on domain expertise we can conclude that some of the mistakes might be due to the classification measure, and some of them raise suspicion on behalf of the learned model.

The second way to check the viability of the model is to compare the results obtained with this classifier with respect to the three class classification problem, i.e. identifying first manually the third so-called *risk* group and then checking whether wrongly classified students will be in the risk class (that would indicate that the learned model is actually more accurate and also that it has difficulties in predicting the students who are difficult to classify into success or failure categories per se). However, we observe that only 25% of the misclassified instances are in this category. It should be noted that this is still twice as much as the risk students ratio in the total dataset. Therefore, this also indicates that the learned model should be improved. Furthermore, 25% of the instances in the false positive class would be classified as *good* using the three-class classification thus indicating a real difference between two classifiers. So from this test we can also conclude that the model as well as the classification criterion should be revised.

After the analysis of errors, the misclassified sets are looked up in the database to search for meaningful patterns manually. A very clear pattern popped up immediately: almost all misclassified students did not have a database entry concerning *LinAlgAB* (and therefore were mapped to zero). Checking out different students showed that there are many possible reasons now to have a zero value in the *LinAlgAB* record: a) a student might be of a cohort in which the *LinAlgAB* exam was in January or later; b) a student might have not shown up during the exam; and c) a student might have taken another way to get its *LinAlgAB* grade: in some years it was possible to bypass the regular exam by doing the subexams *LinAlg1*, *LinAlg2*, *LinAlg3*, *LinAlg4* and *LinAlg5*. A student succeeding in taking this path can well be an excellent student, but gets a zero mark for the *LinAlgAB* attribute. Due to this effect, 216 of the 516 students do have a zero entry in their *LinAlgAB* record (of which 155 instances were classified as unsuccessful and 61 instances as successful). Moreover, the same effect will play a role for the other courses too. Given the dominant position of the *LinAlgAB* attribute in the decision trees generated in section 3.3, attempts in completing the data-set should be considered worthwhile.

5 Conclusions and Future work

Student drop out prediction is an important and challenging task. In this paper we presented a data mining case study demonstrating the effectiveness of several classification techniques and the cost-sensitive learning approach on the dataset from the Electrical Engineering department of Eindhoven University of Technology.

Our experimental results show that rather simple classifiers give a useful result with accuracies between 75 and 80% that is hard to beat with other more sophisticated models. We demonstrated that cost-sensitive learning does help to bias classification errors towards preferring false positives to false negatives.

Surprisingly (according to the student counselor) the strongest predictor of success is the grade for the Linear Algebra course, which has in general not been seen as the decisive course. Other strong predictors are grades for Calculus, Networks and the mean grade for VWO Science courses. The most relevant information is collected at the university itself: the pre-university data can be summarized into a few attributes.

The in depth model evaluation pointed to three major improvements that can be assessed. Firstly, a key improvement in this dataset would be to find a solution for the changing course organization over the set. Aggregating the available information about student performance for a course in a way that can be used for all students in the dataset might prevent the type of misclassifications that is now strongly prevalent. A second, related improvement would be a better way to encode grades in general. Mapping all unknown or not available information to zero showed to be not effective. Specifically, Linear Algebra grades should be available. A more advanced solution dealing with missing values also can be considered in this respect.

The quality of the classification criterion is the third improvement that might be considered. The simple binary classification as used in this study has some disadvantages: a negative classification can only be given after three years, and there is no guarantee that a student who does not get his propedeuse after three years will be not successful in the long run. Also, students who do not receive a propedeutical diploma, should not necessarily be “disqualified”: they may have had different motives to discontinue their studies. This touches on a more fundamental topic: it is not easy to find an objective way of classifying students. In this paper we experimented with the so-called 0/1 loss and cost-sensitive classification. AUC optimization is also one of the directions of further work.

As a final remark we would like to point out that this study shows that learning a model on less rich datasets (i.e. having only pre-university and/or first-semester data) can be also useful, provided the data preparatory steps are carried out carefully.

6 References

[1] Herzog, S. Measuring determinants of student return vs. dropout/stopout vs. transfer: A first-to-second year analysis of new freshmen. In *Proc. of 44th Annual Forum of the Association for Institutional Research* (AIR), 2004.

[2] Herzog, S. Estimating student retention and degree-completion time: Decision trees and neural networks vis-a-vis regression, *New Directions for Institutional Research*, p. 17-33, 2006.

[3] Lassibille, G., Gomez, L. N. Why do higher education students drop out? Evidence from Spain, *Education Economics* 16(1), p. 89-105, 2007.

[4] Luan, J. Data mining and its applications in higher education. *New Directions For Institutional Research*, p. 17-36, Spring 2002.

[5] Parmentier, P. La reussite des etudes universitaires: facteurs structurels et processuels de la performance academique en premiere annee en medecine. PhD thesis, Catholic University of Louvain, 1994.

[6] Pechenizkiy, M., Calders, T., Vasilyeva, E., De Bra, P. Mining the student assessment data: Lessons drawn from a small scale case study. In Proc. of the 1st Int. Conf. on Educational Data Mining (EDM’08), p. 187-191, 2008.

[7] Romero, C., Ventura, S. Educational data mining: a survey from 1995 to 2005, *Expert Systems with Applications* 33, p. 135-146, 2007.

[8] Romero, C., Ventura, S., Espejo, P. G., Hervas, C. Data mining algorithms to classify students. In *Proc. of the 1st Int. Conf. on Educational Data Mining* (EDM’08), p. 187-191, 2008.

[9] Superby, J., Vandamme, J.-P., Meskens, N. Determination of factors influencing the achievement of the first-year university students using data mining methods. In *Proc. of the Workshop on Educational Data Mining* at ITS’06, p. 37-44, 2006.

[10] Terenzini, P. T., Lorang, W. G., Pascarella, E. T. Predicting freshman persistence and voluntary dropout decisions: a replication, *Research in Higher Education* 15(2), p. 109-127, 1981.

[11] Thai Nge, N., Janecek, P., Haddawy, P. A comparative analysis of techniques for predicting academic performance. In *Proc. of 37th Conf. on ASEE/IEEE Frontiers in Education*, 2007.

[12] Tinto, V. Limits of theory and practice in student attrition, *Journal of Higher Education* 53, p. 687-700, 1982.

[13] Touron, J. The determination of factors related to academic achievement in the university: implications for the selection and counseling of students, *Higher Education* 12, p. 399-410, 1983.

[14] Witten, I. H., Frank, E. *Data Mining: Practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann, 2 ed., 2005.

Appendix A. Attributes in the pre-university dataset.

Attributes	Type	Remarks
IDNR	numerical	Used only to check data sanity
VWO Year	nominal	Major changes in Dutch education system, {1..4, ’n/a’}
VWO Profile	nominal	The pre-university education curriculum, {1..5, ’n/a’}
VWO nCourses	numerical	The number of courses taken.
VWO mean	nominal	{ n/a, poor, average, above average, good, excellent }
VWO Science nCourses	nominal	{ n/a, < 3, 3, >3 }
VWO Science mean	nominal	As VWO mean
VWO Math nCourses	nominal	{n/a, 0,1,2}
VWO Math mean	nominal	As VWO mean
HO Education	nominal	{n/a, electrical, technical, other}
HO Year	nominal	Same categories as VWO Year
HO Grade	nominal	As VWO mean
GapYear	nominal	{n/a, < -1, -1, 0, 1, >1 }
Classification	nominal	{-1, 1}

Using Learning Decomposition and Bootstrapping with Randomization to Compare the Impact of Different Educational Interventions on Learning

Mingyu Feng, Joseph E. Beck and Neil T. Heffernan
{mfeng, josephbeck, nth}@wpi.edu
Computer Science Department, Worcester Polytechnic Institute

Abstract. A basic question of instructional interventions is how effective it is in promoting student learning. This paper presents a study to determine the relative efficacy of different instructional strategies by applying an educational data mining technique, learning decomposition. We use logistic regression to determine how much learning is caused by different methods of teaching the same skill, relative to each other. We compare our results with a previous study, which used classical analysis techniques and reported no main effect. Our results show that there is a marginal difference, suggesting giving students scaffolding questions is less effective at promoting student learning than providing them delayed feedback. Our study utilizes learning decomposition, an easier and quicker approach of evaluating the quality of ITS interventions than experimental studies. We also demonstrate the usage of computer-intensive approach, bootstrapping, for hypothesis testing in educational data mining area.

1 Introduction

The field of Intelligent Tutoring Systems (ITS) is often concerned with what type of educational intervention is more effective on promoting student learning. A handful of studies [e.g. 8, 11, 14, 15] have been conducted on comparing different variants of tutoring and feedback strategies, such as worked-out examples, tutored problem solving. One popular method of determining whether one type of instruction is more effective than the other is to run a randomized controlled study. Although the method is shown to be useful, a major problem with the controlled study approach is that it can be expensive. A study could involve many users (in each condition), be of considerable duration, and require the administration of pre/post tests. To address this problem, Beck [2] introduced an approach called learning decomposition, an easy recipe to enable researchers to answer questions such as what type of practice is most effective for helping student to learn a skill. Instead of focusing on performance gain from pretest to posttest, learning decomposition leverages item-level data during a study and is concerned with how student performance changes while students are using the tutor. This approach is a modification of the learning curve analysis technique [12] that has been used in evaluating the efficacy of instructional contents. For instance, Koedinger and Mathan [9] compared learning outcomes associated with two types of feedback in the context of a spreadsheet tutor. Martin et al. [10] evaluated ITS using learning curves, and they also described the impact of changes in system's setup on the results of such analysis.

The ASSISTment system [13] is an online system that presents math problems to students who range from approximately 12 to 16 year olds in middle school or high school to solve. When a student has trouble solving a problem, the system usually provides instructional assistance to lead the student through by breaking the problem into scaffolding steps, or displaying hint messages on the screen, upon student request. Time-stamped student answers are logged into our database. In the ASSISTment system, when the authors create the instructional content, they may use different tutoring strategies. Razzaq et al. [14] reported a randomized controlled experiment that examined effects of the level of tutor-student interaction on helping students learn math skills. In this paper, we take a second look at the study and use a different approach to analyze the experiment: learning decomposition and bootstrapping with randomization test.

The goal of this paper includes 1) Comparing the relative impact of various educational interventions in the ASSISTment system by doing an item-level analysis. 2) Presenting a case study of applying the learning decomposition technique to a domain, mathematics, other than reading where the technique has been shown to be valuable [1, 2, 3]. 3) There has been little prior use of bootstrapping with educational data [1]. We show how bootstrapping can be used with learning decomposition.

2 Methods

2.1 Experimental design

As mentioned in section 1, the experiment reported in [14] compared the efficacy of interventions with various levels of interactions. The experiment included three conditions: *scaffolding + hints*; *hints on demand*; *delayed feedback*. When a problem first appears on the screen, we refer to this as the "main question." If students answered the main question wrong, the "scaffolding + hints" (referred to as scaffold condition) condition forced them to do the scaffolding questions, which would ask them to complete each step required to solve a problem, and they must answer all scaffolding questions correctly to proceed. While in the "hints on demand" (referred to as hint condition) these students only received a message indicating their answer was wrong, and the hint messages, which would tell them the same information without expecting an answer to each step, would only appear when they press the Hint button on the screen. The third condition was a delayed feedback condition (referred to as delayed condition) where students got no immediate feedback from the tutor (even if they answered the question wrong) until they have finished all of the problems in the experiment, whereupon they received worked out solutions to all of the problems.

In this experiment students were presented an assignment with two pretest problems organized in one pretest section, four experiment problems in one experiment section, and four post-test problems in the posttest section that addressed the topic of interpreting linear equations, an 8th-grade (approximately 13-year old) math skill. Two of the pretest problems were repeated in the post-test. Problems in the same section were shown in random order. Students were randomly assigned to the three conditions with equal probability. There were 366 eighth grade students from the Worcester Public Schools in Worcester, Massachusetts who participated in the experiment: 131 students were in honors level classes and 235 were in regular math classes. For the analysis in this paper, we exclude students who got both pretest problems correct (assuming they have mastered the skill), and those who did not finish all problems in the experiment. This leaves 300 students in our data set, with 101 in the delayed condition, 106 in the hint condition and 93 in the scaffold condition. We check to make sure students in all three conditions do not differ on their incoming knowledge. The mean and 95% confidence interval of average pretest and posttest scores for the three groups are listed in Table 1, and Table 2.

Table 1. Statistics of students' performance on pretest

Condition	Mean	Std. Err	95% confidence interval
Delayed	0.342	0.023	[0.297, 0.387]
Hint	0.354	0.022	[0.311, 0.397]
Scaffold	0.323	0.025	[0.274, 0.372]

Table 2. Statistics of students' performance on posttest

Condition	Mean	Std. Err	95% confidence interval
Delayed	0.381	0.025	[0.332, 0.430]
Hint	0.368	0.024	[0.321, 0.415]
Scaffold	0.341	0.025	[0.292, 0.390]

2.2 Approach

2.2.1 Introducing learning decomposition

Beck [2] introduced the idea of learning decomposition that extends the classic exponential learning curve by taking into account the heterogeneity of different learning opportunities for a single skill. The standard form of exponential learning curve can be seen in Equation 1. In this model, parameter *A* represents students' performance on the first trial; *e* is the numerical constant (2.718); parameter *b* represents the learning rate of a skill, and *t* is the number of practice opportunities the learner has at the skill.

$$performance = A * e^{-b*t}$$

Equation 1. Standard exponential learning curve model

$$performance = A * e^{-b*(B*t_1+t_2)}$$

Equation 2. Learning decomposition model

The model as shown in Equation 1 does not differentiate different types of practice, but just counts up the total number of previous opportunities. In order to investigate the difference two types of practice (I and II), the learning opportunities are "decomposed" into two parts in the model in Equation 2 in which two new variables t_1 and t_2 are introduced in replace of *t*, and $t = t_1 + t_2$[1]. t_1 represents the number of previous practice opportunities at one type I; and t_2 represents the number of previous opportunities of type II. The new parameter *B* characterizes the relative impact of type I trials compared to type II trials. The estimated value of B indicates how many trials that one practice of type I is worth relative to that of type II. For example, a B value of 2 would mean that practice of type I is twice as valuable as one practice of type II, while a B value of .5 indicates a practice of type I is half as effective as a practice of type II. The basic idea of learning decomposition is to find an estimate of weight B that renders the best fitting learning curve. Equation 2 factors the learning opportunities into two types, but the decomposition

[1] Interestingly, t1 + t2 does not have to equal t, as shown in [16] and as we will show in this paper.

technique can generalize to *n* types of trials by replacing t with $B_1*t_1 + B_2*t_2 + \ldots + t_n$. Thus, parameter B_i represents the impact of a type *i* trial relative to the "baseline" type *n*.

2.2.2 Decomposing learning opportunities

Now that we have described the model of learning decomposition, we want to "decompose" students' learning opportunities in our data set in order to fit such a model. Various metrics can be used as an outcome measurement of student performance. For instance, Beck [4] chose to model student's reading time since it is a continuous variable. Although one may argue for other indicators, e.g. students' help requests and response times, we simply choose to use the correctness of student's first attempt to a problem as an outcome measure of their performance. A "1" in the data indicates the student got a problem correctly on the first attempt, and thus proceeded to the next problem without getting any instructional assistance, while a "0" means he failed on the first try and received certain type of tutoring from the system, depending on which condition the student has been assigned into.

When it comes to a nominal variable, in our case, dichotomous (0/1) response data, a logistic model should be used. Now learned performance, (i.e. *performance* in Equation 2), is reflected by odds ratio of success to failure. Equation 3 represents a logistic regression model for learning decomposition.

$$performance = \frac{P(correct\_answer)}{P(wrong\_answer)} = A * e^{-b*(B*t_1+t_2)} = e^{\alpha+\gamma*(B*t_1+t_2)}$$

Equation 3. Logistic models for learning decomposition

Equation 3 can be transformed to an equivalent form as below:

$$\ln(\frac{P(correct\_answer)}{P(wrong\_answer)}) = \alpha + \gamma * (B * t_1 + t_2)$$

Where α, γ are the new representation of students' initial knowledge and their learning rates of a skill on the logistic scale. Now that we have determined our outcome variable and functional form of the model, all that remains is to decompose learning opportunities into components. We split student trials into four groups largely on the basis of experimental condition as below. Therefore, the number *n* is equal to 4 in this analysis.

- *hint_wrong_trial* (t_h) indicates the number of prior wrong trials that a student in the hint condition had encountered
- *scaffold_wrong_trial* (t_s) counts the number of prior wrong trials that a student in the scaffold condition had made before.
- *delayed_wrong_trial* (t_d) is similar to the other two variables but for students in the delayed condition. However, it is specially calculated such that the prior encounters will not increase until the student was presented the explanations for all the problems in order to address the fact that the learning actually happened at the moment when the explanations were shown. Note that by doing so we assume that simple exposure to the content does not cause learning. It is also worth pointing out that although the approach of learning decomposition itself does not require the administration of pretests and posttests, in this particular analysis, we do need the results of posttest to be able to detect the impact of explanations (in the delayed feedback condition) on student learning.

- *Others* (t_o). Because what we really care about is the relative effectiveness of the different tutoring interventions during the experiment, we did not differentiate students' practice trials on pretest, posttest and trials where they gave a correct answer to the experiment problem. Instead, all these trials are combined together into the group *others*. Actually, since the number trials on pretest and posttest are the same for all students, it is the correct trial on experiment problems that matter in this group.

For those readers who are familiar with ASSISTments vocabulary, it is also worth pointing out that although in the experiment there are three versions of the experiment problems with different associated interventions, one for each condition, we created one unified problem ID for all the three versions, since the main questions are the same.

Table 3. Decomposed response data of student A

Student ID	Section	Problem ID	Correct?	Previous trials (t)	Decomposed previous trials			
					Hint_ wrong_ trial (t_h)	Scaffold_ wrong_ trial (t_s)	Delayed_ wrong_ trial (t_d)	Others (t_o)
A	Pretest	Pre-1	1	0	0	0	0	0
A	Pretest	Pre-2	0	1	0	0	0	1
A	Exp	Exp2	0	2	0	0	0	2
A	Exp	Exp4	1	3	0	1	0	2
A	Exp	Exp1	0	4	0	1	0	3
A	Exp	Exp3	1	5	0	2	0	3
A	Posttest	Pre-1	1	6	0	2	0	4
A	Posttest	Post-2	0	7	0	2	0	5
A	Posttest	Pre-2	1	8	0	2	0	6
A	Posttest	Post-1	1	9	0	2	0	7

Table 3 shows a sequence of time-ordered trials of a student who was assigned in the *scaffolding* condition. The student finishes all three sections, fails on one of the pretest problems, but learns to solve the problem during the experiment as suggested by a correct answer to the same problem in the posttest. The right part of Table 3 shows the corresponding data after the trials are decomposed into component parts. Since the student is in the scaffolding condition, all values in the hint_wrong_trial and delayed_wrong_trial are zero. He solves the first encountered experiment problem wrong (row 3), which cause an increase on the value of scaffold_wrong_trial from zero to 1 (row 4). Again, he gets the third experiment problem wrong (row 5), and then the value of scaffold_wrong_trial increases from 1 to 2 in row 6. The value of trial for others just increases by one whenever a pretest problem, a posttest problem or a correct trial was encountered. For instance, the student answers the second encountered experiment problem correct (row 4), and thus the value of others increased by 1 (row 5). Limited by space, we only demonstrate the decomposition process for a student in the scaffold condition; the process for the hint condition would be identical. For the delayed feedback condition, since the student would not see the feedback until after all of the experimental trials, it is necessary to model that differently. In the delayed condition, the number of delayed-wrong trials would stay as zero until it jumps to be 2 in row 7, since the student would have seen the two explanations after finishing the experimental questions. This problem requires a rather novel use of learning decomposition, and some care in accounting for when the learning opportunities actually occur.

2.3 Results

We fit the model shown in Equation 3 to the decomposed data in the statistics software package R (see www.r-project.org). To account for variance among students and items, student IDs and unified problem IDs are also introduced as factors. By taking this step we account the fact that student responses are not independent of each other, and properly compute statistical reliability and standard errors. Also, by fitting our model in this manner we do not suffer the scaling problems mentioned by [10] since all three conditions have the same intercept (i.e. *A* parameter). After the model is fitted, it outputs estimated coefficients for every condition, as shown in Table 4. The result suggests that the delayed feedback, estimated coefficient being 0.720, is more effective at helping student learn the skill than the other two conditions, esp. the scaffolding condition for which the coefficient estimate is 0.633. In prior work with this experiment [14], the authors reported that they did not find any main effect. It is possible to use the estimated coefficients (B) and standard errors in Table 4 to perform a statistical z-test, as we did in [7]. However, there is a bit of serendipity: the first author was conducting some exploratory analyses using resampling to see how stable the parameter estimates really were. It appeared that there was little overlap between the estimates for the scaffold and delayed conditions. Therefore, we decide to test this approach formally using bootstrapping [5] and randomization tests [6].

Table 4. Coefficients of logistic learning decomposition model

Coefficients	Estimate (B)	Std. Error	z value	Pr(>|z|)
Others	-0.235	0.034	-6.816	9.33e-12 ***
Hint_wrong_trial	0.706	0.091	7.760	8.52e-15 ***
Scaffold_wrong_trial	0.633	0.103	6.175	6.62e-10 ***
Delayed_wrong_trial	0.720	0.054	13.224	< 2e-16 ***

Bootstrapping is a modern, computer-intensive, general purpose approach to statistical inference, falling within a broader class of resampling methods [5]. It involves the construction of a number of resamples of the observed dataset by random sampling with replacement from the original data set; and each resample is independent (conditioned on the original sample) and identically distributed. Although bootstrapping was developed as techniques for parameter estimation, it can be used for hypothesis testing as well. In general, first we make a null hypothesis. Then we draw repeated samples from the original data set under the condition that the null hypothesis is true, and then we reject the null hypothesis if the statistic computed from the observed dataset is unlikely under the null hypothesis, or otherwise retain the null. In this particular analysis, the hypothesis we would like to test is "The delayed feedback strategy promotes learning more or less effectively than the scaffold (or hint) strategy." Correspondingly, the null hypothesis would be “There is no difference on learning promotion between the delayed and scaffolding strategies.”

Specifically, we follow the following steps to test our hypothesis.

Step 1: Decide on a metric to measure the relative effectiveness between delayed feedback and scaffolding strategies. For this example, we choose the difference between the estimated coefficients of Delayed_wrong_trial and Scaffold_wrong_trial.

Step 2: Calculate the metric on the original data. The results in Table 3 provides B(Delayed_wrong_trial) – B(Scaffold_wrong_trial), equal to .087.

Step 3: Bootstrap the original data with randomization to construct samples where the null hypothesis is true

```
Repeat $N$ times {
    Repeat M times ($M$ = the number of students in our original data set) {
            Sample data of one student (with replacement) from the original data;
            Randomly allocate the student into one of the three conditions: delayed, scaffold, or hint by changing the “Condition” label of each data point
            Re-compute the number of prior trials for the student according to the newly assigned condition;
    }
    Train logistic learning decomposition model on the re-sampled data, and record B(Delayed_wrong_trial) – B(Scaffold_wrong_trial) ;
}
```

In our case, we pick the repeated times N to be 500, and M is 300 as there are 300 students in our data set.

Step 4: Check how likely our original result is under the null hypothesis, and reject or retain the null hypothesis. After the bootstrapping process, we obtain a list of difference between Delayed_wrong_trial and Scaffold_wrong_trial, totally 501 cases including our original result. Then we rank the list descending, and found that the original result was at the 95 percentile, the 25th in the ranking order, which suggests that the probability of the original result has a probability of less than 5%. Although it is tempting to think we have $p < 0.05$, this methodology is actually conducting a one-tailed test. Thus, the two-tailed value is $p = 0.1$. Therefore, we have a marginally reliable result that *delayed feedback* is better than *scaffold* + *hint*, and giving students delayed feedback seems causing more learning than requesting them to finish a series of scaffolding questions.

To complete the story, we repeat the same process compare the other two pairs: delayed vs. hint conditions, and scaffold vs. hint conditions, but find that they are comparable to each other at helping students learning the math skill in ASSISTments.

3 Conclusion

This paper explored the research question of measuring the instructional effectiveness of different tutoring interventions, using the learning decomposition technique. We found that presenting students with delayed feedback works better than breaking problems into scaffolding questions. We also used bootstrapping with randomization to test the statistical reliability of the finding.

Typically, there are two reasons for the usage of learning decomposition (or any educational data mining technique). The first is repurposing a previous experiment's data to answer a new question. The second is using EDM techniques to "zoom in" and detecting subtle effects that previous approaches failed to report. Previous works on learning decomposition [3, 4, 16] have been focusing on the first reason, while in this paper we focus on the second reason through an item level analysis and bootstrapping.

One open question is why bootstrapping plus randomization gives different results than the parametric method of using estimated coefficients and standard errors to derive an analytic p-value. We did a z-test using estimated coefficients and standard errors given in Table 4 and obtained p = .4. Typically computationally intensive techniques are *less* powerful than parametric ones, unless one or more of the parametric tests' assumptions have been violated. We are not sure where the problem lies, but suggest caution in interpreting standard error terms from logistic regression models using learning decomposition.

The contribution of the paper lies in three aspects. First, we found that there is a main effect in a randomized controlled study that delayed feedback tutoring strategy is more effective than giving students scaffolding questions in ASSISTments. While previous analysis using ANOVA failed to detect such an effect, we were able to do so by conducting an item level analysis using EDM techniques. Second, we showed how learning decomposition can be applied in the domain of mathematics to use observational data to estimate the effectiveness of different tutoring strategies. It provides evidence that the learning decomposition is not domain specific. This simple, low cost approach is generally applicable to a variety of ITS that focus on different domains for identifying variances in educational effectiveness of interventions. Also, our use of learning decomposition is novel in that we are careful to consider when various aspects of an intervention occur, and do not give credit for a learning opportunity that has not yet happened (the delayed-wrong condition). Third, the process described in this paper serves as a demonstration of how bootstrapping approach and randomization tests can be employed in the educational data mining field.

Acknowledgements

This research was made possible by the U.S. Department of Education, Institute of Education Science (IES) grants #R305K03140 and #R305A070440, the Office of Naval Research grant # N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the

Spencer Foundation. All the opinions, findings, and conclusions expressed in this article are those of the authors, and do not reflect the views of any of the funders.

References

[1] Beal, C. and Cohen, P. (2005). Comparing apples and oranges: Computational methods for evaluating student and group learning histories in intelligent tutoring systems. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th International Conference on Artificial Intelligence in Education,* pp. 555-562. Amsterdam: ISO Press.

[2] Beck, J.E. (2006). Using learning decomposition to analyze student fluency development. ITS2006 Educational Data Mining Workshop 2006. Jhongli, Taiwan.

[3] Beck, J.E. (2007). Does learner control affect learning? In *Proceedings of the 13th International Conference on Artificial Intelligence in Education*. pp. 135-142.

[4] Beck, J.E. (2008). How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students. *Proceedings of the 9th Intelligent Tutoring System Conference*. pp. 353-362.

[5] Davison, A. C.; Hinkley, D. (1997). *Bootstrap Methods and their Applications.* Cambridge: Cambridge Series in Statistical and Probabilistic Mathematics.

[6] Edgington E.S. (1995). Randomization tests, 3rd ed. New York: Marcel-Dekker, 1995.

[7] Feng, M., Heffernan, N.T., Beck, J. (In press). Using learning decomposition to analyze instructional effectiveness in the ASSISTment system. In Graesser, A., Dimitrova , V., Mizoguchi, R. (Eds.). Proceedings of the 14th International Conference on Artificial Intelligence in Education (AIED-2009). Brighton, UK, 2009.

[8] Kim, R, Weitz, R., Heffernan, N. & Krach, N. (accepted). Tutored Problem Solving vs. “Pure”: Worked Examples. Accepted by Cognitive Science Society Annual 2009 Conference.

[9] Koedinger, K.R. and Mathan, S. (2004). Distinguishing qualitatively different kinds of learning using log files and learning curves. in ITS 2004 Log Analysis Workshop. 2004. Maceio, Brazil. p. 39-46.

[10] Martin, B. Koedinger, K., Mitrovic, A. and Mathan, S. (2005). On Using Learning Curves to Evaluate ITS. Proceedings of the 12th international conference on Artificial Intelligence in Education, AIED2005 Amsterdam, pp. 419-426

[11] Mathan, S. & Koedinger, K. R. (2003). Recasting the Feedback Debate: Benefits of Tutoring Error Detection and Correction Skills. In Hoppe, Verdejo & Kay (Eds.), Artificial Intelligence in Education:Shaping the Future of Learning through Intelligent Technologies. Proceedings of AI-ED 2003 (pp. 39-46). Amsterdam, IOS Press.

[12] Newell, A. & Rosenbloom, P.S. (1981). Mechanisms of skill acquisition and the law of practice. In J.R. Anderson (Ed.). Cognitive skills and their acquisition. Lawrence Erlbaum Associates: Hillsdale, NJ. p.1-56.

[13] Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R, Walonoski, J.A., Macasek. M.A., Rasmussen, K.P. (2005). The Assistment Project: Blending Assessment and Assisting. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) Proceedings of the 12th International

Conference on Artificial Intelligence in Education, pp. 555-562. Amsterdam: ISO Press.

[14] Razzaq, L., Heffernan, N. T., Lindeman, R. W. (2007). What Level of Tutor Interaction is Best? In Luckin & Koedinger (Eds.) Proceedings of the 13th Conference on Artificial Intelligence in Education. IOS Press. pp 222-229.

[15] Sweller, J. & Cooper, G. A. (1985). The Use of Worked Examples As a Substitute For Problem Solving In Learning Algebra. Cognition and Instruction, 2, 59–89.

[16] Zhang, X., Mostow, J. & Beck, J.E. (2007). All in the (word) family: Using learning decomposition to estimate transfer between skills in a Reading Tutor that listens. Educational Data Mining Workshop at the 13th International Conference on Artificial Intelligence in Education.

Does Self-Discipline impact students' knowledge and learning?

Yue Gong, Dovan Rai, Joseph E. Beck and Neil T. Heffernan
Computer Science Department, Worcester Polytechnic Institute

Abstract. In this study, we are interested to see the impact of self-discipline on students' knowledge and learning. Self-discipline can influence both learning rate as well as knowledge accumulation over time. We used a Knowledge Tracing (KT) model to make inferences about students' knowledge and learning. Based on a widely used questionnaire, we measured students' level of self-discipline. When we analyzed the relation of students' self-discipline with their knowledge attributes, we found that high self-discipline students had significantly higher initial knowledge, but there is no consistent relationship of learning while using the tutor. Moreover, higher self-discipline students seemed more careful with respect to making careless mistakes.

1 Introduction

Intellectual attributes (e.g., long term memory, ability to think abstractly) and nonintellectual attributes (e.g., motivation, self-discipline) both contribute to a student's academic performance [1]. Intelligent Tutoring Systems (ITS) have focused on cognitive aspects over last 25 years and are now becoming increasingly aware of non-cognitive traits like motivation, engagement, flow etc. [5,6]. However, self-discipline is still not a major area of exploration in ITS though it has been one of the key areas in psychology and sociology [8,9]. Given that a lot of such large-scale psychosocial studies have been able to demonstrate a positive relation of self-discipline with performance and achievement, we were interested in two questions:

- Does self-discipline have a significant impact when it comes to knowledge acquisition within ITS?
- Does the ITS community need to consider self-discipline while designing ITS?

In this paper, we are trying to use educational data mining technique with fine grained models to get a crisper look at the impact of self-discipline on students' cognitive aspects. We used Knowledge tracing (KT) [3], an established approach to model student knowledge. We can observe students' performance in an ITS over a period of time and make inferences about their latent characteristics like knowledge level and learning across the time. Once we detect those attributes, we can see the impact of self-discipline on the immediate learning and prior accumulation. Besides learning, self-discipline can influence other attributes like consistency and carefulness that can improve performance given the same content knowledge.

2 Methodology

For this study, we used data from ASSISTment, a web-based math tutoring system. We used the data from 171 twelve- through fourteen-year old 8th grade students in urban school districts of the Northeast United States. These data consisted of 74,394 log records of ASSISTment during the period Jan 2009-Feb 2009. We recorded performance records of each student across time slices for 106 skills (e.g. area of polygons, Venn diagram, division, etc).

2.1 Measuring self-discipline

For exploring student individual differences in self-discipline, we employed a questionnaire survey, Brief Self-Control Scale (BSCS; [9]) in December 2008 before the students used the tutor. BSCS is a 13-item questionnaire to measure self-regulatory behavior in four domains: thoughts, emotions, impulses, and performance.

Each question (e.g. “I am lazy”, “I am good at resisting temptation”) asks the respondent to choose from a 5-point Likert scale answer list: a. Very much like me, b. Mostly like me, c. Somewhat like me, d. A little like me, e. Not like me at all. We assigned each response -2, -1, 0, +1, +2 points respectively. We dropped an original survey question (“I wish I had more self-discipline.”) as we find difficult to interpret whether agreeing this statement would imply high self-discipline or low.

2.2 Lie test

While using self-report measures, we have no way of ensuring that respondents don’t lie or answer haphazardly. Therefore, we created three criteria to detect lies and out of total 171 students we dropped 31 from our analyses.

1. The questionnaire asked students for their gender. Twelve students gave an incorrect response. Suspecting them not being serious about the survey, we excluded those students from our study.
2. Some students might be randomly picking answers and therefore we checked for consistency in their answers. Among 12 questions in the survey, for 8 of them “Very much like me” implies low self-discipline (e.g. “I have a hard time breaking bad habits”), and for 4 of them, “Very much like me” implies high self-discipline (e.g. “I am good at resisting temptation”). For both types of questions we used the scoring system in Section 2.1. If a student answered “Very much like me” for a question of the first type, he received -2 points. If he answered “Not like me at all” for a question of the second type, he received +2 points. The two responses consistently tell that he has low self-discipline. The sum is zero. But if he had answered “Very much like me” in the second type, the answers are not consistent and the sum of responses is -4. Similarly, if he had answered “Not like me at all” in both questions, that would be still inconsistent and sum would be +4.
3. For each student, we took average of points in both types of questions (since the groups are of unequal size) and summed the two averages and calculated the absolute value. The sum value can range from 0 (completely consistent) to 4 (completely inconsistent). Based on the questionnaire composition and distribution of the sum from our data, we found 1.6 to be a reasonable cut point and dropped 11 students with sum greater than 1.6.
4. We selected two pairs of questions which are basically asking same trait in opposite ways. For example “I do certain things that are bad for me, if they are fun” and “I refuse things that are bad for me” state the same trait. We cropped students who are saying “very much like me”/ “Mostly like me” or “not like me at all” in both questions. There were 19 such students among which 5 were already excluded from step 2.

Finally, our dataset narrowed down to 134 students, with their 68285 log records. We excluded 10% of the records and 20% of the students. For each student, we had 12-dimensional vectors representing their responses corresponding to each survey question. We performed a factor analysis to reduce data dimensions and we used the strongest factor as the student's self-discipline score.

2.3 Knowledge tracing model

We used knowledge tracing in Dynamic Bayesian Networks (DBN), see Figure 1, to make inferences about student knowledge based on his performance.

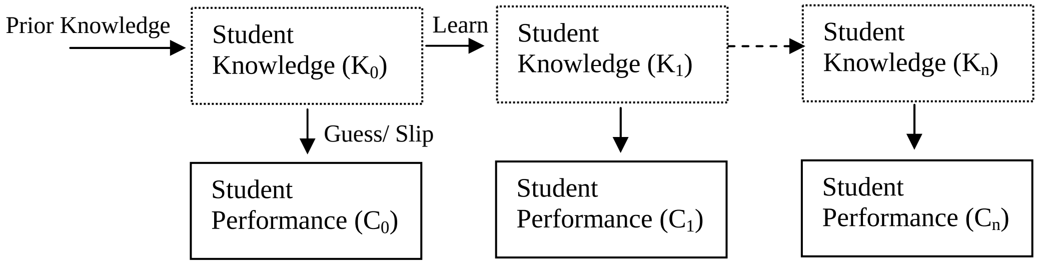

Prior Knowledge
Student Knowledge (K_0)
Learn
Student Knowledge (K_1)
Student Knowledge (K_n)
Guess/ Slip
Student Performance (C_0)
Student Performance (C_1)
Student Performance (C_n)

Figure 1. Knowledge tracing model: Dynamic Bayesian network

Student performance is assumed to be a noisy reflection of student knowledge, mediated by two performance parameters guess and slip. The guess parameter represents the fact that the student may sometimes generate a correct response in spite of not knowing the correct skill. For example, some ASSISTment items are multiple choice, so even a student with no understanding of the question could generate a correct response for those. The slip parameter acknowledges that even students who understand a skill can make an occasional careless mistake [3]. The learning rate parameter estimates the probability that student learns new knowledge that he has not known before.

$Prior\ Knowledge = Pr\ (K_0{=}True)$

$Guess = Pr\ (C_n{=}True \mid K_n{=}False)$

$Slip = Pr\ (C_n{=}False \mid K_n = True)$

$Learning\ rate\ = Pr(K_n = True \mid K_{n-1}{=}False\ )$

We used Bayes Net Toolkit for Student Modeling (BNT-SM [4]), which inputs data and a compact XML specification of a Bayes net model to describe causal relationships among student knowledge and observed behavior. BNT-SM gives us knowledge parameters, prior knowledge and learning as well as performance parameters, guess and slip.

3 Results

3.1 Knowledge tracing model per skill

Based on self-discipline score, we divided students into three equal-sized groups having relatively high, medium and low self-discipline level. For each subgroup, we trained separate knowledge tracing models, and thus estimated knowledge and performance parameters that corresponded to each group. We trained a knowledge tracing model for each of the 106 skills. I.e. observe all the training data across all students for each skill and derive a set of parameters (Prior knowledge, learning, guess, slip) for each skill. Then, for each self-discipline subgroup, we calculated the median values across all the skills (see Table 1). We report median rather than mean values to avoid unnecessarily weighting outliers. However, in accordance with standard convention, our statistical analyses are based on the means rather than medians.

Table 1: Knowledge and performance parameters for self-discipline groups

	High		**Medium**		**Low**	
	Value	**P-value (vs. medium)**	**Value**	**P-value (vs. Low)**	**Value**	**P-value (vs. High)**
Prior Knowledge	0.56	4.96E-8	0.48	0.45	0.49	3.62E-6
Learning	0.13	5.68E-6	0.17	0.001	0.14	0. 186
Guess	0.38	0.015	0.36	2.38E-6	0.32	1.72E-8
Slip	0.16	8.37E-18	0.20	0.591	0.21	1.09E-18

From Table 1, we see that for prior knowledge, the high self-discipline students are statistically higher than the medium group, and the medium and low groups are statistically tied. Meanwhile, high self-discipline students made more correct guesses and fewer slips relative to their lower self-discipline peers. A higher guess parameter should not be viewed as a bad thing. Consider that guess means the ability to answer a question despite not having mastered the skill. Consider two students with similar partial knowledge and one takes more care to figure the right answer while other quickly asks for help. The model will treat this as a guess by the first student. Such behavior seems related to self-discipline. Similarly, students who are more careful and detail-oriented will make fewer slips (keep in mind that a "slip" is defined as making a mistake in spite of the skill being known). The result shows that higher self-discipline students have more prior knowledge and they are more concerned and careful on their task.

However, we received an inconsistent pattern in the learning parameter. The learning rate of the medium self-discipline group is higher than both the high and low groups. We were concerned with the possibility of overestimating the learning parameter in the medium group by giving the guess parameter less weight, while underestimating it in the high group by giving guess more weight. This concern is due to problems with estimating knowledge tracing parameters [8]. For example, a high "guess" parameter can result in students performing well, but allegedly having little knowledge. Since student knowledge is not directly observable, it is hard to validate the parameter estimates and we are left trusting our model that two groups could perform equally well but one group knows less (see [8] for a fuller discussion of the problems of underdetermined models). To guard against this concern, we also plotted student performance as a function of practice opportunity so that we can see the cumulative effect of the knowledge and performance parameters in students' future performance for each level of self-discipline.

By using the four parameters of each subgroup and the knowledge tracing equations listed below, we computed the theoretical performance curves for each of them. Specifically, we initialize *knowledge* to be K_0. After each practice opportunity, we update *knowledge* in formula I (below) as the new likelihood of the student knows the skill after the previous practice. Also we compute *performance*, the probability of the student will respond correctly in the current practice opportunity, by using formula II to combine the estimated knowledge with the slip and guess parameters. Intuitively, the probability of making correct response is dependent on student's knowledge given that he does not slip and also on his probability to make right guess in absence of the knowledge.

*I: Knowledge=previous knowledge + (1-previous knowledge)*learning rate*

II: Performance=knowledge(1-slip) + (1-knowledge)*guess*

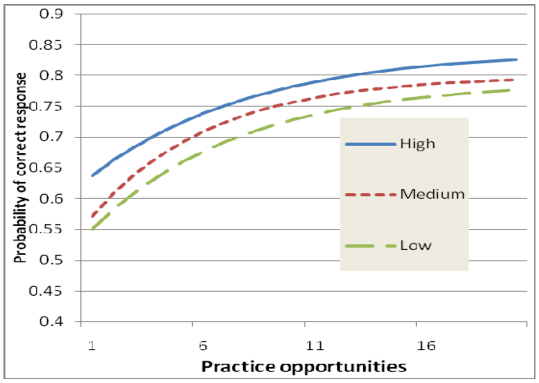


Figure 2a. Theoretic performance curve of three self-discipline groups

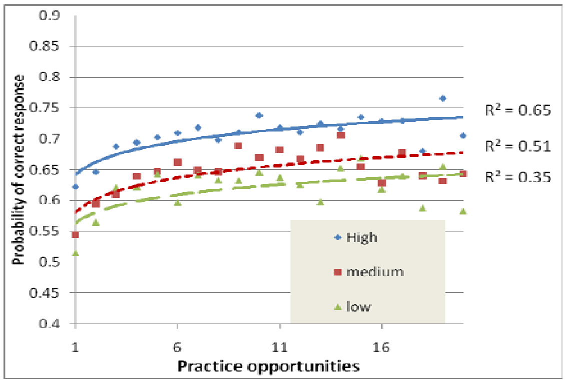


Figure 2b. Real performance curve of three self-discipline groups

From the performance curve in Figure 2a, we see that the combined effects of the four model parameters result in higher self-discipline students performing better. The real performance curve in Figure 2b also showed a similar trend. One interesting observation is to examine the best-fit power curves for each group. The high self-discipline students are performing more lawfully (i.e. higher R^2) than those with low self-discipline, suggesting students with higher self discipline are more consistent. Simply looking at the learning parameters does not tell the whole story. High group students might be learning slower but they are better able to use their partial knowledge to perform better—at least that is what our model is suggesting.

Based on all these findings, we built a causal model that unifies cognitive and non-cognitive aspects of our students. While knowledge parameters like prior knowledge and learning are cognitive attributes, the performance parameters, guess and slip are more related to non-cognitive attributes. This model accounts for the results in Table 1, and suggests the performance parameters might be an interesting avenue of research in their own right (typically the knowledge parameters are of more interest).

3.2 Knowledge tracing model per student

While training a KT model per skill is the regular approach, it is also possible to instead train one model per student by observing his responses in all questions across skills. The model then estimates a set of parameters (prior knowledge, guess, slip and learning) for each student which represents his aggregate performance across all skills. We then looked for a relationship between the student's self-discipline score and his knowledge parameters (prior knowledge and learning). As seen in Table 2, self-discipline is positively correlated with student's prior knowledge (K0), but again there is no statistically reliable correlation with the learning parameter. In the other words, students with higher self-discipline have more incoming knowledge than their lower self-discipline classmates. However, self-discipline seems not to contribute student's ability to learn more in each learning opportunity within the tutor. Perhaps higher self-discipline results in having more learning opportunities rather than learning more from each one?

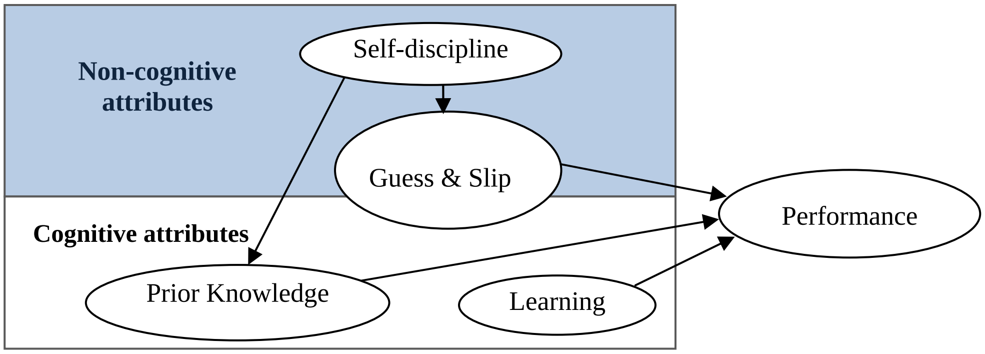


Figure 3: Causal model of cognitive and non-cognitive attributes for academic performance

Table 2: Correlation of self-discipline and knowledge parameters

Correlation of self-discipline with	Correlation	P-value	N
Prior knowledge (K_0)	0.29	0.001	134
Learning	0.13	0.127	134

We also found an interesting observation that self-discipline is highly correlated with the number of problems solved. We were then confronted with two possibilities: either higher self-discipline students are more on task and solve more problems, or students with higher self-discipline have higher knowledge and so need less help and solve problems more quickly. When we did partial correlation within these three variables, as seen in Table 3, we found evidence for the latter possibility. Once we account for prior knowledge, there is no relationship between self-discipline and number of problems solved.

We built a causal model, Figure 4, based on the finding that the higher self-discipline students in fact solved more problems as they were equipped with more knowledge and, perhaps surprisingly, not because they were on task more. The direct correlation between self-discipline and knowledge is 0.29, and between knowledge and number of problems solved is of 0.55. The partial correlations are more interesting. The partial correlation of self-discipline and number of problems, partialing out knowledge was only 0.11. Thus, there is not a direct relation between the two. The partial correlation of knowledge and number of problems solved, partialing out self-discipline is 0.52, i.e. the correlation is relatively unaffected. Thus, knowledge appears to be the direct causal link for number of problems solved, and self-discipline is causally upstream of knowledge.

Table 3: Partial correlation of self-discipline, prior knowledge and # of problems solved

	Correlation	p-value
# of problems solved vs. Self-discipline (prior knowledge as control)	0. 11	.22
# of problems solved vs. prior knowledge (Self-discipline as control)	0. 52	.000

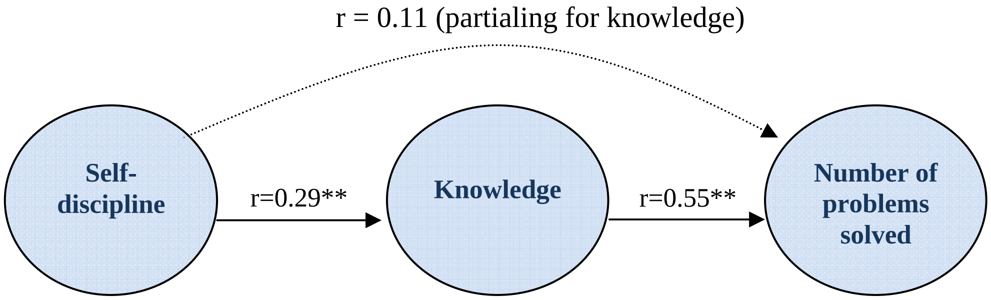

r = 0.11 (partialing for knowledge)
Self-discipline
r=0.29**
Knowledge
r=0.55**
Number of problems solved

Figure 4: Causal model of self-discipline, knowledge and number of problems solved

3.3 Model validation

KT model parameters can be sensitive to erroneous factors like wrong priors, insufficient data, etc. Therefore, we were curious to try some validation of our model parameters with external data.

To validate our model, we used results from a pretest and posttest on the same set of students. The pretest consisted of a 33-item algebra quiz on the subset of knowledge components that we are using in our models. After a month, the students were presented with posttest with exactly the same questions as in the pre-test. The pretest was performed when the students started using the tutor, and the student's score is used to indicate the amount of incoming knowledge before using ASSISTment. Therefore it works as a standard against which to validate student prior knowledge (K_0) that we estimated in our models.

Also, we calculated students' estimated performance after 8 practice opportunities (P_8) as they practice 8 times on average for each skill during the one month period. We estimate performance from prior knowledge, learn, guess and slip parameters as given by knowledge and performance equations mentioned in 3.1. The correlation of P8 with post-test can be a measure of validation of the other three parameters.

There is strong positive correlation between the student pretest scores and model's estimation of their prior knowledge. P_8 and posttest scores are also reliably correlated even when we partial out initial knowledge (K_0). I.e. our performance measure is capturing student learning, not just the student's overall level of knowledge. In addition, Figure 5 shows student learning between pretest and posttest.

The gains in Figure 5 are consistent with our KT model results. The high self- discipline group has higher incoming knowledge than both groups and their final performance is also highest. But when it comes to learning, the medium group appears to have the highest gain. So, we considered some possibilities for the explanation of lower learning in high group. One reason for lower learning in high group could be due to the fact that they already have high knowledge and it is harder to have more gain when we start from higher value. For example going from 50 to 60 is easier than going from 80 to 90.

Table 4: Correlation of prior knowledge (K_0) and P_8 vs. pre and post-test respectively

	Correlation	p-value	N
K_0 vs. Pretest	0.80	3.80E-31	134
P_8 vs. Posttest	0.77	2.57E-25	123
P_8 vs. Posttest (partialing out K_0)	0.34	1.58E-15	123

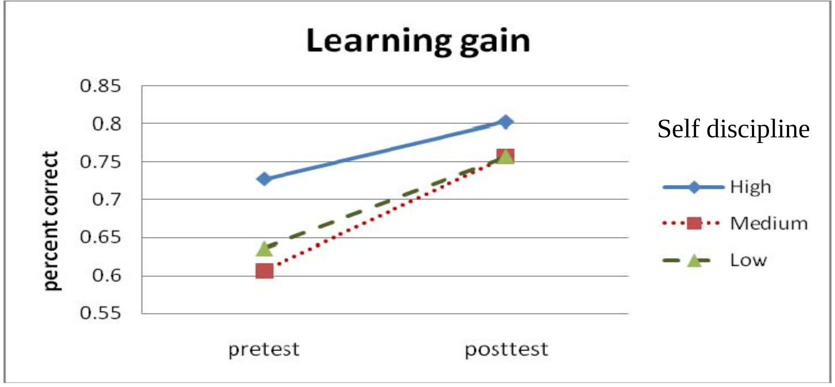


Figure 5: Comparison of learning gain

To test this possibility, we divided pre-test scores into three bins and performed an ANOVA. We treated pretest and self-discipline as factors in our model since we did not necessarily expect a linear effect (as would be implied by treating them as covariates). Table 5 shows the estimated marginal means of gain score for each level of the factors.

Table 5: ANOVA analysis gains by pretest and self-discipline

Pretest	Self-discipline	Mean gain	Std. Error	95% Confidence Interval	
				Lower Bound	Upper Bound
(>80%)	High	0.03	0.03	-0.03	0.10
	medium	0.06	0.07	-0.07	0.19
	low	-0.03	0.05	-0.14	0.08
(40-80%)	high	0.07	0.03	0.02	0.13
	medium	0.15	0.02	0.10	0.20
	low	0.12	0.03	0.07	0.18
(<40%)	high	0.15	0.05	0.04	0.25
	medium	0.25	0.05	0.16	0.34
	low	0.05	0.04	-0.03	0.13

From the result in Table 5, we see that medium group has higher learning in all bins (i.e. they are learning faster no matter what their starting level in pre-test is). Therefore, it appears that the medium group indeed has higher learning and maybe having a balance of self-discipline and some spontaneity helps in having better learning gains. However, their lower incoming knowledge makes the idea of a higher learning rate difficult to reconcile. We choose to leave this as an open discussion for further experimentations in future.

4 Contributions

Psychosocial studies have been based on performance measures like report cards, GPA, income, college admission, etc. [1,8,9]. But, our fine grained model gives us the tools to measure their performance and also latent attributes like knowledge, learning, and even guess and slip. We have found that the impact of self-discipline on students using computers is complex, and appears to influence knowledge and performance while using the tutor. We have constructed a causal model of the impacts of cognitive and non-cognitive attributes on performance within an ITS, and showed that the variability in performance is not only dependent upon cognitive attributes, but also on other non-cognitive aspects like carefulness and self-discipline.

We modified the regular approach to train KT model with data per skill and instead estimated per-student parameters. Although a per-student model trained on prior users is not useful to ITS designers (since it does not apply to new students using the system!), performing parameter estimation at the individual level can open new ways to make different analyses with other individual characteristics. With this new approach, we were able to make correlations of students' pre-test and post test with their knowledge and performance estimations, thus validating the model parameters.

5 Future work and conclusions

Our current method of estimating self-discipline relies upon a self-reported survey administered once. There can be problems of both over- and under-reporting. We could take advantage of the continuous data students generate, and construct a more robust estimate of self-discipline. It may also be possible to consider self-discipline a latent construct, similar to what we do for knowledge in knowledge tracing, and simultaneously estimate both parameters. Broadening the stream of ITS information to include observable measures like homework submission, attendance, usage of tutor, opinion of teachers and parents, etc. would make this possible.

In conclusion, high self-discipline students have higher incoming knowledge, as substantiated from both KT model parameters and pre-test score. However, the impacts do not appear to be substantial, and tutor designers probably do not have to explicitly account for self-discipline. The higher self-discipline group makes better guesses and makes fewer slips, which implies that the higher self-discipline group is more careful and detail oriented. The cumulative effect of learning, slip and guess makes the performance of higher self-discipline students better than that of their peers.

Acknowledgements

We would like to thank all of the people associated with creating the ASSISTment system listed at www.ASSISTment.org. We would also like to acknowledge funding from the National Science Foundation, the Fulbright Program for funding the second author and the US Department of Education and the Office of Naval Research for funding the other authors. All of the opinions expressed in this paper are those solely of the authors and not those of our funding organizations.

References

[1] Angela L. Duckworth and Martin E.P. Seligman, *Self-Discipline Outdoes IQ in Predicting Academic Performance of Adolescents*, Psychological Science, 16, 939-944

[2] Ivon Arroyo, Beverly Park Woolf, *Inferring learning and attitudes from a Bayesian Network of log file data*, 12th international conference on artificial intelligence in education, 2005.

[3] Albert T. Corbett and John R. Anderson, *Knowledge tracing: Modeling the acquisition of procedural knowledge, User modeling and User adapted interaction* 4:253-278, 1995.

[4] Joseph E. Beck, Kai-min Chang, Jack Mostow, Albert T. Corbett, *Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology. Intelligent Tutoring Systems* 2008: 383-394.

[5] Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A. & Koedinger, K. *Why students engage in "Gaming the System" behavior in interactive learning environments.* Journal of Interactive Learning Research. Chesapeake, VA: AACE. 19(2), 185-224

[6] Angel de Vicente and Helen Pain, *Informing the Detection of the Students' Motivational State: An Empirical Study*, 6th International Conference, ITS 2002.

[7] Joseph E. Beck and Kai-min Chang , *Identifiability: A Fundamental Problem of Student Modeling*, 11th International Conference, User Modeling, 2007.

[8] Borghans, Lex, Meijers, Huub and Ter Weel, Bas,*The Role of Noncognitive Skills in Explaining Cognitive Test Scores*(November 2006). IZA Discussion Paper No. 2429. Available at SSRN: http://ssrn.com/abstract=947088

[9]Tangney, J.,P., Baumeister, R.F., & Boone, A.L.(2004). *High self-control predicts good adjustment, less pathology, better grades, and interpersonal success.* Journal of personality, 72, 271-322.

Consistency of Students' Pace in Online Learning

Arnon Hershkovitz and Rafi Nachmias
{arnonher, nachmias}@post.tau.ac.il
Knowledge Technology Lab, School of Education, Tel Aviv University, Israel

Abstract. The purpose of this study is to investigate the consistency of students' behavior regarding their pace of actions over sessions within an online course. Pace in a session is defined as the number of logged actions divided by session length (in minutes). Log files of 6,112 students were collected, and datasets were constructed for examining pace rank consistency in three main situations: day/night sessions, beginning/end (for both situations, sessions of the same learning mode were taken), and a comparison between sessions from different learning modes. For each dataset, students were ranked twice, according to their pace in the two sub-groups, and these ranks were correlated. Results obtained with this study's data suggest that pace is sometimes not consistent, hence might not be considered as a characterizing measure for the whole learning period. A discussion of this study and further research is provided.

1 Introduction

Log files are the essential basis for many Data Mining research, however raw data from these files are usually being transformed into variables on which algorithms and statistical tests might be applied. In EDM research, all levels of aggregation into variables should be considered: keystroke level, answer level, session level, student level, classroom level, and school level [3]. While discussing individual differences between users (i.e., aggregating or estimating in student level), a question might arise: Do variables taken into consideration indeed characterize the learner (even regarding the limited context of domain and environment)? Not only that such a variable (e.g., session length, response time, intense of activity, preferred tasks) might introduce a large variance when repeatedly measured for the same student, there is also a possibility that this inconsistency represents a non-trait measure, hence this variable does not and should not represent a student.

In this study, we chose to examine the pace of actions within a Web-based learning environment. It is a time-related variable occasionally being calculated in the student level. However, in configurations where students have the freedom to choose when, where and what/how to learn, and while their sessions might extend over a long period (days or weeks) – it is not clear that a student has a "characterizing pace", and that we can try to compare students by their pace.

Moreover, pace measuring is just one example from a large set of variables often being used in student models, and an important purpose of this study is to shed light on some obstacles for using such variables.

2 Background

Logged data for calculating pace of activity in a learning environment, was studied – probably for the first time – almost twenty years ago in a Computer-based Instruction

(CBI) configuration [7]. The results suggested that "students exhibit a characteristic rate of responding or way of approaching CBI activities". Although this conclusion treats pace as measuring response or approach to activities, it seems that the basic definition of pace, as the researcher had defined it - number of activities completed, divided by total time on task – tend to be more cognitive than behavioral.

In fact, pace (also referred as *speed*, *rate*) is somehow a slippery term in EDM research, as it might relate to two different phenomena: a) Pace of learning – measured by completion rate per time-unit [7] or by time taken to complete a task – e.g., in [10, 16] (notice the difference in units between these two measures); b) Pace of action – measured by number of actions per time-unit [13, 14]. These two measurement are, of course, not independent, as pace of action might affect pace of learning, and vice versa: If we take, for example, two students with the very same cognitive skills needed for a given task but with different values of pace of action, the student which is more speedy has an advantage in completing the task quicker; on the other hand, student's pace of action might be affected by learning occurred or knowledge application needed between consecutive actions.

Although pace (in either interpretation) might change noticeably between tasks, it is sometimes being treated as characterizing the student for the whole learning period. Therefore, parameters measuring pace are being averaged over multiple sessions (as was previously done by the authors in [13]) or being calculated on the whole learning period level in the first place [8].

Considering pace as representing students might lead to a calculation of relative pace. For example, Beck's disengagement model [4] has a student-specific parameter of *reading speed*, for accounting inter-students variability; this parameter fine-tunes the model by considering the student's speed relative to the class' average, and is calculated and applied across all question types. Another relative calculation of time-related measuring is presented in [18], where student's *working time* was calculated as the ratio between the student's completion time for a given task divided by the class' average completion time. Both these studies rely on the hidden assumption that student's rank, regarding her or his activity's speed or time, is consistent over tasks and/or over time. The examination of that hidden assumption is the core of this research.

3 Methodology

To determine whether pace of action does characterize learners, we examined consistency of pace ranking, i.e., of students' ranking by their pace. If pace does characterize students, pace ranking is expected to be consistent (to a certain measure) over different situations. The following three situations were examined:

a) Day/night – median pace for each student is considered for calculating her or his rank in day/night sessions within the same learning mode
b) Over time – pace ranks are based on pace measures for beginning and last sessions within each learning mode. Second session was chosen to represent the beginning, since pace in first session might be greatly biased

c) Across learning modes – median pace in each mode serves as the basis for pace ranks.

In addition, we examined another situation, which is quite more technical: Pace ranks are based on median pace in two randomly-divided groups of sessions for each student (first, in general, and then within each learning mode).

Different datasets were constructed for each of the above situations, as will be described in section 3.4. Following is a description of the learning environment, the log file, the data collection and preprocessing, and the datasets construction.

3.1 The Learning Environment

A simple yet very intensive online learning unit was chosen as the research field. This fully-online environment focuses on Hebrew vocabulary and is accessible for students who take a face-to-face preparatory course for the Psychometric Entrance Exam (for Israeli universities). The online system is available for the participants from the beginning of the course and until the exam date (between 3 weeks and 3 months in total).

The system includes a database of around 5,000 words/phrases in Hebrew and, offers the students with a few learning modes: a) Memorizing, in which the student browses a table of the words/phrases along with their meanings; b) Practicing, in which the student browses the table of the words/phrases without their meaning. The student may ask for a hint or for the explanation for each word/phrase; c) Gaming; d) Self-testing, in the same format of the exam the students will finally take; and e) Searching for specific word/phrase. The first two modes (Memorizing, Practicing) have a very similar interface of a multi-page table in each row of which there is a word/phrase; while in the Memorizing mode, the meaning of that word/phrase is shown, in the Practicing mode it is hidden and will be revealed only upon the student's request.

3.2 Log File Description

The researched system logs the students' activity, thus each student is identified by a serial number. Each row in the log file documents a session, initiated by entering the system and ended with closing the application window. For each session, the following attributes are kept: starting date, starting/ending time, ordered list of actions and their timestamps; actions documented are every html/asp page in the system, not including actions within Java/Flash applets.

3.3 Data Collection and Preprocessing

For examining the research hypothesizes, we used logged data from April 2006 – May 2007. The original data included 181,111 sessions of 11,068 students. Cleaning was done for keeping only the following: a) active sessions – session that lasted at least one minute and less than one hour, and that had at least five documented actions; b) active students – students who had at least three active sessions. The cleaned log had 64,700 (active) sessions of 6,112 (active) students. Pace for each session was calculated as the number of actions in the session, divided by the session length (in minutes).

Next, we mapped and coded the actions within each session to one of the four learning modes: Memorizing, Practicing, Self-testing, Searching; gaming was not coded because most of the gaming-related pages are implemented in Java, and therefore they were not documented. Then, each session was coded into one of the four modes if at least 60% of its actions were of that same mode. It turned out that about 30% of the sessions were coded as "Memorizing", 20% were coded as "Practicing", only about 1% of the sessions were "Searching", and only a few sessions were "Self-testing"; the rest were not categorized to any of the modes (i.e., they were mixed sessions). Therefore, our study is focused only in the two eminent modes.

3.4 Constructing the Datasets for Testing the Hypotheses

Eight different datasets were constructed, in order to investigate the consistency of pace rank between day/night sessions, between beginning/end sessions, across learning modes, and among random divisions of the sessions. A detailed description is given in Table 1.

Table 1. Description of the datasets for investigating pace rank consistency

Dataset	Learning Mode(s)	Sessions Were Included for Students With…	Total Students	Total Sessions	Pace calculation for student-group
Dataset1$_M$ Day/night	Memorizing	at least 3 sessions in each group of day/night sessions	331	3,823	Median
Dataset1$_P$ Day/night	Practicing	at least 3 sessions in each group of day/night sessions	285	4,389	Median
Dataset2$_M$ Beginning/end	Memorizing	at least 3 Memorizing sessions	2,650	16,724	One sample
Dataset2$_P$ Beginning/end	Practicing	at least 3 Practicing sessions	1,358	11,409	One sample
Dataset3 Across modes	Memorizing + Practicing	at least 3 sessions of each mode (Memorizing, Practicing)	768	12,593	Median
Dataset4$_A$ Random division	All	no limitations	6,112	64,700	Median
Dataset4$_M$ Random division	Memorizing	at least 3 sessions in each of two randomly divided sub-groups of the sessions	758	8,445	Median
Dataset4$_P$ Random division	Practicing	at least 3 sessions in each of two randomly divided sub-groups of the sessions	526	7,739	Median

For each dataset, we sorted the students twice, according to their pace in the relevant sub-groups (the student with the highest pace was ranked as "1", the student with the second-highest pace was ranked as "2", and so on). These two ranks were correlated using Spearman's rho (ρ) and Kendall's tau (τ), two common alternatives for non-parametric correlation coefficients ([-1,1]) which are often being compared, however without a sharp recommendation towards neither of them [9, 12, 17]; it is known that the Kendall's coefficient is usually lower than the Spearman's.

4 Results

Day/Night Consistency

Results for $Dataset1_M$ and $Dataset1_P$, in which day/night situation was examined in the two learning modes, are given in Table 2. It might be concluded from the results that there is a significant relatively high correlation between pace ranks between day and night in both modes. It was also found that there is a significant difference when comparing means of pace values between day and night groups: Mean pace over night sessions was higher than the mean pace over day sessions; *t* values were 2.11* (*df=330*) for $Dataset1_M$, and 2.33* (*df=284*) for $Dataset1_P$.

Table 2. Day/night consistency of pace rank

Dataset	N (Students)	Mode	Group 1	Group 2	ρ	τ
$Dataset1_M$	331	Memorizing	Day	Night	0.59**	0.43**
$Dataset1_P$	285	Practicing	Day	Night	0.53**	0.39**

* p<0.05, ** p<0.01

Beginning/end Consistency

Results for $Dataset2_M$ and $Dataset2_P$, examining consistency of pace ranks over time, are given in Table 3. As might be seen, correlation coefficients are pretty low. On average, beginning and last sessions are differed by pace of action within them: Students tend to work faster at the end, as shown by *t* values of 3.33** (*df=2,649*) for $Dataset2_M$, and 3.64** (*df=1,357*) for $Dataset2_P$.

Table 3. Over time consistency of pace rank

Dataset	N (Students)	Mode	Sample 1	Sample 2	ρ	τ
$Dataset2_M$	2,650	Memorizing	2nd session	Last session	0.26**	0.18**
$Dataset2_P$	1,358	Practicing	2nd session	Last session	0.20**	0.14**

** p<0.01

Another way of looking at these results is to scatter plot a two-dimension representation of the students according to their ranks in both groups, and to look at the four quadrants formed by the median lines. If pace rank is consistent, it is anticipated that the faster students will be faster in both dimensions, and same for the slower students, hence quadrants I (top-right) and III (bottom-left) should be occupied with most of the dots (students).

For example, let's take a look at such a scatter plot for *Dataset2$_P$*, which relates to the beginning/end situation for the Practicing learning mode. The examination of pace rank consistency for this dataset showed a low yet significant correlation (ρ=0.20**). The scatter plot for this example is presented in Figure 1. According to our calculations, the first and the third quadrants each holds 30% of the dots, which means that the second and fourth quadrants hold together 40% of the students.

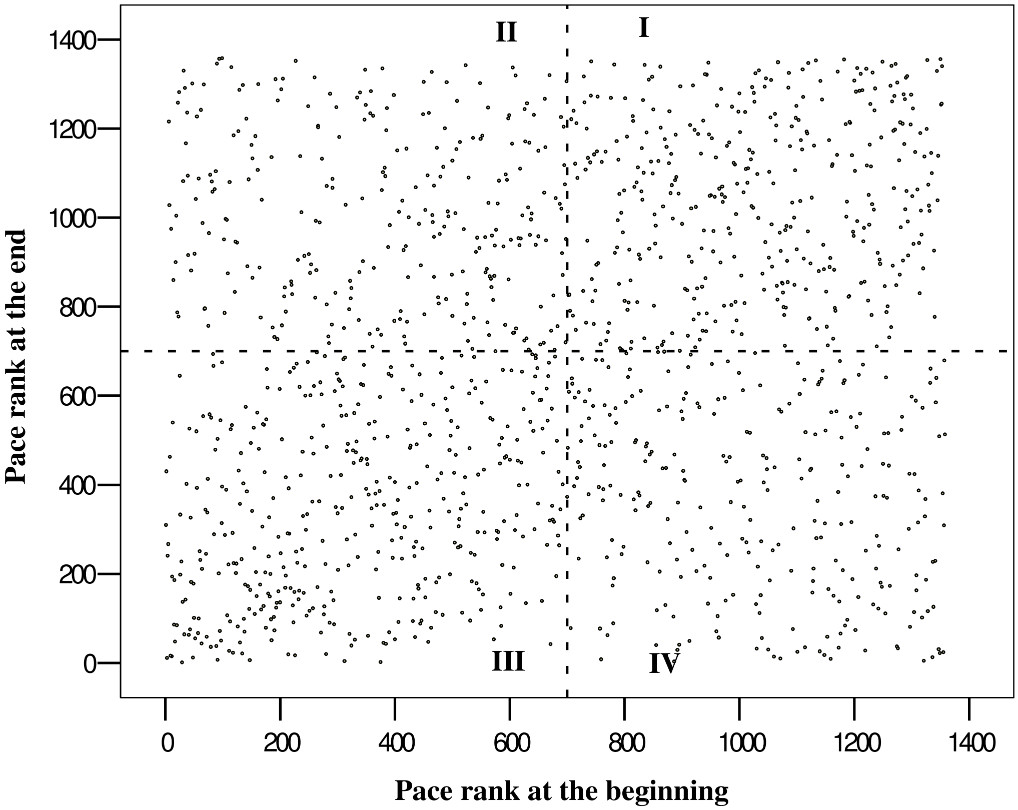


Figure 1. Scatter plot of pace ranks at the beginning (*x*) and the end (*y*) for *Dataset2$_P$* (Practicing learning mode), N=1,358

Across Modes Consistency

Results for *Dataset3* are given in Table 4, representing the examination of pace rank consistency across learning modes. Correlation coefficients are relatively low for this situation. Furthermore, there is a significant difference between the means of the two groups: On average, Memorizing sessions were faster than Practicing sessions with *t(767)*=7.99**.

It is a good point to recall the similarities and differences between the two learning modes being discussed here. While Memorizing and Practicing modes share a very similar GUI, and work according to the same principle (browsing over pages each consisting of a 10-row table of words/phrases), the main difference is that the Memorizing tables show the meaning of the term, while the Practicing tables hide it. As suggested by the results, students spend more time on Memorizing pages than on Practicing pages, and pace ranks across modes have a low correlation. This might imply that pace of action is affected by a set of skills needed for progressing in either of the modes.

Table 4. Across modes consistency of pace rank

Dataset	N (Students)	Group 1	Group 2	ρ	τ
Dataset3	768	Memorizing	Practicing	0.34**	0.23**

** p<0.01

Random Division Consistency

Results for *Dataset4*$_A$, *Dataset4*$_M$ and *Dataset4*$_P$ are given in Table 5. These three datasets relate to a more technical situation than the previous ones: random division of each student's sessions to two groups, and examination of pace rank consistency between these two groups. While *Dataset4*$_A$ takes into consideration all the sessions from the log file, *Dataset4*$_M$ and *Dataset4*$_P$ relate only to Memorizing and Practicing sessions, accordingly.

Table 5. Random division consistency of pace rank

Dataset	N (Students)	Mode	Group 1	Group 2	ρ	τ
Dataset4$_A$	6,112	All	Random	Random	0.36**	0.25**
Dataset4$_M$	758	Memorizing	Random	Random	0.62**	0.45**
Dataset4$_P$	526	Practicing	Random	Random	0.56**	0.41**

** p<0.01

It might be seen that for the general case – correlation is relatively low, however when examining pace ranks within the same learning mode, correlation is resulted with relatively high values of coefficients. Also, no significant difference was observed in the means between the two groups within each of the datasets.

To conclude the results of this study, there were only two situations in which pace rank was found to be consistent with relatively high values of correlation coefficients: a) Day/night division within the same learning mode; and b) Random division of each student's sessions within the same learning mode. In all the other situations - namely: over time, across modes, and all-inclusive random division - pace rank consistency was found to be relatively low, with correlation coefficients (ρ) between 0.20^{**} and 0.36^{**}.

5 Discussion

Many EDM studies often handle fine-grained data in the action/session level, like pace measures. However, when examining the student level, mainly since vector variables are not easy to cope with while applying data mining algorithms, scalar measures of these variables are often being used (e.g., average or median pace over different sessions). Time-related variables (usually describing the time taken for answering a question or for completing a task) are quite common in EDM research [1, 8, 11], but others are also often being averaged, for example: attempts for answering a question [1, 11], hint/help usage (usually per question) [1], and intense of activity (usually in terms of number of actions per session or frequency of certain activities) [6, 15]. While doing this, a hidden assumption – regarding the variable in question being a trait – is lying behind the calculations. It is our obligation to deeply investigate the consistency of each variable before projecting it on a 1-dimensional measuring scale and assuming it is of a trait type, as was clearly presented by Baker [2].

This is why we choose a rather primitive variable, namely pace of actions, in order to study its consistency. As the results obtained with our data suggest, correlation between pace ranks in different situations was sometimes very low. The minimal correlation coefficient (for $Dataset2_P$) was 0.20^{**}, which is almost a zero correlation. The maximal correlation coefficient (for $Dataset4_M$) was 0.62^{**}, which is relatively high but still quite far from a perfect correlation.

To be honest, these results was, at first, very surprising, as we expected to see much higher correlation values. The fact that for one situation (beginning/end consistency, Practicing mode) 40% of the students were located at the second and fourth quadrants of the pace ranks scatter plot (Figure 1) – indicating they were above the median rank in the beginning and below it in the end, or vice versa – is thought-provoking, and explicitly shedding light on the questionability of the assumption of pace rank consistency.

Furthermore, the surprisingly low correlations might imply that our choice of pace was not at all of a simple variable as we first thought, as pace of actions depicts different kinds of processes in which the online student is involved while learning, e.g., reading, memorizing, recalling previous knowledge, thinking, processing, typing, and navigating. Besides the clear effect of different learning components on learning time/pace, individual components also heavily affect it, such as ability to understand instruction or quality of instruction events, as was seminally proposed by Carroll [5]. Considering that pace measurement embodies different task-related and/or student-related components (and potentially others), it is clear that replicating this study with different learning systems and/or with different pace metrics is necessary before generalizing any conclusion regarding the consistency phenomenon.

In general, many educational studies investigate all kinds of students' attributes; however, EDM researches often analyze data drawn from relatively long periods of time, therefore our hand on the reduction trigger is likely to be more itchy. Further research and a deeper investigation is needed in order to better understand which behavioral attributes in online learning are indeed students' traits and which are heavily situation dependent.

References

[1] Arroyo, I. and Woolf, B.P. (2005). Inferring learning and attitudes from a Bayesian Network of log file data. *Proceedings of the 12th International Conference on Artificial Intelligence in Education*. Amsterdam, The Netherlands.

[2] Baker, R. (2007). Is gaming the system state-or-trait? Educational Data Mining through the multi-contextual application of a validated behavioral model. *Proceedings of Workshop on Data Mining for User Modeling at the 11th International Conference on User Modeling*. Corfu, Greece.

[3] Baker, R.S.J.d., *Data Mining for Education*, in *International Encyclopedia of Education (3rd edition)*, B. McGaw, P. Peterson, and E. Baker, Editors. In Press, Elsevier: Oxford, UK.

[4] Beck, J.E. (2004). Using response times to model student disengagement. *Proceedings of ITS2004 Workshop on Social and Emotional Intelligence in Learning Environments*. Maceio, Brazil.

[5] Carroll, J.B. (1963). A model of school learning. *Teachers College Record, 64*(8), 723-733.

[6] Chen, G.-D., Liu, C.-C., Ou, K.-L., and Liu, B.-J. (2000). Discovering decision knowledge from Web log portfolio for managing classroom processes by applying decision tree and data cube technology. *Journal of Educational Computing Research, 23*(3), 305-332.

[7] Clariana, R.B. (1990). Rate of activity completion by achievement, sex and report in computer-based instruction. *Journal of Computing in Childhood Education, 1*(3), 81-90.

[8] Cocea, M. and Weibelzhal, S. (2006). Can log files analysis estimate learners' level of motivation? *Proceedings of 14th Workshop on Adaptivity and User Modeling in Interactive Systems*. Hildesheim, Germany.

[9] Daniel, W.W., *Applied Nonparametric Statistics*. 1978, Boston, MA: Houghton Mifflin.

[10] de Vicente, A. and Pain, H. (2002). Informing the detection of the students' motivational state: An empirical study *Proceedings of The Sixth International Conference on Intelligent Tutoring Systems (ITS 2002)*. Biarritz, France.

[11] Feng, M., Beck, J., Heffernan, N., and Koedinger, K. (2008). Can an Intelligent Tutoring System predict math proficiency as well as a standardized test? *Proceedings of First International Conference on Educational Data Mining*. Montreal, Canada.

[12] Gibbons, J.D., *Nonparametric Measures of Association*. Quantitative Applications in the Social Sciences. 1993, Newbury Park, CA: Sage Publications.

[13] Hershkovitz, A. and Nachmias, R. (2008). Developing a log-based motivation measuring tool. *Proceedings of First International Conference on Educational Data Mining*. Montreal, Canada.

[14] Kornbrot, D. and Macleod, M. (1990). Monitoring and Analysis of Hypermedia Navigation. *Proceedings of INTERACT '90, The 3rd IFIP Conference on HCI*. Cambridge, UK.

[15] Perera, D., Kay, J., Yacef, K., and Koprinska, I. (2007). Mining learners' traces from an online collaboration tool. *Proceedings of Educational Data Mining Workshop at the 13th International Conference on Artificial Intelligence in Education*. Marina del Ray, CA.

[16] Qu, L. and Johnson, W.L. (2005). Detecting the learner's motivational states in an interactive learning environment. *Proceedings of Artificial Intelligence in Education*. Amsterdam, The Netherlands.

[17] Sprent, P., *Applied Nonparametric Statistical Methods*. 2000, London, UK: CRC Press.

[18] Zhang, G., Cheng, Z., He, A., and Huang, T. (2003). A WWW-based learner's learning motivation detecting system. *Proceedings of International Workshop on Research Directions and Challenge Problems in Advanced Information Systems Engineering*. Honjo City, Japan.

Student Consistency and Implications for Feedback in Online Assessment Systems

Tara M. Madhyastha[1] and Steven Tanimoto[2]
madhyt@u.washington.edu, tanimoto@cs.washington.edu
[1]Department of Psychology, University of Washington
[2]Department of Computer Science, University of Washington

Abstract. Most of the emphasis on mining online assessment logs has been to identify content-specific errors. However, the pattern of general "consistency" is domain independent, strongly related to performance, and can itself be a target of educational data mining. We demonstrate that simple consistency indicators are related to student outcomes, and suggest how consistency might be used in an online assessment framework to provide scaffolding to help students in need.

1 Introduction

Online assessment systems have the potential to supply detailed information about how students interact with them, which can be used to provide useful feedback. Much of the effort in mining this data has focused on identifying student misconceptions and partial understandings, in an effort to build upon their existing knowledge and direct them towards corrective interventions. However, there is a more general type of information available from such systems that may be valuable to assess, which we call student consistency. Specifically, we refer to the ability of a student to self-appraise his or her performance while, in our context, interacting with a computer to complete some untimed task or assessment. For example, a student is inconsistent when he or she executes a command, obtains a result that differs from what students are told to expect, and takes no further action to resolve the problem. The evidence indicates that the student made a mistake, but the student does not acknowledge this through his or her actions. Because markers of consistency do not necessarily require pedagogical content models, they may be easier to define than sequences of logged actions that describe a certain misconception. They be used to give students helpful feedback or scaffolding, without detailed pedagogical content models.

This paper outlines a set of consistency markers developed in the context of a laboratory exercise conducted in a college-level course: Artificial Intelligence for Nonmajors. We demonstrate that consistency is related to outcomes, both specifically to the score obtained on the laboratory exercise, and more generally to performance on the final exam. We suggest how consistency markers might be used to provide formative feedback to the student.

2 Background

2.1 Consistency

Cognitive consistency was an important area of research in the 1950's and 1960's that sought to understand how people would behave as a function of the dissonance caused when an individual simultaneously held two conflicting cognitions (e.g. beliefs, opinions). Theories posited that the conflicting cognitions would cause a quantifiable tension that the individual would seek to resolve in some way, recreating a consistent view. Unfortunately, consistency theories failed to fully explain behavior – individuals varied greatly in their ability to tolerate dissonance and their behavior was usually highly situation dependent [1].

Nevertheless, the idea of consistency is an important subtext in education. Teachers assume that student knowledge is consistent and that challenging their incorrect knowledge will cause students to attempt to reconcile the dissonance and learn. To this end, it is recommended that teachers should check the extent to which students hold erroneous concepts throughout instruction, ideally to deliver personalized feedback to students. This process is called "formative assessment" and feedback intended to help learners improve their performance is called "formative feedback". This approach usually requires a detailed model of the content domain and the ways in which students interact with it correctly and incorrectly. Constructing these models is time-consuming and difficult. Furthermore, there is little consensus on what constitutes appropriate formative feedback [2]. Complex, unspecific, or confusing feedback can have a negative effect on learning[3].

One reason feedback may be confusing may be that when students have not achieved mastery of a topic, their understanding is inconsistent. This was Sleeman's hypothesis for why addressing algebraic misconceptions was no more effective than reteaching (which is a simpler approach) [4]. Indeed, mathematical models that align specific errors to a linear ability scale find that groups of errors are "clustered", corresponding to a specific level of mastery [5]. Within each cluster of errors, students vary in the specific error they display. For example, a student at a relatively low level of mastery of graph reading may be equally likely at different times to interpret a straightforward graph of speed versus time with a constant positive slope as "the object is getting faster" (higher is faster) or "the object is getting slower" (higher is slowing down, as in up a hill).

Consistency itself is related to ability. This idea has been exploited to develop consensus-based assessments, where the correct response is defined as the one that most people agree on, and an individual score is measured as a distance from the consensus response. This approach has been used both to score people on situational judgment tests, and to measure intelligence [6]. When United States Military Academy students were asked to respond to a survey about their political beliefs (ranging from liberal to conservative) it was found that their consistency in these responses correlated with their SAT or ACT scores [7]. In research on physics misconceptions, we found that students who held any concept consistently, whether correct or incorrect, had higher math ability than students who were inconsistent [8]. The underlying logic for a relationship between ability and consistency is simple; a consistent response is one that many people can share by using similar underlying reasoning processes, but errors in reasoning and knowledge yield inconsistent, differing, responses. Furthermore, a consistent reasoning strategy requires a significant amount of knowledge about the topic and an ability to reconcile situations where one's beliefs do not match those of others (e.g., people who attempt to explain other's dissenting beliefs by creating for themselves a clear explanation of the opposing view are likely to converge on consistent reasoning than those who dismiss alternative points of view).

Therefore, in this paper we conducted a study to examine how consistency, which requires no complex models of pedagogical content knowledge to measure, is domain-independent, and is easy to identify, is related to performance on online activities and assessments. We describe the implications that inconsistency has on the kind of feedback that might be appropriate in an online assessment system.

2.2 The PixelMath Software.

PixelMath is an educational image processing system. It was developed as a web-oriented successor to the "Pixel Calculator" program that, in turn, was developed by Tanimoto and associates as part of the NSF-funded project "Mathematics Experiences Through Image Processing" [8]. The purpose of PixelMath is to empower students to manipulate digital images using a mathematics-oriented, rather than an artist-oriented, interface. It provides special affordances that reveal the mathematical structure of each image and that can interpret mathematical formulas as image transformations and syntheses. PixelMath also provides a scripting facility using Python, making it possible to teach and learn programming in an authentic, "on-demand" context. PixelMath is both a tool for teaching and learning and a tool for educational research. It is hosted within an online learning environment called INFACT, which serves in part to collect activity data from students as they are learning using online tools. Whenever a student performs an operation in PixelMath, such as selecting a menu item, zooming in on the image, or running a mathematical formula to transform an image, a description of the operation is sent to the INFACT server, and it is stored in a database together with the user ID of the student and the time and date of the event. This makes it possible to perform formative assessment and/or data mining for the student activities without having the students take tests.

3 Method

We describe the research method used in this study.

3.1 Subjects

Subjects were students in one of the authors' Introduction to Artificial Intelligence (for nonmajors) course at the University of Washington in Winter quarter 2008. The University of Washington is highly selective, admitting approximately 100 students into the major each year, so there is a huge demand for courses for non majors from overlapping departments. We asked students to give consent to link INFACT log files from a laboratory exercise with course grades. Of the 40 students in the class, 34 completed the laboratory exercise. Thirty of those who completed the laboratory consented to participate in the study. Three did not submit the laboratory worksheet. Therefore, the final sample consisted of 27 students, including 2 females and 25 males of college age. This highly skewed gender distribution is typical among engineering classes.

The majority of these students were majors in the Applied and Computational Math Sciences program (N=17) or Electrical Engineering majors (N=5) and the remainder were divided among varied other non-computer science majors.

3.2 Procedure

Students were instructed to work individually, in a laboratory classroom with individual computers, to complete a series of exercises using PixelMath to understand some key concepts of image processing. These activities covered concepts of sampling, histograms, thresholding and morphology. They were allowed to ask questions of the instructor, which were answered individually, and to talk to each other. The lab was intended to take approximately an hour to complete. We observed little unrelated activity (e.g. web browsing or messaging) once students began working on the laboratory. The laboratory exercises included very specific instructions, including formulas to use to accomplish critical goals and to use as starting points for inquiry. Students were given participation points for turning in a completed worksheet. The laboratory session was held at noon on a Friday and students were given until Monday afternoon to turn in their work. Most students turned in their assignments by the end of the hour.

3.3 Data Logging and Coding

PixelMath logs a variety of timestamped activities, including file manipulations, image cloning, mouse clicks (zoom operations), and transformation formulas that students enter. From this data we extracted variables related to the amount of time students spent using PixelMath, errors made, and student consistency. The time variables were total time spent, number of logged events, and average time between events. Parse errors were totaled for each student to create a count of errors.

We identified seven Boolean consistency indicators that we categorized in three groups as follows:

- Matches worksheet (2 indicators)

At various points, students were asked to write in a response on the worksheet and use that value in a subsequent calculation, or to try to accomplish some task and then report some value that they found. For example, in the sampling activity, students are asked to determine the minimum number of pixels required, theoretically, to represent a long, white, picket fence. They are then asked to use this number of pixels to create a downsampled image. (Figure 1 is a screen shot that shows the original image in one window, the isolated picket fence in another, and the downsampled picket fence in a third; the PixelMath calculator is also visible.) Later, students are asked to report the sampling factor that resulted in the minimum number of pixels necessary to view the picket fence with minimum loss of pixels. We record a Boolean flag for each of the two worksheet responses: true if the worksheet matches a corresponding command (at any place in the log), and false if it does not. Note that “matching” does not indicate correctness; students often had matching consistency for incorrect answers in the log and worksheet.

- Logical consistency (1 indicator)

Students are given a formula to downsample an image and are asked to revise it so that they can view the image using the minimum number of pixels. Increasing the scaling factor reduces the number of pixels, and decreasing the scaling factor increases the number of pixels. Therefore, an ideal searching behavior would converge upon the ideal scaling factor (ideally obtained through calculation) by moving in subsequently smaller steps around the goal. Inconsistent behavior would be represented by repeatedly trying identical scaling factors and/or taking repeated steps in the incorrect direction. Repeated behavior is defined as 2 or more times. We code this sequence as true if the student is consistent and false if they are not, or if they do not attempt the exercise.

- Recognized expected outcome (5 indicators)

In several places, the student is instructed to execute a specific command, such as to open a specific file or to perform some image transformation. If the student does not execute this command correctly, the output will not be expected and subsequent instructions will not make any sense. This is a consistency error of failure to reconcile the differing information. Ideally, the student should recognize this inconsistency and re-execute the command, or ask a fellow student or the instructor for help and re-execute the command correctly. We coded failure to do so at any point in the lab as a false and a correct execution as true.

3.4 Outcome measures

We defined three outcome measures. The first was the grade on the worksheet. This was a score of 1-3 where 1 indicated an incomplete assignment (some questions were unanswered), 2 indicated some partial understanding of the key concepts, and 3 indicated a solid understanding of the concepts.

The second outcome measure was performance on questions related to image processing on the final exam for the course. The final exam consisted of a multiple choice section (Final Part 1) and an open ended section (Final Part 2) with three questions. The first question (Final Part2.Q1) was a direct analogue to the first activity involving sampling. The second question (Final Part2.Q2) was more difficult and required a deeper understanding of the concepts. The third question (Final Part2.Q3) was an in depth question on unrelated material. The final exam was administered 2 weeks after the laboratory session.

The third outcome measure was a metric reflecting general performance in the class. We obtained this measure by doing an unrotated principal components factor analysis on the subscores of all parts of the final. Two factors had eigenvalues over 1, resulting in a two factor solution that accounted for 70.37% of the variance in scores. The first factor loaded .89 on the score for Final Part2.Q1, .83 on the score for Final Part2.Q2, and .56 on the Final Part 1. The second factor loaded .96 on Final Part2.Q3. This pattern of loading suggests that the first factor represents general knowledge gained in the course, separate from the format of the exam (e.g., open-ended questions versus multiple choice). We use the first factor as an outcome measure of general class knowledge.

4 Results

Table 1 shows the mean and standard deviation for the outcome measures, measures of time spent completing the laboratory exercise, and the consistency measures. There was clear variation among the students on all measures, including the consistency measures that may seem obvious (for example, writing the same response on the worksheet that was used in the exercise). This variation is particularly substantial considering that these students are highly selected into a competitive state university, and are expected to have developed skills that would result in higher consistency measures than the population as a whole.

There is also significant variation in scores on the final exam. We note that the average score on Final Part2.Q2, the more difficult of the two questions dealing with image processing, is lower than the average score of Final Part2.Q1. This suggests that the questions were difficult enough to avoid ceiling effects.

Table 1. Summary statistics for outcome measures, general logfile measures, and consistency measures. Maximum score or range of scores is given, where appropriate, in parentheses following the measure.

Measure	Mean (SD)
Outcome measures	
Worksheet Score (1-3)	2.57 (.69)
Final Part 1 (50)	30.60 (7.01)
Final Part2.Q1 (10)	6.97 (3.64)
Final Part2.Q2(10)	5.87 (4.17)
Final Part2.Q3(10)	7.67 (2.63)
Other Logfile Measures	
Number log events	273.60 (262.87)
Average time between events (in minutes)	.21 (.08)
Number parse errors	2.13 (2.33)
Consistency measures	
Matches worksheet (2)	1.48 (.64)
Recognized expected outcome (5)	4.13 (1.66)
Logical consistency (1)	.37(.49)

To examine the differences in outcome measures based on consistency, we split the students by the median sum score of consistency (7) forming two groups. We call these the low consistency group (N=14) and the high consistency group (N=13). We conducted a one-way ANOVA to determine whether outcome measures differ across the two groups. Results are summarized in Table 2.

On average, the high consistency group scored higher on all outcome measures (though not all differences were significant). The high consistency group obtained significantly greater worksheet scores (p=.002). The difference in scores is particularly noteworthy, because the consistency measures do not measure correct or incorrect responses, whereas the worksheet scoring does. Furthermore, the high consistency group scored marginally significantly higher on the Final Part2.Q1 (p=.083) and significantly higher on the Final Part2.Q2 (p=.049). These two questions of the final were designed to measure the same concepts covered by the worksheet, and were administered two weeks afterward. The high consistency group also scored significantly higher on the class knowledge measure extracted from the midterm scores (p=.018). The effect size (measured by Cohen’s d) was quite large.

Differences in consistency cannot be attributed solely to “carelessness” or greater or lesser time spent on the assignment. The high consistency group logged fewer events (not significant) with more parse errors (marginally significant, p=.062). The average time spent between logged events was virtually identical among the two groups.

Table 2. Analysis of variance.

	High Consistency (N=13)		Low Consistency (N=14)				
	Mean	SD	Mean	SD	F	Sig.	Cohen’s d
Worksheet Score	2.92	.28	2.21	.80	9.12	**.006**	**1.02**
Number Log Events/Student	198.31	82.55	326.71	357.81	1.59	.219	
Average Time Between Events (minutes)	.23	.08	.20	.09	.71	.409	
Number Parse Errors/Student	3.15	2.82	1.43	1.70	3.81	.062	
Final Part 1 (50)	33.23	6.09	29.43	7.00	2.26	.146	
Final Part2.Q1 (10)	8.46	2.26	6.14	4.07	3.27	.083	
Final Part2.Q2(10)	7.92	3.62	4.79	4.21	4.28	**.049**	**0.86**
Final Part2.Q3(10)	8.15	1.52	7.93	2.53	.08	.783	
Class Ability Measure (Factor extracted from Final Subtests)	.56	.58	-.28	1.04	6.46	**.018**	**0.84**

5 Implications for Feedback

We have proposed some rough indicators of consistency and shown that the low consistency students perform worse on several important class outcomes than the high consistency students. It is particularly remarkable that we find such variation in consistency in such a highly selected population.

We do not know the reasons why students are inconsistent, and we have not demonstrated whether higher consistency results in higher performance or whether it is a side-effect of something else (e.g., low level of engagement or interest in the class). This is a topic for future research. However, the inconsistencies that we have coded are rather blatant. When a student repeatedly types in the wrong command and does not recognize discrepancies in the results or attempt to reconcile them, it is reasonable to believe that such events may provide a learning opportunity. Furthermore, the needs of this student in this circumstance are not based on a model of how the student interacts with the pedagogical content of the assignment.

We suggest that consistency across several dimensions may be dynamically monitored and used to adaptively control scaffolding (additional guidance) for the student. Scaffolding involves (a) reducing the number of steps required to solve a problem by simplifying the task (b) keeping the student motivated (c) marking discrepancies between actions taken and the desired solution (d) controlling frustration and (e) demonstrating an idealized version of the task [9]. It is a technique used primarily when students are learning new material, and cannot handle complex problems. Graesser et al showed that use of scaffolding, including good questions and answers, could promote deep inquiry, which students tended to avoid without prompting [10].

For example, suppose students did not follow the directions on the worksheet correctly and did not realize it. An ideal student would have recognized that “something was wrong” and taken some action to resolve the cognitive dissonance, checking the last command executed or re-entering the command. If still confused, the student could ask a classmate or the instructor for help. As part of an online assessment system, we wish to encourage such behavior. An appropriate first step would be to emphasize the cognitive dissonance. In the assignment in the experiment, expectations are outlined in the text, e.g., by writing “Apply a formula that makes a monochrome image in which the cedar foliage is white and everything else is black” before providing the formula. However, the discrepancy could be highlighted further by showing an image of the expected result and asking the student, “Does your image look like this?” before proceeding.

In the extreme case when the same error is repeated, we can assume that the mistake is not inconsistency but represents some higher order misconception. For example, in the PixelMath interface there are buttons for common functions, such as XOR. Other functions can be typed in the window. One command asked students to apply the BXOR formula to exclusive-OR two images on a bit-by-bit basis, and many incorrectly applied the XOR function. It is possible that students who do not correct their error with appropriate scaffolding may not understand the difference between bitwise-exclusive-OR and straight exclusive-OR, or may not realize that they can type equations directly into the PixelMath formula area without clicking buttons. This might require specific formative feedback that reteaches these concepts to the student.

A common misconception about image processing systems such as PixelMath is that the formula tells where to move each pixel of the source image. In reality, the formula describes, for each destination pixel location, where or how to get its value. This “push-instead-of-pull” notion is exhibited by students asked to come up with a formula to, say, reduce the size of an image by a factor of two. Instead of writing Source1(x*2, y*2) which is correct, they write Source1(x/2, y/2). Similarly, to shift an image 5 pixels to the left, they should write Source1(x+5, y), but they put down Source1(x-5, y). After seeing a consistent pattern of such incorrect formulas, an automatic feedback system could provide scaffolding to specifically address the "push-instead-of-pull" misconception.

6 Conclusions

We have identified a dimension of student performance, consistency, that is content independent, easily mined from educational logs, and that is related to performance outcomes. We suggest that because consistency is an assumption that underlies many educational interventions, the significance of lack of consistency may be overlooked as a potential opportunity to provide scaffolding. We give some suggestions for how an adaptive learning system might exploit consistency measures to scaffold instruction, and to identify when consistency of incorrect responses suggests moving to an intervention based on more sophisticated models of the learner, content, and their interaction.

References

[1] R. Abelson, *Theories of cognitive consistency: a sourcebook.*, Chicago: Rand McNally, 1968.

[2] V.J. Shute, "Focus on Formative Feedback," *Review of Educational Research,* vol. 78, Mar. 2008, pp. 153-189.

[3] R.L. Bangert-Drowns, C.C. Kulik, J.A. Kulik, and M. Morgan, "The Instructional Effect of Feedback in Test-Like Events," *Review of Educational Research,* vol. 61, Jan. 1991, pp. 213-238.

[4] D. Sleeman, A. E. Kelly, R. Martinak, R. D. Ward, and J. L. Moore, "Studies of diagnosis and remediation with high school algebra students," Jul. 2005.

[5] K. Scalise, T. Madhyastha, J. Minstrell, and M. Wilson, "Improving Assessment Evidence in e-Learning Products: Some Solutions for Reliability," *International Journal of Learning Technology (IJLT)*, In press. .

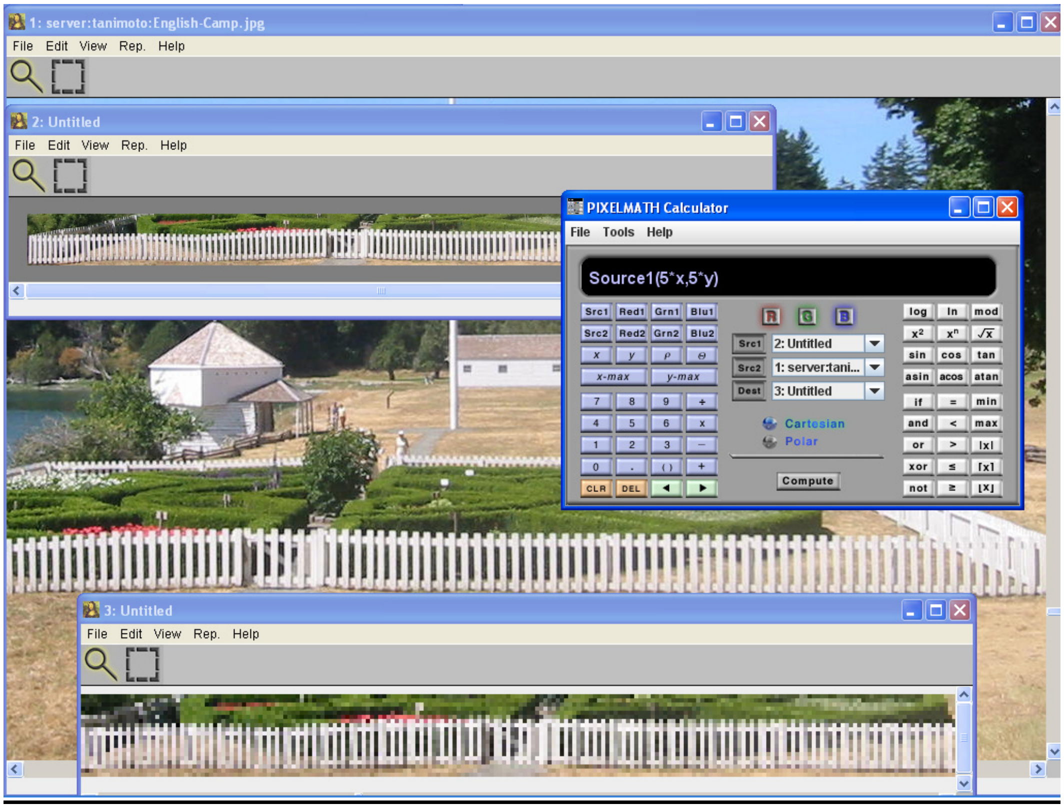
[6] P. Legree, J. Psotka, T. Tremble, and D. Bourne, "Applying Consensus-Based Measurement to the Assessment of Emerging Domains," Jan. 2005.

[7] J. Psotka, "Psychophoresis and Intelligent Self Assessment (ISA) Scales," *Annual Meeting of the Society for Intelligence Research (ISIR),* Decatur, GA: 2008.

[8] T.M. Madhyastha, "The Relationship of Coherence of Thought and Conceptual Change to Ability. ," *Annual Meeting of the American Educational Research Association,* San Francisco: 2006.

[9] J. Bransford and National Research Council (U.S.).;National Research Council (U.S.)., *How people learn : brain, mind, experience, and school,* Washington D.C.: National Academy Press .

[10] A.C. Graesser, D.S. McNamara, and K. VanLehn, "Scaffolding Deep Comprehension Strategies Through Point&Query, AutoTutor, and iSTART.," *Educational Psychologist,* vol. 40, Fall2005. 2005, pp. 225-234.



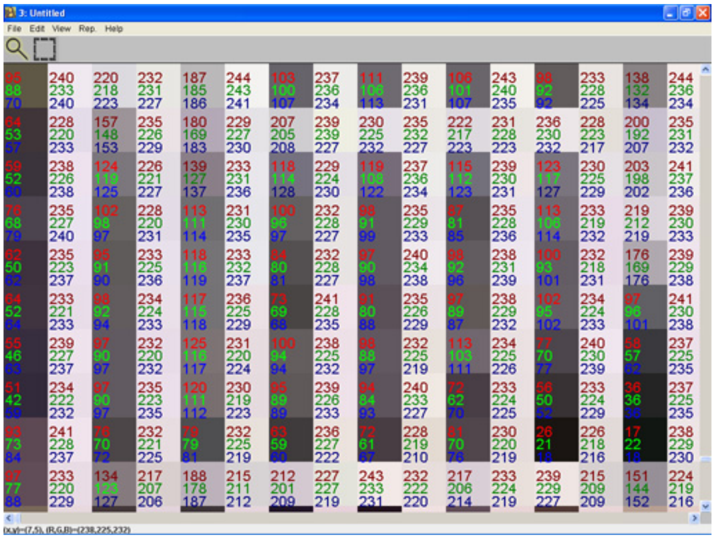
Source1(5*x,5*y)

Src1 2: Untitled

Src2 1: server:tani...

Dest 3: Untitled

Figure 1. The background window contains the original image used by students for the laboratory activity. The fence has been extracted into another window (above, and zoomed out by a factor of 2) and downsampled into another (below). The PixelMath calculator can also be seen here with the correct formula for the downsampling: Source1(5*x, 5*y).



95 88 70	240 233 240	220 218 223	232 231 227	187 185 186	244 243 241	103 100 107	237 236 234	111 106 113	239 236 231	106 101 107	243 240 235	98 92 92	233 228 225	138 132 134	244 236 234
64 53 57	228 220 233	157 148 153	235 226 229	180 169 183	229 227 230	207 205 208	239 239 227	230 225 232	235 232 227	222 217 223	231 228 223	236 230 232	228 223 217	200 192 207	235 231 232
59 52 60	238 226 238	124 119 125	226 221 227	139 127 137	233 231 236	118 114 128	229 224 230	119 108 122	237 236 234	115 112 123	239 230 231	123 117 127	230 225 229	203 198 202	241 237 236
76 68 79	235 227 240	102 98 97	228 220 231	113 111 114	231 230 235	100 96 97	232 228 227	98 91 99	235 229 233	87 81 85	235 228 236	113 106 114	233 219 232	219 212 219	239 230 233
62 50 62	235 223 237	95 91 90	233 225 236	118 116 119	233 232 237	84 80 81	232 228 227	97 90 98	240 234 238	98 92 96	238 231 239	100 93 101	232 218 231	176 169 176	239 229 238
64 52 64	233 221 233	98 92 94	234 224 233	117 115 118	236 225 229	73 69 68	241 228 235	91 80 88	235 226 229	97 89 87	238 229 232	102 95 102	234 224 233	97 96 101	241 230 238
55 46 63	239 227 237	97 90 97	232 220 232	125 116 117	231 220 224	100 94 94	238 225 232	98 88 97	232 225 219	113 103 111	234 225 226	77 70 77	240 230 239	58 57 62	237 225 235
51 42 59	234 222 232	97 90 97	235 223 235	120 111 112	230 219 223	95 89 89	239 226 233	94 84 93	240 233 227	72 62 70	233 224 225	56 50 52	233 224 229	36 36 36	237 225 235
93 73 84	241 228 237	76 70 72	232 221 225	79 79 81	232 225 219	63 59 60	236 227 222	72 61 67	228 219 210	81 70 76	230 220 219	26 21 18	226 218 216	17 22 18	238 229 230
97 77 88	233 220 229	134 123 127	217 207 206	188 178 187	215 211 212	212 201 209	227 227 219	243 233 231	232 222 220	217 206 214	233 224 219	239 229 227	215 209 209	151 144 152	224 219 216

(x,y)=(7,5), (R,G,B)=(238,225,232)

Figure 2. Detail within the downsampled picket fence showing the effect of sampling at the Nyquist rate. Also, PixelMath's display of the RGB values of each pixel can be seen.

Edu-mining for Book Recommendation for Pupils

Ryo Nagata[1], Keigo Takeda[1], Koji Suda[2], Junichi Kakegawa[2], and Koichiro Morihiro[2]
rnagata@center.konan-u.ac.jp
[1]Konan University
[2]Hyogo University of Teacher Education

Abstract. This paper proposes a novel method for recommending books to pupils based on a framework called Edu-mining. One of the properties of the proposed method is that it uses only loan histories (pupil ID, book ID, date of loan) whereas the conventional methods require additional information such as taste information from a great number of users which is costly to obtain. To achieve this, the proposed method solves the book recommendation problem as a problem of loan date prediction, relying solely on loan histories. Experiments show that the proposed method achieves an accuracy of 60% and outperforms the method (weighted slope open collaborative filtering) used for comparison. In addition to the performance, the proposed method has the following two advantages: (i) it is inexpensive compared to the conventional methods and (ii) reading level is adjustable.

1 Introduction

Reading is one of the most essential intellectual activities for pupils. Librarians in school libraries play an important role in facilitating the activity by recommending proper books to them. However, it is often the case that there are not enough librarians in school. Besides, it is impossible for a librarian to remember all the book information and the preferences of every single pupil in a school.

To solve the problem, there has been work on automatic book recommendation. Whichbook [7] is a book recommendation system that helps the reader find books based on mood and style parameters, such as *happy or sad*, s/he specifies. A drawback of the system is that it requires collecting training data to tune mood and style parameters, which is costly and time-consuming. Besides, it is questionable whether or not pupils or even teachers can find proper books that have good educational effects by using the system since it has more than 100 parameter settings.

Another way is collaborative filtering [1]. Collaborative filtering is a general way of recommending items including books [3], movies [2], news articles [5], and music albums [6]. However, collaborative filtering is also costly in that it normally requires collecting taste information from a number of users. In book recommendation, readers have to rate books they have read. This becomes especially problematic when readers are pupils as in our task. Pupils are not capable of rating books properly in some cases. As an example, imagine a situation where a teacher recommends a book that is a bit difficult for a pupil in order to give him/her a chance to read books of a higher reading level. S/he will be likely to rate the book as *not-good* or *not-interesting* because it is a bit difficult for him/her even if it has good effects on his/her intellectual development.

In view of this background, this paper proposes a novel method for recommending books to pupils based on the Edu-mining framework [4] which favors inexpensive methods (See Nagata et al. [4] for the details of Edu-mining). The proposed method relies solely on loan histories to recommend books. Here, a record of a loan history consists of triples: user ID (ID for a pupil), book ID (ID for a book that the pupil borrowed), and date of loan. A loan history consists of the triples for all books a pupil borrowed. Normally, loan histories are registered in a database in the school library. Thus, there is no need for collecting training data or taste information in the proposed method, which means that the proposed method is much more inexpensive than the conventional methods.

Given loan histories, the proposed method solves the book recommendation problem as a problem of loan date prediction. In other words, it predicts when the target pupil will borrow the books that s/he has not borrowed yet from loan histories. This solution yields further two advantages. One is that the proposed method can estimate the reading levels of books it recommends. This means that reading levels are adjustable depending on the target pupil in the book recommendation of the proposed method. The other is that the preferences of pupils are implicitly included in the recommendation results.

The rest of this paper is structured as follows. Section 2 discusses the basic concept of Edu-mining. Section 3 introduces the basic idea of the proposed method. Section 4 describes the proposed method. Section 5 describes experiments conducted to evaluate the proposed method. Section 6 discusses the experimental results.

2 Edu-mining

Edu-mining [4] is based on data mining and text mining but differs from them in the following three points. First, the target data are peculiar to education. A good example of this is the writing of pupils where the distributions of words, mechanics, and style are quite different from those of adults. These differences affect the performance of tools such as a morphological analyzer that is often used in text mining to extract words, and thus they provide poor information on the data. Another example can be seen in our task. As already mentioned, collaborative filtering predicts user preferences based on previous user preferences (taste information). In the book recommendation for pupils, taste information is not as reliable as that of adults; pupils are not capable of rating books properly in some cases as exemplified in Section 1. These facts imply that Edu-mining has to solve the problems that arise from the differences between educational data and normal data in its own scheme.

Second, it prefers simple and inexpensive techniques. It should be implemented at moderate cost since it mainly aims at the use in school. Also, the target users of Edu-mining are mainly teachers and/or students (including pupils). If the used techniques are simple, the target users are likely to use them easily. Besides, they may sometimes be able to give feedback on the techniques.

Third and finally, whereas data/text mining aims at improving the quality of the mined knowledge, it is not necessarily the case in Edu-mining; its ultimate goal is to achieve good educational outcomes. In case of normal book recommendation, the ultimate goal is to find and recommend books that the user wishes to read (and probably to purchase). By contrast, in case of our task, this is not the ultimate goal; the ultimate goal is to facilitate their intellectual development by recommending proper books.

This is the basic concept of Edu-mining. The next section describes the basic idea of the proposed method based on Edu-mining.

3 Basic Idea

So far, we have seen the basic concept of Edu-mining and its relation to book recommendation for pupils. This section describes the basic idea of the proposed method based on Edu-mining. In book recommendation for pupils, the peculiarity of the data is that taste information obtained from pupils may be unreliable as Section 1 describes. The proposed method overcomes the problem by not using taste information. Instead, it solves the book recommendation problem as a problem of loan date prediction. It uses a simple and intuitive way to predict loan dates.

Before describing the basic idea of the proposed method, let us introduce a new loan date called absolute loan date. Loan date normally has the form of date, month, and grade of the pupil (e.g., 1st Sep. 1st grade). This form of loan date is not suitable for the calculation used in the proposed method as we will see below. So, absolute loan date is used instead of the normal loan date. Absolute loan date is a simple mapping of the normal loan date. The first day of the first grade is the base date and mapped to 0. Other loan dates are simply mapped to the corresponding absolute loan dates of which distance from the base date is given by the number of days from the first day of the first grade. For example, a month later from the first day is mapped to 30 (or 31), the first day of the second grade is mapped to 365, and so on. Figure 1 illustrates the mapping between normal loan dates and absolute loan dates.

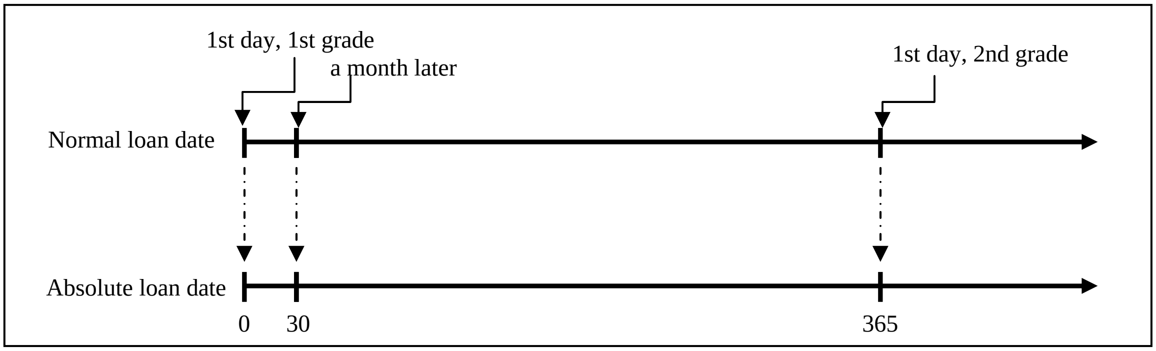


Figure 1. Mapping between normal loan date and absolute loan date

Here, it is worthwhile to note that absolute loan dates roughly correspond to reading levels. Namely, first grade pupils tend to borrow books of low reading levels whereas upper grade pupils tend to borrow books of higher reading levels. This implies that if one can predict absolute loan dates, s/he can also estimate reading level. This is why reading level is adjustable in the recommendation of the proposed method.

Now let us describe the basic idea of the proposed method. The proposed method solves the book recommendation problem as a problem of loan date prediction as already mentioned. This is equivalent to saying that the proposed method predicts absolute loan dates from loan histories. Once it predicts absolute loan dates, it can easily recommend books to the target pupil because it knows when s/he will borrow the books s/he has not borrowed yet. Simply, it recommends books which are predicted to be borrowed at the day of the recommendation or near the day. Or, if one wishes to recommend a book of a higher reading level, it can recommend books which are predicted to be borrowed some days (say, a half year later) after the day of the recommendation; the opposite can also be done.

To see how the proposed method predicts absolute loan dates, suppose that we have loan histories shown in Figure 2 where loan dates are expressed by absolute loan dates. Figure 2 shows, for example, that *pupil A* borrowed *book A* on the absolute date 365 (equivalently, the first day of the second grade). Further suppose that we are predicting the absolute loan date of *book B* for *pupil A* (the question mark in Figure 2 denotes that *pupil A* has not borrowed *book B* yet). If we look at the loan history of *pupil B*, we will notice that s/he borrowed *book B* 370 days after *book A*. Based on this, it is natural to predict the absolute loan date of *book B* for *pupil A* to be 370 $(= 670 - 300)$ days after the loan date 365 of *book A* for *pupil A*, or equally 735 $(= \{365 + (670 - 300)\})$. Similarly, based on the loan history of *pupil C*, it is natural to predict the absolute loan date of *book B* for *pupil A* to be 725 $(= \{365 + (710 - 350)\})$. To obtain the final prediction, we take the average of the two absolute loan dates, that is, $(735 + 725)/2 = 730$ (equivalently, the first day of the third grade). It should be noted that adding the average of the differences between the loan dates to the loan date of the base book gives the same result. For instance, 730 = 365 + {(670 - 300) + (710 - 350)}/2.

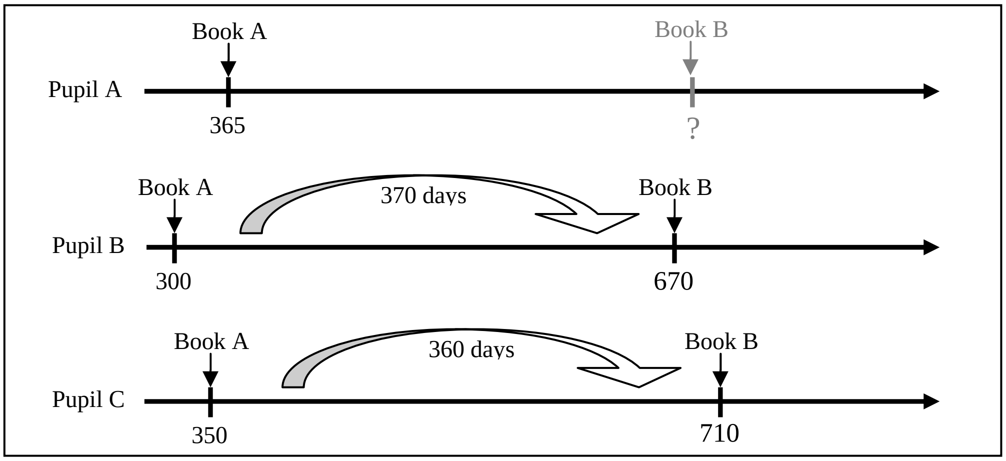


Figure 2. Example of loan histories

Although the loan histories in Figure 2 involves only three pupils and two books for illustration purpose, actual loan histories often involves far more pupils and books. Therefore, the average is taken over the relevant books and the relevant pupils in actual use. A rough definition of relevant pupils and relevant books is as follows (the next section will describe the strict definition). A relevant pupil is those who have borrowed the following two books: (a) one of the books the target pupil borrowed and (b) the book of which absolute loan history is to be predicted. A relevant book is the book that is borrowed by (i) the target pupil and (ii) one or more of the relevant pupils.

This is the basic idea of how the proposed method predicts absolute loan dates from loan histories. The next section describes the prediction method in detail.

4 Proposed Method

To formalize the prediction method, we will use the symbol p and b to denote a pupil and a book, respectively, in the given loan histories. We will also use the symbol $d_{p,b}$ to denote the absolute loan date when the pupil p borrowed the book b; if the pupil p has not borrowed the book b yet, then $d_{p,b}$ is set to -1.

Now, let p be the target pupil (target for book recommendation) and b be the book of which absolute loan date is to be predicted. Then, a relevant pupil is those who have borrowed both b and one of the books the target pupil borrowed. Thus, a set of relevant pupils is defined by

$$P(b,b') = \{p' | d_{p',b} \neq -1, d_{p',b'} \neq -1\} \qquad (1)$$

where b' denotes one of the books the target pupil borrowed. Also, a relevant book is a book that satisfies the following two conditions: (i) a book that the target pupil borrowed, and (ii) a book of which relevant pupil exists[1]. Using Equation (1), a set of relevant books is defined by

$$B(p,b) = \{b' | d_{\mathrm{p},b'} \neq -1, |P(b,b')| > 0\}\,. \qquad (2)$$

Using Equation (1) and Equation (2), absolute loan dates are predicted by

$$\hat{d}_{p,b} = \frac{\sum_{b'\in B(p,b)} \sum_{p'\in P(b,b')} \{\mathrm{d}_{p,b'} + (\mathrm{d}_{p',b} - \mathrm{d}_{p',b'})\}}{\sum_{b'\in B(p,b)} |P(b,b')|}. \qquad (3)$$

Here, $\{\mathrm{d}_{p,b'} + (\mathrm{d}_{p',b} - \mathrm{d}_{p',b'})\}$ corresponds to the simple prediction of absolute loan dates discussed in the basic idea in Section 2 (for instance, $\{365 + (670 - 300)\}$). The sums in the numerator are the total sum of the simple predictions over the relevant pupils and the relevant books; in the case of the same example, the sums correspond to $725 + 730$. The denominator is the number of simple predictions. Hence, Equation (3) gives the average of the simple predictions. In case of $|P(b,b')| = 0$ for any b', $\hat{d}_{p,b}$ is set to -1 meaning that the proposed method cannot predict the absolute loan date.

Intuitively, books whose absolute loan date is given by Equation (3) are similar, in terms of the topic, to the books that the target pupil borrowed because the average is taken over the relevant books and relevant pupils; the average is taken over the relevant pupils who have borrowed some of the same books as the target pupil and over the books that the relevant pupils have borrowed. In other words, the book preferences of the target pupil are implicitly included in the prediction through the relevant pupils and the relevant

[1] Note that a relevant book is not a book borrowed by the target pupil and anyone who borrowed the book b.

books. Furthermore, the similarity of each relevant book is considered in Equation (3). This can be seen by noting that Equation (3) can be rewritten as

$$\hat{d}_{p,b} = \frac{\sum_{b' \in B(p,b)} \left\{ |P(b,b')| d_{p,b'} + \sum_{p' \in P(b,b')} \left(d_{p',b} - d_{p',b'} \right) \right\}}{\sum_{b' \in B(p,b)} |P(b,b')|}.$$

In the rewritten version of Equation (3), the base date $d_{p,b'}$(the first term in the numerator) is weighed by the factor $|P(b,b')|$ which denotes the number of pupils that borrowed both b and b'. It is reasonable to think that the more pupils borrow two books, the more similar the two are, and in turn it is reasonable to give a higher weight to such a pair in the prediction. Equation (3) exactly does this.

Also, it should be noted that the denominator can be regarded as the credibility of the prediction because it denotes the number of relevant pupils and relevant books involved in the prediction. The prediction is not reliable if it is made based on few relevant pupils and few relevant books. Considering this, predictions whose $\sum_{b' \in B(p,b)} |P(b,b')| < N$ where N denotes a certain threshold are discarded in the book recommendation.

Once absolute loan dates are predicted for the books that the target pupil has not borrowed yet, the proposed method recommends books to the target pupil as follows. It recommends M books which are predicted to be borrowed at the day of the recommendation or near the day; here ($M = 5$ or $M = 10$, for example). Or, if a teacher wishes to recommend (or the target pupil wishes to read) books of a higher reading level, it recommends books which are predicted to be borrowed some days after the day of the recommendation. If one wishes the opposite, it recommends books which are predicted to be borrowed some days before the day of the recommendation. The amount of days can be chosen by an intuitive way to specify reading level. Recall that absolute loan date is simply the one to one mapping of normal loan date. If one sets the amount to 365 days after, it corresponds to specifying a one-grade-higher reading level.

5 Evaluation

For evaluation, we collected loan histories of pupils in an elementary school where the grades range from first to sixth. Table 1 shows the statistics on the loan histories.

Table 1. Statistics on the loan histories used for evaluation

Term of collection	Number of pupils	Total number of loaned books
30th Aug. 2007 to 1st Sep. 2008	619	14383

In the evaluation, we conducted two experiments. In the first, we evaluate how accurately the proposed method can recommend books similar to the books that the target pupil borrowed, which is described in 4.1. In the second, we evaluate the capability of the proposed method in estimating reading level, which is described in 4.2.

5.1 *Experiment on Book Recommendation Accuracy*

The experimental conditions and procedures are as follow. First, we randomly selected 10 target pupils (two for each grade, from first to fifth grade) from the loan histories; pupils in sixth grade ware not included in the experiment because of the limitation of the proposed method which will be discussed in Section 5. Second, we predicted absolute loan dates for the target pupils using the proposed method; the threshold N, which was discussed in Section 3, was set to five. Third, we selected five most difficult books and five easiest books for each pupil according to the predicted loan date. Then, the 10 books were shown to two elementary school teachers together with the corresponding loan history. Finally, the two teachers separately rated each book as *similar* (to one or more of the books in the loan history in terms of its topic), *not-related*, or *unknown* referring to the corresponding loan history. The performance of the proposed method was measured by accuracy. Accuracy was defined by

$$\frac{\text{total number of books rated as } related}{\text{total number of books}}.$$

For comparison, we implemented the weighted slope one collaborative filtering [2], which had been shown to be effective in item recommendation. To fully implement the weighted collaborative filtering, we need taste information for each book as described in Section 1. However, normal loan histories such as the ones used in this evaluation, do not contain taste information. For this reason, we implemented the weighted collaborative filtering with the loan histories in which an equal rating was given to all books. Doing so, it can recommend related books but cannot rank recommended books; all books are equally favored. So, 10 books were randomly chosen from the recommended books and shown to the two teachers for evaluation. The performance was measure by accuracy as in the proposed method.

Table 2 shows the results. It shows that the proposed method achieves an accuracy of 0.600. This means that on average, six out of the 10 books recommended by the proposed method are related to the books that the target pupil borrowed. It seems to be not so difficult for teachers or even pupils to select related books from the recommended books which are actually related 60% of the time.

Table 2. Evaluation on book recommendation accuracy

Method	Accuracy
Proposed method	0.600
Slope one collaborative filtering	0.435

Table 2 also shows that the proposed method outperforms the weighted slope one collaborative filtering. Indeed, the difference between the two is significant (normal approximation to the binomial test, $p < 0.01$). The performance of the weighted slope one collaborative filtering implies that its recommendation may confuse teachers and pupils because more than half of the recommended books are not relevant.

5.2 Experiment on Reading Level Estimation

The experimental conditions and procedures are as follow. First, we made five pairs of books by randomly selecting a book from the five most difficult books and a book from the five easiest books which the proposed method recommended to each target pupil (50 pairs in total). Second, we randomly labeled the two books in each pair as *A* and *B*. Third, four human raters (undergraduate students) separately determined which book in each pair was more difficult by referring to a book search system that retrieves book information including the title, the author(s), the number of pages, the picture(s) (if available), the reading level (if available), and the synopsis (if available). They separately gave each pair either $+1$, -1, or 0 meaning *A is more difficult*, *B is more is difficult*, and *indistinguishable*, respectively. Then, we merged the results. If the sum is equal to or greater than 3, then A is determined to be more difficult. Similarly, the sum is equal to or smaller than -3, then B is determined to be more difficult. If the sum is -2 or2, the first and second authors joined the four human raters in the evaluation giving newly the pair $+1$, -1, or 0. If the new sum is equal to or greater (smaller) than 3 (-3), then A (B) is determined to be more difficult; otherwise, indistinguishable. Also, if the sum is between -1 and $+1$, the pair is determined to be indistinguishable.

As the results, 34 out of 50 pairs were distinguishable in terms of the reading level. For the 34 pairs, the predictions of the proposed method agreed with the decisions of the human raters 62% of the time (21 Out of 34 pairs). Although the results show that the predictions of the proposed method roughly agree with the decisions of the human raters, the agreement is not as high as we expected. We will discuss the reason in the next section.

6 Discussion

The evaluation has shown that the proposed method is effective in recommending books related to the books that the target pupil borrowed. The reason is that the proposed method predicts absolute loan dates from the relevant books and the relevant pupils.

The effects can be seen in the results of the recommendation. The proposed method is capable of recommending books in series as Table 3 shows. As underlined, the proposed method recommended *Astronomical observation 1*, *4*, and *9* to the pupil who borrowed *Astronomical observation 8*. Information about books in series is useful for recommendation since teachers or book database systems do not necessarily have the information. More importantly, the results show that the proposed method is effective in recommending related books. For instance, it recommended *Constellation observation 1*, which is highly related to *Astronomical observation 8*, *The wonder of the Earth*, *The birth of the great telescope Subaru*, and *Journey in the space*. Table 4 shows another example.

By contrast, the performance of the proposed method concerning reading level is not as high as we expected. As already described in the previous section, the differences in reading level were indistinguishable in 32% of the 50 pairs. For the rest, the predictions of the proposed method agreed with those of the human raters 62% of the time. One of the major reasons is that we used loan histories whose term is one year in the evaluation (or should we say we could only collect that amount?). This means that the difference in reading level is a one-grade higher or lower at most and often much less than one-grade. This explains why 32% of the 50 pairs were indistinguishable in terms of reading level. Considering this, the proposed method will improve in the reading level prediction with longer term loan histories. Another reason is related to the problem of evaluation. It is not so easy to accurately evaluate reading level. It was sometimes difficult for the human raters to determine which book was more difficult by only referring to the book search system. It is possible to take another way of evaluation, which will be our future work.

Table 3. Example of book recommendation (books in series)

Borrowed Books	Recommended Books
Astronomical observation 8: Sun and stars	Astronomical observation 1: Spring constellation
Let's do experiments about air	Astronomical observation 4: Winter constellation
Wonderful science for pupils 10	Astronomical observation 9: Earth, moon, and planets
Journey to the West, vol. 1	Constellation observation 1: Find spring constellations
The wonder of the Earth	Wanamuke the Witch
The birth of the great telescope Subaru	Seton's Wild Animals 8
Questions about earthquakes 1	Football
Experiments about the light and the sight	Brother Bear
Science games	The great maze of Triceratops
Journey in the space	Zorori and a mysterious, magical girl

Table 4. Example of book recommendation (related books)

Borrowed Books	Recommended Books
Kon and Aki	Wonders of bats
The wonder of cold	Stag beetle
Invisible man Samgury in the Zokuzoku village	Mantis
The wonder of brain	Bat
Big feet of Mr. alligator	Mickey mouse: The sorcerer's apprentice
Questions for body functions	Kenta and rabbit
Swallowtail	Zorori: The great operation of ghost
Morning glory	What and what questions from 100 pupils in first grade
Ladybug	Penguin patrol party
Oh, Grandma!	Mrs. Cat's guest
The wonder of teeth and mouth	
Deko-chan	
Firefly	
Cabbage white butterfly	

This section has discussed the effectiveness of the proposed method in book recommendation. Here, it is also worthwhile to discuss the limitations of the proposed method. One of the limitations is that the proposed method is not effective in recommending books to pupils during the last days of school. It is often impossible to take the differences of absolute loan dates in Equation (3) because there are no pupils in higher grades. This is why we excluded pupils in sixth grade from the evaluation. Another limitation is that the proposed method is not capable of recommending books that have never been borrowed; other systems based collaborative filtering have the same limitation. By contrast, teachers or especially librarians can properly recommend such books. It requires other techniques to achieve this.

7 Conclusions

This paper proposed a novel method for book recommendation based on Edu-mining. It has three advantages over the conventional methods: (i) it is inexpensive, (ii) it can recommend books related to the books that the target pupil borrowed, and (iii) reading level is adjustable. The evaluation reveals that the proposed method achieves an accuracy of 60% in recommending related books and outperforms the weighted slope open collaborative filtering. The evaluation also reveals that the reading level predicted by the proposed method roughly agrees with the reading level determined by human raters.

For future work, we will investigate how the prediction of reading level can be evaluated more accurately. We will also investigate how other sources of information can be used to improve the proposed method.

References

[1] Goldberg, D., Nichols, D., Oki B.M., and Terry D. Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*, 1992. 35(12) p. 61-70.

[2] Lemire, D., and Maclachlan, A. Slope One Predictors for Online Rating-Based Collaborative Filtering. *Proceedings of the 5th SIAM International Conference on Data Mining*, 2005. p. 471-475.

[3] Linden, G., Smith, B., York, J. Amazon.com Recommendations: Item-to-item Collaborative Filtering. *IEEE Internet Computing*, 2003, 7(1). p. 76-80.

[4] Nagata, R., Suda, K., Kakegawa, J., Morihiro, K., and Showji, K. Edu-mining for finding keywords to improve message-production skills. *Proceedings of 7th International Conference on Advanced Learning Technologies*, 2007. p. 416-420.

[5] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Proceedings of the 1994 ACM conference on Computer supported cooperative work, 1994*. p. 175-186.

[6] Shardanand, U., and Maes, P. Social Information Filtering: Algorithms for Automating “Word of Mouth”. *Proceedings of the ACM Special Interest Group on Computer-Human Interaction conference on Human factors in computing systems, 1995*. p. 210-217.

[7] Whichbook: http://www.whichbook.net/

Conditional Subspace Clustering of Skill Mastery: Identifying Skills that Separate Students

Rebecca Nugent[1], Elizabeth Ayers[1], and Nema Dean[2]
{rnugent, eayers}@stat.cmu.edu, {nema}@stats.gla.ac.uk
[1]Department of Statistics, Carnegie Mellon University
[2]Department of Statistics, University of Glasgow

Abstract. In educational research, a fundamental goal is identifying which skills students have mastered, which skills they have not, and which skills they are in the process of mastering. As the number of examinees, items, and skills increases, the estimation of even simple cognitive diagnosis models becomes difficult. We adopt a faster, simpler approach: cluster a *capability matrix* estimating each student's individual skill knowledge to generate skill set profile clusters of students. We complement this approach with the introduction of an automatic subspace clustering method that first identifies skills on which students are well-separated prior to clustering smaller subspaces. This method also allows teachers to dictate the size and separation of the clusters, if need be, for practical reasons. We demonstrate the feasibility and scalability of our method on several simulated datasets and illustrate the difficulties inherent in real data using a subset of online mathematics tutor data.

1 Introduction

One of the most important classroom objectives in educational research is identifying students' current stage of skill mastery (complete/partial/none). A variety of cognitive diagnosis models address this problem using information from a student response matrix and an expert-elicited assignment matrix of the skills required for each item [10, 13]. However, even simple models become more difficult to estimate as the numbers of skills, items, and students grow [10]. Faster methods that scale well with large datasets and provide immediate feedback in the classroom are needed. In addition, these methods also need to be able to incorporate practical information from and be interpreted by classroom teachers.

In previous work [1], we introduced a *capability matrix* showing for each skill the proportion correct on all items tried by each student involving that skill (extending the sum-score work of [4,8]) and applied two standard clustering methods to identify students with similar skill set profiles. This approach gives faster, comparable results to common cognitive diagnosis models, scales well to large datasets, and adds flexibility in skill mastery assignment (allowing for partial mastery). However, the use of clustering algorithms usually requires assumptions about the number, size, and shape of the clusters which may be unknown. Moreover, standard techniques do not allow for easy incorporation of user-specified separation and size thresholds.

In this paper, we complement our previous work by proposing an alternative approach, an automatic conditional subspace clustering algorithm that takes advantage of obvious group separation in one or more dimensions (skills). Users do not need to specify a number of clusters nor a particular cluster shape. The method only requires a separation threshold (i.e. how far apart groups of students should be before they would be considered different) and a size threshold (i.e. what size would warrant the implementation of an additional strategy).

After describing the use of the capability matrix (Section 2), we introduce an algorithm in Section 3 that identifies skills with clearly separated groups of students (if any) and correspondingly partitions the feature space. In Sections 4, 5, we demonstrate the approach on simulated data from a common cognitive diagnosis model as well as data from the Assistment Project [7], an ongoing IES funded online mathematics tutor development research project. Finally we conclude with comments on current and future work in Section 6.

2 Skill Set Profile Clustering

After estimating the students' skill knowledge via the capability matrix (or other appropriate estimate), we use clustering methods to partition the students into similar skill set profiles. In recent cognitive diagnosis clustering work, hierarchical clustering, k-means, and model-based clustering have all been utilized. We do not detail the methods here (see e.g. [5, 6]) but instead briefly define and highlight strengths/weaknesses. Also, this paper's focus is the description of an automatic conditional subspace clustering algorithm; detailed comparisons of estimates' and algorithms' performances are elsewhere [2].

2.1 The Capability Matrix

The capability matrix is constructed using an item-skill dependency matrix Q and a student response matrix Y. The Q-matrix, also referred to as a transfer model or skill coding [3, 13], is a $J \times K$ matrix where $q_{jk} = 1$ if item j requires skill k and 0 if it does not, J is the total number of items, and K is the total number of skills. The Q-matrix is usually an expert-elicited assignment matrix. This paper assumes the given Q-matrix is known and correct. Student responses are assembled in a $N \times J$ response matrix Y where y_{ij} indicates both if student i attempted item j and whether or not they answered it correctly and N is the total number of students. If student i did not answer item j, then $y_{ij} = NA$ (i.e. $I_{y_{ij} \neq NA} = 0$). If student i attempted item j ($I_{y_{ij} \neq NA} = 1$), then $y_{ij} = 1$ if they answered correctly (0 if not).

In [1], we define an $N \times K$ *capability matrix B*, where B_{ik} is the proportion of correctly answered items involving skill k that student i attempted,

$$B_{ik} = \frac{\sum_{j=1}^{J} I_{y_{ij} \neq NA} \cdot y_{ij} \cdot q_{jk}}{\sum_{j=1}^{J} I_{y_{ij} \neq NA} \cdot q_{jk}}$$

where y_{ij} and q_{jk} are the corresponding entries from the response matrix Y and Q-matrix. The vector B_i estimates student i's skill set knowledge and then maps student i into a K-dimensional hypercube. For each dimension, zero indicates no skill mastery, one is complete mastery, and values in between are less certain. The 2^K hypercube corners correspond to the true skill set profiles $C_i = \{C_{i1}, C_{i2}, ..., C_{iK}\}, C_{ik} \in \{0, 1\}$. This skill knowledge estimate accounts for the number of items in which the skill appears as well as for missing data.

If $B_{ik} = NA$, we impute an uninformative value (e.g., 0.5, mean, median). Exploring this choice is ongoing. Here we assume the data are complete or correctly imputed. Similarly to [4,8], we find groups of students with similar skill set profiles by clustering the B_i.

2.2 Hierarchical Agglomerative Clustering

Hierarchical agglomerative clustering (HC) "links up" groups in order of closeness to form a tree structure (dendrogram) from which a cluster solution can be extracted. The user-defined distance measure is most commonly Euclidean distance. Briefly, all observations begin as their own group. The distances between all pairs of groups are found (initially just the distance between all pairs of observations). The closest two groups are merged; the inter-group distances are then updated. We alternate the merging and updating operations until we have one group containing all observations. The results are represented in a tree structure where two groups are linked at the height equal to their inter-group distance. The algorithm requires *a priori* how to define the distance between two groups. Here we use the common complete linkage method. Complete linkage defines the distance between two groups as the largest distance between a pair of observations, one from each group, i.e. $d(C_k, C_l) = \max_{i \in C_k, j \in C_l} ||(\underline{x}_i - \underline{x}_j)||^2$. It tends to partition the data into spherical shapes.

Once constructed, we extract G clusters by cutting the tree at the height corresponding to G branches; any cluster solution with $G = 1, 2, ..., N$ is possible. In [4], extraction of $G = 2^K$ clusters is suggested. This choice may not always be wise. First, if not all skill set profiles are present in the population, we may split some profile clusters incorrectly into two or more clusters. Moreover, if $N < 2^K$ (a reasonable scenario for many end-of-year assessment exams), we will be unable to extract the desired number of skill set profiles.

2.3 K-means

K-means is a popular iterative descent algorithm for data $X = \{\underline{x}_1, \underline{x}_2, ..., \underline{x}_n\} \in R^K$. It uses squared Euclidean distance as a dissimilarity measure and tries to minimize within-cluster distance and maximize between-cluster distance. For a given number of clusters G, k-means searches for cluster centers m_g and assignments A that minimize the criterion $\min_A \sum_{g=1}^{G} \sum_{A(i)=g} ||\underline{x}_i - \bar{\underline{x}}_g||^2$. The algorithm alternates between optimizing the cluster centers for the current assignment (by the current cluster means) and optimizing the cluster assignment for a given set of cluster centers (by assigning to the closest current center) until convergence (i.e. cluster assignments do not change). It tends to find compact, spherical clusters and requires the number of clusters G and a starting set of cluster centers.

A common method for initializing k-means is to choose a random set of G observations as the starting set of centers. In our hyper-cube, another natural set of starting cluster centers could be the 2^K skill set profiles at the corners. If students mapped closely to their profile corners, k-means should easily locate the nearby groups. Again, $G = 2^K$ has been suggested [4]. However, again if we are missing representatives from one or more skill set profiles in our population, forcing 2^K clusters may split some clusters into sub-clusters unnecessarily. In [1], this issue was addressed by allowing k-means to have empty clusters.

2.4 Model-Based Clustering

Model-based clustering (MBC) [5, 11] is a parametric statistical approach that assumes: the data $X = \{\underline{x}_1, \underline{x}_2, ..., \underline{x}_n\}$, $\underline{x}_i \in R^K$ are an independently and identically distributed sample from an unknown population density $p(\underline{x})$; each population group g is represented by a (often Gaussian) density $p_g(\underline{x})$; and $p(\underline{x})$ is a weighted mixture of these density components, i.e. $p(\underline{x}) = \sum_{g=1}^{G} \pi_g \cdot p_g(\underline{x}; \theta_g)$ where $\sum \pi_g = 1$, $0 < \pi_g \leq 1$ for $g = 1, 2, ..., G$, and $\theta_g = (\mu_g, \Sigma_g)$ for Gaussian components. The method chooses the number of components G by maximizing the Bayesian Information Criterion (BIC) and estimates the means and variances (μ_g, Σ_g) via maximum likelihood. While it may assume Gaussian components, its flexibility on their shape, volume, and orientation allows student groups of varying shapes and sizes. MBC also often fits overlapping components in an effort to improve fit; users are not able to specify cluster separation information and are also required to give a range of possible numbers of clusters. If multiple students map to the same hypercube location, MBC may overfit the data by using spikes with near singular covariance in these locations. To alleviate this concern (and improve visualization), we jitter the B_i a small amount (0.01). The effect on our results is minimal.

In all three cases, the algorithm returns a set of cluster centers and an assignment vector mapping each B_i to a cluster. A cluster center represents the skill set profile for that subset of students. Note that cluster centers are not restricted to be in the neighborhood of a hypercube corner (although they could be assigned to one). Returning cluster centers rather than their closest corners gives more conservative estimates of skill mastery (vs. 0/1).

As a small illustrative example, we use a subset of 26 items requiring three skills from the Assistment System online mathematics tutor [7]. The Q-matrix is unbalanced; Skill 1 (Evaluating Functions) appears in eight items (six single, two triple), Skill 2 (Multiplication) in 20 items (18 single, two triple), and Skill 3 (Unit Conversion) in two items (both triple). Overall, 551 students answered at least one item. Figure 1 shows the corresponding 3-D cube, each corner one of eight true skill set profiles. Since Unit Conversion appears in only two items, $B_{iUC} \in \{0, \frac{1}{2}, 1\}$; students are mapped to three well-separated planes.

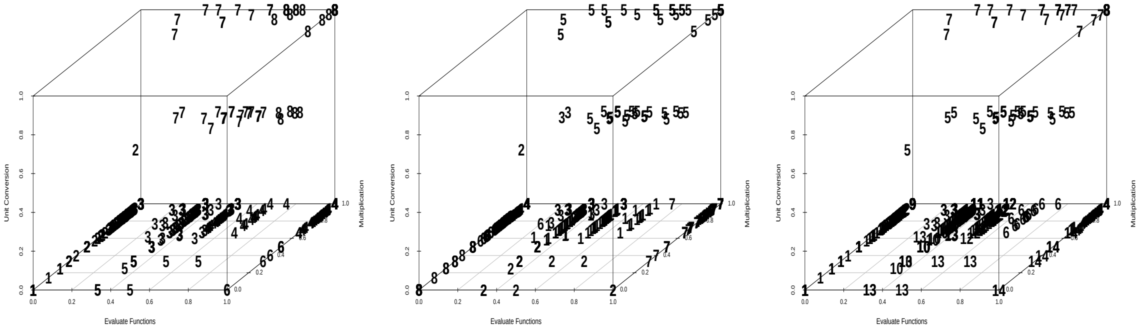


Figure 1: Cluster Assignments: a) HC, Complete G=8; b) K-means G=8; c) MBC G=14

Figures 1a-c) show the clusters found by HC (complete), k-means, and MBC respectively. We set G=2^K=8 for both HC and k-means; MBC searched over G=1 to 25, choosing 14. Only MBC separated the students in the three Unit Conversion planes (B_{iUC}=0:1-4, 6, 9-14; B_{iUC}=0.5: 5; B_{iUC}=1: 7, 8). Both HC and k-means combined students with (arguably very) different Unit Conversion capability across planes into clusters. In contrast, MBC assigns one cluster to the students with B_{iUC}=0.5 and two clusters to those with B_{iUC}=1.0 (the corner cluster contains multiple students). In all three solutions, the B_{iUC}=0 students are split among several clusters defined by their B_{EF} and B_M capabilities. In the HC and k-means results, these clusters include one to three students with B_{iUC}=0.5.

Updating the clusters with new items, skills, etc requires minimal computational time; for example, MBC required ≈ 21 seconds. Classroom teachers can quickly see the changes in the students' skill knowledge over time. However, none of the three solutions seems the obvious winner. In addition, the user was only able to dictate the number of clusters (and somewhat restrict shape); no guarantees were made about their separation and size.

3 Conditional Subspace Clustering or "Valley-Hunting"

In general, clusters are chosen according to a criterion or measure of closeness. Often the user has to define the number of clusters in advance which could be useful to a teacher with fixed resources. For example, he/she might ask for three groups of students clustered on their skill knowledge. However, three clusters may not represent the class well. There may be more or fewer unique skill set profiles. Moreover, the three clusters might be very similar or very different sizes (which both may be impractical). A more useful definition of a cluster might be a well-separated group of students larger than some size threshold.

While any skill's marginal distribution will always have a finite number of unique values, the marginal distribution of some skills may show very well-separated groups of students. We can take advantage of these skills by partitioning the hypercube along their marginal separations. This subsetting alone may be enough to divide students into appropriate clusters. However, it may be the case that there is multivariate cluster structure not detectable by examining the marginal distributions. As such, we advocate using this algorithm either alone or as a dimension reduction tool for other clustering methods. That is, we could first use the marginal distributions to select skills with obvious group structure and then cluster (if needed) the resulting subspaces. Reducing the dimensionality prior to clustering can greatly improve efficiency and/or results [11]. While the Figure 1 hyperplane separation is clear, it could be very difficult to identify obvious separations in a higher dimensional hypercube with noisier marginal distributions. A method to automatically find candidate skills for partitioning (and alert teachers to skills that separate the class) is more desirable.

Akin to the nonparametric clustering notion that a density's mode corresponds to a group in the population [6] and the discretization of continuous variables, we condition on a skill if its marginal distribution contains one or more "significant valleys", a non-trivial area of low density between two high density areas. This decision is made by investigating the marginal distribution's contours. Scanning from zero to one, the low density area must be preceded by a descent and followed by an ascent, both of gradient larger than a specified depth threshold (cluster size), and must be wider than a specified width threshold (cluster separation). There are at least two ways in which low density areas might occur. A skill only occurs in a few items and so has few possible B_{ik} values, or the B_{ik} might be centered around only a few values. If one or more significant valleys are found, we partition the hypercube at the minimum density point of each significant valley. (Other choices could be made, e.g. the halfway point between the two peaks.) In practice, we initially search for significant valleys in all skills' marginal distributions to select skills for partitioning (if any). The resulting subspaces consisting of dimensions (skills) without obvious separations are then clustered if desired; the results can be combined into one final clustering solution.

Let τ_d, τ_w be the respective depth and width separation thresholds (user-specified). These thresholds can be constant or differ over skills (τ_{dk}, τ_{wk}). For computational ease, we use histograms to represent each skill's marginal distribution. The user may also choose a histogram bin width. The automatic subspace partitioning algorithm is as follows:

```
For each skill $k$:
    Calculate the probability histogram for the given bin width. Let $\lambda_i$ = height of Bin $i$.
    Define the gradient $\gamma_{i,i+1}$ as the difference in the percent of students in bins $i, i+1$.
    Let $\gamma_N = \lambda_i - \lambda_j$ be the total descent gradient from a peak (Bin $i$) to a valley (Bin $j$).
    Let $\gamma_P = \lambda_i - \lambda_j$ be the total ascent gradient from a valley (Bin $i$) to a peak (Bin $j$).
    Let $L_m$ be the location of the mode preceding the current valley (scan's startpoint).
    Let $L_v$ be the location of the lowest height of the current valley.
    Initialize $L_m = L_v$ = Bin 1.

**1)** Scan $\gamma_{i,i+1}$ until $\gamma_{i,i+1} < 0$.
    If no such gradient exists, there are no remaining valleys.
**2)** Else, scan $\gamma_{i,i+1}$ until $\gamma_{i,i+1} \geq 0$ (end of valley) or out of bins; compute $\gamma_N$.
        If $|\gamma_N| > \tau_d$, have found a "significant" descent. Set $L_v$ = Bin $i+1$.
        **3)** Scan $\gamma_{i,i+1}$ until $\gamma_{i,i+1} < 0$ (end of peak) or out of bins; compute $\gamma_P$.
            If $|\gamma_P| > \tau_d$, we have found a "significant" ascent. Find valley width $w$.
                If $w > \tau_w$, significant valley; store mode locations. Else, do not store.
                In either case, set $L_m = L_v$ = Bin $i+1$. Scan for next valley **(return to 1)**.
            Else, have not found significant ascent.
                Scan $\gamma_{i,i+1}$ until $\gamma_{i,i+1} \geq 0$ (end of next valley) or out of bins.
                If $\lambda_{i+1} < \lambda_{L_v}$, current valley is lower than valley at $L_v$.
                    Set $L_v$ = Bin $i+1$. **(return to 3)**
                Else, current valley is higher than valley at $L_v$; have "hiccup mode".
                    **(return to 3)**
        Else, have not found a significant descent.
            Scan $\gamma_{i,i+1}$ until $\gamma_{i,i+1} < 0$ (end of next peak) or out of bins.
            If $\lambda_{i+1} > \lambda_{L_m}$, current peak is higher than peak at $L_m$.
                Set $L_m$ = Bin $i+1$. Scan for next valley **(return to 1)**.
            Else, current peak is lower than peak at $L_m$; have "hiccup mode". **(return to 2)**
```

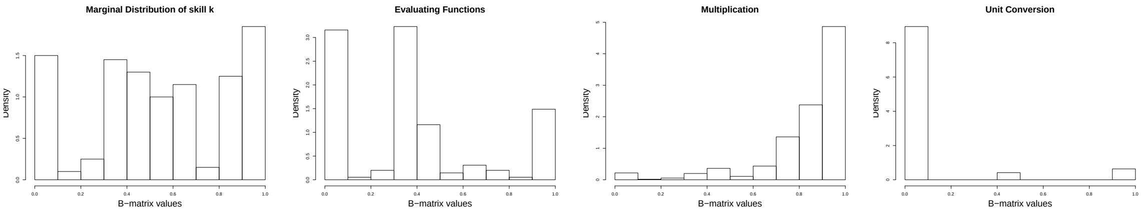


Figure 2: Marginal Skill Distributions: Illustrative Example, Three Assistment Skills

The spirit of our algorithm is similar to mode-hunting (e.g. [12]) excepting that we only want to identify modes that are separated by a valley of substantial depth and width. In a sense, we are "valley-hunting". For example, if while searching for a descent of substantial depth we find a "hiccup mode" where the marginal distribution slightly increases and then continues to decrease, the algorithm does not view that small valley to be important. (A "hiccup mode" might similarly be found when searching for a substantial ascent.) Figure 2a contains an example marginal distribution of Skill k, a histogram with bin width = 0.10. For example, say a teacher will only adapt classroom strategies for groups of students who are at least 10% of the class and whose capability values are separated by at least 20%. Given $\tau_d = 0.1, \tau_w = 0.2$, we start at Bin 1 and immediately find a descent of 0.14 $(1.5 \cdot 0.10 - 0.1 \cdot 0.10)$. We know that there is at least one bin in the preceding mode with at least 10% of the students (our depth threshold). We continue scanning to find a total ascent of 0.135 $(1.45 \cdot 0.10 - 0.1 \cdot 0.10)$ at Bin 4, evidence that the next mode also has at least 10% of the students. As both gradients exceed τ_d, we check that the valley is wide enough by measuring the distance between the two modes (0.0, 0.3). Since $0.3 > 0.2 = \tau_w$, both modes are separated by at least 20% capability, and we have identified a "significant valley". Continuing to scan, we find another descent and valley at Bin 6. In this case, the descent is not large enough yet to indicate a well-separated group (Bin 7 is a "hiccup mode"). A large enough descent is eventually found between Bin 4 and Bin 8, followed by a significant ascent. The next significant valley is then from Bin 4 to Bin 10. We partition the skill at Bin 2 (0.15) and Bin 8 (0.75) to create three groups of students of size at least 10% of the class separated by at least 20% capability on Skill k. If our thresholds were $\tau_d = .045, \tau_w = 0.10$, four groups would have been found (cutpoints: 0.15, 0.55, 0.75).

Figure 2 also includes the three Assistment skill marginal distributions. While Unit Conversion (Figure 2d) has three well-separated peaks, given reasonable depth/size thresholds, our algorithm would not partition this skill since two non-zero bin counts are very small (i.e. modes of trivial mass). We also would likely not partition the skewed Multiplication distribution. Given τ_d=0.1, τ_w=0.2, we do partition Evaluate Functions at 0.15, 0.75 for three groups of students and cluster the three subsequent two-dimensional subspaces. Figure 3 shows the methods' respective results. There is less cross-plane clustering in HC and k-means without partitioning Unit Conversion (Figures 3a,b). MBC again chose 14 total with similar results; however, the subspace clustering (including both finding the partitions and clustering the subspaces) took ≈ 6 seconds (vs. 21) for computational savings of 71%.

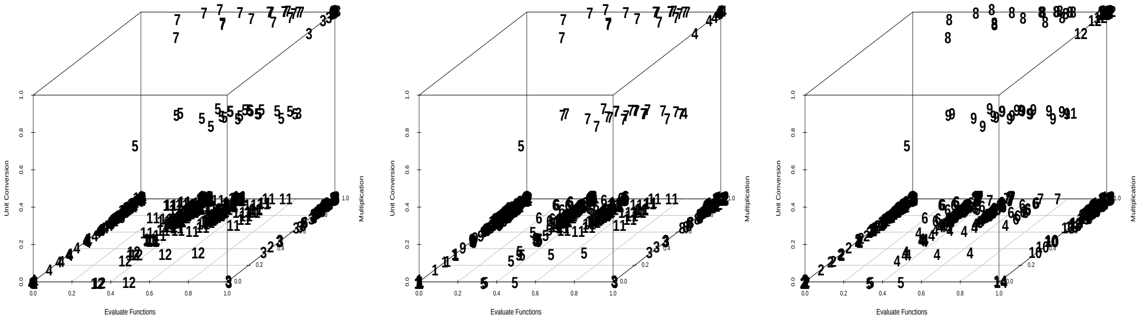


Figure 3: Cluster Assignments: a) HC, Complete G=3 · 2^2; b) K-means G=3 · 2^2; c) MBC G=14

4 Recovering the True Skill Set Profiles

In this section, we simulate data from the DINA model, a common educational research model, to compare the methods' ability to recover the students' true skill set profiles. The deterministic inputs, noisy "and" gate model (DINA) is a conjunctive cognitive diagnosis model used to estimate student skill knowledge [10]. The DINA model item response form is $P(y_{ij} = 1 \mid \eta_{ij}, s_j, g_j)$= $(1 - s_j)^{\eta_{ij}} g_j^{1-\eta_{ij}}$ where $\alpha_{ik} = I_{\{\text{Student } i \text{ has skill } k\}}$ and $\eta_{ij} = \prod_{k=1}^{K} \alpha_{ik}^{q_{jk}}$ indicates if student i has all skills needed for item j; s_j= $P(y_{ij}$=0 | η_{ij}=1) is the slip parameter; and g_j= $P(y_{ij}$=1 | η_{ij}=0) is the guess parameter. If student i is missing any of the required skills for item j, $P(y_{ij} = 1)$ decreases due to the conjunctive assumption. Prior to simulating the y_{ij}, we fix the skills to be of equal medium difficulty with an inter-skill correlation of either 0 or 0.25 and generate true skill set profiles C_i for each student. In our work thus far, only a perfect inter-skill correlation has a non-negligible effect on the results. These parameter choices evenly spread students among the 2^K natural skill set profiles. We randomly draw our slip and guess parameters ($s_j \sim$ Unif(0,0.30); $g_j \sim$ Unif(0,0.15)). Given the true skill set profiles and slip/guess parameters, we generate the student response matrix Y. Then, using a fixed Q matrix, we calculate and cluster the corresponding B matrix.

For the first three methods, no partitioning is done (HC, k-means: $G = 2^K$; MBC: searches from 1 to $G > 2^K$). In conditional subspace clustering, we initially use τ_d= 0.1, τ_w= 0.2 and then cluster the resulting subspaces (if any). To gauge performance, we calculate their agreement to the true profiles using the Adjusted Rand Index (ARI), a common measure of agreement between two partitions [9]. Under random partitioning, E[ARI] = 0, and the maximum value is one. Larger values indicate better agreement.

Table 1 presents selected simulations for $K = 3, 7, 10$ for varying J, N. In the Cond (MBC) column, the first ARI corresponds to the partitioning alone, the second to the clustering of the partitioned subspaces (with MBC). We also vary the Q-matrix design to include only single skill items, only multiple skill items, or both. In addition, the Q-matrix was balanced (bal) or unbalanced (unbal). If balanced, all skills and skill combinations occur the same number of times. Unbalanced refers to uneven representation of or missing skills (miss).

Table 1: Comparing Clustering Methods with the True Generating Skill Set Profiles via ARIs

K	J	N	Q Matrix Design	HAC	K-means	MBC	Cond (MBC)	Selected Skills
3	30	250	Single	1.000	1.000	0.970	1.000	3
3	30	250	Both, bal	0.792	0.615	0.939	0.531 (0.402)	2
3	30	250	Both, unbal, uneven	0.541	0.625	0.703	0.241 (0.641)	1
3	30	250	Both, unbal, miss	0.582	0.578	0.707	0.249 (0.713)	1
3	30	250	Multiple, bal	0.414	0.419	0.416	0.222 (0.495)	1
3	30	250	Multiple, unbal, uneven	0.350	0.504	0.515	—	0
3	30	250	Multiple, unbal, miss	0.235	0.242	0.194	—	0
7	40	300	Single	0.746	0.553	0.987	0.982	7
7	40	300	Both, unbal, miss	0.333	0.308	0.386	0.290	3
10	100	2500	Single	0.876	0.786	0.062	0.958	10

Excepting the multiple unbalanced design, the subspace algorithm selected one or more skills for partitioning (in some cases, all skills were correctly selected). In almost all simulations, MBC was comparable to or better than HC and k-means for true skill set profile recovery. The partitioning method coupled with using MBC on the reduced subspaces gave comparable or better results in all cases except the balanced single and multiple skill design. In addition, subspace partitioning/MBC was always faster than MBC alone.

Table 2: Comparison of Depth, Width Thresholds

τ_d	τ_w	Cond (MBC)	Selected Skills
0.1	0.2	0.249 (0.713)	1
0.1	0.1	0.249 (0.713)	1
0.05	0.2	0.569 (0.510)	2
0.05	0.1	0.569 (0.510)	2
0.025	0.1	0.629 (0.694)	3

In addition, for the fourth K= 3, J= 30 Q matrix design, we vary the depth and width thresholds. Smaller values of τ_d, τ_w will find narrower, shallower separations; in addition, smaller isolated clusters will be found. In this particular example, we found that as we decreased the depth threshold, more skills were (correctly) selected, and the performance of the partitioning by itself improved. While the parameters are designed to be user-specified, we are currently exploring their behavior in order to make good default suggestions.

5 Thirteen Skill Assistment Example

Finally, we briefly look at a higher dimensional Assistment example with K=13 skills, N=344 students, and J=135 items. This data set included multiple skill items and a large amount of missing response data. HC and k-means are not appropriate choices; finding 2^{13}=8192 clusters is unreasonable (without, say, allowing for empty clusters as in [1]); MBC will largely depend on choosing an appropriate search range. The conditional subspace clustering algorithm, however, searches the space for obvious separation and partitions 9 of the 13 skills for a total of 221 subspaces (1 sec). All subspaces contained $\leq$ 13 students and so could likely be used alone or as subspaces for further clustering if needed.

6 Conclusions

We presented a conditional subspace clustering algorithm for use with the capability matrix (or similar skill knowledge estimate). The method selects skills that separate students well and reduces dimensionality for subsequent clustering. Our work so far shows that for most Q-matrix designs, the recovery of true skill set profiles is comparable or better than other clustering methods while also including skill selection. Since the true profiles in the Assistment examples are unknown, we cannot judge their recovery. However, visual inspection indicates that the partitions and skill selection seem sensible. To our knowledge, work in this area has not adequately addressed the need to analyze high-dimensional Q-matrices. The approach presented, while allowing for real time estimation of student skill set profiles, can handle large numbers of skills as well as incorporate practical user specifications.

References

[1] Ayers, E, Nugent, R, Dean, N. “Skill Set Profile Clustering Based on Student Capability Vectors Computed from Online Tutoring Data”. *Educational Data Mining 2008: 1st International Conference on Educational Data Mining, Proceedings* (refereed). R.S.J.d. Baker, T. Barnes, and J.E. Beck (Eds), Montreal, Quebec, Canada, June 20-21, 2008. p.210-217.

[2] Ayers, E, Nugent, R, Dean, N. “A Comparison of Student Skill Knowledge Estimates”. *Educational Data Mining 2009: 2nd International Conference on Educational Data Mining*, accepted.

[3] Barnes, T.M. (2003). *The Q-matrix Method of Fault-tolerant Teaching in Knowledge Assessment and Data Mining*. Ph.D. Dissertation, Department of Computer Science, NCSU.

[4] Chiu, C (2008). *Cluster Analysis for Cognitive Diagnosis: Theory and Applications*. Ph. D. Dissertation, Educational Psychology, University of Illinois at Urbana Champaign.

[5] Fraley, C. and Raftery, A. Mclust: Software for model-based cluster analysis. *Journal of Classification*, 1999, 16, 297-306.

[6] Hartigan, J.A. *Clustering Algorithms*. Wiley. 1975.

[7] Heffernan, N.T., Koedinger, K.R. and Junker, B.W. *Using Web-Based Cognitive Assessment Systems for Predicting Student Performance on State Exams.* Research proposal to the Institute of Educational Statistics, US Department of Education. Department of Computer Science at Worcester Polytechnic Institute, Worcester County, Massachusetts, 2001.

[8] Henson, J., Templin, R., and Douglas, J. Using efficient model based sum-scores for conducting skill diagnoses. *Journal of Education Measurement*, 2007, 44, 361-376.

[9] Hubert, L. and Arabie, P. Comparing partitions. *Journal of Classification*, 1985, 2, 193-218.

[10] Junker, B.W., Sijtsma K. Cognitive Assessment Models with Few Assumptions and Connections with Nonparametric Item Response Theory. *Applied Psych Measurement*, 2001, 25, 258-272.

[11] Raftery, A and Dean, N. *Variable Selection for Model-Based Clustering*. Journal of the American Statistical Association, Vol. 101, No. 473 (March 2006), pp. 168-178.

[12] Silverman, B. W. (1981) Using kernel density estimates to investigate multimodality.*J. Royal. Statistical Society Series B*. 43: 97-99.

[13] Tatsuoka, K.K. (1983). Rule Space: An Approach for Dealing with Misconceptions Based on Item Response Theory. *Journal of Educational Measurement*. 1983, Vol. 20, No. 4, 345-354.

Determining the Significance of Item Order In Randomized Problem Sets

Zachary A. Pardos[1] and Neil T. Heffernan
{zpardos, nth}@wpi.edu
Worcester Polytechnic Institute

Abstract. Researchers who make tutoring systems would like to know which sequences of educational content lead to the most effective learning by their students. The majority of data collected in many ITS systems consist of answers to a group of questions of a given skill often presented in a random sequence. Following work that identifies which items produce the most learning we propose a Bayesian method using similar permutation analysis techniques to determine if item learning is context sensitive and if so which orderings of questions produce the most learning. We confine our analysis to random sequences with three questions. The method identifies question ordering rules such as, question A should go before B, which are statistically reliably beneficial to learning. Real tutor data from five random sequence problem sets were analyzed. Statistically reliable orderings of questions were found in two of the five real data problem sets. A simulation consisting of 140 experiments was run to validate the method's accuracy and test its reliability. The method succeeded in finding 43% of the underlying item order effects with a 6% false positive rate using a p value threshold of <= 0.05. Using this method, ITS researchers can gain valuable knowledge about their problem sets and feasibly let the ITS automatically identify item order effects and optimize student learning by restricting assigned sequences to those prescribed as most beneficial to learning.

1 Introduction

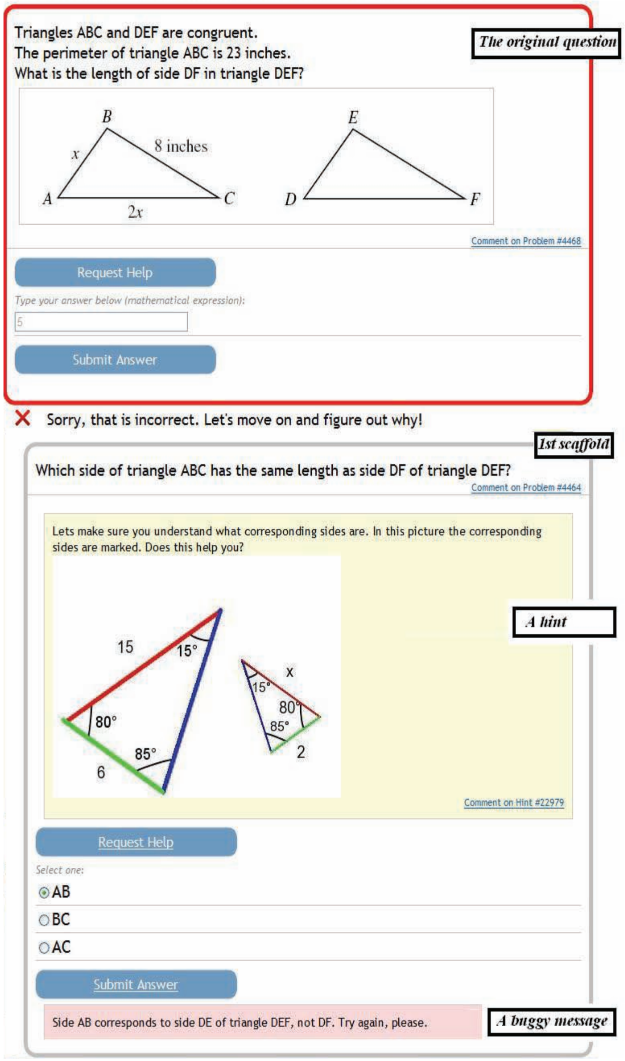
Corbett and Anderson style knowledge tracing [3] has been successfully used in many tutoring system to predict a student’s knowledge of a knowledge component after seeing a set of questions that used that knowledge component. We present a method that allows us to detect if the learning value of an item might be dependent on the particular context the question appears in. We will model learning rates of items based on what item comes immediately after it. This will allow us to identify rules such as; item A should come before B, if such a rule exists. Question A could also be an un-acknowledged prerequisite for answering question B. After finding such relationships between questions, a reduced set of sequences can be recommended. The reliability of our results is tested with a simulation study in which simulated student responses are generated and the method is tasked with learning the underlying parameters of the simulation.

We presented a method [5] that used similar analysis techniques to this one, where an item effect model was used to determine which items produced the most learning. That method had the benefit of being able to inform Intelligent Tutoring System (ITS) researchers of which questions, and their associated tutoring, are or are not producing learning. While we think that method has much to offer, it raised the question of whether the learning value of an item might be dependent on the particular context it appears in. The method in this paper is focused on learning based on item sequence.

[1] National Science Foundation funded GK-12 Fellow

1.1 The Tutoring System and Dataset

Our dataset consisted of student responses from The ASSISTment System, a web based math tutoring system for 7th-12th grade students that provides preparation for the state standardized test by using released math items from previous tests as questions on the system. Figure 1 shows an example of a math item on the system and tutorial help that is given if the student answers the question wrong or asks for help. The tutorial helps the student learn the required knowledge by breaking the problem into sub questions called scaffolding or giving the student hints on how to solve the question.


Triangles ABC and DEF are congruent.
The perimeter of triangle ABC is 23 inches.
What is the length of side DF in triangle DEF?

Sorry, that is incorrect. Let's move on and figure out why!

Which side of triangle ABC has the same length as side DF of triangle DEF?

Lets make sure you understand what corresponding sides are. In this picture the corresponding sides are marked. Does this help you?

Side AB corresponds to side DE of triangle DEF, not DF. Try again, please.

Figure 1. An ASSISTment item

The data we analyzed was from the 2006-2007 school year. Subject matter experts made problem sets called GLOPS (groups of learning opportunities). The idea behind the GLOPS was to make a problem set where the items in the problem set related to each other. They were not necessary strictly related to each other through a formal skill tagging convention but were selected based on their similarity of concept according to the expert. We chose the five three item GLOPS that existed in the system each with between 295 and 674 students who had completed the problem set. Items do not overlap across GLOP problem sets. Our analysis can scale to problem sets of six items but we wanted to start off with a smaller size set for simplicity in testing and explaining the analysis method. The items in the five problem sets were presented to students in a randomized order. Randomization was not done for the sake of this research in particular but rather because the assumption of the subject matter expert was that these items did not have an obvious progression requiring that only a particular sequence of the items be presented to students. In other words, context sensitivity was not assumed. We only analyzed responses to the original questions which meant that a distinction was not made between the learning occurring due to answering the original question and learning occurring due to the help content. The learning from answering the original question and scaffolding will be conflated as a single value for the item.

1.2 Knowledge Tracing

The Corbett and Anderson method of "knowledge tracing" [3] has been useful to many intelligent tutoring systems. In knowledge tracing there is a set of questions that are assumed to be answerable by the application of a particular knowledge component which could be a skill, fact, procedure or concept. Knowledge tracing attempts to infer the probability that a student knows a knowledge component based on a series of answers. Presumably, if a student had a response sequence of 0,0,1,0,0,0,1,1,1,1,1,1 where 0 is an incorrect first response to a question and 1 is a correct response, it is likely she guessed the third question but then learned the knowledge to get the last 6 questions correct. The Expectation Maximization algorithm is used in our research to learn parameters from data such as the probability of guess.

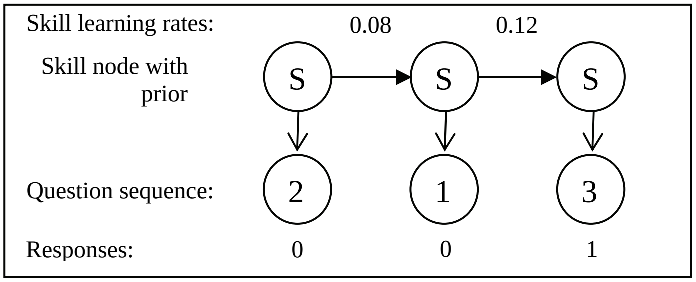


Figure 2. Bayesian network model for question sequence [2 1 3]

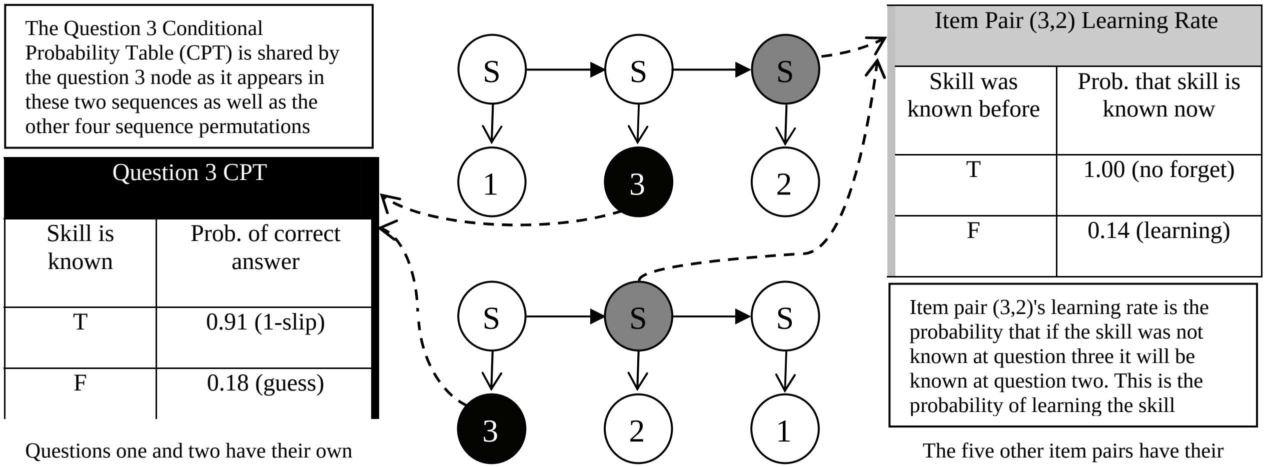
Figure 2 depicts a typical knowledge tracing three question static Bayesian network. The top three nodes represent a single skill and the inferred value of the node represents the probability the student knows the skill at each opportunity. The bottom three nodes represent three questions on the tutor. Student performance on a question is a function of their skill knowledge and the guess and slip of the question. Guess is the probability of answering correctly if the skill is not known. Slip is the probability of answering incorrectly if the skill is known. Learning rates are the probability that a skill will go from "not known" to "known" after encountering the question. The probability of the skill going from "known" to "not known" (forgetting) is fixed at zero. Knowledge tracing assumes that the learning on a piece of knowledge is independent of the question presented to students, that is that all questions should lead to the same amount of learning. The basic design of a question sequence in our model is similar to a dynamic Bayesian network or Hidden Markov Model used in knowledge tracing but with the important distinction that the probability of learning is able to differ between opportunities. This ability allows us to model different learning rates per question which is essential to our analysis. The other important distinction of our model is the ability to model permutations of sequences with parameter sharing, discussed in the next section.

2 Analysis Methodology

In order to represent all the data in our randomized problem sets of three items we must model all six possible item sequence permutations. If six completely separate networks were created then the data would be split into six which would degrade the accuracy of parameter learning. This would also learn a separate guess and slip for each question in each sequence despite the questions being the same in each sequence. In order to leverage the parameter learning power of all the data and define an individual question's guess and slip values we will use parameter sharing[2] to link the parameters across the different sequence networks. This means that question one as it appears in all six sequences will share the same guess and slip conditional probability table (CPT). The same will be true for the other two questions. This will give us three guess and slip parameters total and the values will be trained to reflect the questions' non sequence specific guess and slip values. In our item order effect model we also link the learning rates of item sequences.

2.1 *The Item Order Effect Model*

In the model we call the item order effect model we look at what effect item order has on learning. We set a learning rate for each pair of items and then test if one pair is reliably better for learning than another. For instance, should question A come before question B or vice versa? Since there are three items in our problem sets there will be six ordered pairs which are (3,2) (2,3) (3,1) (1,3) (2,1) and (1,2). This model allows us to train the learning rates of all six ordered pairs simultaneously along with guess and slip for the questions by using shared parameters to link all occurrences of pairs to the same learning rate conditional probability table. For example, the ordered pair (3,2) appears in two sequence permutations; sequence (3,2,1) and sequence (1,3,2) as shown in Figure 3.



The Question 3 Conditional Probability Table (CPT) is shared by the question 3 node as it appears in these two sequences as well as the other four sequence permutations

Question 3 CPT	
Skill is known	Prob. of correct answer
T	0.91 (1-slip)
F	0.18 (guess)

Questions one and two have their own shared CPTs as well

Item Pair (3,2) Learning Rate	
Skill was known before	Prob. that skill is known now
T	1.00 (no forget)
F	0.14 (learning)

Item pair (3,2)'s learning rate is the probability that if the skill was not known at question three it will be known at question two. This is the probability of learning the skill

The five other item pairs have their own CPTs in the full network

Figure 3. A two sequence portion of the Item Order Effect Model (six sequences exist in total)

2.2 *Reliability Estimates Using the Binomial Test*

In order to derive the reliability of the learning rates fit from data we employed the binomial test[3] by randomly splitting the response data into 10 by student. We fit the model parameters using data from each of the 10 bins separately and counted the number

[2] Parameter sharing was accomplished in the Bayesian network model using equivalence classes from Kevin Murphy's Bayes Net Toolbox, available at: http://bnt.sourceforge.net/

[3] The binomial test was run with the MATLAB command: binopdf(successes, trials, 1/outcomes)

of bins in which the learning rate of one item pair was greater than its reverse, (3,2) > (2,3) for instance. We call a comparison of learning rates such as (3,2) > (2,3) a rule. The null hypothesis is that each rule is equally likely to occur. A rule is considered statistically reliable if the probability that the result came from the null hypothesis is <= 0.05. For example, if we are testing if ordered pair (3,2) has a higher learning rate than (2,3) then there are two possible outcomes and the null hypothesis is that each outcome has a 50% chance of occurring. Thus, the binomial test will tell us that if the rule holds true eight or more times out of ten then it is <= 0.05 probable that the result came from the null hypothesis. This is the same idea as flipping a coin 10 times to determine the probability it is fair. The less likely the null hypothesis, the more confidence we can have in the result. If the learning rate of (3,2) is greater than (2,3) with p <= 0.05 then we can say it is statistically reliable that question three and its tutoring followed by question two better help students learn the skill than question two and its tutoring followed by question three. Based on this conclusion it would be recommended to give sequences where question three comes before two. The successful detection of a single rule will eliminate half of the sequences since three comes before two in half of the sequence permutations. Strictly speaking the model is only reporting the learning rate when two comes directly after three however in eliminating half the sequences we make the pedagogical assumption that question three and its tutoring will help answer question two even if it comes one item prior such as in the sequence (3, 1, 2). Without this assumption only the two sequences with (2,3) can be eliminated and not sequence (2,1,3).

2.3 Item Order Effect Model Results

We ran the analysis method on our problem sets and found reliable rules in two out of the five problem sets. The results below show the item pair learning rate parameters for the two problem sets in which reliable rules were found. The 10 bin split was used to evaluate the reliability of the rules while all student data for the respective problem sets were used to train the parameters shown below.

Table 1. Item order effect model results

		Learning probabilities of Item Pairs						
Problem Set	**Users**	*(3,2)*	*(2,1)*	*(3,1)*	*(1,2)*	*(2,3)*	*(1,3)*	**Reliable Rules**
24	403	0.1620	0.0948	0.0793	0.0850	0.0754	0.0896	(3,2) > (2,3)
36	419	0.1507	0.1679	0.0685	0.1179	0.1274	0.1371	(1,3) > (3,1)

As shown in Table 1, there was one reliable rule found in each of the problem sets. In problem set 24 we found that item pair (3,2) showed a higher learning rate than (2,3) in eight out of the 10 splits giving a binomial p of 0.0439. Item pair (1,3) showed a higher learning rate than (3,1) also in eight out of the 10 splits in problem set 36. Other statistically reliable relationships can be tested on the results of the method. For instance, in problem set 36 we found that (2,1) > (3,1) in 10 out of the 10 bins. This could mean that sequence (3,1,2) should not be given to students because question three comes before question one and question two does not. Removing sequence (3,1,2) is also supported by rule (1,3) > (3,1). In addition to the learning rate parameters, the model simultaneously trains a guess and slip value for each question. Those values are shown below in Table 2.

Table 2. Trained question guess and slip values

	Problem Set 24		**Problem Set 36**	
Question #	*Guess*	*Slip*	*Guess*	*Slip*
1	0.17	0.18	0.33	0.13
2	0.31	0.08	0.31	0.10
3	0.23	0.17	0.20	0.08

3 Simulation

In order to determine the validity of the item order effect method we chose to run a simulation study exploring the boundaries of the method's accuracy and reliability. The goal of the simulation was to generate student responses under various conditions that may be seen in the real world and test if the method would accurately infer the underlying parameter values from the simulated student data. This simulation model assumes that learning rates have distinct values and that item order effects of some magnitude always exist and should be detectable given enough data.

3.1 Model design

The model used to generate student responses is a six node static Bayesian network as depicted in Figure 2 from section 1.2. While the probability of knowing the skill will monotonically increase after each opportunity, the generated responses (0s and 1s) will not necessarily do the same since those values are generated probabilistically based on skill knowledge and guess and slip. Simulated student responses were generated one student at a time by sampling from the six node network.

3.2 Student parameters

Only two parameters were used to define a simulated student, a prior and question sequence. The prior represents the probability the student knew the skill relating to the questions before encountering the questions. The prior for a given student was randomly generated from a distribution that was fit to a previous year's ASSISTment data [6]. The mean prior for that year across all skills was 0.31 and the standard deviation was 0.20. In order to draw probabilistic parameter values that fit within 0 and 1, an equivalent beta distribution was used. The beta distribution fit an α of 1.05 and β of 2.43. The question sequence for a given student was generated from a uniform distribution of sequence permutations.

3.3 Tutor Parameters

The 12 parameters of the tutor simulation network consist of six learning rate parameters, three guess parameters and three slip parameters. The number of users simulated was: 200, 500, 1000, 2000, 4000, 10000, and 20000. The simulation was run 20 times for each of the seven simulated user sizes totaling 140 generated data sets, referred to later as experiments. In order to faithfully simulate the conditions of a real tutor, values for the 12 parameters were randomly generated using the means and standard deviations across 106 skills from a previous analysis [6] of ASSISTment data. Table 3 shows the distributions that the parameter values were randomly drawn from and then assigned to questions and learning rates at the start of each run.

Table 3. The distributions used to generate parameter values in the simulation

Parameter type	Mean	Std	Beta dist α	Beta dist β
Learning rate	0.086	0.063	0.0652	0.6738
Guess	0.144	0.383	0.0170	0.5909
Slip	0.090	0.031	0.0170	0.6499

Running the simulation and generating new parameter values 20 times gives us a good sampling of the underlying distribution for each of the seven user sizes. This method of generating parameters will end up accounting for more variance than the real world since standard deviations were calculated for values across problem sets as opposed to within. Also, guess and slip have a correlation in the real world but will be allowed to independently vary in the simulation which means sometimes getting a high slip but low guess, which is rarely observed in actual ASSISTment data. It also means the potential for generating very improbable combinations of item pair learning rates.

3.4 Simulation Procedure

The simulation consisted of three steps: instantiation of the Bayesian network, setting CPTs to values of the simulation parameters and student parameters and finally sampling the Bayesian network to generate the students' responses.

To generate student responses the six node network was first instantiated in MATLAB using routines from the Bayes Net Toolbox package. Student priors and question sequences were randomly generated for each simulation run and the 12 parameters described in section 3.3 were assigned to the three questions and item pair learning rates. The question CPTs and learning rates were positioned with regard to the student's particular question sequence. The Bayesian network was then sampled a single time to generate the student's responses to each of the three questions; a zero indicating an incorrect answer and a one indicating a correct answer. These three responses in addition to the student's question sequence were written to a file. A total of 140 data files were created at the conclusion of the simulation runs, all of which were to be analyzed by the item order effect detection method. The seeded simulation parameters were stored in a log file for each experiment to later be checked against the method's findings. An example of an experiment's output file for 500 users is shown in Table 4 below.

Table 4. Example output from data file with N=500

Simulated User	Sequence identifier	1st Q	2nd Q	3rd Q
1	5	0	1	1
⋮	⋮	⋮	⋮	⋮
500	3	1	0	1

Each data file from the simulation was split into 10 equal parts and each run separately through the analysis method just as was done in analysis of real tutor data. This analysis step would give a result such as the example in Table 5 below.

Table 5. Example output from item order effect analysis

	(3,2)	(2,1)	(3,1)	(1,2)	(2,3)	(1,3)
Split 1	0.0732	0.0267	0.0837	0.0701	0.0379	0.642
⋮	⋮	⋮	⋮	⋮	⋮	⋮
Split 10	0.0849	0.0512	0.0550	0.0710	0.0768	0.0824

In order to produce a p value and determine statistical reliability to the $p < 0.05$ level the binomial test is used. The method counts how many times (3,2) was greater than (2,3) for instance. If the count is greater than eight then the method considers this an identified rule. Even though there are six item pairs there is a maximum of three rules since if (3,2) > (2,3) is a reliable rule then (3,2) < (2,3) is not. In some cases finding two rules is enough to identify a single sequence as being best. Three rules always guarantee the identification of a single sequence. The method logs the number of rules found and how many users (total) were involved in the experiment. The method now looks "under the hood" at the parameters set by the simulation for the item pair learning rates and determines how many of the found rules were false. For instance, if the underlying simulated learning rate for (3,2) was 0.08 and the simulated learning rate for (2,3) was 0.15 then the rule (3,2) > (2,3) would be a false positive (0.08 < 0.15). This is done for all 140 data files. The total number of rules is three per experiment thus there are 420 rules to be found in the 140 data files.

3.5 Simulation Results

The average percent of found rules per simulated user size is plotted in Figure 2 below. The percentage of false positives is also plotted in the same figure and represents the error.

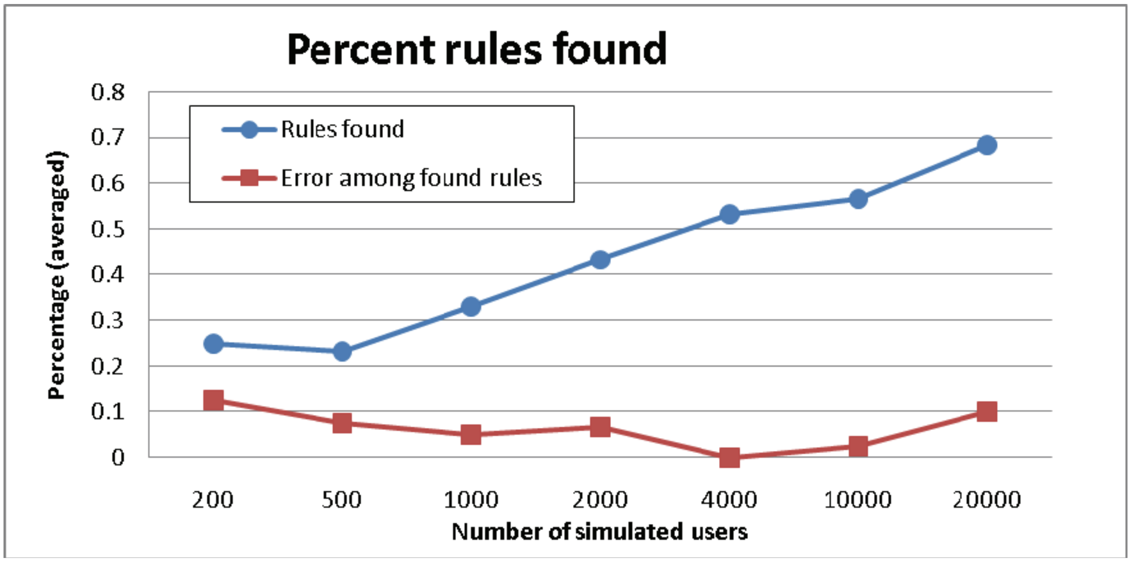


Figure 4. Results of simulation study

Figure 4 shows that more users allows for more rules about item order to be detected. It also shows that the false positive rate remains fairly constant, averaging around the 6% mark. From 200 users to 1,000 users the average percentage rules found was around 30% which would correspond to about 1 rule per problem set (0.30 * 3). This percentage rises steadily in a linear fashion from 500 users up to the max number of users tested of 20,000 where it achieves a 69% discovery rate which corresponds to about two rules per problem set on average. The error starts at 13% with 200 users and then remains below 10% for the rest of the user sizes. The overall average percent of rules found across users sizes is 43.3%. The overall average false positive rate is 6.3% which is in line with the binomial p value threshold of 0.05 that was used and validates the accuracy of the method's results and dependability of the reported binomial p value.

Limitations and Future Work

One of the limitations of this permutation analysis method is that it does not scale gracefully. The number of network nodes that need to be constructed is exponential in the number of items. For a three item model there are six nodes per sequence and six sequences. For a seven item model there are fourteen nodes per sequence and 5,040 sequences (70,560 nodes). One potential optimization would be to only construct sequences for which there is data, which will be at most the number of students.

The split 10 procedure has the effect of decreasing the amount of data the method has to operate on for each run. A more efficient sampling method may be beneficial, however, our trials using resampling with replacement for the simulation instead of splitting resulted in a high average false positive rate (>15%). A more sensitive test that takes into account the size of the difference between learned parameter values would improve reliability estimates. The binomial accuracy may also be improved by using a Bonferroni correction as suggested by a reviewer. This correction is used when multiple hypotheses are tested on a set of data (i.e. the reliability of item ordering rules). The correction suggests using a lower p value cut-off.

There is a good deal of work in the area of trying to build better models of what students are learning. One approach [1] uses a matrix of skill to item mappings which can be optimized [2] for best fit and used to help learn optimal practice schedules [7] while another approach attempts to find item to item knowledge relationships [4] such as prerequisite item structures using item tree analysis. We think that the item order effect method introduced here and its accompanying paper [5] have parallels with these works and could be used as a part of a general procedure to try to learn better fitting models.

Contribution

This method has been shown by simulation study to provide reliable results suggesting item orderings that are most advantageous to learning. Many educational technology companies [8] (i.e. Carnegie Learning Inc. or ETS) have hundreds of questions that are tagged with knowledge components. We think that this method, and ones built off of it, will facilitate better tutoring systems. In [5] we used a variant of this method to figure out what items are causing the most learning. In this paper, we presented a method that allows scientists to see if the items in a randomly ordered problem set produce the same learning regardless of context or if there is an implicit ordering of questions that is best for learning. Those best orderings might have a variety of reasons for existing. Applying this method to investigate those reasons could inform content authors and scientists on best practices in much the same way as randomized controlled experiments do but by utilizing the far more economical means of investigation which is data mining.

Acknowledgements

We would like to thank the Worcester Public Schools and the people associated with creating ASSISTment listed at www.ASSISTment.org including investigators Kenneth Koedinger and Brian Junker at Carnegie Mellon and also Dave Brown and Carolina Ruiz at Worcester Polytechnic Institute for their suggestions. We would also like to acknowledge funding from the U.S. Department of Education’s GAANN and IES grants, the Office of Naval Research, the Spencer Foundation and the National Science Foundation.

References

[1] Barnes, T. (2005). Q-matrix Method: Mining Student Response Data for Knowledge. *Proceedings of the AAAI-05 Workshop on Educational Data Mining*, Pittsburgh, 2005. (AAAI Technical Report #WS-05-02).

[2] Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning factors analysis - A general method for cognitive model evaluation and improvement. *In Proc. the 8th International Conference on Intelligent Tutoring Systems*. pp. 164-175

[3] Corbett, A. T., Anderson, J. R. & O'Brien, A. T. (1995) Student modeling in the ACT programming tutor. In P. Nichols, S. Chipman, & R. Brennan (Eds.), *Cognitively diagnostic assessment* (pp. 19-41). Hillsdale, NJ: Erlbaum.

[4] Desmarais, M. C., Meshkinfam, P. & Gagnon, M. (2006). Learned student models with item to item knowledge structures. *User Modeling and User-adapted Interaction*, 16(5), 403–434.

[5] Pardos, Z. A., Heffernan, N. T. In Press (2009) Detecting the Learning Value of Items In a Randomized Problem Set. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education. Brighton, UK. IOS Press.*

[6] Pardos, Z. A., Heffernan, N. T., Ruiz, C. & Beck, J. (2008) Effective Skill Assessment Using Expectation Maximization in a Multi Network Temporal Bayesian Network. *The Young Researchers Track at the 20th International Conference on Intelligent Tutoring Systems*. Montreal, Canada. pp. 31-40

[7] Pavlik, P. I., Jr., Presson, N., & Koedinger, K. R. (2007). Optimizing knowledge component learning using a dynamic structural model of practice. *In proceedings of the 8th International Conference on Cognitive Modeling*. Ann Arbor, Michigan, USA.

[8] Stevens, R. H., & Thadani, V. (2006) A Bayesian Network Approach for Modeling the Influence of Contextual Variables on Scientific Problem Solving. In M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): *ITS 2006*, LNCS 4053, Springer-Verlag. pp.71-84.

Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models

Philip I. Pavlik Jr.[1], Hao Cen[2], and Kenneth R. Koedinger[1]
{ppavlik, hcen}@andrew.cmu.edu, and koediger@cmu.edu
[1]Human Computer Interaction Institute, Carnegie Mellon University
[2]Machine Learning Department, Carnegie Mellon University

Abstract. This paper describes a novel method to create a quantitative model of an educational content domain of related practice item-types using learning curves. By using a pairwise test to search for the relationships between learning curves for these item-types, we show how the test results in a set of pairwise transfer relationships that can be expressed in a Q-matrix domain model. Creating these Q-matrices for various test criteria we show that the new domain model results in consistently better learning curve fits as shown by cross-validation. Further, the Q-matrices produced can be used by educators or curriculum designers to gain a richer, more integrated perspective on concepts in the domain. The model may also have implications for tracing student knowledge more effectively to sequence practice in tutoring/training software.

1 Introduction

Because of the complexities involved in curriculum design, and because of the possibility of expert blind spots in the design of curricula [1], an alternative to human sequenced curricula might be desirable. One way to achieve this goal is to create a model that explicitly captures the pairwise knowledge component (KC, which may refer to skills, procedures, concepts or facts) relationships between item-types in the domain, what might be called a “transfer model”. While a model that captures transfer may have many forms, here the transfer model we will investigate maps to a Q-matrix, which is a matrix in which rows represent item-types and columns represent KCs. In such a matrix, a 1 value indicates the item-type uses the KC, while a 0 indicates the KC is not involved in the item-type performance. Such a model is desirable because it allows us to make determinations about the optimal order of problems (sequence of repetition and presentation) since it allows prediction of which item-type will cause learning of KCs that transfer to the other item-types most efficiently. (Throughout this paper we use the term item-type to signify a collection of practice items that are either identical or only slightly different, e.g. instantiated with different numbers of the same relative magnitude but involving the same numerical operations or concepts.)

Many others have worked on such models of domains using testing data results [e.g. 2, 3]; however, this paper attempts to broaden the problem by simultaneously addressing the issue of what learning data results imply for the domain model found. It seems that a solution to this problem would have to propose some sort of “transfer function” that captures how the learning and performance of item-type A causes effects on the learning and performance of item-type B. Indeed, learning transfer function models already exist (logistic regression models that predict item-type B as a function of prior practice of either A and/or B) and have been used with human generated KC sets for the purpose of refining the model of individual KCs [4, 5]. A similar issue regarding domain structure and learning curves has been addressed by others who have looked at how to use human derived domain structures to combine learning curves to modify adaptive instruction so that feedback is based on combined KCs rather than the individual KCs [6, 7].

By combining these two approaches (machine derived domain modeling with learning curve analysis of transfer) we hope to produce a technique that both avoids the need for possibly error-prone, time-consuming human labeling of the domain and avoids the problems inherent in proposing an automatic domain model using observations that are non-static (i.e. they change with time as a function of learning). For this paper we will constrain our focus to explaining our pairwise-method of transfer testing, and how this can be used to automatically determine an overall linked (transfer) model of a domain using learning curves. We will then compare this linked model (with various criteria for linkage) with an unlinked model. Cross-validation will be used to establish the superiority of the linked model at various linkage criteria.

1.1 POKS and LiFT (Learning Factors Transfer)

This work is conceptually similar to work with knowledge spaces using the partial order knowledge structure (POKS) method [8]. In both cases we construct a partial order directed acyclic graph using pairwise tests to determine linkages and their directionality. In POKS the relationship between 2 item-types (A and B) is written A→B, and allows inferences of the type "if A is known, then B must be known" and "if B is unknown, then A is unknown". In contrast, in the Learning Factors Transfer (LiFT) test a directional relationship is expressed in set notation so an analogous link is written $A \supset B$ and expresses the inference that the KCs required for A are a proper superset of the KCs required for B. For instance, in the dataset we examine, item-type A might be “What is 1/1 in percent? (Answer: 100%)”, which requires an understanding of whole number fractions and the percent conversion procedure, whereas item-type B is “What is 1/1 in decimal? (Answer: 1)” and only requires the understanding of whole number fractions. Because this example represents the superset relationship, it encodes a situation where learning of item-type B transfers fully to A, but where learning of item-type A transfers only partially to B. This example describes a model with 2 KCs for A and 1 KC for B. Thus, practice of B benefits A partially, while practice of A benefits both A and B.

The POKS test is not fully applicable to repeated practice opportunities for a specific item-type because it requires single observations for each item from each subject to compute in standard form (the test assumes that observations are independent). In previous work we showed how it was possible to use the average performance rather than a single observation to compute implication relationships using POKS [9]. While this POKS analysis provided interesting information about the domain, like many previous works describing domain structure, the result necessarily abstracted over learning effects which the LiFT test explicitly analyzes. This inclusion of the effect of learning is a key advantage of our new method, since the LiFT test (assuming it shows transfer between 2 item-types) should therefore allow us to better answer questions such as which item-type should be practiced first and how much it should be practiced before it is optimal to switch to the other. For example, in some cases the quantitative model (Section 2.1) will predict, given $A \supset B$, that B should be practiced because initial performance of A will be poor without some practice of B first. In other cases, A might be easy enough or the prior learning of B might be strong enough that A should be practiced first. Additionally, it seems plausible to suggest that if our test function does not include a learning component then a domain search using it will be less accurate when our data contain significant amounts of learning.

2 Learning Factors Transfer (LiFT) Analysis

LiFT analysis takes the form of a basic pairwise test that establishes the likelihood that any pairwise relationship is better represented as a transfer relationship, or whether it appears the item-types are unrelated. While the results of the pairwise test may be useful for human curriculum designers to consider, the LiFT test can also be employed to mine large datasets with many item-types being learned simultaneously. We will describe how this algorithmic usage can be performed, and compare the model fit for the transfer relationships discovered at various criterion settings with 3 alternative models: a single item/KC model, a independent item/KC model, and a random transfer models with the number of links in the Q' matrix yoked to the LiFT found transfer models, but placed at random. Because transfer is a within-subject effect, occasionally we will see it occur due to general correlation among item-types caused by latent variables such as motivation, general intelligence, or generally better prior learning. This possibility may be minimized by setting a strict criterion for our transfer test.

2.1 PFA Item-type model

The model equation we will use here is similar to a model equation that has been recently described and shown to fit better than either the standard logistic regression equation used in LFA (Learning Factor Analysis) or the standard version of Bayesian knowledge tracing [10]. For this paper we have made a slight modification which specifies that prior learning parameters are assigned at the item-type level rather than the KC level. The model equation, which can be referred to as the Item-type PFA equation, is a logistic regression model of performance on each trial that includes a parameter to capture the initial strength for each item-type and 2 parameters that track the learning and performance with each KC. The PFA Item-type equation is shown in Equation 1, where *m* is a logit value representing the accumulated learning for a student *i* on one item-type *k* using one or more KCs *j*. The easiness of the item-types is captured by the β parameter for each item-type. The effect of learning and performance is captured by *s*, which tracks the prior successes for the KC for the student and *f*, which tracks the prior failures for the KC for the student. The 2 parameters γ and ρ scale the effect of these observation counts for each KC as a function of the *s* or *f* of prior observations for student *i* with KC *j*. Equation 2 is the logistic function used to convert *m* strength values to predictions of observed probability. It is useful to note that the model always assumes that each item-type has a "base KC" that matches to each item-type and the LiFT test tries to go beyond that base KC to propose other KCs that might be transferred to the item-type to better model performance.

$$m(i, j \in KCs, k \in Items, s, f) = \beta_k + \sum_{j \in KCs} (\gamma_j s_{i,j} + \rho_j f_{i,j}) \qquad (1)$$

$$p(m) = \frac{1}{1 + e^{-m}} \qquad (2)$$

The assignment of KC's to item-types is described by a Q-matrix which describes which KC's influence which item-types. To represent an independent component for each item-type we have specified that each KC is matched to a specific item-type. Therefore, our Q-matrix will be a square matrix with item-types as rows and matched KCs as columns. Since every item-type has its matching KC, the diagonal will be all 1s, indicating that each KC is present in its corresponding item-type. To distinguish this kind of Q-matrix (square where every item-type is assigned at least 1 KC) from the larger set of standard Q-matrices, henceforth we will refer to it as the Q'-matrix.

2.2 LiFT test

The LiFT test takes as input the sequence of practice data for 2 item-types in a tutor (their learning curves) and computes the relative likelihood that they have an overlapping KC by comparing the weights of the likelihood difference of alternative models. In the version we are presenting here, we considered the 2 alternative models, $A \supset B$ and A~B (where A and B do not share a KC) for each order pair of item-types (thus pair (X, Y) is distinguished from (Y, X) and the test is run on both pairs). While we consider these 2 models, there are other transfer assumptions that might be tested but these are beyond the scope of this paper. The $A \supset B$ model asserts that the A item-type is controlled by the 2 KCs, while the B item-type is only controlled by 1 KC (the 2x2 Q'-matrix is filled with 1's on the diagonal and the upper right corner is also filled with a 1 to indicate item-type A shares the same KC as item-type B). In contrast, the A~B model supposes that each item-type is independent (the Q'-matrix is only filled with 1's on the diagonal since each item-type has a single KC).

When this test is computed it determines whether we can get an improvement in model fit by proposing that learning for one item-type transfers to the performance of the other item-type. At the same time as it determines whether the item-types share a KC, the directionality of the test determines which of the two item-types contains an additional independent KC, thus indicating that it contains a superset of the skills required relative to the other item-type. To compute the test, we fit these 2 models (each with 6 parameters) and compared them according to their BIC (Bayesian Information Criterion) weights to determine the evidence ratio in favor of $A \supset B$. (Because model complexity was equal, this was equivalent to using AIC weights or likelihood ratio.) This evidence ratio gives us the likelihood of $A \supset B$ expressed as a probability. This use of BIC weights to compute evidence ratios has been described in detail previously [11]. Because the BIC weight test requires observations to be independent, we minimized BIC using the average loglikelihood for each subject rather than for each observation. This procedure is conservative since it overcompensates for the only partial dependence between observations within a single subject.

2.3 LiFT algorithm

Many possibilities exist for how this test could be applied to a dataset to determine the relationships between item-types. For this first attempt we did an exhaustive search for pairwise relationships, accepting those BIC weight test results that resulted in improvements above a probability criterion. Acceptance of the result of any pairwise test meant that the superset transfer implication was added to the Q'-matrix. For example, imagine that the criterion is .43 and we test $A \supset B$ and get a probability value of .32. In this case we have not passed criterion and we do not alter the Q'-matrix row for item-type A. However, if the $A \supset B$ test arrived at a result of .87 (thus passing the criterion), we would enter a 1 in the Q'-matrix at item-type row A and KC column B.

This LiFT test is applied to a multi-item-type dataset according to the following steps.

1. Create diagonal matrix with 1's on the diagonal assuming rows and columns equal the number of item-types.

2. Compute the pairwise test for all non-diagonal entries, entering a 1 in the matrix for any tests that pass.

3. Use the Q'-matrix from step 2 and maximize the likelihood of the entire dataset.

Because we wanted to get a perspective on what was an effective criterion, we tested criteria from 0 to 1 in .01 increments. We used 10 fold cross validation of the mean absolute deviation averaged by subject to compare the full models we tested. In applying this algorithm we used the following dataset.

3 Data

The dataset we used was gathered from a middle school in Florida which uses the Bridge to Algebra Cognitive Tutor by Carnegie Learning Inc. As part of a larger investigation we had supplemented the tutor with 9 problems sets each with 34 item-types. These supplemental units were closely matched to the tutor units that followed them, and future analysis will look at the potential for transfer into the Bridge to Algebra tutor from the supplemental units. For this investigation we choose 1 of these supplemental units (Unit 5, Fraction, Decimal and Percent Conversions) to investigate how our algorithm would work to improve the predictions of the model by filling in the Q'-matrix. The item-types were ideally sequenced for an analysis of this sort since they were randomized into a 4 by 2 within subjects design where there were 4 levels of practice (0, 1, 2 or 4 repetitions) and 2 levels of spacing (3 or 15 intervening trials). These conditions (with a few buffer trials) resulted in total of 64 practices for each supplemental lesson. 361 students produced valid data. The data for each subject took the form of sequential lists of which item-type was practiced and whether the result for that practice was correct or incorrect.

4 Results

Figure 1 shows the size of the resulting Q'-matrices found by the LiFT algorithm run on Florida dataset. At a criterion of 0, every BIC weight test passes and the matrix is saturated with 1 values. In this case the algorithm fits a model with 34 item-type parameters (β) and 34 γ and ρ parameters which are identical for every KC because they are shared across every item-type (a square Q' matrix filled with 1s). As the criterion is made more stringent, Figure 1 shows how fewer and fewer links are proposed until at a criterion of 1, none of the BIC weight tests pass, and the algorithm proposes 34 item-types, each with an associated KC, each represented by its own β, γ, and ρ (a diagonal Q' matrix of 1s).

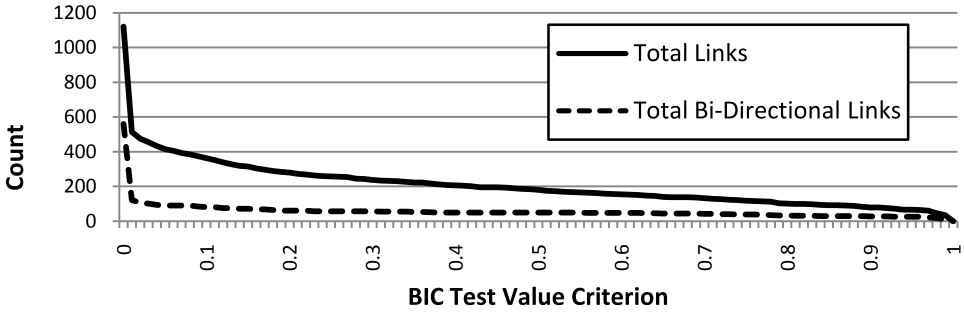


Figure 1. Number of total links and number of bidirectional links A ⊃ B found with pairwise BIC weight test.

Figure 2 shows 10 fold cross validated estimates of mean absolute probability deviation (averaged by subject) for the model fit at each criterion. For comparison, the 3 alternative models are also presented on this figure. The 1 KC/item model comparison is a 3 parameter model which assumes that all of the 64 practices for each student are actually best modeled as a single item-type with one KC. The no transfer model comparison is a 102 parameter model with 34 KCs/items and is represented by a diagonal Q'-matrix which assigns 1 KC to each item-type (equivalent to criterion = 1). The yoked random control comparison is a model calculated with a random square Q'-matrix yoked to the number of links in the LiFT found Q'-matrix at that criterion, but with those links placed randomly. Note: Just as with the found Q'-matrices, we began with the assumption that each item-type was associated with at least a single KC. The yoked random control is an important comparison since it helps to establish that selection using the LiFT test is causing the advantage seen, rather than it being caused merely by the presence of links in the Q'-matrix.

The result shown in Figure 2 establishes that the transfer model (LiFT found Q') fits the data better than the 1 KC/item, no transfer, or yoked random Q'-matrix models. Further, the cross validated comparison establishes that the result is likely to generalize to similar populations. Of some interest for further research is why the improvement in fit is relatively large even when using an extremely liberal BIC weight test value criterion. It seems likely that averaging of multiple low criterion transfer relations (multiple 1s in a Q'-matrix row) causes a reduction of the error compared to the pairwise test models.

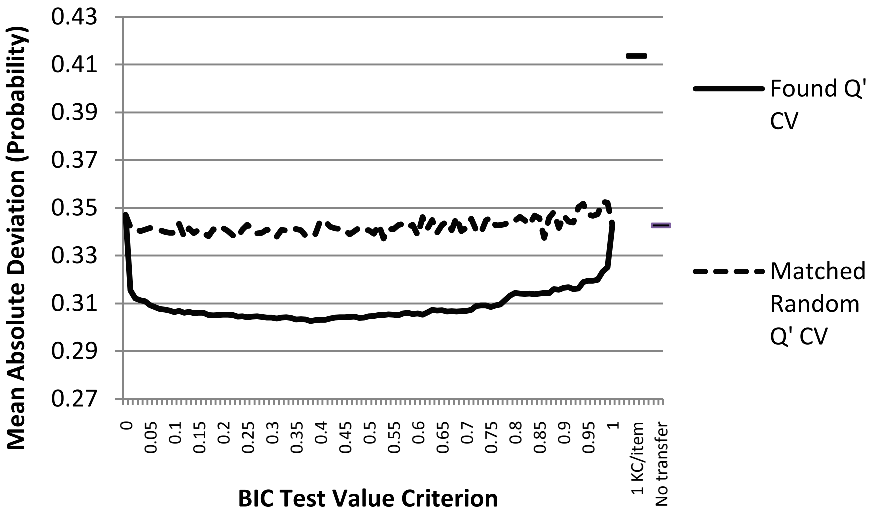


Figure 2. 10-fold cross-validated fit of the model and yoked random control across the range of BIC weight test criterion values.

4.1 Interpretation of Results

In explaining the LiFT test, we provided an example where the two item-types showed an hypothetical subset relationship with one item-type requiring 1 KC and the other requiring that KC plus an additional KC. However, we did not find any clear superset relationships when we looked at the results. In contrast to this theory, our results were more varied but showed several specific patterns. We examined these patterns at a few test value criteria finding quantitative but not qualitative differences. The following description is given for a criterion of 0.6 on the BIC weight test.

Group 1 (4 item-types) was the smallest item-type group where item-types were left completely unlinked. Because these item-types were unlinked, the found Q'-matrix models of these item-types (determined by β, γ, and ρ and Equation 1) is identical to the no transfer model of these item-types. This set includes questions that are relatively difficult for most students (a low β parameter). However, these item-types also tend to have high learning and performance parameters (a high γ and ρ). Based on these results we might suppose that the problems are badly worded, tricky, or beyond the level of the average student. For instance, the item-type (instantiated with different numerals in 6 versions that were randomly selected from with replacement) "If we are given that 4% of y is 1, one fraction of our proportion is 4/100 and the other is what? (Answer: 1/y)" was found to be in this category. Compared to other item-types it is wordy and highly complex involving fraction KCs, percent KCs, and proportion solving KCs.

Group 2 (15 item-types) was similar to Group 1 in that these item-types did not share their KC component with any other item-types. Unlike Group 1 however, these item-types had between 2 and 10 input KC's that other item-types shared with them. As we will see, Group 3 provides these inputs KCs. Group 2 item-types are distinguished by being primarily repetition based vocabulary practice item-types. Further, we see less fundamental conversions (“What is 1/1000 in percent? (Answer: 0.1%)” and “What is 10/1 in percent? (Answer: 1000%)”) are included in this group, perhaps because their performance and ability to be learned depends on understanding more fundamental frequency conversions (e.g. 1/10 in decimal), while the converse is less true.

Group 3 (15 item-types) was composed of item-types that were strongly connected with other item-types both by sharing their matched KC with between 2 and 22 item-types and by receiving KC input from between 3 and 11 item-types. Group 3 is composed of item-types that start out at moderately higher βs (logit greater than 0, i.e. >50% initial performance) and have much lower γ and ρ parameters for their matched KCs. These item-types appear to describe 3 conceptual categories that can be distinguished by the dominant sharing of inputs and outputs within each category. Essentially what has happened in each case is that inputs and outputs form loose conceptual categories by sharing that is primarily within category. Category 3a includes two item-types “What is a ratio of the amount of decrease to the original value, written as a percent? (Answer: percent decrease)” and the corresponding question about an increase. This category seems to depend on the general form of the question (identical) which allows direct transfer of the solution pattern. Category 3b is similar, but has to do with 3 questions that required the partial solution of proportions (e.g. “Given the proportion 1/y = 3/4, what is the product of the extremes? (Answer: 4)”) One of these questions also interacted heavily with category 3c. Category 3c can most closely be aligned with a fundamental proficiency in the domain, or with general intelligence, since these item-types all shared their KC with at least 13 item-types and received inputs from at least 8 item-types. While each of these item-types had its own β parameter, this extensive sharing means that learning and performance change for these item-types occurs nearly as a unit. 8 of these item-types involved simple place value problems such as “What digit is in the 10s place in the number 25046.37189? (Answer: 4)“ and “What is 1/10 in decimal? (Answer: 0.1)” Also in this category was a simple pre-algebra problem: “What is y in the equation 3 × y = 9? (Answer: 3) (with 6 versions). Finally, the item-type “If 54 can be completed in 10 hours, what is the amount completed in 1 hour (as a decimal)? (Answer: 5.4)” (with 6 versions) appeared to be part of both 3b and 3c, which seems consistent, since it involves aspects of both proportions and place value (the set of item-types always required a power of ten calculation).

The three groups found show that understanding the model has interesting implications for the curriculum designer. First, it appears that item-types in Group 1 need to be improved in some way. While we cannot tell for sure what the problem is, the lack of connections to the other groups establishes that these item-types are either in a different domain, too complex to relate to the more simple item-types in the set, or badly worded so that students cannot transfer in related knowledge. Group 2, items on the other hand receive KC from Group 3 item-types, so we can surmise that they are at least marginally related to the domain. The fact that these questions did not share their KCs might have more to do with the limitations of the question set overall (which did not require much application of these mostly vocabulary-based item-types in other questions) or the model (see Discussion) rather than any specific lack in the items themselves. Finally, Group 3 items are useful to consider because the model here suggests that the 3 conceptual categories (3a, 3b and 3c) are each better modeled as single units rather than as independent skills. Knowing these item-type clusters are closely related allows curriculum designers to have more information as they make curriculum design decisions.

5 Discussion

In general, the results were a qualified success because the model found produced a better fit that generalized and because the model structure provided other reflections on the content subdivisions in the domain of item-types studied. Despite this, there were problems with the current logistic regression model. Specifically, because of the way the equation uses the Q'-matrix to share KCs as whole units, the parameter magnitudes changed as a function of whether or not each item-type shared KCs with other item-types. Item-types that shared their KC's (especially when they provided their KC to many other item-types) had lower performance parameter values (γ and ρ) than when they were fit in the no transfer model. The reason for this is simply that when the probability is determined from multiple inputs each input must be scaled down so the total growth still resembles the situation with one KC. However, one unfortunate consequence of this is that the model is then insensitive to repetition effects for a single item-type and therefore predicts much slower growth when the same item-type is repeated. This is because the item-type is no longer being controlled by a single KC, but rather has become tied to a collection of KCs that control it. One solution to this problem in future work may be to take a knowledge decomposition approach that does not insist on this KC sharing through the Q'-matrix [12]. Such an approach might instead propose that the magnitude of transfer for each KC is different depending on what item-type the KC transfers to. While this would add parameters to our model, it might also greatly improve the fit of the model.

This work may apply directly to the educational problem of sequencing item-types to maximize learning because the resulting model captures learning, is adaptive to performance, and captures the domain structure together in a single model. Unlike other automatically determined domain models, which determine performance dependencies and might be used for ordering practice, our model explicitly tracks learning also. By adding learning to our domain model, our model has the potential to answer not just the question of what item-type is best next, but also the question of how much more should the current item-type be practiced.

Acknowledgements

This research was supported by the U.S. Department of Education (IES-NCSER) #R305B070487 and was also made possible with the assistance and funding of Carnegie Learning Inc., the Pittsburgh Science of Learning Center and DataShop team (NSF-SBE) #0354420, and Ronald Zdrojkowski.

References

[1] Koedinger, K., Nathan, M.J.: The real story behind story problems: Effects of representation on quantitative reasoning. *Journal of the Learning Sciences,* 2004, 13(2), p. 129-164.

[2] Barnes, T.: The Q-matrix Method: Mining Student Response Data for Knowledge. *American Association for Artificial Intelligence 2005 Educational Data Mining Workshop*, 2005.

[3] Spacco, J., Winters, T., Payne, T.: Inferring use cases from unit testing. *AAAI Workshop on Educational Data Mining*, 2006. ACM Press.

[4] Leszczenski, J.M., Beck, J.E.: What's in a Word? Extending Learning Factors Analysis to Model Reading Transfer. *13th International Conference on Artificial Intelligence in Education, Educational Data Mining Workshop*, 2007. Los Angeles, CA.

[5] Cen, H., Koedinger, K.R., Junker, B.: Learning Factors Analysis - A general method for cognitive model evaluation and improvement. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 2006. Springer Berlin / Heidelberg, p. 164-175.

[6] Martin, B., Mitrovic, A.: The effect of adapting feedback generality in ITS. In: Wade, V., Ashman, H., Smyth, B. (Eds.) *AH 2006*, 2006. p. 192-202.

[7] Martin, B., Mitrovic, A.: Using learning curves to mine student models. *10th International Conference on User Modelling*, 2005. Edinburgh, p. 79-88.

[8] Desmarais, M.C., Maluf, A., Liu, J.: User-expertise modeling with empirically derived probabilistic implication networks. *User Modeling and User-Adapted Interaction,* 1996, 5(3-4), p. 283-315.

[9] Pavlik Jr., P.I., Cen, H., Wu, L., Koedinger, K.R.: Using Item-type Performance Covariance to Improve the Skill Model of an Existing Tutor. In: Baker, R.S., Beck, J.E. (Eds.) *Proceedings of the 1st International Conference on Educational Data Mining*, 2008. Montreal, Canada, p. 77-86.

[10] Pavlik Jr., P.I., Cen, H., Koedinger, K.R.: Performance Factors Analysis -- A New Alternative to Knowledge Tracing. In: Dimitrova, V., Mizoguchi, R. (Eds.) *The 14th International Conference on Artificial Intelligence in Education*, 2009, accepted. Brighton, England.

[11] Wagenmakers, E.-J., Farrell, S.: AIC model selection using Akaike weights. *Psychonomic Bulletin & Review,* 2004, 11(1), p. 192-196.

[12] Zhang, X., Mostow, J., Beck, J.E.: All in the (word) family: Using learning decomposition to estimate transfer between skills in a Reading Tutor that listens. *AIED2007 Workshop on Educational Data Mining*, 2007. Marina del Rey, CA.

Detecting and Understanding the Impact of Cognitive and Interpersonal Conflict in Computer Supported Collaborative Learning Environments

David Nadler Prata[1], Ryan S.J.d. Baker[2], Evandro d.B. Costa[3], Carolyn P. Rosé[2], Yue Cui[2],
Adriana M.J.B. de Carvalho[2]
ddnprata@gmail.com,
[1]Campus Arapiraca - Pólo Penedo, Universidade Federal de Alagoas
[2]Pittsburgh Science of Learning Center, Carnegie Mellon University
[3]Departamento de Computação, Universidade Federal de Alagoas

Abstract. This paper presents a model which can automatically detect a variety of student speech acts as students collaborate within a computer supported collaborative learning environment. In addition, an analysis is presented which gives substantial insight as to how students' learning is associated with students' speech acts, knowledge that will significantly influence how this model is utilized by running learning software. Within Piagetian theory, the cognitive conflict of ideas between students is seen as beneficial for learning. Which sorts of interpersonal behaviors lead to most effective learning, however, is open to debate, with some researchers arguing that cooperation is most effective and others arguing that interpersonal conflict is a natural part of collaborative learning. We find that, in fact, interpersonal conflict is associated with positive learning, a finding that must be taken into account, in designing interventions that rely upon detectors of students' speech acts in CSCL environments.

1 Introduction

In recent years, there has been a substantial increase in the use of Computer Supported Collaborative Learning (CSCL) environments to promote learning of key educational concepts and skills, in a variety of domains. During collaboration, students engage in a wide variety of collaborative and learning behaviors, which impact each students' learning in a variety of ways [5, 14, 19]. Bringing collaboration online creates the possibility of automatically detecting differences in students' collaborative behavior and responding automatically to differences in students' behaviors [9].

However, adapting appropriately depends on achieving three key sub-goals. First, we need to know which collaborative behaviors merit intervention. Second, we have to know how to intervene appropriately in those cases. Third, we have to be able to accurately detect those behaviors so that we can respond to them. In this paper, we present work that makes a contribution to the first and third of these goals. We study the learning associated with a set of four theoretically interesting collaborative behaviors, within an ecologically valid CSCL environment for fractions. We then present a machine-learned model, developed within TagHelper [20], which can accurately distinguish a set of collaborative behaviors, and performs especially effectively within categories previously determined to be of interest. We conclude with a discussion of how the study findings impact how the detectors we have developed should be used, to adapt to students' behavior during CSCL.

1.1 Cognitive and Social Conflict in CSCL

Within this paper, we focus on cognitive and social conflict within computer supported collaborative learning, categories found to have relevant impacts on student learning within non-computer-mediated collaborative learning.

Studies from the 1970's have shown that cognitive conflict promotes cognitive development [11,13,21,23,25], in line with Piaget's [16] writings on the equilibration process. Piaget claimed that one source of progress in the development of knowledge is found in the imbalance that forces a subject to seek new equilibrium through assimilation and accommodation.

Many researchers have found results indicating that cognitive conflict and learning emerges from the process of collaboration, when students mutually engage to co-construct shared knowledge [5,14,19]. In fact, Moshman and Geil [12] and Kruger [10] have argued that the conceptualization of cognitive change as either a process of conflict or a process of cooperation is a false dichotomy, claiming that productive cognitive conflicts take place solely within a cooperative context, and not via competition or interpersonal conflict. In Moshman and Geil's view, productive cognitive conflict does not emerge from students arguing in favor of their own views, but from co-constructing a consensus solution. Similarly, Howe [7] suggested a separation between types of conflict that involves transactive [3] dialogues (e.g. cognitive conflict) and interpersonal conflict that involves aggression. These studies argued that these two types of interaction occur in distinct groups, depending on students' gender and temperament.

However, other studies have provided evidence suggesting that cognitive conflict does not solely occur in purely collaborative and consensus-based process. For example, Arsenio and Lover [2] and Shantz [22], give evidence that the conflict of ideas often leads to interpersonal conflict.

Hence, it appears to still be an open question whether productive cognitive conflict, and the learning that emerges from it, only occurs when students show collaborative behaviors, or whether it still occurs in conjunction with interpersonal conflict.

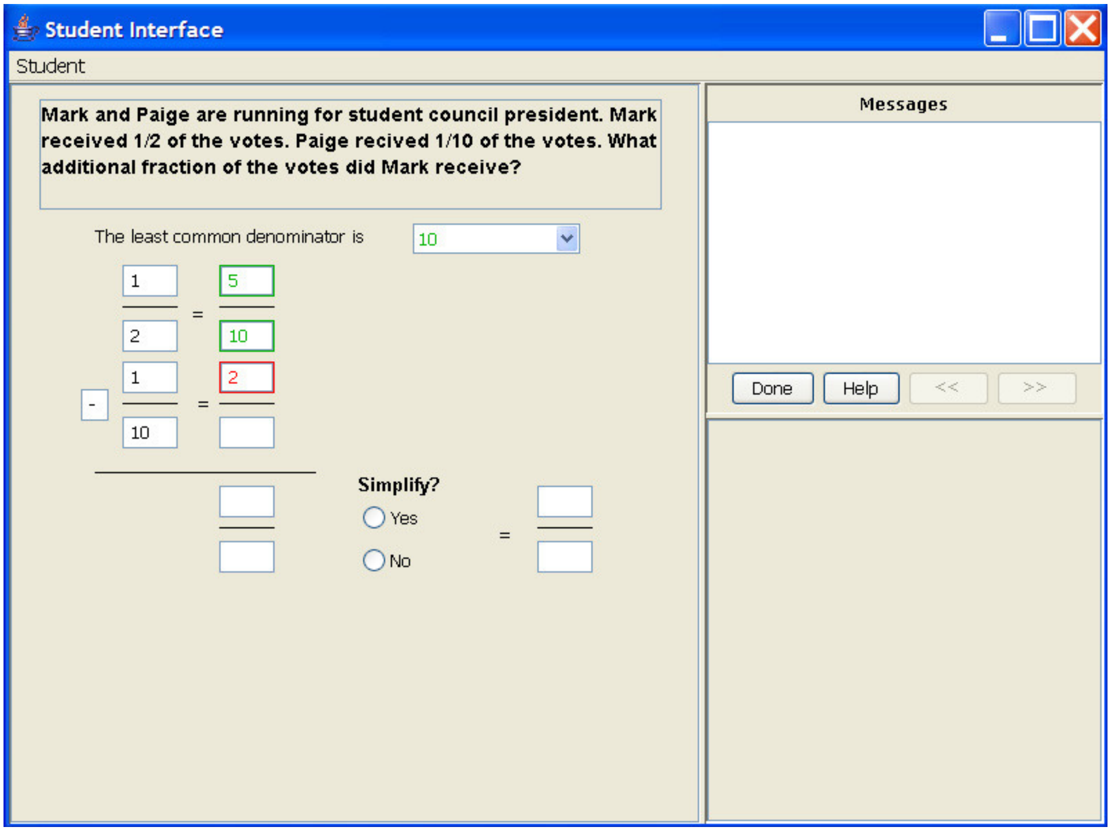
This question is especially relevant within the context of online collaborative learning. Online learning has different affordances than the face-to-face learning settings where much prior collaborative learning research has taken place – in particular, online collaboration, due to the anonymity potentially afforded, is prone to a very high rate of insults, often called "flaming" in the online collaboration literature (e.g. O'Sullivan & Flanagin [15]). To investigate these questions, we consider the relationship between learning and the different behaviors students display in anonymous online collaborative learning, focusing on the insults indicative of interpersonal conflict. Our hypothesis is that the cognitive conflict which occurs in online learning is highly likely to produce interpersonal conflicts, demonstrated by insults; however the online medium may also reduces the social consequences associated with insults, meaning that insults may not impede learning gains.

2 Analysis of Learning Associated With Speech Acts

2.1 Methods

Twenty four sixth-grade students from a suburban elementary school near Pittsburgh, PA, participated in this study. The study was conducted in a genuine setting of learning, involving authentic learning materials. Because one of the students did not use the chat interface during the two lab days, that student was excluded from analysis (that student's partner was still included in the sample because this student used the chat interface to discuss the problem solving and complain about his partner's lack of interactivity), and the sample was reduced to 23 students. Each student made, on average, 84.3 utterances, for a total of 1940 utterances.

Each student used a mathematics tutoring program covering problems on fraction addition, subtraction, multiplication, and division [9], in collaborative pairs mediated through TuTalk. TuTalk [8] is a collaborative problem solving interface that include two online panels: a chat, and a collaborative interface for the problem solving built in the CTAT authoring tool [1].



Mark and Paige are running for student council president. Mark received 1/2 of the votes. Paige recived 1/10 of the votes. What additional fraction of the votes did Mark receive?

The least common denominator is 10

$\frac{1}{2} = \frac{5}{10}$

$- \frac{1}{10} = \frac{2}{\ }$

Simplify?

○ Yes

○ No

Figure 1. Problem-solving interface [9].

The students worked in their school lab computer, in pairs, using TuTalk, with their chat dialogues and problem solving contributions (within the interface) logged for later analysis. The arrangement of the lab was designed so that the students could not easily talk with their pairs outside of the chat interface, with the identity and the seat location of the collaborating pairs hidden from their partners.

Each student individually took nearly isomorphic pre-tests and post-tests, covering knowledge of the material covered in the tutor, during a 30 minute period during on separate days from tutor usage. The students collaborated in learning fractions within TuTalk within two lab sessions, each lasting 45 minutes. This design enabled us to investigate the student's knowledge gains based on the pre- and post-tests, and to analyze students' collaborative and individual learning behaviors.

2.2 Analytical Method

In analyzing these dialogues, we divided student behavior into a selected number of categories relevant to our analyses. Cognitive communicative categories were split in accordance with Youniss and Damon's [26] interpretation of Piaget's views on social relations in the individual construction of knowledge, where cognitive conflict and knowledge construction can occur either through disagreement, where one student perceives a misconception or other error in his partner's thinking and disagrees, attempting to express why it is wrong (called disagree with concept in our coding scheme). Within this type of disagreement, a student is arguing in favor of his or her own views, an ultimately competitive act. In one example, one student said, "i dont think thats the common denominator". By contrast, a student may also refine their partner's ideas by expressing their perspective on an idea, and informing the partner as to their beliefs (called inform belief in our coding scheme), attempting to co-construct a solution – a more collaborative manner of expression. One example of this within our corpus is, "the common denominator is 54".

These two categories are shown in Table 1. The two categories do not form an exhaustive list of possible cognitive communicative acts, but are particularly relevant to the analysis presented here. A fuller taxonomy of speech acts is given in the first author's doctoral dissertation [17].

Table 1. Description of the cognitive communicative categories

Context	Category	Description
More Cooperative	Inform Belief	The speaker informs his/her partner about his beliefs about the concept (*the common denominator is 54)*
Less Cooperative	Disagree with Concept	The speaker disagrees with his partner's belief about the concept. *(i dont think thats the common denominator)*

Table 2 Description of social communicative categories

Offer collaborative act	The speaker offers to do something for his or her partner towards the problem-solving goals	*i do the botttom now* (student 2b)
Insult	The speaker insults his or her partner by calling them an obscene or offensive word.	*you loser* (student 14b)

The offer collaborative act and insult categories are social communicative categories (Table 2). A student who offers collaborative act offers to do something to forward the problem-solving goals, generally without having specifically been asked to do so. As such, offer collaborative act is a reflection of the peers' social collaboration. By contrast, an insult reflects interpersonal conflict within the dialogue, and is in contrast to social collaborative behavior.

This set of four categories was coded by the first and sixth authors. Both authors coded a subset of 225 utterances made by students during collaborative learning. Cohen's [4] Kappa was 0.80, indicating good inter-rater reliability. Afterwards, the protocol analyses were based on the first author's codes for the entire corpus. We also developed a machine-learned model that was able to accurately code these categories, discussed later in the paper – however, for this analysis human coding was used, as a tractable gold-standard.

2.3 Results

We analyzed the correlations between pre- and post-test learning gains and the frequency of each category of our coding scheme in each pair's dialogue. The overall pattern of results is shown in Table 3. As can be seen, the number of inform belief speech acts a student made or received was not significantly correlated with learning gains, respectively t(22)=-0.31, p=0.75, t(22)=-0.10, p=0.92 (all tests reported are two-tailed). The number of offer collaborative act speech acts a student made or received also was not significantly correlated with learning gains, t(22)=0.00, p=0.99, t(22)=0.64, p=0.52, for a two-tailed t-test.) Hence, neither of the two cooperative behaviors coded were associated with significantly higher learning gains.

By contrast, the two non-cooperative behaviors were associated with positive learning – but only in the student being non-cooperative. The amount a student disagreed with concept was associated with statistically significantly higher learning gains for the disagreeing student, r=0.53, t(22)=2.93, p<0.01, but not for their partner, t(22)=-0.38, p=0.70. The disagree with concept act has the intention to alter the peer's reasoning with conflicting ideas, but appears to have been more of a marker of the disagreeing student's learning than a learning opportunity for their partner.

Interestingly, the amount a student made insults was also associated with significantly higher learning gains, r=0.70, t(22)=4.53, p<0.001, but receiving insults from another student was not associated with higher learning gains, r=0.26, t(22)=1.26, p=0.21.

Hence, students who acted in ways that create or indicate interpersonal conflict appeared to have higher learning gains in this study. Students who behaved in a more cooperative fashion did not appear to have higher gains. However, the mechanism explaining this result is not clear. Did students learn more because they allowed cognitive conflict to move into interpersonal conflict, or did students engage in interpersonal conflict because they had learned more than their partner, and were impatient with them? It is possible, in particular, that the anonymity of the online learning system facilitated students who had just learned the material in choosing to insult their partner rather than help them.

Table 3. The relationship between learning gains and different dialogue acts (p values shown). Statistically significant results (p<.05) in boldface.

Context	Cognitive			Social		
	S	**H**	**Category**	**S**	**H**	**Category**
More Cooperative	0.8	0.9	inform belief	0.9	0.5	Offer collaborative help
Less Cooperative	**0.008**	0.2	disagree with concept	**0.0001**	0.2	insult

S – Speaker, H - Hearer

3 Development of Collaboration Behavior Detectors

Having coded a significant number of utterances, the next step was to determine whether it would be possible to develop a machine learned model that could automatically detect these four categories. Such a detector could be used to drive automated interventions by the CSCL environment. (Possible interventions will be discussed in the discussion section).

These four categories were combined with additional data coded with twenty-eight other categories, representing a wide span of possible dialogue acts within collaborative learning. The full coding scheme is discussed in detail in the first author's doctoral dissertation [17]. In total, there were 170 utterances coded with the 4 speech acts discussed above, and a total of 1940 utterances coded with the full set of 32 speech acts.

Rosé et al's [20] TagHelper tool kit was used to develop a machine learned model that could identify the set of speech acts in students' utterances. TagHelper provides text classification services, designed for use with several languages, and access to a variety of metrics for validating model goodness. It also automatically distills features previously found to be useful for linguistic analyses, such as bigrams and the presence of “stop” words. We used TagHelper to develop models, and to quantify our success in terms of agreement with the hand-coded gold standard corpus. The Naïve Bayes classification algorithm was selected, and applied to the 32 speech acts on the dialogue data. The Kappa [4] statistic was used, in combination with 10-fold cross validation, to assess reliability of the model’s coding. Non-stratified cross-validation was used, under the assumption that multiple utterances by a single student on a single topic are unlikely to be highly correlated to each other (as opposed to other types of behavior, where a student’s responses may be more characteristic), especially when all students are discussing the same mathematical topics. For instance, terms used to discuss fractions, or to disagree about fractions, are likely to be similar between students.

The model was successful at classifying student utterances. Within the whole set of 32 speech act categories, kappa was a respectable 0.65. Within the set of four utterances that were previously thought to be particularly relevant for modeling and understanding learning (two were indeed found to be statistically significantly associated with learning), kappa was an excellent 0.91. This was better than our human judges’ degree of agreement, suggesting that the model was highly successful.

4 Discussion and Conclusions

In the previous section, an automated detector of a variety of speech acts was presented and shown to be reasonably effective at distinguishing between a variety of speech acts, including the four categories discussed in detail in this paper: inform belief, disagree with concept, offer collaborative act, and insult. With Kappa values between 0.65 (all categories) and 0.91 (categories discussed in this paper), it seems quite feasible to use the model for detecting and responding to different types of speech acts.

However, using the model to drive appropriate interventions depends on understanding the implications of each type of speech act, which leads to a need for analyses such as the one presented here. In the analysis in section 2 of this paper, learning gains were correlated with speech acts. Understanding this gives us an important first piece in the puzzle of deciding how a CSCL system should respond to those acts. Developing a full understanding of what prompts different speech acts will help us even further.

Within this study, learning gains were positively related to interpersonal conflict. Arsenio and Lover [2] and Shantz [22] previously found, in face to face collaboration, that the conflict of ideas can lead to aggressive behavior, including the types of interpersonal conflict observed here. In those studies, the aggressive behavior harmed students’ interpersonal relationships. However, the anonymity of communication in our study may have enabled students to insult each other with lower interpersonal cost, eliminating one of the negative factors associated with aggressive behavior in collaborative learning. In broader usage of such a CSCL system, this anonymity could persist within internet usage while not persisting in classroom usage (because over time, students can determine who they are collaborating with in a classroom, eliminating anonymity).

This does not mean, however, that insults are to be encouraged. Insults are clearly perceived as problematic in online communication (e.g. O'Sullivan and Flanagin [15]), and may be associated with the abandonment of usage of online learning environments (cf. Reinig et al. [18]).

In general, this work supports the hypothesis that positive cognitive conflict can coincide with interpersonal conflict. It is not at all clear from our results that the interpersonal conflict had a positive impact on cognitive conflict or learning – for instance, it may have been a side-effect of one student's greater learning, with no positive impact on the other student. Studying this issue in richer depth will require a combination of methods, including time series analysis on a significantly larger corpus of data, and perhaps experimentally manipulating interpersonal conflict via not transmitting students' insults, in order to determine insults' causes and impacts on learning.

One clear implication of our results is that insults and interpersonal conflict play a prominent role in collaborative learning, which cannot be safely ignored. An overly harsh response to student insults may also interfere with the positive learning that insults appear to be associated with. One approach may be to attempt to develop designs which guide students in moderating their comments to others, without disrupting the cognitive conflict which insults appear to be associated with. However, if insulting another student produces pleasure for the insulting student and increases the insulting student's desire to persist in the use of a learning environment [cf. 24], it may be feasible for a CSCL environment to automatically strip out insults from the text the insulted student actually receives. Further research on how software that supports CSCL can optimally handle insults and other interpersonal conflict behaviors, given the ability to detect those acts, appears to be warranted.

5 Acknowledgements

This work was funded in part by NSF grant REC-043779 to "IERI: Learning-Oriented Dialogs in Cognitive Tutors: Toward a Scalable Solution to Performance Orientation".

References

[1] Aleven, V., Sewall, J., McLaren, B. M., & Koedinger, K. R. Rapid authoring of intelligent tutors for real-world and experimental use. In Kinshuk, R. Koper, P. Kommers, P. Kirschner, D. G. Sampson, & W. Didderen (Eds.), *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006)*, 847-851.

[2] Arsenio,W. F., & Lover, A. Emotions, conflicts, and aggression during preschoolers' free play. *British Journal of Developmental Psychology*, 1997, 15, 531–542.

[3] Berkowitz, M., & Gibbs, J. Measuring the developmental features of moral discussion. *Merrill-Palmer Quarterly*, 1983, 29, 399-410.

[4] Cohen, J. A. Coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 1960, 20, 37–46.

[5] Dillenbourg, P., Baker, M., Blaye, A. & O'Malley, C. The evolution of research on collaborative learning. In E. Spada & P. Reiman (Eds) *Learning in Humans and Machine: Towards an interdisciplinary learning science*. (pp. 189-211), 1995. Oxford: Elsevier.

[6] Hamel, R. *Over het denken van de architect (On the thought processes of architects),* 1990. Amsterdam: AHA Books.

[7] Howe, C. & McWilliam, D. Opposition in social interaction between children: why intellectual benefits do not mean social costs. *Social Development*, 2006, 15, 205-231

[8] Jordan, P., Hall, B., Ringenberg, M., Cui, Y., Rosé, C. P. Tools for Authoring a Dialogue Agent that Participates in Learning Studies. *Proceedings of Artificial Intelligence in Education 2007.*

[9] Kumar, R., Rosé, C. P., Wang, Y. C., Joshi, M., Robinson, A. Tutorial Dialogue as Adaptive Collaborative Learning Support. *Proceedings of Artificial Intelligence in Education 2007.*

[10] Kruger, C. Cognitive aspects of re-use in industrial design engineering. In W. Visser (Ed.), *Proceedings of the Workshop of the Thirteenth International Joint Conference on Artificial Intelligence "Reuse of designs: an interdisciplinary cognitive approach",* 1993, Chambéry, France.

[11] Miller, S.A. and C.A. Brownell. Peers, persuasion, and Piaget: dyadic interaction between conservers and nonconservers. *Child Development,* 1975, 46, 992-997 .

[12] Moshman, D. and Geil, M. Collaborative reasoning: evidence for collective rationality. *Thinking and Reasoning*, 1998, 4(3), 231-248.

[13] Murray, F. B., Ames, G., & Botvin, G. The acquisition of conservation through cognitive dissonance. *Journal of Educational Psychology*, 1997, 69, 519-527.

[14] Nastasi, B. K., & Clements, D. H. Social-cognitive behaviors and higher-order thinking in educational computer environments. *Learning and Instruction*, 1992, 2, 215-238.

[15] O'Sullivan and A.J. Flanagin, Reconceptualizing 'Flaming' and Other Problematic Messages, *New Media & Society*, 2003, 5 (1), 69–94.

[16] Piaget, J. The Role of Action in the Development of Thinking. In W.F. Overton & J.M Gallagher (Eds.), *Advances in Research and Theory*, 1997. New York: Plenum Press.

[17] Prata, D.N. *Modelo de Análise de Conflitos em Diálogos em Aprendizagem Colaborativa (Analytical Model of Conflicts in Collaborative Learning Dialogues).*

Unpublished doctoral dissertation, Universidade Federal de Campina Grande (Federal University of Campina Grande), 2008, Campina Grande, Brazil.

[18] Reinig, B.A., Briggs, R. O.,Brandt, S. A., and Nunamaker, J. F., Jr. The electronic classroom on fire: why it happens and how to put out the flames. *Proceedings of the Thirtieth Hawaii International Conference on System Sciences*, 1997. Maui, HI: IEEE Computer Society Press.

[19] Roschelle, J. & Teasley S.D. The construction of shared knowledge in collaborative problem solving. In C.E. O'Malley (Ed), *Computer-Supported Collaborative Learning*, 1995, pp. 69-197. Berlin: Springer-Verlag

[20] Rose, C. P., Wang, Y.C., Cui, Y., Arguello, J., Fischer, F., Weinberger, A., Stegmann, K. Analyzing Collaborative Learning Processes Automatically: Exploiting the Advances of Computational Linguistics in Computer-Supported Collaborative Learning. *International Journal of Computer Supported Collaborative Learning*, 2008, 3 (3), 237-271.

[20] Rosenthal, T. L., & Zimmerman, B. J., Modeling by exemplification and instruction in training conservation. *Developmental Psychology*, 1972, 6, 392-401.

[22] Shantz, D.W. Conflict, aggression, and peer status: An observational study. *Child Development,* 1986, 57, 1322–1332.

[23] Silverman, I.W. and E. Geiringer. Dyadic interaction and conservation induction: a test of Piaget's equilibration model. *Child Development*, 1973, 44, 815-821 .

[24] Suler, J.R., Phillips, W. The bad boys of cyberspace: deviant behavior in multimedia chat communities. *CyberPsychology & Behavior,* 1998, 1, 275–294.

[25] Waghorn, L. & Sullivan, E.V. The exploration of transition rules in conservation of quantity (substance) using film mediated modeling. *Acta Psychologica,* 1970, 32, 65-80.

[26] Youniss, J., & Damon, W. Social construction in Piaget's theory. In H. Berlin & B. Pufal (Eds.), *Piaget's theory: Prospects and possibilities*, 1992, pp. 267-286. Hillsdale, NJ: Erlbaum.

Using Dirichlet priors to improve model parameter plausibility

Dovan Rai, Yue Gong, and Joseph E. Beck
{dovan, ygong, josephbeck}@wpi.edu
Computer Science Department, Worcester Polytechnic Institute

Abstract. Student modeling is a widely used approach to make inference about a student's attributes like knowledge, learning, etc. If we wish to use these models to analyze and better understand student learning there are two problems. First, a model's ability to predict student performance is at best weakly related to the accuracy of any one of its parameters. Second, a commonly used student modeling technique, knowledge tracing, suffers from having multiple sets of parameters providing equally good model fits. Furthermore, common methods for estimating parameters, including conjugate gradient descent and expectation maximization, suffer from finding local maxima that are heavily dependent on their starting values. We propose a technique that estimates Dirichlet priors directly from the data, and show that using those priors produces model parameters that provide a more plausible picture of student knowledge. Although plausibility is difficult to quantify, we employed external measures to show the parameter estimates were indeed improved, even if our model did not predict student behavior any more accurately.

1 Introduction

The goal of student modeling is to take observations of a student's performance and use those to estimate the student's knowledge, goals, preferences, and other latent characteristics. In general, student models are used to adapt instruction and are evaluated by how well they predict the student's behavior. However, with the advent of educational data mining, it is becoming more common to use model parameters to answer scientific questions (e.g. [1]). Unfortunately, just because a model is an accurate predictor of student behavior, that does not mean we are justified in interpreting the model's parameters to make claims about student learning. This paper focuses on examining this issue, investigates techniques for finding more plausible model parameters, and proposes methods for evaluating parameters for plausibility. First, we provide some background into our student modeling framework, knowledge tracing, and the statistical approach we use to bias model fitting, Dirichlet priors.

1.1 Knowledge tracing model

Knowledge tracing [2], shown in Figure 1, is an approach for taking student observations and using those to estimate the student's level of knowledge. There are two parameters *slip* and *guess*, which mediate student knowledge and student performance. These two parameters are called the performance parameters in the model. An assumption of the model is that even if a student knows a skill, there is a chance he might still respond incorrectly to a question that utilizes that skill. This probability is the *slip* parameter.

There are a variety of reasons for an incorrect response, for example, the student could have made a simple typo (e.g. typed ‘12’ instead of ‘21’ for “7 x 3”).

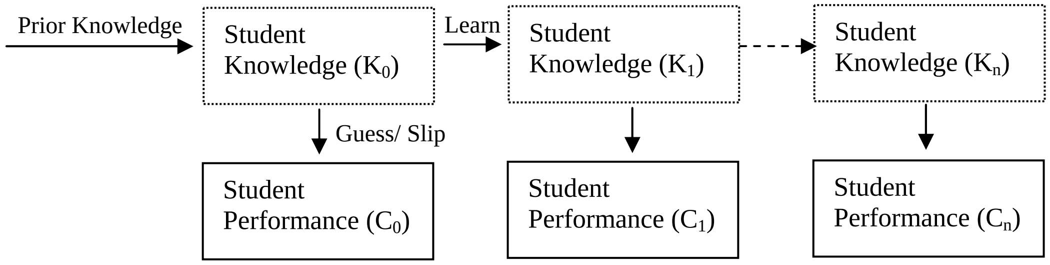

Prior Knowledge
Student Knowledge (K_0)
Learn
Student Knowledge (K_1)
Student Knowledge (K_n)
Guess/ Slip
Student Performance (C_0)
Student Performance (C_1)
Student Performance (C_n)

Figure 1. Knowledge tracing model

Prior Knowledge = Pr (K_0*=True)*

Guess = Pr (C_n*=True |* K_n*=False)*

Slip = Pr (C_n*=False |* K_n *=True)*

Learning rate = Pr (K_n *=True |* K_{n-1}*=False)*

Conversely, a student who does not know the skill might still be able to generate a correct response. This probability is referred to as the *guess* parameter. A guess could occur either through blind chance (e.g. in a 4- choice multiple choice test there is a ¼ chance of getting a question right even if one does not understand it), or the student being able to utilize a weaker version of the correct rule that only applies in certain circumstances.

In addition to the two performance parameters, there are two learning parameters. The first is *prior knowledge* (K_0), the likelihood the student knows the skill when he first uses the tutor. The second learning parameter is *learning*, the probability a student will acquire a skill as a result of an opportunity to practice it. Every skill to be tracked has these four parameters, *slip*, *guess*, K_0, and *learning*, associated with it.

1.2 The problem

One issue is how to estimate the model parameters. One approach is to use the expectation maximization (EM) algorithm to find parameters that maximize the data likelihood (i.e. the probability of observing our student performance data). However, in EM, we have to start with some initial value of the parameter, and final parameter estimations are sensitive to those initial values. Furthermore, one flaw of a knowledge tracing model is that it has multiple global maxima. That is to say, there can be more than one set of learning/performance parameters that fit the data equally well.

Consider the three sets of hypothetical knowledge tracing parameters shown in Table 1, the *knowledge* model reflects a set of model parameters where students rarely guess. The *guess* model assumes that 30% of correct responses are due to randomness. This limit of 30% is the maximum allowed in the knowledge tracing code used by the Cognitive Tutors [2]. The third model has parameters similar to data from Project Listen’s Reading Tutor [3].

By using the four parameters and the knowledge tracing equations, we can compute the theoretic learning and performance curves for each model. Specifically, we initialize P(know) to be K_0. After each practice opportunity, we use formula I to update P(know) as the new likelihood of the student knows the skill after the previous practice. Also we compute P(correct), the probability of the student will respond correctly in the current practice opportunity, by using the knowledge tracing formula to combine the estimated knowledge with the *slip* and *guess* parameters shown in formula II.

$$P(know) = P(know) + (1 - P(know)) * learning \qquad (I)$$

$$P(correct) = P(know) * (1\text{-}slip) + (1 - P(know)) * guess. \qquad (II)$$

For example, the knowledge model's prior knowledge (K_0) is 0.56. At the second practice opportunity the knowledge model would have a P(know) of 0.56 + (1 – 0.56) * 0.1 = 0.604. Furthermore, the likelihood for the student making a correct response would be 0.604 *(1-0.05) + (1-0.604) * 0.00 = 0.574. As seen in Figure 2, the three models have identical student performance (in the left graph), but their estimates of student knowledge (right graph in Figure 2) are very different.

Table 1. Parameters for three hypothetical knowledge tracing models

Parameter	Model		
	Knowledge	*Guess*	*Reading Tutor*
Prior Knowledge	0.56	0.36	0.01
Learning	0.1	0.1	0.1
Guess	0.00	0.3	0.53
Slip	0.05	0.05	0.05

Given the same set of performance data, we have presented three knowledge tracing models that fit the data equally well, i.e. all three sets of estimated parameters have equally good predictive power. Unfortunately, for drawing conclusions about student learning, they make very different claims. Statistically there is no justification for preferring one model over the others, since all three of the sets of parameters fit the observed data equally well. This problem of multiple (differing) sets of parameter values that make identical predictions is known as identifiability [4].

1.3 Proposed solution: Dirichlet priors

Dirichlet prior is an approach used to initialize conditional probability tables when training a Dynamic Bayesian network. Dirichlet distributions are specified by a pair of numbers (α,β). Figure 3 shows an example (the dashed line) of the Dirichlet distribution for (9,6). If this sample distribution were of K_0, it would suggest that few skills have particularly high or low knowledge, and we expect students to have a moderate probability of mastering most skills. Conceptually, one can think of the conditional probability table of the graphical model being as seeded with 9 instances of the student knowing the skill initially and 6 instances of him not. If there is substantial training data, the parameter estimation procedure is willing to move away from an estimate of 0.6. If there are few observations, the priors dominate the process. The distribution has a mean of α/(α+β). Note that if both α and β increase, as in the solid curve in Figure 3, the mean of the distribution is unchanged (since both numerator and denominator are multiplied by 3) but the variance is reduced. Thus, Dirichlets enable researchers to not only specify the most likely value for a parameter but the confidence in the estimate.

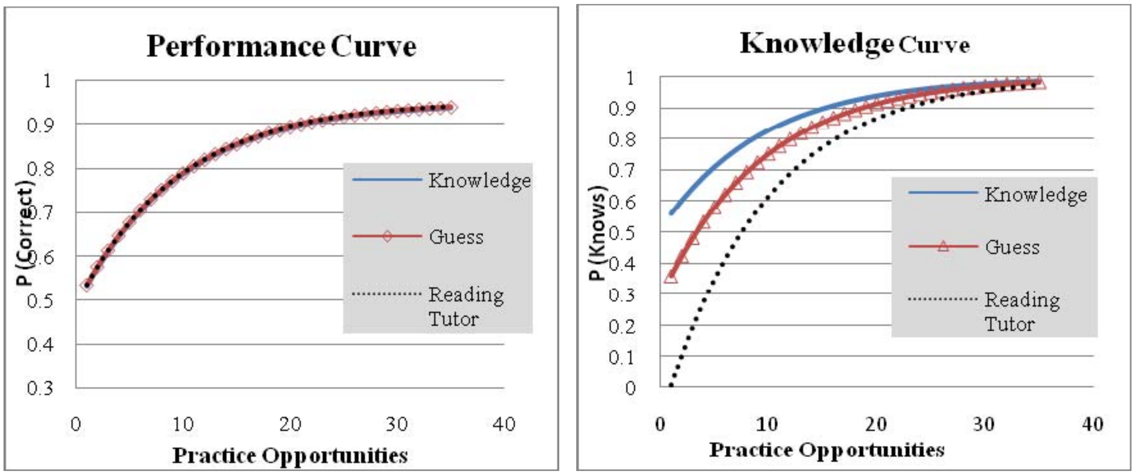


Figure 2 performance & learning curve

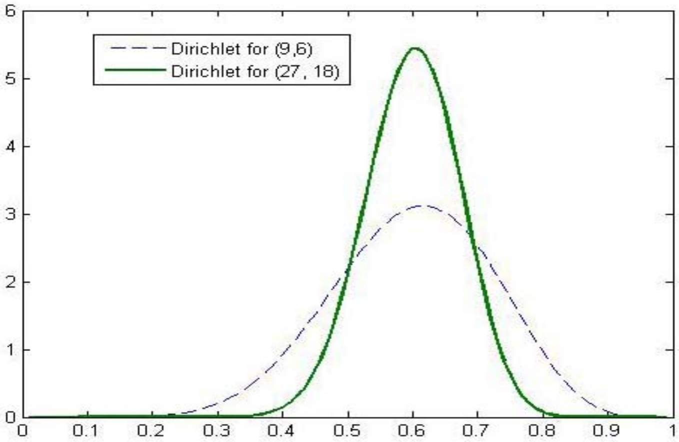


Figure 3. Sample Dirichlet Distributions demonstrating decreasing variance

Dirichlets provide bias towards the mean of the distribution. Since we estimated a set of parameters for each skill, for models with few training data, the parameter estimates can get wacky, since sparse data provide few constraints on the parameters. Hence, those parameters are sometimes estimated as extreme values. In this situation, we prefer to have parameters which are more similar to other, better-estimated, skills. With Dirichlet priors, the observations for each case are weighted against prior α, β values, i.e. models with few data are more influenced by the priors towards the mean. Therefore, we expect those estimates will be become more reasonable.

It is important to note that researchers can use Dirichlets to set confidence on priors. If the variance is less, we are surer about the priors, whereas if the variance is high, we are less sure about the priors. Each of the four parameters will not only have different mean values, but different degrees of certainty. Suppose, in a group of students if they start with similar incoming knowledge but have variable learning. Then Dirichlet prior will set higher confidence in students' prior knowledge (e.g.: α, β = 20, 34) but lower confidence in students' learning (e.g.: α, β = 1, 4). As a result, prior knowledge parameter estimation will be more biased towards prior or distribution's mean whereas learning will have more tendency to move away from prior value.

2 Methodology

There are several sources of setting Dirichlet prior values. One approach is using knowledge of the domain [e.g. 4]. If someone knows how quickly students tend to master a skill or the likelihood of knowing a skill, that knowledge can be used to set the priors. One complaint is that such an approach is not necessarily replicable as for different domains and different subjects, different experts may give different answers.

2.1 An automatic approach for selecting priors

To compare estimations from fixed and Dirichlet prior models, we trained two KT models initialized with fixed and with Dirichlet priors. We used the following approach:

1. Initialize EM with fixed priors from our rough estimates of the domain. Then use EM to estimate the model parameters for each skill in the domain

2. For all four parameters (guess, slip, K_0, learning)

- Compute the mean (μ) and variance (σ^2) of the parameter estimates
- Weight the mean and variance by the number of cases (n) of each skill. Specifically, for each parameter P of skill i,
    - $\text{weight}_i = \sqrt{n_i}$
    - $\mu' = \sum Pi * \text{weight}_i / \sum \text{weight}$
    - $\sigma^{2\prime} = \sum \text{weight}_i * (P_i - \mu_p)^2 / \sum \text{weight}$
- Select α and β to generate a Dirichlet with the same mean and variance as the estimates Specifically, solve for α and β such that:
    - $\alpha = (\mu'^2 / \sigma^{2\prime}) * (1 - \mu') - \mu'$
    - $\beta = \alpha * ((1/ \mu') - 1)$

3. We now have one Dirichlet distribution described by (α, β) for each of the four parameters

4. Reestimate two kinds of knowledge tracing models: a fixed prior model with initial value of μ' and Dirichlet prior model using the (α, β) pairs.

We calculated the mean and variance of the data. Based on those two values, we calculated α, β parameters (using the equations in step #2). However, simply calculating the mean gives all data points equal weight. This can be problematic, since as we mentioned earlier, skills with few cases are susceptible to error: going to extreme values such as getting 0 as student's learning parameter. Therefore, we weight each estimate by the square root of the number of cases used to generate the estimate, since √N is how the standard error decreases.

2.2 *Iterating the algorithm*

Rather than just stopping after step #4, it is possible to loop back to step #2. We were interested to see how the parameter estimates change by iterating the algorithm with new prior values. We ran a number of iterations on both fixed and Dirichlet prior.

Table 2. Results of iterating automatic process approach for K_0 and slip parameters

		Prior Knowledge (K_0)			**Slip**		
		Iteration 1	**Iteration 2**	**Iteration 3**	**Iteration 1**	**Iteration 2**	**Iteration 3**
Fixed Prior	Mean, Variance	0.473, 0.025	0.471, 0.025	0.468, 0.025	0.205, 0.006	0.205, 0.006	0.203, 0.005
Dirichlet Prior	Mean, Variance	0.478, 0.019	0.477, 0.017	0.476, 0.016	0.207, 0.003	0.208, 0.002	0.208, 0.002
	α, β	5.76, 6.3	6.66,7.3	6.86,7.55	11.21, 42.82	14.5, 55.2	16.63, 63.31

As shown from Table 2 , the parameters do not change much across iterations, although the variance decreases. The amount of bias towards the mean is proportional to how large α and β are, which is inversely related to the population variance. That is, if the population has a high variance then there is a small bias. Conversely, if a parameter value is already tightly clustered, there will be a strong bias towards the mean. Therefore, at each iteration estimates will move towards the mean, and the values of α, β will increase. We discuss this problem further in the future work section.

3 Validating the models

For this study, we used data from ASSISTment, a web-based math tutoring system. The data are from 199 twelve- through fourteen- year old 8th grade students in urban school districts of the Northeast United States. They were from two classes, each of which only lasted one month. These data consisted of 92,319 log records of ASSISTment during January 2009 to February 2009. Performance records of each student were logged across time slices for 106 skills (e.g. area of polygons, Venn diagram, division, etc). We split our data into training set and test set with the proportion of 2:1.

Using our approach, we ran the fixed prior model and the Dirichlet prior model for a number of successive iterations and compared their predictive accuracy and parameter plausibility.

3.1 *Predictive Accuracy*

Predictive accuracy is the measure of how well the instantiated model fits the data. We used two metrics to examine the model performance on test set: AUC (Area Under Curve) and Summed Squared Error (SSE).

As seen in We also computed the SSE = Σ (observed performance – P (correct)) $^{2.}$ We found the first iteration of the Dirichlet prior model shows a slightly better, but not meaningfully better SSE than the first iteration of fixed prior model: 8008 vs. 8016. With more iteration, SSE marginally decreases for fixed prior whereas it increases in Dirichlet.

Table 3, the AUC values don't show any difference in performance of fixed prior model and Dirichlet prior model. The values remain unchanged even for successive iterations. We also computed the SSE = Σ (observed performance – P (correct)) ^{2}. We found the first iteration of the Dirichlet prior model shows a slightly better, but not meaningfully better SSE than the first iteration of fixed prior model: 8008 vs. 8016. With more iteration, SSE marginally decreases for fixed prior whereas it increases in Dirichlet.

Table 3. Comparison of SSE and AUC

	AUC		**SSE**	
	Fixed	Dirichlet	Fixed	Dirichlet
iteration #1	0.66	0.66	8016	8008
iteration #2	0.66	0.66	8015	8010
iteration #3	0.66	0.66	8015	8012

These results show that predictive accuracy is not meaningfully better with Dirichlet priors and the accuracy does not seem to be improving with successive iterations.

3.2 Parameter plausibility

Predictive accuracy is a desired property, but EDM is also about interpreting models to make scientific claims. Therefore, we prefer models with more plausible parameters when we want to use those for scientific study. Unfortunately, quantifying parameter plausibility is difficult since there are no well-established means of evaluation. In our study, we explored two metrics for this analysis.

For our first metric, we inspected the number of practice opportunities required to master each skill in the domain. We assume that skills in the curriculum are designed to neither be so easy to be mastered in three or fewer opportunities nor too hard as to take more than 50 opportunities. We define mastery as the same way as was done for the mastery learning criterion in the LISP tutor [5]: students have mastered a skill if their estimated knowledge is greater than 0.95. Based on students' prior knowledge and learning parameters and knowledge tracing equations described before, we calculated the number of practice opportunities required until the predicted value of P(know) exceeds 0.95. Then, we compared the number of skills with unreliable extreme values in both cases (fewer than 3 and more than 50).

As seen in Table 4, fixed priors result in more extreme cases than Dirichlet priors. This result implies that Dirichlet prior model estimates more plausible parameters. . With more iteration, the extreme cases remain constant with fixed prior whereas the number slightly decreases with Dirichlet priors. The skills that are found implausible by Dirichlet are a subset of those found by fixed priors. Hence, Dirichlet is fixing the implausibility of fixed priors and is not introducing new problems of its own.

Along with this method, we had tried to make an evaluation based on the correlation between estimated the model's K_0 and the skill difficulty. We consulted two domain experts to rate skill difficulties. But their ratings were not consistent (correlation <0.4) with each other and so we abandoned this approach.

Table 4. Comparison of extreme number of practice until mastery

	# of skills with # of practices >=50		**# of skills with # of practices <=3**	
	Fixed	Dirichlet	Fixed	Dirichlet
iteration #1	29	17	2	0
iteration #2	29	16	2	0
iteration #3	29	15	2	0

Next, we tried to model students instead of skills since we it is easier to objectively rate characteristics of students rather than skills. We trained KT model per student by observing his responses in all questions across skills. The model then estimated a set of parameters (prior knowledge, guess, slip and learning) for each *student* (rather than for each skill) which represents his aggregate performance across all skills.

The students in our study had taken a 33-item algebra pre-test just before using the tutor. Taking the pre-test as external measure of incoming knowledge, we calculated the correlation between students' prior knowledge (K_0) as estimated by KT models and their pretest scores. In Table 5, we can see that the Dirichlet prior model produces slightly stronger, but not reliably so, correlations than the fixed prior. Neither method improves with more iterations.

Table 5 Comparison of correlation between prior knowledge and pretest

	Fixed prior model	**Dirichlet prior model**
iteration #1	0.76	0.80
iteration #2	0.73	0.81
iteration #3	0.73	0.81
iteration #4	0.72	0.81

4 Contributions

This paper extends prior work in automatically generating Dirichlet priors [6] in several ways. First, this study has been scaled up both in terms of more students and more skills. Prior work found a small positive, but non-reliable, gain in predictive accuracy from using Dirichlets. This paper provides evidence that the improvement was illusory. We have also improved the estimation of the α and β parameters by weighting the parameter estimates by the number of observations we have for the skill. In this way we reduce the effect of skills that only have few estimates of skewing the mean and increasing the variance.

This paper also presents a new method for evaluating student models for parameter plausibility. Although prior work [4,6] in this area proposed and used a variety of metrics, there is still a need for additional methods. Our new method was to essentially swap the knowledge tracing problem, and estimate a set of model parameters for the students rather than the skills. We then correlated the K_0 parameter for each student with his pretest score. There are many ways of estimating how much knowledge students have, and many research efforts will have approaches for doing this. Therefore, we expect this technique to have broad applicability.

Finally, we are able to extend the result that EM produces more predictive models than Conjugate Gradient Descent [8], the approach used to estimate parameter in the CMU cognitive tutors. We are now able to say that EM + Dirichlet priors is better than EM alone. Using Dirichlets we are not able to predict student behavior any better, but the parameters are generally more plausible than with fixed priors.

5 Future work and Conclusions

There are several interesting open issues regarding the estimation of Dirichlet priors. First, our method of weighting the parameter estimation process by $\sqrt{N}$, although inspired by the relative standard error of each skill's parameters, could use more theoretic grounding. Second, neither the current nor past attempt [6] at automatically extracting α and β values from the data have shown improvements in model predictive performance. However, the single attempt at human-generated Dirichlet priors [4] did show such gains. Perhaps people have useful knowledge to bring to bear on this task? Some means of incorporating human experts, and perhaps combining their insight with computer-suggested priors could be a positive step.

The notion of iterating our process of fitting the data, estimating α and β, and refitting the data seems like it should work, and was in fact inspired by the expectation maximization recipe. That it did not work was something of a disappointment, but we think we understand why: at each iteration the population variance shrinks, increasing α and β, which further shrinks the population variance on the next iteration. We need some mechanism of preventing α and β from increasing arbitrarily high, or some better metric that suggests what a "good" value of those parameters would look like.

Finally, the assumption that we can estimate the shape of the Dirichlet distribution from which the parameters were drawn is certainly more relaxed than the standard assumption that we can correctly estimate the parameter values for each skill, however it is still somewhat naïve. For example, consider the initial knowledge of a skill. It is plausible that some skills will not have been covered in class by the students: those skills could be described by a Dirichlet with a low average. Other skills, that were covered in class, could be well described by a Dirichlet with a high average. There is no single distribution that would handle both cases. Therefore, it might be productive to consider mixtures of Dirichlets.

This paper has shown that automatically generated Dirichlets are a method for generating more plausible parameters. We found that, with Dirichlets, fewer skills were estimated to require too many or too few practice opportunities to master. We have also introduced a new evaluation technique for evaluating parameter plausibility, and expect this technique to be widely applicable.

Acknowledgements

We would like to thank all of the people associated with creating the ASSISTment system listed at www.ASSISTment.org. We would also like to acknowledge funding from the National Science Foundation, the Fulbright Program for funding the first author and the US Department of Education and the Office of Naval Research for funding the second and third authors. All of the opinions expressed in this paper are those solely of the authors and not those of our funding organizations.

References

[1] Joseph E. Beck, Kai-min Chang, Jack Mostow, Albert T. Corbett, *Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology. Intelligent Tutoring Systems* 2008: 383-394.

[2] Corbett, A. and J. Anderson, *Knowledge tracing: Modeling the acquisition of procedural knowledge*. User modeling and user-adapted interaction, 1995. 4: p. 253-278.

[3] Mostow, J. and G. Aist, Evaluating tutors that listen: An overview of Project LISTEN, in Smart Machines in Education, K. Forbus and P. Feltovich, Editors. 2001, MIT/AAAI Press: Menlo Park, CA. p. 169-234.

[4] Beck, J. E., & Chang, K.-m. (2007, June 25-29). *Identifiability: A Fundamental Problem of Student Modeling.* Proceedings of the 11th International Conference on User Modeling (UM 2007), Corfu, Greece.

[5] Corbett, A.T. Cognitive computer tutors: *Solving the two-sigma problem. in International Conference on User Modeling.* 2001. p. 137-147.

[6] Beck, J. E. (2007, July 9). *Difficulties in inferring student knowledge from observations (and why you should care).* Proceedings of the AIED2007 Workshop on Educational Data Mining, Marina del Rey, CA, 21-30.

[7] Kimberly Ferguson, Ivon Arroyo, Sridhar Mahadevan, Beverly Woolf and Andy Barto: *Improving Intelligent Tutoring Systems: Using Expectation Maximization to Learn Student Skill Level.* Intelligent Tutoring Systems: Volume 4053/2006

[8] Kai-min Chang, Joseph Beck, Jack Mostow and Albert Corbett *: A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems* : Intelligent Tutoring Systems: Intelligent Tutoring SystemsVolume 4053/2006

Reducing the Knowledge Tracing Space

Steven Ritter[1], Thomas K. Harris[2], Tristan Nixon[1], Daniel Dickison[1], R. Charles Murray[1] and Brendon Towle[1]
[1]Carnegie Learning
[2]EDalytics, LLC

Abstract. In Cognitive Tutors, student skill is represented by estimates of student knowledge on various knowledge components. The estimate for each knowledge component is based on a four-parameter model developed by Corbett and Anderson [Nb]. In this paper, we investigate the nature of the parameter space defined by these four parameters by modeling data from over 8000 students in four Cognitive Tutor courses. We conclude that we can drastically reduce the parameter space used to model students without compromising the behavior of the system. Reduction of the parameter space provides great efficiency gains and also assists us in interpreting specific learning and performance parameters.

1 Introduction

Since their start over 15 years ago, Cognitive Tutors [9] have used Corbett and Anderson's [4] knowledge tracing algorithm as a method for estimating student knowledge. The knowledge tracing algorithm models student understanding as a collection of knowledge components (also called skills). Task performance depends on whether students have the requisite knowledge and whether they are able to exhibit that knowledge within the task. Knowledge components are assumed to be either known or unknown, and the system's task is to estimate the probability that each of the target knowledge components are known. The model uses two knowledge parameters: *pinitial*, the probability that the knowledge component was known prior to instruction within the software; and *plearn*, the probability than an unknown knowledge component will transition to the known state, given an encounter with a task requiring that knowledge. The model also incorporates two performance parameters, which are meant to explain why performance of a task does not exactly match the state of student knowledge. The two performance parameters are *pslip*, the probability that a student will make an error when the knowledge component is known; and *pguess*, the probability that the student will provide the correct answer when the knowledge component is unknown.

At each opportunity to use a skill, *pknown*, the system's estimate of the probability that a particular knowledge component is known, is updated as a Bayesian function of the four parameters (*pinitial* being the initial *pknown*). Since *pknown* at any point is dependent only on the prior *pknown* and the three other knowledge tracing parameters, this model is a variant of a hidden Markov model, and we can use various techniques to estimate the best-fitting parameters for each knowledge component [7].

The benefits of setting knowledge tracing parameters based on student data were empirically demonstrated by Cen et. al. [3], who fit knowledge tracing parameters based on data collected for one cohort of students, and used the new parameter settings within an optimized version of the tutor. Students using the optimized tutor were able to reach mastery in 12% less time, relative to an identical system without the optimized parameters, while maintaining equivalent performance on both immediate and delayed post-tests.

This result demonstrates the value of using educational data to improve the performance and efficiency of Cognitive Tutors. Our goal in this paper is to explore the sensitivity of the Cognitive Tutor to the particular parameters used to model students in order to see if we can reduce the search space of knowledge tracing parameters. In particular, we would like to determine whether we can achieve the benefits of setting learning and performance parameters from student data without exploring the full parameter space.

There are several reasons for our interest in this topic. First, as a practical matter, we are collecting data on over 50,000 students from curricula containing thousands of skills. Although there are several good algorithms for optimizing the search through the parameter space [7], finding the best fit can be computationally expensive. Different methods will typically find different parameters, and so it is important to understand whether these differences are large enough to have practical effects on the system's effectiveness.

Sensitivity to particular parameter fits may also affect generality. If the behavior of the system is relatively insensitive to the particular parameters used, then we might expect relatively little variability in these parameters as we model different cohorts of students. On the other hand, if we found extreme sensitivity, we might benefit from exploring whether different parameter sets for different groups of students might be an appropriate method to refine our modeling.

Areas of relative insensitivity within the parameter space can be used to reduce the variations in parameters that we consider. In the extreme case, if we were able to find a small number of parameter sets that provide good fits across a wide range of data, then we can exhaustively search through these parameter sets to find the best fit. Using a small number of parameter sets within the tutors, rather than searching a large space of parameters may also help us to more accurately estimate initial parameters for new units of instruction and more quickly adapt the system based on student data.

Perhaps the most interesting reason to reduce the parameter space is that it has the potential to allow us to interpret the fits that we find. The knowledge tracing algorithm can simply be thought of as a Markov process with four parameters, but we do ascribe meaning to the parameters: one represents prior knowledge, one ease of learning, one ease of guessing the answer and one the probability of slipping. When we find that the best fitting parameter set for a particular skill has a high probability of being learned, there is a tendency to believe that the data tells us that the skill is easily learned. But that interpretation could be misleading. It could be the case that the second-best fit to the data indicates a relatively low probability of being learned (with a compensating high probability of being initially known, for example). Such a case would not be a concern if there is a large difference in the quality of fit between the two parameter sets, but in an insensitive parameter space, it is quite possible that the second-best fit is almost as good a fit as the first. If that is the case, then what basis do we have for saying that the skill is easily learned (or is more difficult to learn)? If we reduce the search space, it is much easier to recognize whether there is a likely second-best fit for the skill that leads to a different interpretation. This inability to choose between parameter sets that more-or-less fit the data equally well has been called the identifiably problem [2].

Supporting interpretation of skill parameters brings us to the point where we can use parameter fitting for reasons other than optimizing knowledge tracing. For example, if we can depend on the interpretation of the *plearn* parameter, then we can identify skills that are not learned (or learned slowly) within the tutor, which gives us a metric for identifying particular skills or units of instruction that could be improved.

2 Examining the parameter space

Our data for these explorations comes from 8341 students who used at least one of four Cognitive Tutor courses (Bridge to Algebra, Algebra 1, Geometry or Algebra 2) in the 2007-08 school year. Across these four curricula, there are 2400 skills.

Our first step was to understand how the fitted parameters cover the knowledge tracing parameter space.

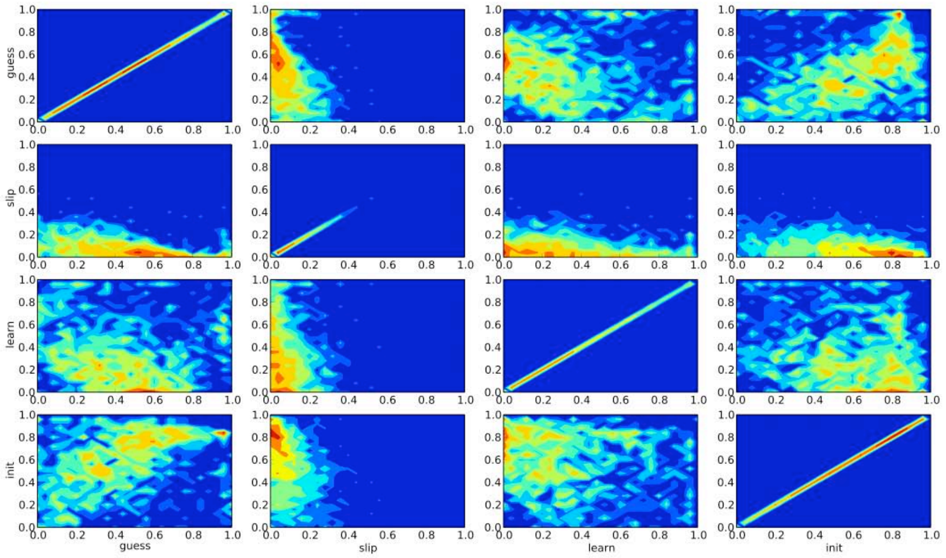


Figure 1: Heat maps showing the distribution of parameters, based on best fits of 2400 skills without constraining pguess. Each graph shows the number of skills occupying a particular position in a two-dimensional cut of the parameter space. Dark blue areas indicate regions of the space where no fits were found. Yellow and red show regions where a large number of parameter fits reside.

Figure 1 shows the results of fitting parameters on all 2400 skills. Each graph shows a two-dimensional space, defined by two of the knowledge tracing parameters. Parameters were found using an exhaustive search of the space, assuming two decimal places for each parameter (i.e. there are 100,000,000 possible parameter sets for each skill). The color in the graph indicates how many skills have a best-fitting parameter set in that region of the space. Dark blue spaces indicate areas with no skills. Yellow and red indicate areas with a large number of skills.

Inspection of Figure 1 gives us a good sense of the general shape of the parameter space. For example, it is evident that *pslip* tends to be fairly low for all skills. Skills that are judged likely to be known prior to using the tutor (with high *pinitial*) tend to have particularly low *pslip* values.

This result is promising for our goal of interpreting parameters. High values of *pslip* would be problematic for interpretation. If *pslip* exceeds 0.5, that means that the student has a greater than 50% chance of getting the item wrong, even if they know the answer. While this is logically possible, it would probably indicate a user interface where the student's intent and the student ability to express that intent in the interface are seriously compromised. If we can trust the interpretation of these parameters, then the low values of *pslip* that we see may be an indication that users generally are able to follow their intentions within the user interface.

Pguess, however, varies across the range. This is problematic. The meaning of *pguess* is the probability of being able to provide the correct answer, without having knowledge of the underlying skill. By this definition, it is hard to see how *pguess* could be greater than 0.5, because the interface never presents a case where the correct answer can be guessed with greater than a 0.5 probability. The easiest-to-guess cases in the software are ones where the student is given a two-alternative choice (0.5 probability), and those are very rare. There may be other methods of coming to a correct answer without knowledge, but they either assume that the skill model is very poor or that students generally have access to a source of answers other than their own knowledge. Baker et. al [1] call models with large values of *pguess* or *pslip* "degenerate" and also take .5 as the maximum reasonable value for these parameters. In practice, when parameters are set initially (prior to student use), we tend to fix the *pguess* parameter based on the type of question the student is being asked, with a default setting between 0.2 and 0.3.

Since part of our goal is to explore the semantics of knowledge tracing parameters, we decided to repeat this fit exercise, after constraining *pguess* to values less than 0.5. Figure 2 shows the resulting parameter space. Constraining pguess this way amounts to searching a space with 1,000,000 possible points (100 values for each of three parameters). Despite the reduction in the search space, the parameters cluster even more tightly after constraining *pguess* (that, is, there is more empty deep blue space).

The relationship between *pinitial* and *plearn* may be the most interesting, since those are the knowledge (as opposed to performance) parameters. In Figure 2, it is evident that the range of *plearn* values tends to increase as *pinitial* increases. At high values of *pinitial*, there are skills along the full range of *plearn*, and there are large clusters of skills at both very high and very low *plearn* values. This follows from the fact that high values of *pinitial* are associated with tasks that have very low error rates. If errors are infrequent, it is difficult to tell whether a particular skill is learned easily, since there are few observations of a student moving from an unlearned to a learned state. Thus, when *pinitial* is high, *plearn* can vary widely. Another way to think about this is to say that when *pinitial* is high, we cannot get a reliable estimate of *plearn*.

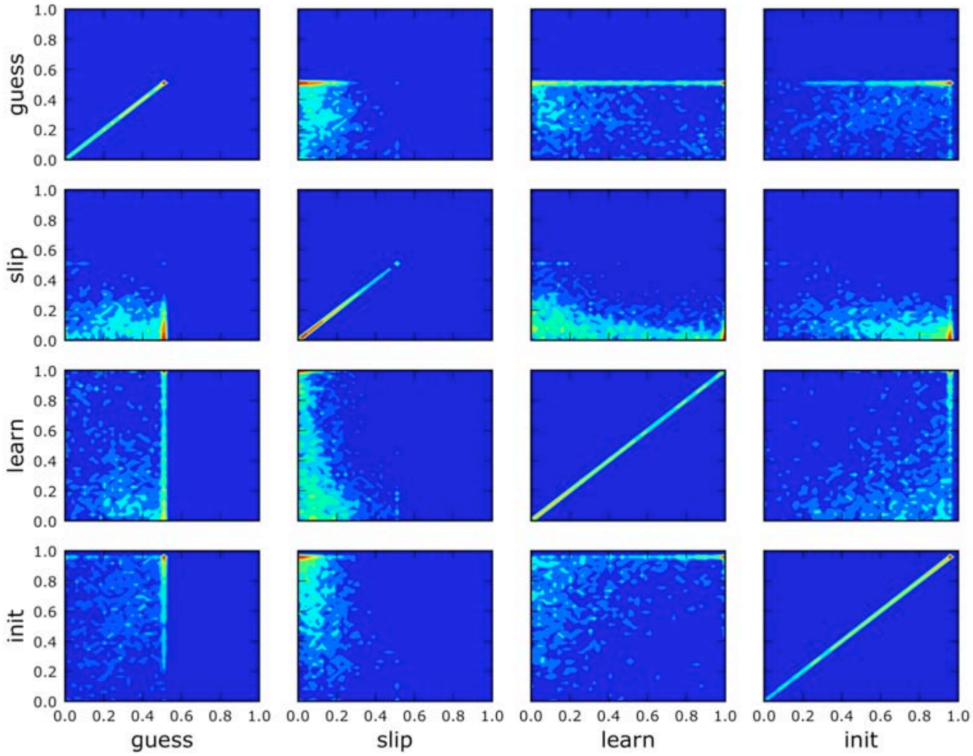


Figure 2: Parameter space with *pguess* and *pslip* constrained to be ≤ 0.5

It is also interesting to look at the relationship between *pslip* and the knowledge parameters. Although *pslip* is always low (as it was in the unconstrained fits), when *pinitial* is high, *pslip* tends to be particularly low. This finding makes sense under the assumption that skills which have been previously learned are well learned and thus relatively resistant to careless errors. Skills with both high and low *plearn*, in contrast, are in the process of being learned and thus may lend themselves more readily to slips, as is shown in the graph of the tradeoffs between those parameters.

The fact that so much of the parameter space is not used gives us hope that we will be able to find good fits to the student data using a small cluster of parameters.

3 Clustering

Building on these preliminary investigations, we set out to find the smallest group of parameter sets that could model the data sufficiently well. As a practical matter, we wanted to find a small enough number of clusters that we could imagine giving them semantically meaningful names (e.g. “not previously known but easy to learn”, “hard to learn but easy to guess”, etc.).

This approach represents a different solution to the identifiability problem than using Dirichlet priors [2]. Instead of biasing our fits based on prior beliefs about reasonable parameters, we are fitting the data using only a small number of parameter sets that provide a good fit for a large number of skills. Our assumption is that there are likely to be only a small number of semantically distinct parameter sets and that we can fit the data well using only these few sets.

In many forms of data analysis, it is assumed that a set of data was generated by some finite number of distinct processes (typically Gaussian). Clustering algorithms are a family of maximum likelihood estimation procedures for identifying these underlying processes from the set of data that they produce. The resulting model for the data consists of the set of parameters used to represent the clusters. In the current context, we are not attempting to identify the underlying generative processes (which in any event would involve complex psychological models), but rather groups of skills which behave the same with respect to the best knowledge-tracing representation. In terms of the algorithm, this turns out to mean that we are trying to identify groups of skills which project to the same regions of the p-parameter space.

In order to accomplish this, we used a k-means clustering to the fitted skills. K-means [8] is an iterative expectation-maximization [5] procedure that represents each cluster as the mean point in the parameter space. In the expectation phase, each data point is assigned to the closest cluster center. Then, in the maximization phase, each cluster center is moved to the mean point of its assigned data points. Starting with *k* cluster centers initialized at random positions throughout the parameter space, k-means converges to its final cluster positions in approximately 200-400 iterations. We used a “strict” k-means algorithm, in which the assignment of skills to clusters is an all-or-nothing relationship. This has the advantage of having a clear stopping condition – if there are no further changes in skill-to-cluster assignment, then the cluster means will not change, and the model has converged.

The K-means clustering minimizes the Euclidean distance, in the parameter space, between data points and cluster centers. This is differs from the fitting algorithm which minimizes the MSE of the predicted *pknown*, established by the model parameters, to the observed student data. Thus, it is possible to force skills into clusters that do not fit well, even though the skill is not far from the centroid of the cluster in parameter space. In theory it is possible to choose clusters that minimize the MSE to the data, rather than the distance in parameter space; in practice, however, this turns out to be computationally impractical. One avenue for future work we are looking at is ways to reduce this computational load. Since the Euclidean distance is continuous and monotonically decreasing everywhere, it is a good approximation so long as the MSE is at least locally smooth and decreasing. An informal examination of the MSE-space for a small sample of the skills indicated that this was the case, however a more in-depth examination is warranted. Using Euclidean distance has the further benefit of producing clusters that are non-disjoint in the parameter space. It would be much more difficult to justify the semantic relevance of a cluster comprised of two or more non-overlapping regions of the parameter space.

We initialized this process with *k* = 50, which converged to 23 distinct non-empty clusters. Since the assignment of skills to clusters in this particular variant of k-means is an all-or-nothing assignment, it gives the algorithm some freedom to “prune” away unnecessary clusters by assigning no data points to them. Essentially this gives the algorithm a degree of flexibility in estimating the best number of clusters needed to explain the data. Experiments with larger initial numbers of clusters (up to *k* = 100) also consistently resulted in between 20 and 25 non-empty clusters. Although the random initialization of cluster centers does introduce some variation in how the clusters converge, we found the resulting cluster centers to be very stable.

4 Interpreting the clusters

Figure 3 plots the 23 clusters that were found in the parameter space. Each cluster is represented by a circle, and the size of the circle is proportional to the number of skills that are contained in the cluster. The largest cluster contains 393 skills, and the smallest has only a single skill.

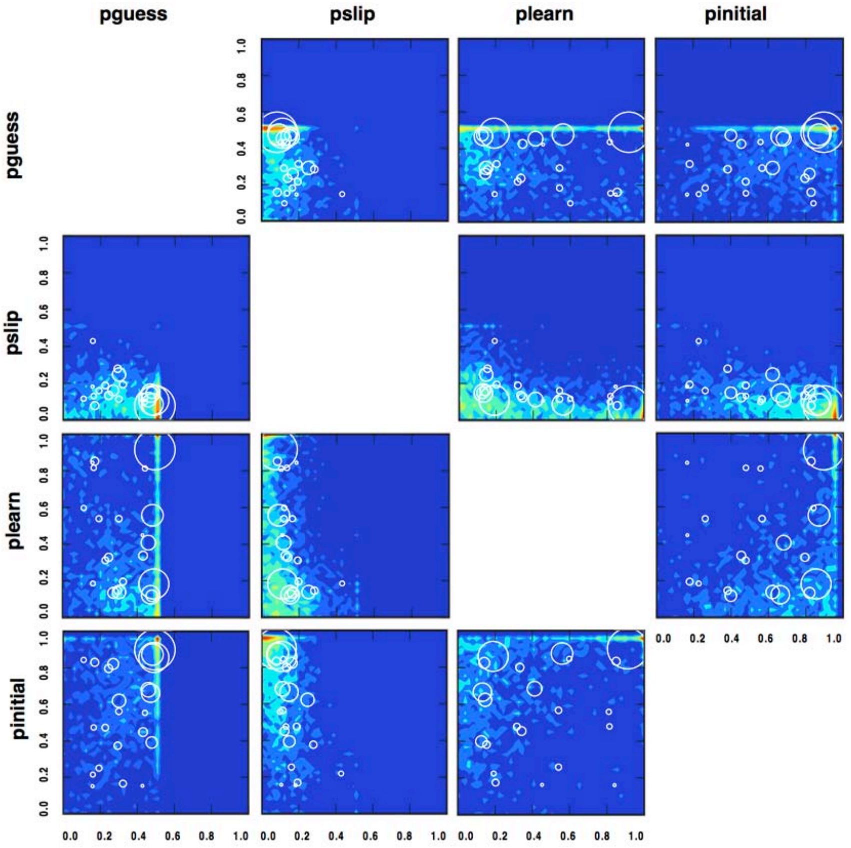


Figure 3: Positions of the final 23 clusters in the parameter space superimposed over the heatmaps. Each cluster is represented by a circle. The size of the circle is proportional to the number of skills contained in the cluster.

Using the best-fit parameters, the mean squared error (MSE) is 0.1204. Using clustered parameters increases MSE slightly, to 0.1245. MSE for the parameters delivered with the software (only some of which were set based on fits to prior years' data) was substantially higher, at 0.187. Clustering with only 23 clusters thus appears to provide a very good fit to the data, but it is difficult to understand whether even this small increase in MSE has significant effects on the behavior of the system. Since skills are bundled within problems, some skills may be presented to students even after the system has determined that the student is at mastery. For those skills, the difference between the best fit and the clustered fit may amount to nothing.

Figures 1-3 show a large number of skills with high *pinitial*. This is not surprising, since many Cognitive Tutor sections build on previous work (copying portions of a task while adding some new objectives). In these sections, skills may be repeated, and these repeats count as new skills within our model. There is little adverse effect of having skills with high *pinitial*; students will be able to master them very quickly, and their ability to master sections of the curriculum that contain a large number of skills will depend on those skills that do not have high *pinitial*. This highlights the fact that skills that are mastered quickly have little influence on system behavior. The system should be particularly insensitive to the behavior of these skills, since problem selection and mastery does not often depend on them.

For this and other reasons, Dickison et. al [6] developed a procedure for "replaying" logs of actual student behavior using fitted parameters. This algorithm takes into account skill bundling and the problem selection algorithm to determine how many problems each student in the dataset would have needed to do if the delivered parameters matched the fitted parameters. Since our goal was to predict performance in the 2008 version of the software, we used the 2008 problem selection algorithm (which changed somewhat from 2007). This necessitated dropping some sections that either incorporated changes to the skills tracked between 2007 and 2008 or that were dropped or renamed in 2008. We also excluded sections on which we had data from fewer than 10 students. For this reason, these analyses include 182 sections with a median of 177 students per section.

In order to test whether the clustered parameter sets produced substantially different system behavior than the best-fit parameter sets, we compared the median number of problems that students would need to do under best-fit parameters to the number they would need to do under the parameter sets found through clustering. The median problem counts per section using best-fit parameters were highly correlated with those using clustered parameters ($R^2 = 0.977$) suggesting that the changes in parameters made by the clustering process are negligible. The most prominent effect of clustering was that the clustered parameters often slightly reduced the change in problem count in relation to delivered parameters. The mean absolute change in median problem count (relative to the delivered parameters) was 1.95 for the fitted parameters and 1.63 for the clustered parameters. A paired t-test showed a significant difference: $t(181) = 3.2$, $p < 0.01$. This may be due to the fact that clustering tends to move parameters away from extreme values, bringing them closer to delivered parameters, which generally avoid extremes.

Another advantage of clustering is to avoid overfitting with smaller amounts of data. To test this, we developed 23 new clusters, using 1561 skills and 1312 students. We then found the best-fitting cluster for each of the 275 skills that were not used in developing the clusters, using varying numbers of students. We also found best-fitting parameters for these 275 skills on the subsets of students and tested the fit with another set of 200 students. As Figure 4 shows, when there are a small number of students contributing to the data, the clusters provide a substantially better fit to the data than the best-fit estimates. This provides evidence both that clusters developed with one set of skills will generalize to another set and that, with small amounts of student data, clusters can help prevent overfitting.

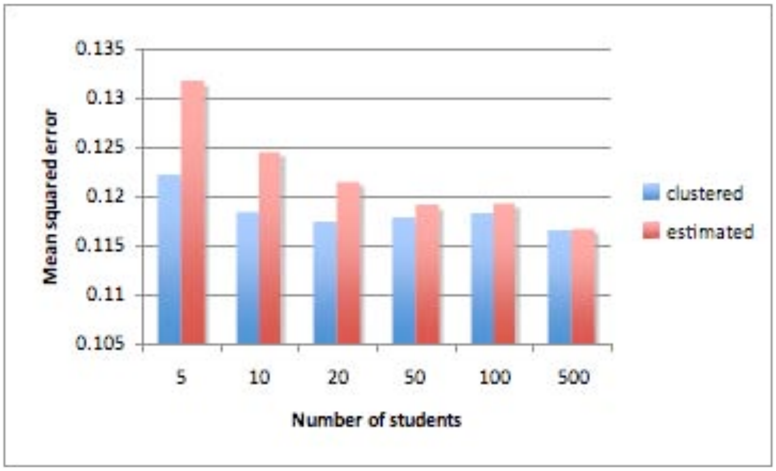


Figure 4: Comparison of clustered vs. best-fit estimates with differing numbers of students

5 Conclusion

Previous work has shown that modeling student learning and performance parameters based on prior-year student data results in improved system efficiency. This paper explored the issue of how sensitive such effectiveness is to the particular sets of parameters used. Our results have shown that tutor performance is relatively insensitive to the particular parameter sets that are used. We were able to show that, using only 23 sets of parameters, we could produce virtually the same system behavior as we would see if we had used parameters found through exploring the full parameter space. This result does not argue against fitting these parameters based on data; rather it suggests that a quick estimate of such parameters can be sufficient to produce near-optimal behavior.

It is worth pointing out that the parameters we are setting act as population parameters, which would likely benefit from adjustment for individual differences [1]. Indeed, these results may suggest that a more profitable route to accurate student modeling is to focus on individual differences, rather than population characteristics. We see clustering as complementary to both the Dirichet priors approach [2] and the use of contextual guess and slip [1].

The fact that we can model student behavior with a very small set of parameters helps us to extend the knowledge tracing model beyond simply a mathematical model of student behavior; we now have a better chance to interpret individual parameters within the set. For any knowledge component, we could calculate the goodness of fit to the data for each of the 23 parameter clusters. If we only see a good fit to one cluster, and that cluster has a high *plearn* parameter, then we can reasonably conclude that that the knowledge component is easily learned. Such a conclusion would be computationally expensive to reach in the full parameter space since, since we would need to explore a large part of the space before we could conclude that there is an almost-as-good fit to the data to be found with a low-*plearn* parameter set.

Clustering parameters thus provides us a way to quickly examine knowledge components and determine which ones are problematic. Knowledge components with low plearn might suggest areas where we should refine our instruction. Ones with high *pguess* or high *pslip* might indicate areas where we need to reconsider the user interface. Ones with high *pinitial* might indicate areas where instruction is unneeded. We are optimistic that our work in reducing the parameter space for knowledge tracing will provide us with new ways to more quickly and confidently use knowledge tracing parameters to interpret student behavior.

6 References

[1] Baker, S. J. d., Corbett, A. T. and Aleven, V. More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Estimation. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems,* 2008, pp. 406-415.

[2] Beck, J. E. and Chang, K. M. Identifyability: A fundamental problem of student modeling. *Proceedings of the 11th International Conference on User Modeling*, 2007, pp. 137-146.

[3] Cen, H., Koedinger, K.R., Junker, B. Is Over Practice Necessary? – Improving Learning Efficiency with the Cognitive Tutor using Educational Data Mining. In Lucken, R., Koedinger, K. R. and Greer, J. (Eds). *Proceedings of the 13th International Conference on Artificial Intelligence in Education,* 2007, pp. 511-518.

[4] Corbett, A.T., Anderson, J.R. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 1995, 4, 253-278.

[5] Dempster, A.P., Laird, N.M., & Rubin, D.B. Maximum Likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 1977, 39(1), 1-38.

[6] Dickison, D., Ritter, S., Harris, T. and Nixon, T. A Method for Predicting Changes in User Behavior in Cognitive Tutors. *Workshop on scalability issues in AIED 2009.*

[7] Harris, T. H., Ritter, S., Nixon, T. and Dickison, D. Hidden-Markov Modeling Methods for Skill Learning. Carnegie Learning Technical Report. 2009

[8] Lloyd, S. P., Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 1982, 28:129-137

[9] Ritter, S., Anderson, J.R., Koedinger, K.R., & Corbett, A. The Cognitive Tutor: Applied research in mathematics education. *Psychonomics Bulletin & Review*, 2007, 14(2), pp. 249-255.

Automatic Detection of Student Mental Models During Prior Knowledge Activation in MetaTutor

Vasile Rus[1], Mihai Lintean[1], and Roger Azevedo[2]
{vrus, mclinten, razevedo}@memphis.edu
[1]Computer Science Department, The University of Memphis
[2]Psychology Department, The University of Memphis

Abstract. This paper presents several methods to automatically detecting students' mental models in MetaTutor, an intelligent tutoring system that teaches students self-regulatory processes during learning of complex science topics. In particular, we focus on detecting students' mental models based on student-generated paragraphs during prior knowledge activation, a self-regulatory process. We describe two major categories of methods and combine each method with various machine learning algorithms. A detailed comparison among the methods and across all algorithms is also provided. The evaluation of the proposed methods is performed by comparing the prediction of the methods with human judgments on a set of 309 prior knowledge activation paragraphs collected from previous experiments with MetaTutor on college students. According to our experiments, a content-based method with word-weighting and Bayes Nets algorithm is the most accurate.

1 Introduction

This paper describes automatic methods for detecting students' mental models (MM) during interaction with MetaTutor [5], an intelligent tutoring system that teaches students self-regulatory processes during learning of complex science topics. At the beginning of their interaction with MetaTutor, students are given a learning goal, e.g. *learn about the human circulatory system*, and encouraged to use a number of self-regulatory processes that will eventually help with their learning. One of the important self-regulatory processes in MetaTutor is prior knowledge activation (PKA), which involves students recalling knowledge about the topic to be learned.

During prior knowledge activation, students must write a paragraph which is assumed to reflect students' knowledge with respect to the learning goal. Excerpts from PKA paragraphs corresponding to High (H) and Low (L) mental models with respect to the goal of learning about the circulatory system are given in Table 1. The paragraphs are reproduced as typed by students. Entire paragraphs are not shown due to space reasons.

Table 1. Examples of PKA paragraphs for High (H) and Low (L) mental models (MM).

MM	PKA Paragraph
H	Circulatory system is made up of 3 parts: heart, blood and blood vessels. The heart is a muscle which pumps blood in and out to the rest of the body. ... There are 3 types of blood vessels. Artery, veins and capillaries. The arteries carry blood away from the heart, veins to the heart. ...
L	I know that we all have hearts. The heart is the main source of blood. It is the strongest and most important muscle. I know that there are arteries going (coming) out of the heart. ...

Given such a PKA paragraph, the task is to infer the student mental model. We work with three qualitative mental models: low, medium, and high. We view the task of detecting the student mental models as a standard classification problem. The general approach is to combine textual features with supervised machine learning algorithms to automatically derive classifiers from expert-annotated data. The parameters of the classifiers will be derived using six different algorithms: naive Bayes (NB), Bayes Nets (BNets), Support Vector Machines (SVM), Logistic Regression (LR), and two variants of decision trees (J48 and J48graft, an improved version of J48). These algorithms were chosen because of their diversity in terms of patterns in the data they are most suited for. For instance, naive Bayes are best for problems where independent assumptions can be made among the features describing the data. The assortment of the selected learning algorithms provides some diversity in terms of potential weighting and dependency patterns among the features used to model the task at hand, e.g. naïve Bayes assume total independence among features.

In order to find a good method and algorithm for inferring student mental models based on PKA paragraphs, we have investigated two categories of methods and combined them with the above six machine learning algorithms. In one category of methods, called *content-based*, student-generated PKA paragraphs are automatically compared with various sources of knowledge describing the learning goal. The sources can be (1) a collection of pages that describe the goal, (2) a taxonomy that includes the major concepts related to the goal, or (3) ideal/expected paragraphs, written by human experts, describing the learning goal and its subgoals. The second category of methods, called *word-weighting*, maps student-articulated PKA paragraphs onto a set of features in which individual words act as features and the corresponding values are weights derived using distributional information of the words across a corpus of documents (in our case the PKA paragraphs). This latter method resembles traditional text classification models [14] in that it uses individual words as features (some classification models also use the position of the words in the documents). In addition to all the above methods, we also experimented with two baseline algorithms random guessing and uniform guessing, i.e. guessing all the time the dominant category in the training data.

The rest of the paper is structured as follows. *Background* presents the mental models in MetaTutor and previous work on automatic student input assessment. The subsequent section, *Methods*, describes in detail the methods we proposed whereas *Experimental Setup and Results* presents performance figures, lessons learned, and also outlines plans for the future. The *Conclusions* section ends the paper.

2 Background

MetaTutor is an adaptive hypermedia learning environment that is designed to detect, model, trace, and foster students' self-regulated learning about human body systems such as the circulatory, digestive, and nervous systems [5]. Theoretically, it is based on cognitive models of self-regulated learning [1, 17]. The underlying assumption of MetaTutor is that students should regulate key cognitive and metacognitive processes in order to learn about complex and challenging science topics. The design of MetaTutor is based on extensive research by Azevedo and colleagues' showing that providing adaptive human scaffolding, that addresses both the content of the domain and the processes of self-regulated learning, enhances students' learning about challenging science topics with hypermedia [2, 3, 4, 5, 10]. Overall, their research has identified key self-regulatory processes that are indicative of students' learning about these complex science topics. More specifically, they include several processes related to planning (e.g., generating sub-goals), metacognitive monitoring processes (e.g., feeling of knowing, judgment of learning), learning strategies (coordinating information sources, summarization), and methods of handling task difficulties and demands (e.g., time and effort planning).

2.1 Mental Models

Mental models are mental representations that include the declarative, procedural, and inferential knowledge necessary to understand how a complex system functions. Mental models go beyond definitions and rote learning to include a deep understanding of the component processes of the system and the ability to make inferences about changes to the system. The acquisition of mental models of complex systems can be facilitated through presenting multiple representations of information such as text, pictures, and video in hypermedia learning environments [12]. Therefore, hypermedia environments, such as MetaTutor, with their flexibility in presenting multiple representations, have been suggested as ideal learning tools for fostering sophisticated mental models of complex systems [1, 8].

Detecting mental model shifts during learning is an important step in diagnosing ineffective learning processes and intervening by providing appropriate feedback. One method to detect students' initial mental model of a topic is to have them write a paragraph. Cognitively, this activity allows the learner to activate their prior knowledge of the topic (e.g., declarative, procedural, and inferential knowledge) and express it in writing so that it can be externalized and amenable to computational methods of analysis. A mental model can be categorized qualitatively, and depending on the current state (e.g., simple model vs. sophisticated model), is then used by the hypermedia system to provide the necessary instructional content and learning strategies (e.g., prompt to summarize, coordinate informational sources) to facilitate the student's conceptual shift to the next qualitative level of understanding. Along the way, students can be prompted to modify their initial paragraph and thereby demonstrate any subsequent qualitative changes to their initial understanding of the content. This qualitative augmentation is a key to an intelligent, adaptive hypermedia learning environment's ability to accurately foster cognitive growth in learners. This process continues periodically throughout the learning session.

2.2 Mental Models Coding

Due to their qualitative nature, most researchers develop complex coding schemes to represent the underlying knowledge and most often use categorical classification systems to denote and represent students' mental models. For example, Chi and colleagues' early work [7] focused on 7 mental models of the circulatory system. Azevedo and colleagues [1] extended their mental models classification to 12 to accommodate the multiple representations embedded in their hypermedia learning environment. In this paper, we have re-categorized our existing 12 mental models of the circulatory system (see [10] for the details) into 3 categories of low-, intermediate, and high-mental models of the circulatory system. The rationale for choosing the 3-category mental models approach was to enhance the ability of determining students' mental models shifts during learning with MetaTutor and because the 12 mental models approach would have been too detailed of a grain size to yield reliable classifications and thus to accurately assess "smaller" qualitative shifts in students' models.

2.3 Previous Work on Evaluating Natural Language Student Input in Intelligent Tutoring Systems and Automated Essay Grading

Researchers who have developed tutorial dialogue systems in natural language have explored the accuracy of matching students' written input to a pre-selected stored answer: a question, solution to a problem, misconception, or other form of benchmark response. Examples of these systems are AutoTutor and Why-Atlas, which tutor students on Newtonian physics [9, 16], and the iSTART system, which helps students read text at deeper levels [13]. Systems such as these have typically relied on statistical representations, such as latent semantic analysis (LSA; [11]) and content word overlap metrics [13]. LSA has the advantage of representing texts based on latent concepts (the LSA space dimensions, usually 300-500) which are automatically derived from large collection of texts using singular value decomposition (SVD), a technique for dimensionality reduction. More recently, a lexico-syntactic approach, entailment evaluation [15], has been successfully used to meet the challenge of natural language understand and assessment in intelligent tutoring systems. The entailment approach has been primarily tested on short student inputs, namely individual sentences. Both LSA and the entailment approach pose some challenges for evaluating the PKA paragraphs we have to handle. LSA requires the construction of a LSA space based on a large collection of documents from the domain of interest, i.e. the circulatory system. Collecting such tests is a time consuming task. Also, LSA suffers from the text-length confound which means using it for handling paragraph-length texts would lead to high similarity scores, probably resulting in many false positives. The entailment approach has been designed for sentence-to-sentence relation and thus it is not trivial to extend it to handle paragraph-to-paragraph tasks as it requires the use of a syntactic parser which operates on one sentence at a time. We do plan to extend it to handle paragraph-to-paragraph textual relation detection using coreference resolution components that will link concepts across sentences for a paragraph-level meaning representation. For the time being, we opted instead for a set of methods that combine simple textual overlap features with machine learning algorithms to automatically infer student mental models. We take advantage of the goals and subgoals in MetaTutor when choosing the features to be used in our solution to the student mental model detection problem, as explained later.

The problem of detecting student mental models from PKA paragraphs is related to the task of *automated essay scoring* (AES), i.e. automatically evaluating and scoring written texts. The purpose in AES is to improve time, cost, reliability and generalizability of the process of writing assessment. Dikli [19] gives a fairly comprehensive survey of AES systems. AES systems require training , i.e. human-scored written texts, and rely on form and content features to score written texts. They do not really understand the texts or emulating the human scoring process. One difference between AES and MM detection is that the length of the input is different. Usually, in AES essay-long texts, which are comprised of many paragraphs, are considered while in our task of MM detection we work with smaller, paragraph-length texts. AES systems use the multi-paragraph structure of essays as part of the scoring algorithm while in the MM detection problem this structural information is less important. The content-based components of the AES systems could be used for the MM detection task. Some of our proposed methods resemble some of the content-based methods employed in AES systems (see the word weighting in the vectorial representation used in E-rater, which is described in [19]).

3 Methods

All the methods we implemented, except the baselines, have two major steps. The first step consists of data processing and feature extraction. The details of this step are specific to each method and will be described later. During a second step, we used machine learning algorithms to induce various classifiers for categorizing PKA paragraphs into high, medium, and low mental models. We experimented with the six machine learning algorithms mentioned earlier. It is beyond the scope of this paper to discuss in detail these algorithms (see [14, 18] for details). We used the implementation of the algorithms from WEKA, a machine learning toolkit [18]. The algorithms were run with the default parameters, e.g. SVM was run with the polynomial kernel. There is a large parameter space for these learning algorithms and we plan to tweak these parameters in the future in order to further investigate their behavior for our problem. For this paper, the machine learning phase was used to check the effectiveness of the preprocessing phase and of the chosen set of features and methods.

The performance of all the methods was evaluated using 10-fold cross validation. In k-fold cross-validation the available data set is split into k folds. Then, one fold is kept for testing and the other *(k - 1)* are used for training. This process is repeated for each fold resulting in k trials. The reported performance is then computed as the average of the individual trials' performances. When $k = 10$ we have 10-fold cross-validation. To further increase the confidence in the estimated values of the reported accuracy, we have run 10-fold cross-validation 10 times, each time with a different seed value, which is an input parameter to k-fold cross-validation evaluation. The seed value affects the way instances in the data set are selected for the individual folds. Thus, for each method and learning algorithm we compute $10 * 10 = 100$ performance scores and then take the average. The advantage of running 10-fold cross-validation 10 times with different seed points is that each instance in the original data set is evaluated 10 times. By comparison, a 100-fold cross-validation would result in each instance being evaluated once. We also ran paired t-tests among different methods and learning algorithms in order to check if differences in performance are statistically significant. We report performance in terms of accuracy and kappa coefficient. Accuracy is the percentage of correct predictions out of all predictions. Kappa coefficient measures the level of agreement between predicted categories and expert-assigned categories while also accounting for chance agreement.

3.1 Content-based Methods

The methods in this category rely on the presence of key concepts related to the learning goal in the student-articulated paragraphs. The key concepts are specified in different ways for the three methods in this category and it is in this aspect that the methods differ. The key concepts are specified in the three methods using the following benchmarks, respectively: (1) expert-created domain taxonomy, (2) original pages of content, and (3) expert-generated ideal descriptions of the learning goal and its subgoals.

For all three methods, 8 features are computed: one feature corresponding to the overall learning goal and one feature for each of the 7 subgoals. The value of each feature represents the percentage of words in the entire benchmark (for the feature corresponding to the overall learning goal) or parts of the benchmark corresponding to subgoals (for subgoal features) that are present in the student-generated PKA paragraphs. For instance, for the *taxonomy-based method* (*tax* in Table 2) a taxonomy of concepts is the benchmark. The overall goal, i.e. learn about the circulatory system, is the top node of the taxonomy (see Figure 1). The seven subgoals are the nodes in the *ideal level* in the Figure 1. The parts of the taxonomy benchmark corresponding to subgoals are the subtrees below the subgoals nodes in the taxonomy. We use nodes in these subtrees to compute the values corresponding to the 7 subgoal-related features. The advantage of the taxonomy-based method is its simplicity and small computational costs as the taxonomy only includes several dozen concepts. The trade-off is the expert associated costs to build the taxonomy. In MetaTutor, the taxonomy was needed for assessing and feedback during another self-regulation process, subgoal generation, and thus there is no extra effort to build the taxonomy specifically for mental model detection.

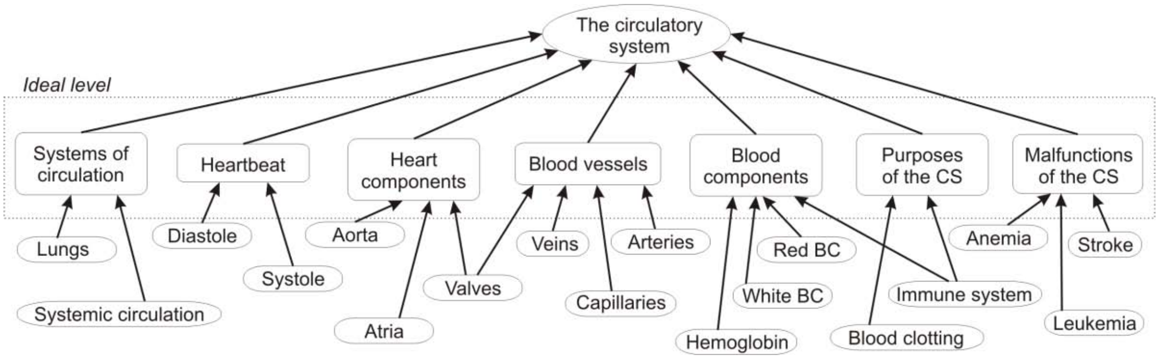


Figure 1. Partial Taxonomy of Topics in Circulatory System.

N-grams methods are very similar to the taxonomy-based method. Instead of using the taxonomy to identify key concepts relevant to the learning goal or subgoals, we used the subset of content pages related to the overall goal or subgoals, respectively. The values for the features are computed as the percentage of N-grams, i.e. sequences of N consecutive words, in the benchmark, or parts of it for subgoal features, that are present in the PKA paragraphs. In this method, it is necessary to know which page is relevant to which subgoal. An expert mapped each individual page onto each subgoal. Also, to generate the N-grams the pages and PKA paragraphs are pre-processed: stop words are eliminated and the remaining words are lowercased and stemmed. Stop words are very frequent words such as determiners, e.g. *the*. Stemming is the process of mapping all morphological variation of a word to its base form, e.g. *hearts* and *heart* are mapped to *heart*. We used both unigrams (*uni*) and bigrams (*bi*) to compute content overlap. We also experimented with a combined method in which both bigrams and unigrams are used (*uni-bi*). Bigrams have the advantage (over unigrams) of capturing some word order, i.e. syntactic information. The N-grams methods have the advantage of needing no extra structures, e.g. expert-built taxonomies, to generate the features. We simply used the original content pages about the circulatory system from Encarta, which are used in MetaTutor. On the other hand, there is need for an expert to specify which content page is relevant to which subgoal. The biggest disadvantage of the N-gram method is their use of too much content to compare against, e.g. bigrams from all the content pages for the overall goal feature, as opposed to a set of well-selected key concepts from a taxonomy as is the case with the taxonomy-based method.

In the last method in this category, called *expectation-based*, we started by asking domain experts to generate ideal descriptions for each of the seven subgoals. These descriptions are short textual paragraphs comprising of 5-7 sentences. The collection of all paragraphs for the 7 subgoals is used to derive the eighth feature corresponding to the overall learning goal. The values of the features are generated using unigram and bigram overlap between the ideal paragraphs and the student PKA paragraphs. In this method (labeled *ip* - ideal paragraphs - in Table 2), there is no need for creating a crisp taxonomy of concepts and decide which concepts is directly related to which concept. The effort to create the ideal paragraphs is less compared to building a taxonomy for instance.

3.2 Word-weighting Methods

In this category of methods, we select from each paragraph all the words that have minimum 4 letters (when all words were used performance results were slightly worse), excluding the stop words. The selected words are then converted to lower case and stemmed. The resulting set of words is used to describe the paragraphs, i.e. they are the features. Each feature is weighted using *tf-idf* (term frequency-inverted document frequency), which captures the importance of the corresponding feature for a given paragraph. Inverted document frequency (idf) is computed as the inverse of document frequency, which is the number of documents a term occurs in from a collection of documents. In our case, document frequency is the number of prior knowledge-paragraphs a term occurs in. Term frequency, *tf*, is the number of occurrences of a term/word in a document, i.e. a PKA paragraph. As a result, a total of 1038 features are extracted and used to describe each instance in data set. Other weighting schemes, besides tf-idf, could be used but the tf-idf proved to be successful in a number of other applications [6] which is the reason we chose it.

4 Experimental Setup and Results

4.1 The Dataset

In this paper, we have experimented with an existing dataset consisting of 309 mental model essays collected from previous experiments by Azevedo and colleagues (based on

[2, 3]). The dataset consisted of entries from senior high school students and non-biology college majors. These mental model essays were classified by two experts with extensive experience coding mental models. Each expert independently re-coded each mental model essay into one of the three categories and achieved an inter-rater reliability of .92 (i.e., 284/309 agreements) yielding the following new dataset for this paper: 139 low mental models, 70 intermediate mental models, and 100 high mental models. The coders included a nurse practitioner and a high school biology teacher.

4.2 Results

We report results for all combinations of methods and learning algorithms mentioned earlier. In Table 2, rows correspond to methods and columns to learning algorithms. An analysis of the results revealed that a tf-idf method combined with Bayes Nets leads to best overall results in terms of both accuracy and kappa values. The second best results were obtained using a combination of unigrams and/or bigrams with SVM or LR. Both SVM and LR are called function-based classifiers as they are both trying to identify a function that would best separate the data into appropriate classes, i.e. mental model types in our case. For the random baseline we obtained (accuracy = 31%, kappa = -0.06 - a kappa close to 0 means chance) based on averaging over 10 random runs while for the uniform baseline, i.e. predicting all the time the dominant class, which is the Low mental model class, we obtained (accuracy = 45%, kappa = 0).

Table 2. Performance results as accuracy(%)/kappa values

Method	NB	BNets	SVM	LR	J48	J48graft
tf-idf	57.70/0.35	76.31*/0.63*	64.12*/0.42	54.21/0.28	68.22*/0.50*	71.19*/0.55*
Tax	61.44/0.39	61.93/0.37	67.18*/0.44	69.61*/0.50*	62.23/0.40	62.65/0.40
Uni	63.65/0.45	62.97/0.44	67.57/0.45	70.03*/0.52	64.65/0.43	64.52/0.43
Bi	66.14/0.47	64.75/0.46	70.09/0.49	70.64*/0.52	63.40/0.41	63.56/0.41
uni-bi	65.43/0.47	63.63/0.45	68.79/0.46	70.22/0.52	68.93/0.49	68.89/0.49
ip-uni	66.39/0.48	66.14/0.48	67.83/0.45	65.62/0.44	65.85/0.47	65.88/0.47
Ip-bi	61.42/0.38	65.18*/0.43	67.21*/0.44	67.05*/0.45	62.14/0.40	62.37/0.40
ip-uni-bi	64.94/0.45	64.53/0.46	67.05/0.43	66.83/0.46	65.40/0.46	65.66/0.46

Based on a more careful analysis of the results in Table 2, we found that given a method the choice of the machine learning algorithm is important. Looking at the results within each group of methods one can notice the relative large range of the performance figures. For instance, the accuracy values for the tf-idf method vary most from 57.70% for naive Bayes to 76.31% for Bayes Nets. For Bayes Nets the Weka's default K-2 search algorithm was used. This variability indicates that this method is more sensitive with respect to the choice of the machine learning algorithm. We call such methods less stable. One possible explanation for the variability of the tf-idf method could be its large number of features used (1038) relative to the number of instances (309). This is not unusual for text classification as, for instance, a typical naive Bayes method [14] uses not only all the words in the documents to be classified but also their positions leading to a very large number of features. The last three groups of methods in Table 2 also show variability but they seem more stable as the range of the values is somehow smaller. The most stable methods are the ideal paragraph-based methods and the unigram/bigram methods. As unigram/bigram methods provide better results than the paragraph-based methods we could say that the former offer the best of performance and stability across various machine learning schemes. We plan to conduct a study on the stability of the tf-idf method once more PKA paragraphs are available from future MetaTutor experiments. Given its best performance overall, if we can show that this method is also stable if more training data is available - as we suspect - it would be a very important finding.

5 Conclusions

We presented and evaluated several methods for detecting student mental models in the intelligent tutoring system MetaTutor. We have found that a tf-idf method combined with a Bayes Nets algorithm provides the best accuracy and kappa values. Bigram-based methods combined with Logistic Regression or Support Vector Machines provide competitive results. In addition, bigram-based methods seem to be less sensitive to the choice of the machine learning algorithm compared to the tf-idf method. It is believed that tf-idf methods would be more stable if more training data would be available.

Acknowledgments

This research was supported by funding from the National Science Foundation awarded to R. Azevedo (0133346, 0633918, and 0731828) and V. Rus (0836259). We thank Amy Witherspoon, Emily Siler, Michael Cox, and Ashley Fike for data preparation.

References

[1] Azevedo, R. (in press). The role of self-regulation in learning about science with hypermedia. In D. Robinson & G. Schraw (Eds.), *Current perspectives on cognition, learning, and instruction*.

[2] Azevedo, R., Greene, J.A., & Moos, D.C. (2007). The effect of a human agent's external regulation upon college students' hypermedia learning. *Metacognition and Learning, 2*(2/3), 67-87.

[3] Azevedo, R., Moos, D.C., Greene, J.A., Winters, F.I., & Cromley, J.C. (2008). Why is externally-regulated learning more effective than self-regulated learning with hypermedia? *Educational Technology Research and Development, 56*(1), 45-72.

[4] Azevedo, R., & Witherspoon, A. Self-regulated learning with hypermedia. (in press). In Graesser, A., Dunlosky, J., & Hacker D. (Eds.), *Handbook of metacognition in education,* in press. Manwah, NJ: Erlbaum.

[5] Azevedo, R., Witherspoon, A., Graesser, A.C., McNamara, D.S., Rus, V., Cai, Z., & Lintean, M. MetaTutor: An adaptive hypermedia system for training and fostering self-regulated learning about complex science topics. *Annual Meeting of Society for Computers in Psychology*, 2008. Chicago, IL.

[6] Baeza-Yates, R. & Ribeiro, B. *Modern Information Retrieval*, Addison-Wesley, 1998.

[7] Chi, M. T. H., Siler, S., Jeong, H., Yamauchi, T., & Hausmann, R. Learning from human tutoring. *Cognitive Science*, 2001, 25, p. 471-534.

[8] Goldman, S. Learning in complex domains: When and why do multiple representations help? *Learning and Instruction*, 2003, 13, p. 239-244.

[9] Graesser, A.C., Hu, X., & McNamara, D.S. Computerized learning environments that incorporate research in discourse psychology, cognitive science, and computational linguistics. In Healy, A., ed., *Experimental Cognitive Psychology and its Applications*, 2005, p. 59-72. Washington, D.C.: APA.

[10] Greene, J.A. & Azevedo, R. A macro-level analysis of SRL processes and their relations to the acquisition of sophisticated mental models. *Contemporary Educational Psychology*, 2009, 34, p. 18-29.

[11] Landauer, T., McNamara, D.S., Dennis, S. & Kintsch, W. (Eds), *Latent Semantic analysis: A road to meaning*, 2007, Mahwah, NJ:Erlbaum.

[12] Mayer, R. *The Cambridge handbook of multimedia learning*, 2005, NY: Cambridge University Press.

[13] McNamara, D.S., Boonthum, C., Levinstein, I.B., & Millis, K., Evaluating self-explanations in iSTART: comparing word-based and LSA algorithms. In Landauer, T., D.S. McNamara, S. Dennis, and W. Kintsch (Eds.), *Handbook of LSA*, Mahwah, NJ: Erlbaum, 2007, p. 227-241.

[14] Mitchell, T. *Machine Learning*, McGraw Hill, 1997.

[15] Rus, V., McCarthy, P.M., Lintean, M.C., Graesser, A.C., & McNamara, D.S. Assessing student self-explanations in an intelligent tutoring system, In D. S. McNamara and G. Trafton (Eds.), *Proceedings of the 29th the Annual Conference of the Cognitive Science Society*, 2007.

[16] VanLehn, K., Graesser, A.C., Jackson, T., Jordan, P., Olney, A., & Rose, C. When are tutorial dialogues more effective than reading? *Cognitive Science*, 2007, 31(1), 3-62.

[17] Winne, P. & Hadwin, A. The weave of motivation and self-regulated learning. In Schunk, D. & Zimmerman, B. (Eds.), *Motivation and self regulated learning: Theory, research and applications*, 2008, p. 297-314. NY: Taylor & Francis.

[18] Witten, I. H. & Frank, E. *Data Mining: Practical machine learning tools and techniques, 2nd Edition*, Morgan Kaufmann, San Francisco, 2005.

[19] Dikli, S. An Overview of Automated Scoring of Essays. *Journal of Technology, Learning, and Assessment*, 2006, 5(1).

Automatic Concept Relationships Discovery for an Adaptive E-course

Marián Šimko and Mária Bieliková
{simko, bielik}@fiit.stuba.sk
Institute of Informatics and Software Engineering,
Faculty of Informatics and Information Technology,
Slovak University of Technology

Abstract. To make learning process more effective, the educational systems deliver content adapted to specific user needs. Adequate personalization requires the domain of learning to be described explicitly in a particular detail, involving relationships between knowledge elements referred to as concepts. Manual creation of necessary annotations is in the case of larger courses a demanding task. In this paper we tackle a concept relationship discovery problem that is a step in adaptive e-course authoring process. We propose a method of automatic concept relationship discovery for an adaptive e-course. We present two approaches based on domain model graph analysis. We evaluate our method in the domain of programming.

1 Introduction

Authoring an adaptive educational system consists of several steps and differs among particular methodology employed. Nevertheless, one step is common: underlying domain model creation. Its purpose is to describe the domain area that is the subject of learning. The description is most commonly provided in the form of a concept map [1]. Interlinked concepts resemble lightweight ontology (refer to Figure 1) where relationship types are limited to those relevant for educational process. Typical example is a *prerequisity* relationship determining that two concepts have to be learned in a given order. Similarly, a *similar-to* relationship represents the fact that concepts are similar to a certain extent (e.g., they represent topics that often appear together in learning objects). Concepts are also connected with educational material. A weighted relationship determines the degree of concept’s relatedness to (or “containess” in) a learning object – educational material portion. Learning objects are not limited to explanatory text (such as chapters or sections from books), but represent also exercises, examples, etc.

Accurate identification of concepts and their relationships is crucial to adaptation quality (e.g., intelligent concept recommendation). However, when authoring a course, a suitable domain model is often not available. Although some standardized domain ontologies exist, they suffer from excessive generalization and only exceptionally fit to the author's needs at the desired level of granularity. In such cases the domain model has to be created “from scratch”. Unfortunately, manual construction is a tedious and time-consuming task even for small domains. If there are dozens of concepts identified, relations are counted by hundreds. Adaptive e-course authoring and maintainability complexity is a major bottleneck of adaptive educational systems.

Our research goal is to support teachers (content authors) in adaptive e-course authoring process. In this paper, we tackle automatic concept relationship discovery problem. We aim at concept similarity computation that is a core step in a relationship creation process. We propose two approaches based on graph algorithms processing underlying domain model portion.

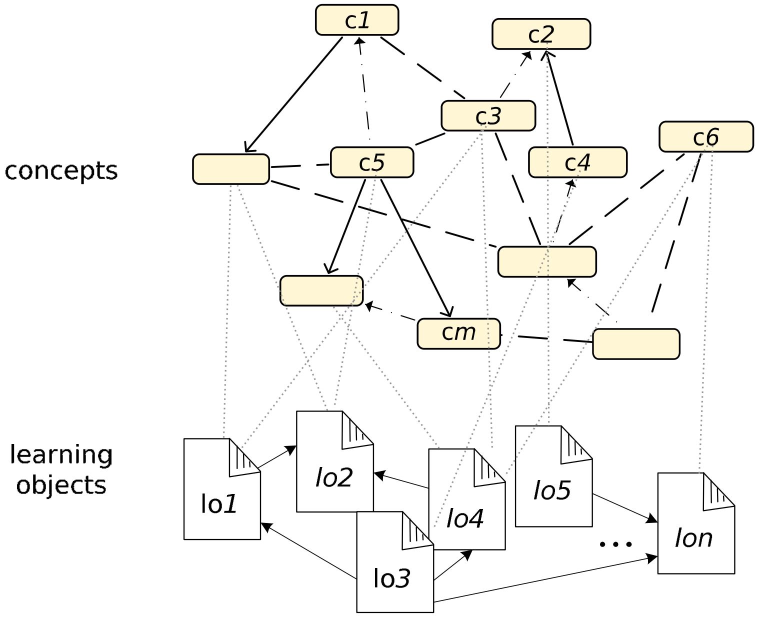


Figure 1. E-course domain model.

The rest of the paper is structured as follows. In section 2 we discuss related work. Section 3 introduces the proposed method, while sections 4 and 5 provide its description in more detail. Section 6 provides evaluation details and the results of performed experiments. In section 7 we sum up our contribution and discuss future work.

2 Related work

The work related to concept relationship discovery in the area of adaptive e-learning is presented in [6]. Concept similarities are computed based on the concept domain attributes comparison. However, the meaning of *concept* in the terminology of Cristea is different from one described by Brusilovsky [1]. Although it contains domain attributes, it also holds textual representation. This should be considered an intentional description from the ontological point of view, but then the reusability of such concepts is arguable. We are not aware of any further evidence of (semi-)automatic concept relationship generation in the adaptive e-learning field.

Finding relations between concepts is a subtask of ontology learning field [10]. The relations being created typically have taxonomic character (*is-a*). Considering text mining, related approaches mainly utilize natural language processing (NLP) techniques. Relations are induced based on linguistic analysis relying on preceding text annotation [2], incorporating formal concept analysis (FCA) [4] or using existing resources such as Wikipedia or WordNet [5]. The drawback of relationship discovery as an ontology learning task (in relation to e-course authoring) is its dependency on precise linguistic analysis. Most of the approaches rely on lexical or syntactical annotations, the presence of powerful POS taggers, existing domain ontologies, huge corpuses or other external semantic resources (e.g. WordNet). As mentioned above, this knowledge is often not available during an e-course authoring. The solution for teachers should involve unsupervised approaches to unburden them from additional work. This need we address in the method we propose.

The task of structuring the concept space is also present in the area of the topic maps. In this field, the elementary units – topics – we can view analogical to concepts. Authors in [7] generate relations between topics by analyzing the HTML structure of Wikipedia documents. The results indicate that learning objects representation structure should be considered when structuring the domain space. Categorization methods are used in [9] where similar topics are discovered by latent semantic indexing (LSI) and K-means clustering. Unsupervised methods serve as guidance in topic ontology building. Similar approach is missing in the area of adaptive e-learning.

Our method is based on statistical unsupervised text processing and graph analysis related to actual knowledge about the domain. We explore generated or existing concept associations in order to reveal hidden semantic relationships. We were motivated by good results of graph analysis employment achieved in the area of Web search. Our method does not depend on external semantic resources allowing to be used in various domain environments. However, additional semantic connections need not to be excluded.

3 Automatic concept relationship discovery

The concept relations discovery is one of several steps in the adaptive e-course authoring process. Prior to this step we assume a teacher has already created (eventually reused) learning objects, put them into reasonable structure (e.g., hierarchical), identified domain concepts and assigned them to learning objects. Concept extraction can also be done semi-automatically as we show later.

Our goal is to utilize the actual knowledge and discover relationships between concepts. As we represent a domain using a graph model, we conduct a graph analysis. We employ two alternative graph algorithms suitable to this task. Before the algorithms are applied we perform learning objects preprocessing. Because we are interested in e-learning domain, we consider several specifics when dealing with the knowledge discovery.

Learning objects are present mostly in a textual form. Thus we employ text mining techniques. We suppose the number of learning objects is known. This enables us to compute inverse document frequency when building term vector-based learning objects representations. Moreover, the learning objects we process are related to one domain area. This fact reduces the concept ambiguity problem unlike when processing heterogeneous sources. Learning objects are often mutually interconnected. We usually know the hierarchical relationships between learning objects in the course or references to other course parts may exist.

Due to the specifics we are able to compose relatively accurate vector representation during the preprocessing. For example, we can employ bag-of-words model with tf-idf weights. Based on the weights we determine the degree of each concept's relatedness to all learning objects.

After the preprocessing step we apply concept relationship discovery method itself. Using selected graph analysis algorithm we (1) compute concept similarity scores and finally (2) create relationships between each concept and his top-*k* most similar neighbors. For graph analysis we employ two alternatives: spreading activation and PageRank-based approach. Both approaches are described in more detail in the following sections.

In the second step, for each concept the most similar neighbors according to computed similarity are chosen. The top-*k* set we define as a set where the most similar neighbors sorted by score accumulate *k*% of all neighbor similarity values. For example, top-20 neighbors accumulate 20% of sum of all neighbors' similarities. We typically set coefficient *k* from <10; 20>. Between each concept and its top-*k* neighbors we create relationships. The relations' weights are normalized with regard to the whole domain model.

4 Spreading activation

The principle of the spreading activation approach is to consider the domain model to be a contextual network. Contextual network is network where several types of nodes exist. We recognize two node types: learning object nodes and concept nodes. In contextual networks, the spreading activation method is often used for similarity search [3]. The queried node is activated with energy E that spreads to neighbor nodes via incident edges. Final energy distribution in the graph determines the similarity of nodes. We use this principle to compute the degree of the concepts' similarity.

Basic steps of the algorithm can be described as follows:

```
For each concept $c_i$:
   1. activate concept with initial energy $E_0$,
   2. spread activation to entire graph,
   3. determine the degree of similarity to all concepts.
```

In step 2 the energy spreads from an activated node to all neighbor nodes proportionally according to outgoing relationships weight. Weights derived from learning object vector representation determine concept relatedness to different learning objects (refer to example depicted in Figure 2).

The crucial step is step 3. After activation spreading finishes, each concept in the network is activated with energy according to its relatedness to concept c_i. The degree of similarity between concept c_i and its neighbor c_j is computed as follows:

$$sim^{sa}_{i,j} = \frac{E_j}{\sum_k E_k} \log(sd_{i,j}) \qquad (1)$$

where $sim^{sa}_{i,j}$ is spreading activation similarity between concepts c_i and c_j, E_j is the energy of concept c_j and $sd_{i,j}$ is the shortest distance between the concepts c_i and c_j in the contextual network graph. The purpose of the formula is to normalize the power-law distributed activation energy values.

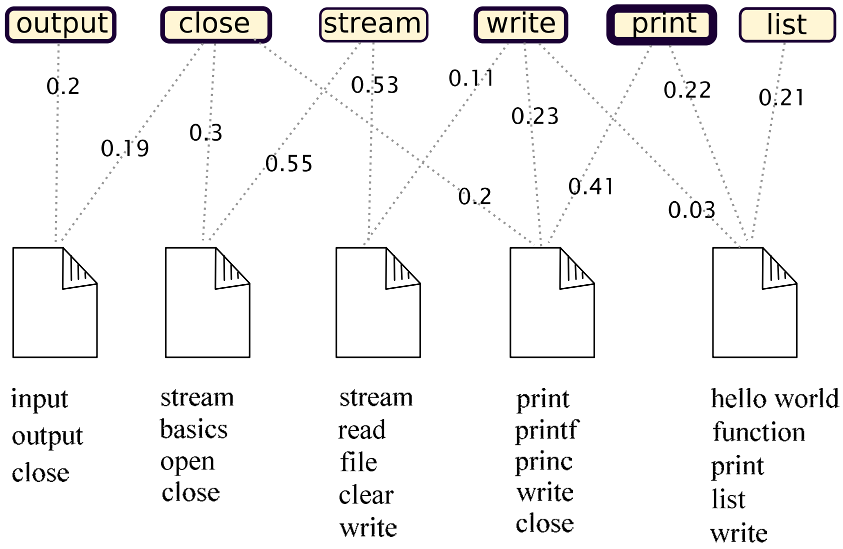


Figure 2. Example of a contextual network with two types of nodes: concepts and learning objects with the simplified vector representation. After activation spreading finishes, each concept in the network is activated with energy depending on the initially activated concept "print" (accumulated activation is visualized by the concepts' border width).

5 PageRank-based Analysis

The graph representation of the domain model forms the basis for the second variant of concept-to-concept similarity computation. This approach builds on the algorithms for estimating relative importance in networks [12]. Such algorithms are used to compute the quantitative measure of node similarity with respect to a given node (or set of nodes). We employ PageRank with Priors in particular, successfully used in categorization systems [8]. We modify Diedrich's and Balke's core idea to allow for the inclusion of weights between learning objects and concepts. The approach principle lies in the propagation of actual domain model topology characteristics into the explicit links between concepts.

The algorithm consists of the following steps:

1. for each concept c_i select concepts connected via exactly one learning object,
2. for concept c_i and selected neighbors compute relation weight $ow_{ij,}$
3. build a temporary domain graph where concepts are nodes and $ow_{i,j}$ weights are assigned to edges,
4. for each concept c_i select the most related co-occurring neighbors,
5. for each concept c_i compute the PageRank scores biasing on the selected neighbors.

For $ow_{i,j}$ computation in step 2 we use the following equation:

$$ow_{i,j} = \sum_{lo_k \in LO} \sqrt{w_{i,k} w_{j,k}} \tag{2}$$

where $ow_{i,j}$ is the relation weight between concepts c_i and c_j, and $w_{i,k}$ is the weight assigned to the association between the concept c_i and learning object lo_k. The idea is that weights are considered to be probabilities that the concept is related to learning object. The resulting probability of concepts' similarity is the mean of individual probabilities.

After building a graph in step 3, the most related co-occurring neighbors are determined in step 4. The most related neighbors we consider neighbors that accumulate top-k % of the sum of all neighbors' similarity scores to a given concept.

In step 5 we use the PageRank analysis algorithm (concepts and temporal links in between are analogical to the Web) to adjust the graph and compute the prestige of nodes. PageRank scores represent similarity $sim^{prp}{}_{i,j}$ towards a biased node set with regard to relations within the whole graph. The sorted set of all graph concepts is subject to top-k selection step in order to obtain only relevant relationships.

6 Evaluation in the programming learning domain

The proposed method we evaluated in the domain of programming learning. For an experiment we used Functional programming course being lectured at the Slovak University of Technology in Bratislava. The Functional programming course is a half-term course consisting of 70 learning objects on the functional programming paradigm and programming techniques in the Lisp language. The learning material is hierarchically organized into chapters and sections according to a textbook used in the course. Learning objects are represented using the DocBook markup language enabling easy processing.

The method results we evaluated against manually constructed functional programming concept map. The course lecturer together with randomly chosen sample of 2007/08 course students were involved. Manual creation of concept map comprised the assignment of weighted values to concept relationships. As assigning continuous values from interval <0; 1> is non-trivial task, possible weight values were limited to set {0, 0.5, 1} implying:

- 0 – concepts are not related to each other (no relation),
- 0.5 – concepts are partially (maybe) related to each other (weak relation),
- 1 – concepts are highly (certainly) related to each other (strong relation).

There were 366 relationships created, 216 were weak relations while 150 were strong relations.

In first step of experiment we preprocessed learning objects and composed their bag-of-words term vector representation. The frequency of the terms presented as domain keywords in the textbook index was boosted. We extracted concepts as most frequent terms and their normalized tf-idf values we used as weights for concept-to-learning object associations. Hereby we executed all necessary steps prior to concept relationship discovery.

After preprocessing we separately applied both proposed method alternatives and obtained concept relationships. The relationships were compared to manually constructed reference concept map using well-known *precision* and *recall* measures and their harmonized mean, the *F-measure*. In order to gain more accurate evaluation, we extended the original recall measure to involve the manually constructed domain model relationship types:

$$R^* = \frac{|\, retrieved \cap (correctA \cup correctB)\,|}{|\, correctA \cup (correctB \cap retrieved)\,|} \tag{3}$$

where *R** is the extended recall measure, *retrieved* is the set of all relationships retrieved by the method, *correctA* is the set of manually created “strong” relationships with weight 1.0 and *correctB* is the set of manually created “weak” relationships with weight 0.5.

The purpose of equation (3) is to take into consideration the fact that “weak” relationships need not necessarily be the part of a domain model. Table 1 sums up the results of the performed experiments.

Table 1. Experimental results. The F* contains the harmonized mean using R* recall.

Variant	P	R	**F**	R*	**F***
Spreading Activation	0.544	0.443	**0.488**	0.784	**0.566**
PageRank-based Analysis	0.501	0.569	**0.532**	0.741	**0.652**

The results show that performance of PageRank-based approach is better than spreading activation. This result is evidence that PageRank-based analysis enables more precise processing of the underlying graph representation.

The resulted F/F* measure we interpret as “completeness” of generated concept map. However, generated relationships not contained in the manually constructed concept space were all considered incorrect that should not reflect the reality. Though manual relationship creators made their best effort to match real-world relations, relationships retrieved automatically need not to be irrelevant. They might represent bindings, which were not explicitly realized even by the most concerned authors. As the proposed method goal is to serve as assistant to a teacher, the amount and accuracy of recommended relationships can be considered helpful.

7 Conclusions

In this paper we presented a method of automatic concept relationship discovery for an adaptive e-course. The method is applied as a step in the process of an adaptive e-course authoring and its goal is to help teacher and contribute to overall authoring automation. We proposed and evaluated two variants of the concept score similarity computation representing two approaches to domain model graph processing. We found that our PageRank-based variant achieves better results and identifies more correct relationships against manually constructed concept map.

The main contribution of this paper is a novel approach to adaptive e-course authoring automation. No similar approaches relying on domain model processing and yielding similar results (F*-metrics 65.2%) are applied in the field of adaptive e-learning. The method we propose is independent "component" of a course authoring process. It has clearly defined interface and does not depend on preprocessing techniques. Prior to relationship discovery the concept extraction and weight assignment techniques may vary: various IR methods can be employed or manual annotations can be used. Both approaches may be eventually combined, which seems reasonable especially when considering social and collaborative aspects of e-course authoring. The method addresses the specifics of e-learning domain and does not rely on external resource presence like similar approaches do. Moreover, it is not dependent on the language of an e-course.

As the results of evaluation seem promising, we currently work on more complex evaluation in a real-world environment to obtain even more objective feedback on discovered relationships accuracy and relevancy during functional programming learning.

The further advantage of our method is that although the variants are targeted at the e-learning domain, they are not limited to it – the presented computations are also applicable to different environments. A similar situation with acute "metadata" need is on the Web. Concept maps constructed over the Web pages should in first step serve as backbone for development of richer semantic descriptions. Involvement of social annotations or folksonomies shifts our method applicability even further.

Acknowledgements

This work was partially supported by the Cultural and Educational Grant Agency of the Slovak Republic, grant No. KEGA 3/5187/07.

References

[1] Brusilovsky, P. Developing adaptive educational hypermedia systems: From design models to authoring tools. In T. Murray, S. Blessing and S. Ainsworth (eds.): *Authoring Tools for Advanced Technology Learning Environment.* Dordrecht: Kluwer Academic Publishers, pp. 377–409.

[2] Buitelaar, P., Olejnik, D., and Sintek, M. A protégé plug-in for ontology extraction from text based on linguistic analysis. In *Proc. of the 1st European Semantic Web Symposium* (ESWS), 2004.

[3] Ceglowsky, M., Coburn, A., Cuadrado, J. *Semantic Search of Unstructured Data using Contextual Network Graphs*. 2003.

[4] Cimiano, P., Hotho, A., Staab, S. Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. In *JAIR - Journal of AI Research*, vol. 24, pp. 305–339, 2005.

[5] Cimiano, P., et al. Learning Taxonomic Relations from Heterogeneous Evidence. In *Proc. of ECAI Workshop on Ontology Learning and Population*, 2004.

[6] Cristea, A. I., de Mooij, A. Designer Adaptation in Adaptive Hypermedia. In *Proc. of Int. Conf. on Information Technology: Computers and Communications* ITCC'03. Las Vegas, 2003. IEEE Computer Society.

[7] Dicheva D., Dichev C. Helping Courseware Authors to Build Ontologies: the Case of TM4L. In *13th Int. Conf. on Artificial Intelligence in Education*, AI-ED 2007, July 9-13, 2007, LA, California, pp. 77–84.

[8] Diedrich, J., Balke, W-T. The Semantic GrowBag Algorithm: Automatically Deriving Categorization Systems. In *Proc. of the 11th European Conf. on Research and Advanced Technology for Digital Libraries*, ECDL 2007, Budapest, Hungary, 2007, pp 1-13.

[9] Fortuna, B., Grobelnik, M., Mladenic, D. Semi-automatic Construction of Topic Ontology. In *Semantics, Web and Mining, Joint Int. Workshop*, EWMF 2005 and KDO 2005, Porto, Portugal, October 3-7, 2005.

[10] Maedche, A., Staab, S. Ontology Learning for the Semantic Web, In *IEEE Intelligent Systems*, Vol. 16, No. 2, pp. 72–79, 2001.

[11] Šimún, M., Andrejko, A., Bieliková, M. Maintenance of Learner's Characteristics by Spreading a Change. In Kendall, M., Samways, B. (eds.). *IFIP Int. Federation for Information Processing*, Vol. 281, Learning to Live in the Knowledge Society, Boston: Springer, 2008. pp. 223–226.

[12] White, S., Smith, P. Algorithms for estimating relative importance in networks. In *Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining*. ACM Press, 2003, pp. 266–275.

Unsupervised MDP Value Selection for Automating ITS Capabilities

John Stamper[1] and Tiffany Barnes[2]
[1]john@stamper.org, [2]Tiffany.Barnes@gmail.com
Department of Computer Science, University of North Carolina at Charlotte

Abstract. We seek to simplify the creation of intelligent tutors by using student data acquired from standard computer aided instruction (CAI) in conjunction with educational data mining methods to automatically generate adaptive hints. In our previous work, we have automatically generated hints for logic tutoring by constructing a Markov Decision Process (MDP) that holds and rates historical student work for automatic selection of the best prior cases for hint generation. This method has promise for domain-independent use, but requires that correct solutions be assigned high positive values by the CAI or an expert. In this research we propose a novel method for assigning prior values to student work that depends only on frequency of occurrence for the component steps, and compare how these values impact automatic hint generation when compared to our MDP approach. Our results show that the utility metric outperforms a classic MDP solution in selecting hints in logic. We believe this method will be particularly useful for automatic hint generation for ill-defined domains.

1 Introduction

Our goal is to simplify the creation of intelligent tutoring systems (ITSs) by augmenting existing computer aided instruction (CAI) with intelligent behaviors, such as adaptive feedback and help, derived using educational data mining on CAI data. In our previous work, we have shown that we can successfully generate appropriate context-specific hints for a logic tutor using a Markov decision process (MDP) built from historical student data [2]. The core element of this work that makes automated hint generation possible is the assignment of relative values or “rewards” to each step in a problem solution. We have proposed that alternate assignment values may allow for hints tailored to specific student needs or readiness to learn. As in a recommender system that makes purchase suggestion based on frequent behaviors, we believe that hints generated based on the frequency of a particular step represent those that the majority of students would understand and be able to apply. Vygotsky’s theory of the zone of proximal development [19] states that students are able to learn new things that are closest to what they already know. Presumably, frequent actions could be those that more students feel fluent using. Therefore, paths based on typical student behavior may be more helpful than optimal or expert solutions, which may be above a student’s current ability to understand.

Based on this idea, and the observation that our MDP method sometimes generates a hint that a typical student would not do, or one that is technically correct but was not necessary to the problem solution, we hypothesized that we may be able to generate hints based on “usefulness” and frequency for a particular step in a student’s attempt. Currently, when we construct our MDP, we connect all correct student attempts to a synthetic goal state, assign this state a high value and errors negative values, and rely on value iteration to assign high values to states that are close to the goal, and lower values to those further away. Using this approach results in high values for expert-like solutions, which are short and have few errors. However, in a few instances our tutor gives hints that suggest a less popular path with additional, unnecessary steps. Close inspection of the MDP showed that the states derived from a single, error-free student's solution could get higher values than a more popular route, but that had a significant number of errors. A metric that more heavily emphasizes frequency can mitigate this issue.

The overall goal of our work is to derive domain-independent ways to add intelligence to tutors. However, our prior approach requires that we can label all data as correct or incorrect. In the logic proofs domain, this is simple but in other domains, especially ill-defined domains, hand grading of all student solutions might be required. For example, it is often difficult to determine if a computer program is complete and correct, but it is possible to extract features that many attempts contain, such as variables or loop structures. It seems reasonable to propose that the more student attempts that contain a particular feature, the more likely it is that this feature is a necessary part of a correct program. To lay the foundation for hint generation in such ill-defined domains, we performed an experiment to verify that we could use an unsupervised utility metric to label and value states in an MDP for logic. We hypothesized that this metric would result in similar hints in the logic domain to those we derive using our MDP method.

2 Background and Related Work

The most successful intelligent tutors require the construction of complex models (of knowledge or constraints) that are applicable only to a specific tutorial in a specific field, requiring the time of experts to create and test. It takes between 100-1000 work hours to create 1 hour of content for an intelligent tutor [16]. In order to bring the benefits of intelligent tutors to a wider audience, we must find a way to simplify their creation. One approach is to use generalized authoring tools to simplify the creation of intelligent tutoring systems. Two of the most widely-known authoring tools, including CTAT [11] for building cognitive tutors and ASPIRE [15] for building constraint based tutors, have been used to successfully create and deploy new tutors. Yet, both of these authoring tools discount the tremendous amount of CAI that already exists and require the construction of new tutors. In our work, we have shown that it is possible to provide intelligent, context specific-hints through educational data mining, that allows us to augment existing CAI with the intelligent behaviors found in other tutoring systems.

There are several ways that researchers have proposed to simplify the creation and improvement of intelligent tutors. CTAT has used demonstrated examples to learn production rules that are problem-solving models for cognitive tutors [1]. For CTAT example-based tutors, teachers work problems in what they predict to be frequent correct and incorrect approaches, and then annotate the learned rules with appropriate hints and feedback. This system has also been used with data to build initial models for an ITS, in an approach called Bootstrapping Novice Data (BND) [13]. However, in both of these approaches, considerable time must still be spent in identifying student approaches and creating appropriate hints. Machine learning has also been used not only to build but also to improve tutoring systems. In the ADVISOR tutor, machine learning was used to build student models that could predict the amount of time students took to solve arithmetic problems, and to adapt instruction to minimize this time while meeting teacher-set instructional goals [5]. In the Logic-ITA tutor, student data was mined to create hints to warn students when they were likely to make mistakes using their current approach [14].

Our research uses past student data to generate Markov Decision Processes (MDPs) that assign numerical values to every state reached by past students solving a problem in an existing CAI. Using these values, we can estimate for any problem state what the “best” next step that any student has taken from the current problem state in the past. Our method of automatic hint generation using previous student data reduces the expert knowledge needed to generate intelligent, context-dependent hints [2], and allows for visualization of student approaches to problem solving [8]. The system is capable of continued refinement as new data is provided. In [2] we performed a feasibility study for hint generation using historical student data, and found that we could have made hints available for 71% of past student steps using one semester of past data. Our results indicated valuable tradeoffs between hint specificity and the amount of data used to create an MDP. In [3], we discussed how we added the method to existing CAI used to teach logic and reported the results of our initial pilot study.

Ill-defined domains, such as medical diagnosis, computer programming and legal reasoning, pose particular problems for ITS developers [12]. In particular, it is difficult to generate feedback for environments where there are many possible ways to solve a problem. Sequential Pattern Matching (SPM) [17] is a data-mining method used in ill-defined domains to extract frequent actions into plans. SPM has been used in a tutor to teach astronauts to use a robotic arm, where the tutor suggested a plan based on their current location in the problem. Like our approach, this method only uses good solutions and takes into account how often different actions occur, but this is specific to the robotic arm control domain.

3 Method

A Markov decision process (MDP) is defined by its state set S, action set A, transition probabilities P, and a reward function R [18]. For a particular point in a student attempt, our method takes the current problem features as the state, and the student’s input as the action. Therefore, each student problem attempt can be seen as a graph, or Markov chain, with a sequence of states (each describing the solution up to the current point), connected by actions. We combine all student solution graphs into a single graph, by taking the union of all states and actions, and mapping identical states to one another. Once this graph is constructed, it represents all of the paths students have taken in working a particular problem. Typically, at this step value iteration is used to find an optimal solution to the MDP. A large initial value is set for the goal state, penalties for incorrect states, and a transition cost for taking each action. Setting a non-zero cost on actions causes the MDP to penalize longer solutions. We apply value iteration to assign reward values to all states in the MDP [18]. Once this is complete, the optimal solution corresponds to taking a greedy traversal approach in the MDP [4]. The reward values for each state then indicate how close to the goal a state is, while probabilities of each transition reveal the frequency of taking a certain action in a certain state.

In our original MDP method, all paths which solved the problem were directed to a goal state which was given a high reward value. The use of a single goal state works well when we know whether each student attempt is correct. Our new utility metric determines the “goodness” of a state by based on the frequency of each component step in the state. Unlike our original method where the goal state was known, the utility method has no known goal states so all terminal states are treated as possible goals. Terminal states are defined as those that are not errors, where no subsequent student actions were taken.

We derive our utility metric using techniques related to Latent Semantic Indexing (LSI), which are used to search large databases of text documents [11]. In LSI, terms refer to words, while for logic proofs, we define a term, or feature, as the statement a student derives in a single problem-solving step. Therefore, each attempt is composed of a sequence of statements. As in LSI, we use a term-document matrix, as shown in Table 2, to show the occurrence of each statement or term in each student attempt, marking a 1 for terms that occur and 0 that do not occur. We then compute the frequency by summing the columns. We set a percentage frequency threshold such that all state features above the threshold had a good potential of being a part of the solution. Setting this threshold can be done automatically or with the help of a domain expert. We discuss selection of the threshold in this experiment in section 4.2.

Once a list of frequent statements is determined, we calculate initial utility values for all terminal states (leaves) in the MDP, which are potential goal states. This replaces our original approach of creating a goal state with a single positive value. The utility value of a terminal state is the sum of the value for each statement (or feature) in the student attempt. The value of each step is positive if it was frequent and negative otherwise. Error states receive a high negative start value, and all other states start at zero. After the initial values are set, value iteration is applied until the state values become stable.

4 Experiment

We applied the utility method on a known dataset from CAI used to teach logic. This dataset had been used in previous research and had state values calculated using the MDP method. We compared the results of both methods paying special attention to states that had different best values.

4.1 Data

We have used the NCSU-Proof1 dataset in [2] and [4]. The data comes from four fall semesters of 2003-2006, where an average of 220 students take the discrete math course each year. Students attend several lectures on logic and then use the Proofs Tutorial to solve 10 proofs. Sixty percent of students used direct proof when solving proof 1. We extracted 537 of students’ first attempts at direct solutions to proof 1. An example attempt of proof 1 is shown in Figure 1.

Statement	Line	Reason
1. a → b		Given
2. c → d		Given
3. ¬ (a → d)		Given
¬ a v d	**3**	**rule IM (error)**
4. a ^ ¬ d	3	rule IM implication
5. a	4	rule S simplification
b	**4**	**rule MP (error)**
b	**1**	**rule MP (error)**
6. b	1,5	rule MP modus ponens
7. ¬ d	4	rule S simplification
8. ¬c	2,7	rule MT modus tollens
9. b ^ ¬c	6,8	rule CJ conjunction

Figure 1. Sample student attempt to NCSU Proof 1

The data were validated by hand, by extracting all statements generated by students, and removing those that 1) were false or unjustifiable, or 2) were of improper format. We also remove all student steps using axioms Conjunction, Double Negation, and Commutative, since students are allowed to skip these steps in the tutorial. After cleaning the data, there were 523 attempts at proof 1. Of these, 381 (73%) were complete and 142 (27%) were partial proofs, indicating that most students completed the proof. The average lengths, including errors, were 13 and 10 steps, respectively, for completed and partial proofs. When excluding errors and removed steps, the average number of lines in each student proof is 6.3 steps. The validation process took about 2 hours for an experienced instructor, and could be automated using the existing truth and syntax-checking program in our tutorial. We realized that on rare occasions, errors are not properly detected in the tutorial (less than 10 premises were removed).

Table 1. Sample states derived from example student attempt in Figure 1

State	State Description	Error	Action	Result State
1	a → b, c → d, -(~(a → d)		IM	2
2	a → b, c → d, -(~(a → d), -~a v d	Yes		1
1	a → b, c → d, -(~(a → d)		IM	3
3	a → b, c → d, -(~(a → d), a ^ -~d		S	4
4	a → b, c → d, -(~(a → d), a ^ -~d, a		MP	5
5	a → b, c → d, -(~(a → d), a ^ -~d, a, b	Yes		4
4	a → b, c → d, -(~(a → d), a ^ -~d, a		MP	6
6	a → b, c → d, -(~(a → d), a ^ -~d, a, b		S	7
7	a → b, c → d, -(~(a → d), a ^ -~d, a, b, -~d		MT	8
8	a → b, c → d, -(~(a → d), a ^ -~d, a, b, -~d, -~c		CJ	9
9	a → b, c → d, -(~(a → d), a ^ -~d, a, b, -~d, -~c, b ^ -~c			

An MDP was created from this data using our MDP method resulting in 821 unique states. Table 1 shows the states created in our MDP for the student attempt shown in Figure 1. In the logic proofs domain, a step in the solution is considered to be a new statement added to the previous state. For example, in state 2, the statement ~a v d is the next "step" in the problem, however, since it is an error detected by the software, this statement is deleted and the problem is returned to state 1.

4.2 Utility Process

If our data are labeled, we simply connect all valid solutions to a synthetic goal state. However, when goal states are unknown, we need a way to label or measure correct attempts. Our proposed utility metric is one way that assumes that frequent features are important in the problem solution. From our 523 attempts, we extracted 50 unique statements (including 3 given statements) and calculated their frequencies. A partial sample of the statement-attempts matrix is shown in Table 2. Note that only the first three attempts and only those statements appearing in those three attempts are shown. The complete statements-attempts matrix would contain all 50 statements in rows and all 523 attempts in the columns. To determine statement frequency, we sum each column.

Table 2. Sample matrix showing the occurrence of elements in student solution attempts.

	Terms								
	a → b	c → d	-(a → d)	a ^ -d	a	b	-d	-c	b ^ -c
Attempt 1	1	1	1	1	1	1	1	1	1
Attempt 2	1	1	1	0	0	1	1	1	1
Attempt 3	1	1	1	1	0	0	1	0	0

We then graphed the frequency of each statement, and the frequencies of statements (number 1-47) with more than 1 usage are shown in Figure 2. Statements 1-22 occurred only once in the data, while statements 43-47 occur in over 370 unique student attempts. Since there is variation in correct solutions, we set a low threshold frequency of 8 attempts for statements we might consider "useful" in a proof, and this is true for statements 29-47 and higher. A logic instructor verified that all the statements 29-47could be expected to occur in correct student solutions, while those with fewer were not as useful. The threshold value could be chosen automatically using the frequency profile.

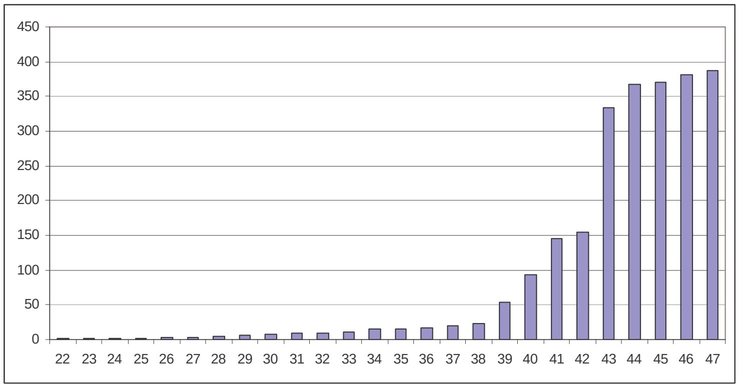


Figure 2. Frequency of Statements in Proof 1

Next we calculate the initial values for MDP states. For the possible goal states (valid terminal states), the initial value was a sum of the individual scores given to the component statements. Each statement score was +5 if its frequency was above the threshold and was -1 for those below. Error states received a value of -2, and all other states started at zero. Finally, after the initial values were set we ran a value iteration algorithm until the state values stabilized. Note that during value iteration, a -1 transaction cost was associated with each action taken.

4.3 Comparing Utility Method to MDP Method

We use an MDP along with its state values to generate hints that provide students with details of the best next state reachable from their current state [3]. To compare the utility method to our traditional MDP method we compared the effects of state values on the choice of the "best" next state. Both methods create the same 821 states, of which 384 were valid, non-error states. From the valid states, 180 states had more than one action resulting in new state. These 180 states are the ones that we focused on since these are the only states that could lead to different hints between the two methods. Comparing the two methods, they agree on the next best state in 163 states out of 180 (90.56%). For the remaining 17 states where the two methods disagreed, experts identified 4 states where the MDP method identified the better choice, 9 states where the utility method identified the better choice, and 4 states where the methods were essentially equivalent. These 17 states can be seen in Table 3, with the highlighted cells marking the expert choice.

Table 3. States where the methods disagree (17 total states)

State	State Description	# of Possible Actions	MDP next State	MDP added Statement	MDP Value	Utility Next State	Utility added Stmt	Utility Value
1	a>b,c>d,-(a>d)	14	53	-d>-c	49.91	2	-(-a+d)	10.57
2	a>b,c>d,-(a>d),-(-a+d)	9	238	b	98.00	579	(a*-d)	14.00
3	a>b,c>d,-(a>d),-(-a+d),a*-d	8	310	-(a*-b)	93.00	310	-(a*-b)	29.00
4	a>b,c>d,-(a>d),-(-a+d),a*-d,b	4	5	-c	87.72	119	-d>-c	38.74
7	a>b,c>d,-(a>d),-a+b	6	780	-d>-c	29.00	780	-d>-c	18.00
8	a>b,c>d,-(a>d),-a+b,-c+d	2	599	b+-c	99.00	10	-(-a+d)	18.02
19	a>b,c>d,-(a>d),-(-d>-a)	2	20	-(d+-a)	27.13	274	a*-d	7.67
36	a>b,c>d,-(a>d),-c+d,-(-a+d),a*-d	2	170	-c	24.33	186	b	6.04
53	a>b,c>d,-(a>d),-d>-c	5	460	-(-a+d)	96.00	684	-b>-a	21.00
82	a>b,c>d,-(a>d),(a*-d),-c	3	84	b	99.00	320	-(a*-b)	14.00
91	a>b,c>d,-(a>d),-(-a+d),a*-d,-d>-c	3	92	(a*-d)>(b*-c)	99.00	473	b	19.33
119	a>b,c>d,-(a>d),-(-a+d),a*-d,b,-d>-c	3	773	-c+d	98.00	120	-c	42.71
156	a>b,c>d,-(a>d),-(-a+d),a*-d,-a+b	2	208	-d>-c	98.00	423	b	29.60
228	a>b,c>d,-(a>d),a*-d,-d>-c	2	288	-c	76.20	619	b	14.00
333	a>b,c>d,-(a>d),a*-d,-c+d	2	334	-a+b	99.00	785	-c	19.00
337	a>b,c>d,-(a>d),-a+b,-(-a+d),a*-d,b	2	646	-c+d	61.67	339	-c	20.20
522	a>b,c>d,-(a>d),-(-a+d),a*-d,b,-c+d,-d>-c	2	766	-c	99.00	523	-c+d	30.00

These results show that the unsupervised utility metric does at least as good a job as the traditional MDP method in determining state values even when it is not known if the student attempt was successful. In all cases, the hints that would be delivered with either method would be helpful and appropriate. We believe that the utility metric provides a strong way to bias our hint selection toward statements derived by a majority of students, which may give students hints at a more appropriate level.

Before we derived the utility metric presented here, we considered modifying MDP values by combining them in a weighted sum with a utility factor after value iteration had been completed. In our first attempt to integrate frequency and usefulness into a single metric, we analyzed all of our attempts to find derived statements that were necessary to complete the proof, by doing a recursive search for reference lines starting from the conclusion back through a student's proof. For each attempt, this "used again" value was set to 1 if a derived statement could be reached backward from the goal, and zero otherwise. We summed the total times a statement was used again, and compared this with the total times a statement occurred in attempts. Table 4 shows the comparison of the frequency and used again values for all statements where used again was more than 1. The values have no real correlation, but most items that were used again had high (>7) frequencies, so we decided that frequency was a relatively good indictor of usefulness in the logic proof domain. The "used again" calculation is possible in the logic domain because students must provide a justification for the current statement using rules and references to prior statements. In other domains, this may not be possible but we believe that frequency of occurrence in student solutions indicates that a step is either needed, or is a very common step that will only skew state values in a consistent way.

Table 4. Comparison of frequency and used again

Statement Number	Statement	Frequency	Used Again
30	(a+c)>(b+d)	8	2
31	-(a*c)+(b*d)	9	2
32	-(d+-a)	9	7
33	(a*-d)>(b*-c)	10	10
34	-(-d>-a)	15	7
35	-b>-a	16	5
36	-(c*-d)	17	6
37	(a*c)>(b*d)	20	4
38	-(a*-b)	23	8
39	(a*-d)	53	44
40	-d>-c	93	71
41	-a+b	145	69
42	-c+d	155	80
43	-(-a+d)	334	300
44	-c	367	344

5 Conclusion and Future Work

The most important feature of the MDP method is the ability to assign a "value" to the states. This allows the tutor to identify the action that will lead to the next state with the highest value. In this research we have shown that the utility metric that assigns values to terminal states based on the component steps in the state can be used to achieve hint-source decisions as one that assigns a single value to all goal states.

The main contribution of this paper is to show how this new utility metric can be used to generate MDP values based on features of student solution attempts. Our results show that the utility metric could be used to achieve equivalent or better hints than our prior single-goal MDP approach. This is significant because the utility metric does not require a known goal state, so it can be applied in domains where the correctness of the student attempts is unknown, or difficult or costly to compute. We believe that this utility metric combined with our MDP method can be used to generate hints for a computer programming tutor. In this domain, it is difficult to say that a program is complete, but it is possible to say whether specific features are represented. The method of using a term-document matrix to determine utility could also be extended into using more complicated LSI techniques which would be a natural fit for tutors using textual answers such as essay response questions. Text based answers are prevalent in legal reasoning and medical diagnosis tutors.

In our future work, we plan to construct and compare traditional and utility-based MDPs for other proofs and for student work in other domains. We also plan to analyze our logic tutor hint data to see if the utility method would result in different hints. This will give an indication of how much the utility technique is needed for our logic tutor. We also plan to analyze log data compiled from a C++ programming course to determine what kind of features we might extract and how well we can calculate the utility of those features.

References

[1] Aleven, V., McLaren, B. M., Sewall, J., & Koedinger, K. (2006). The Cognitive Tutor Authoring Tools (CTAT): Preliminary evaluation of efficiency gains. In M. Ikeda, K. D. Ashley, & T. W. Chan (Eds.), *Intelligent Tutoring Systems* (ITS 2006), (pp. 61-70). Berlin: Springer Verlag.

[2] Barnes, T., Stamper, J. (2008). Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data. In E. Aimeur, & B. Woolf (Eds.) *Intelligent Tutoring Systems (ITS 2008)*, pp. 373-382. Berlin, Germany: Springer Verlag.

[3] Barnes, T., Stamper, J., Croy, M., Lehman, L. (2008). A Pilot Study on Logic Proof Tutoring Using Hints Generated from Historical Student Data. In R. Baker, T. Barnes, J. Beck (Eds.) *Educational Data Mining (EDM 2008)*, pp. 197-201. Montreal, Canada.

[4] Barnes, T. and Stamper, J. Toward the extraction of production rules for solving logic proofs, In *Artificial Intelligence in Education, Educational Data Mining Workshop (AIED2007)*, pp. 11-20. Los Angeles, CA.

[5] Beck, J., Woolf, B. P., and Beal, C. R. ADVISOR: A Machine Learning Architecture for Intelligent Tutor Construction. In: *7th National Conference on Artificial Intelligence*, pp. 552—557. AAAI Press / The MIT Press, 2000.

[6] Conati, C. Gertner, A. and VanLehn, K. (2002). Using Bayesian Networks to Manage Uncertainty in Student Modeling. In User Model. User-Adapt. Interact, vol. 12 (4).

[7] Croy, M. Graphic Interface Design and Deductive Proof Construction, *Journal of Computers in Mathematics and Science Teaching*, 1999, 18(4), 371-386.

[8] Croy, M., Barnes, T., and Stamper, J. Towards an Intelligent Tutoring System for propositional proof construction. In P. Brey, A. Briggle & K. Waelbers (eds.), *Proc. 2007 European Computing & Philosophy Conf.*, Amsterdam: IOS Publishers.

[9] Heffernan, N. and Koedinger, K.(2002). An Intelligent Tutoring System Incorporating a Model of an Experienced Human Tutor. In *Intelligent Tutoring Systems*, 596–608.

[10] Koedinger, K., Aleven, V., Heffernan. T., McLaren, B. & Hockenberry, M. Opening the door to non-programmers: Authoring intelligent tutor behavior by demonstration. *Intelligent Tutoring Systems 2004*.

[11] Landauer, T. K., Foltz, P. W., and Laham, D. (1998). Introduction to Latent Semantic Analysis. *Discourse Processes*, 25, 259-284.

[12] Lynch, C., Ashley, K., Aleven, V., & Pinkwart, N. (2006). Defining Ill-Defined Domains; A literature survey. In V. Aleven, K. Ashley, C. Lynch, & N. Pinkwart (Eds.), *Workshop on Intelligent Tutoring Systems for Ill-Defined Domains at the 8th International Conference on Intelligent Tutoring Systems* (p. 1-10).

[13] McLaren, B., Koedinger, K., Schneider, M., Harrer, A., & Bollen, L. Bootstrapping Novice Data: Semi-automated tutor authoring using student log files, In Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes,. *Intelligent Tutoring Systems (ITS-2004),* Maceió, Brazil, August 30, 2004.

[14] Merceron, A. & Yacef, K.: Educational Data Mining: a Case Study. In *Artificial Intelligence in Education*, Amsterdam, Netherlands, IOS Press, 2005.

[16] Murray, Tom. Authoring intelligent tutoring systems: An analysis of the state of the art. *Intl. J. Artificial Intelligence in Education,* 1999, 10: 98-129.

[17] Nkambou, R., Mephu Nguifo, E., Fournier-Viger, P.: Using Knowledge Discovery Techniques to Support Tutoring in an Ill-Defined Domain. In *E. Aimeur, & B. Woolf (Eds.) Intelligent Tutoring Systems (ITS 2008),* pp. 395-405. Berlin: Springer Verlag.

[18] Sutton, R. & A. Barto. Reinforcement Learning: An Introduction, 1998, The MIT Press, Cambridge, MA.

[19] Vygotsky, L. (1986). *Thought and language*. Cambridge, MA: MIT Press.

Recommendation in Higher Education Using Data Mining Techniques

César Vialardi[1], Javier Bravo[2], Leila Shafti[2], Álvaro Ortigosa[2]
cvialar@correo.ulima.edu.pe, {Javier.Bravo, Leila.Shafti, Alvaro.Ortigosa}@uam.es
[1]Universidad de Lima
[2]Universidad Autónoma de Madrid

Abstract. One of the main problems faced by university students is to take the right decision in relation to their academic itinerary based on available information (for example courses, schedules, sections, classrooms and professors). In this context, this work proposes the use of a recommendation system based on data mining techniques to help students to take decisions on their academic itineraries. More specifically, it provides support for the student to better choose how many and which courses to enrol on, having as basis the experience of previous students with similar academic achievements. For this purpose, we have analyzed real data corresponding to seven years of student enrolment at the School of System Engineering at Universidad de Lima. Based on this analysis, a recommendation system was developed.

1.-Introduction

A university curriculum is generally flexible. The study program to obtain a university degree is conformed by several courses, which are distributed in academic terms. Prior to the beginning of each term, the student should enrol on one or more courses of which some are compulsory and other optional, corresponding to the period according to his/her progress; the succession of courses enrolled in each term made by a student during his/her career is called the student's academic itinerary. An academic itinerary is successful when the student, after realising his/her successive enrolments, obtains good results in each enrolled courses, allowing thus to finish him with his/her career in the exact time and with good results.

In this work the case of the School of System Engineering at Universidad de Lima was analyzed. In this institution, enrolment is done through a Web system. Even though students with better academic performance have priority on choosing groups, there are enough vacancies for all the students. In this sense, students can enrol on some or in all the courses available for his/her study plan or curriculum (a course is available for a student when he/she satisfied all the requirements for enrolment and is able to take the course).

The enrolment of a student in a course only depends on his/her decision. Previously, the student can require advice from a professor with experience, in order to know, based on his/her academic record, how many and which of the available courses he/she should enrol on. Nevertheless, students rarely require these advices from professors; most of the time the enrolment is based only on the student experience and on the information available.

However, many students do not have enough experience for taking enrolment decisions, as they do not know to associate time, effort and intellectual abilities required to successfully culminate each course. Many times the choosing criteria are closely related to the time required to finish the studies, mainly due to students' maturity. Certainly, the university offers the required quantitative information (available courses, sections, classrooms and professors), but qualitative information (implicit information regarding the experience of previous students) is lost.

Collaborative recommendation systems are agents that suggest options [4] for the user to choose among them. They are based on the idea that individuals with approximately the same profile generally select and/or prefer the same things. The systems are highly accepted and offer good outcomes for a large number of applications.

In the education environment, a recommendation system is an intelligent agent that suggests different alternatives to students, having as starting point previous actions from other students with approximately the same characteristics, such as academic performance and other personal information. It is known that before taking a course, the student have to enrol on the course; the most notorious of this process is not enrolment itself, but the previous decision that has to be taken, mainly related to how many and which course are going to be taken. In this work, we show a collaborative recommendation system based on data mining techniques [7] applied to the educational environment. The aim of this work is to offer students key elements to take better decisions in the enrolment process, using as basis the academic performance of other students with similar profiles, in order to obtain good results in each courses pertaining to its academic itinerary.

For this work we had used data of enrolments since 2002. The data is composed of demographic information of each student, enrolment in courses, grades obtained, number of courses taken at each academic term, average grade and cumulative grade per academic term. After filtering and cleaning the data, we applied the learning algorithm C 4.5[13], obtaining rules that are used for the system to suggest the student if his/her enrolment in certain course has good probabilities of success or not [17]. With this information, students will have a supporting tool that will help them taking the best decisions previous to their enrolment.

Evaluations made on the performance of the rules used by the recommender system show that they are expected to predict correctly student results in approximately 80% of the cases.

The rest of this paper is organized as follows: section 2 gives an overview of related works applying data mining in education environments. In section 3 we describe the recommendation systems and their relation with data mining techniques. In section 4 we had described data processing. In section 5 we explain the sequence of experiments required for this domain. Section 6 shows the analysis of results. Finally, section 7 outlines the conclusions and future work.

2.-Related Work

Data mining techniques are useful when huge amounts of data have to be classified and analyzed [9]. Nowadays, it is a very common situation in many scenarios, such as web information exploitation [16]. In the last years, a number of works have focused on the use of data mining techniques in the context of educational environment [14]. The most widespread techniques are: classification algorithms [3] and association rules [2]. Although the interest for using data mining in this context is growing, little work has been done regarding the use of these techniques in education.

The use of data mining is more common in educational environments based on e-Learning, for instance Educational Adaptive Hypermedia (EAH) courses. These techniques are used to discover the patterns used by students in web courses, thus helping professors and students to optimize the use of such systems. In this sense [5] these techniques support the improvement of EAH courses, applying decision tree analysis, finding the most relevant branches of the tree afterwards. These branches are presented to the professor in order to improve the course design.

Many authors have also researched the application of data mining techniques to Recommender Systems. In [1], several examples where data mining techniques are used to learn a user model (based on previous ratings) and classify unseen items are explained. Recommendation systems link users with items [15], associating the content of the recommended item or opinion of other individuals with the actions or opinions of the original users of the system. Recommendation techniques are classified in three different categories [12]: Rule-based Filtering System, Content-filtering System and Collaborative Filtering System.

Many and diverse algorithms can be applied to recommendation systems [1]. The Rule-based Filtering Systems are based on classic filtering techniques, which are information search and retrieve. Differently, collaborative filtering systems use *classification* [3], *clustering* [11], *association* [2] and *sequential patterns* [10] to discover new and interesting models that can help to suggest recommendations based on different user profiles. These systems used for educational environments has been used based on decision making have had little research or development as means to help students in taking decisions. Our system includes courses taken by the students, their grades, the registered courses in the semester, and the grade point average before registering in the course. Data mining techniques are used, in our case, as a basis for the system, since it provides precise recommendations to the students in relation to their academic performance. A similar idea is used in [6], in this case they will introduce *OrieB*, a *CRS* working in the Academic Orientation domain, to support advisors helping students of secondary school to make decisions about their academic future. *OrieB* utilizes the students' grades as input data in order to suggest their academic possibilities by providing qualitative recommendations based on the fuzzy linguistic approach.

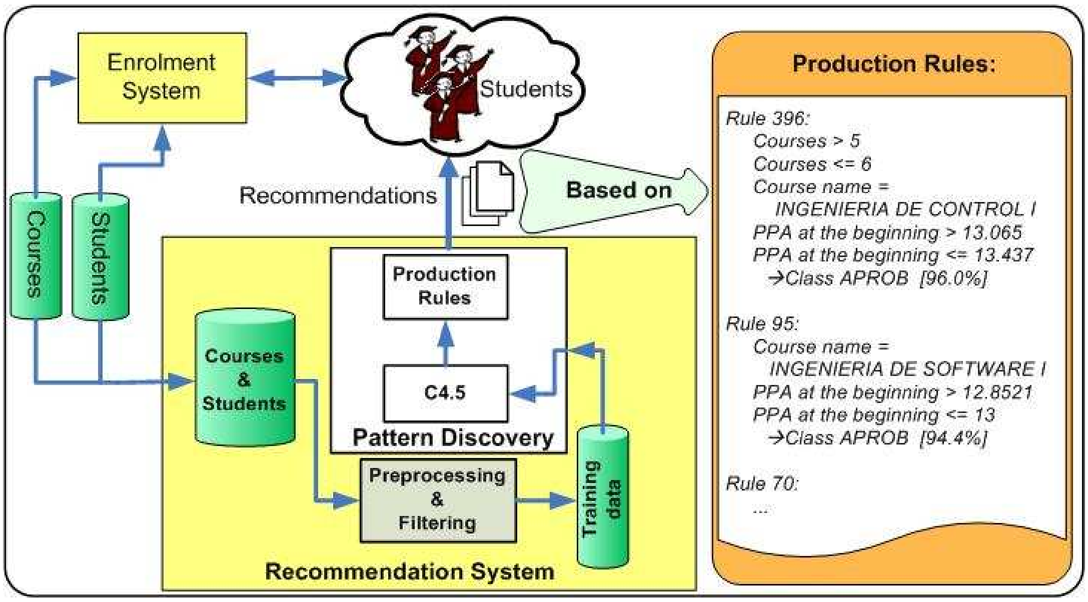
3.-Recommendation using Data Mining

The objective of the system is to predict how convenient it is for a particular student to take a specific course, using as basis results obtained by other students with similar profile who had taken that course. To achieve this, data was organized in a table; each row represents data from a student and a course. In this way, if a certain student had taken C courses, in the table there will be C records with data about this student.

The result of the layout of the data is an *m x n* table (m records and n attributes) students-attribute; the columns have data of when the student took the course: number of courses taken simultaneously, name of course, grade obtained and accumulated grade point average (GPA) of the student until the previous semester. The class to be considered in the application of the supervised learning algorithm is the grade. This has been discretized following current norms of the institution: failure (from 0 to 10.99) and success (from 11.00 to 20).

Likewise, as the number of courses per curriculum is limited, there are not scalability and dispersion problems inherent to this type of representation in traditional collaborative filtering systems [4].

Figure 1 shows the architecture of the Recommendation System in the context of the enrolment system. Firstly, the recommendation system uses the data of the historical database of students and results obtained by them, with the goal of obtaining the rules. These rules are generated by the Pattern Discovery module. In this module, the C4.5 sub-module uses the training data (Pre-processing and Filtering prepares the training data) as an input for generating the decision tree. This tree is used by the Production Rules to generate the rules. Finally, the system provides recommendations based on these rules. An example of a set of production rules is exhibited on the left side of figure 1.



Rule 396:
Courses > 5
Courses <= 6
Course name =
INGENIERIA DE CONTROL I
PPA at the beginning > 13.065
PPA at the beginning <= 13.437
→Class APROB [96.0%]

Rule 95:
Course name =
INGENIERIA DE SOFTWARE I
PPA at the beginning > 12.8521
PPA at the beginning <= 13
→Class APROB [94.4%]

Rule 70:
...

Fig. 1 Recommendation system architecture

4.- Data pre-processing

In the entire data mining process, it is of great relevance the data cleaning process in order to eliminate irrelevant items. The discovery of patterns will be only useful if the data represented in files offers a real representation of the academic performance and the actions and/or decisions taken by the student in the past [8].

Initially, the University provided us a database of 100274 records corresponding to 3230 students. After filtering process of the data 58871 records were left. The data supplied is only from students at the School of Systems Engineering, enrolled through the years 2002 to 2008.

The main objective of our research is to discover patterns that will be used to suggest positive or negative recommendations to a student previously to his/her enrolment at given course, taking as basis grades from other students with similar academic yields. In this sense, knowing the role of each attribute and the implicit relations among them, we have decided to consider that automatic learning will be performed with attributes Courses enrolled in, Name of course, Accumulative GPA starting the term, Grade

5.- Pattern extraction and evaluation

The objective is to develop a system able to predict the failure or success of a student in a course using a classifier obtained from the analysis of a set of historical data related to the academic yield of other students, who took the same course in the past.

We tried out several techniques and configurations, seeking classifiers models to optimize predictions about students' outcomes. Of these tries, two distinctive factors of the work were taken into consideration:

- The method used to learn the classifier should consider that on real situations, conditions could change from year to year and the classifier could reflect "old" patterns. For example, recommendations made on the first term of academic year 2009 will be done using historical data until 2008.

- As with any learnt classifier, it is expected to have a percentage of error in the predictions. It will be worst to recommend a student to enrol in a course that he/she will not pass, than recommend not to enrol in a course that he/she would pass.

These criteria lead the configuration of trials performed to determine patterns for a correct prediction of academic performance. Four assays were performed; following is a description of them. As it was mentioned, these assays were performed in 58871 records after filtering and cleaning the data, corresponding to 2867 students who carried out their enrolment between the years 2002 and 2008.

The first trial was the application of the algorithm C4.5[13] with the training set. This set corresponds to instances between 2002 and 2007, that is to say, 50488 instances were analyzed. It is worth to mentioning that instances representing enrolments during 2008 (8383 instances) were taken as unseen instances. For this reason, they are used to test the accuracy of the model. The model obtained in this trial has the number of false positives significantly greater than the number of false negatives. This fact is crucial for this research since the knowledge of experts in this domain indicates that this recommender system would be useful if the false positives are lower than false negatives. In other words, the number of false positives is more important than the false negatives. Although the accuracy of the resulting model was high (more than 80%), a second trial was needed.

The second assay consisted of using the same algorithm with the same previous conditions, but using re-sampling on the training data set. On this way, the class distribution of the instances was biased towards a uniform distribution. The goal was to increases the chances of negative prediction, even if it would imply a worse performance of the classifier. Luckily the result of this assay showed that false positives were decremented and the accuracy was only 5% less than the accuracy of the previous trial.

The validation of the model was made in the third assay by executing C4.5 algorithm with a supplied experiment set (unseen instances). It is worth to noting that using enrolment data from year 2008 to test the classifier model learnt with data between 2002 and 2007 replicates how the model will be used in real situations: old data is used to predict outcomes of new students. This trial demonstrated that the accuracy of the model is enough to consider it adequate to predict the success or failure in a course by a student.

Finally, the algorithm C4.5 was applied to the entire set of data, from 2002 to 2008, using re-sampling. As a result, patterns that will be effective for the recommendation system were obtained.

6.- Analysis of Results

The analysis of the results of the assays, as well as the main statistics of the prediction model, is explained in this section.

6.1. - First assay

Using the first classifier, 41086 instances were correctly classified (81.38%), and 9402 were incorrectly classified (18.62%). The confusion matrix showed that false positives (students that failed but were classified as passing) were 7217, representing 76.76% of wrongly classified and 14.29% of total of classified students. False negatives (students that passed but were classified as failing) were 2185, representing 23.24% of wrongly classified and 4.32% of the total of classified students.

As the most important in a recommendation system is the effectiveness it achieves with users and giving the particular domain, the experts in the domain consider that in this first assay the value of representing false positives is too high in relation with the false negatives.

6.2.- Second assay

The second model classified correctly 62470 instances (75.80%). This percentage means that accuracy in this trial was lower than in the first one, but it was still adequate. The instances incorrectly classified were 19944 representing 24.20%. The confusion matrix displays that the false positives were 7262 representing 36.41% of incorrectly classified and 8.81% of total classified students. False negatives were 12682, representing 63.59% of incorrectly classified and 15.39% of total classified students.

6.3.- Third assay

The third model classified correctly 6193 instances, representing 73.9%. In this trial accuracy of the model was lower than in the two previous ones. Instances correctly classified were 2190 representing 26.1%; from the confusion matrix, it can be deduced that false positives were 435 representing 19.8% of incorrectly classified and 5.2% of total classified students. False negatives were 1755 representing 80.2% of wrongly classified and 20.4% of total classified students.

6.4.- Fourth assay:

This last trial was performed with the minimum number of items for branches in the C4.5 method set to 20. Table 6 presents a set of interesting production rules.

Table 1. Rules fourth phase results

Rule 198:	Name of course = INTEROPERABILIDAD Y ARQ. DEL SOFTWARE Accumulative GPA starting the term > 10.0256 Accumulative GPA starting the term <= 10.7428 -> class APROB [96.2%]
Rule 396	Courses enrolled in > 5 Courses enrolled in <= 6 Name of course = INGENIERIA DE CONTROL I Accumulative GPA starting the term > 13.0652 Accumulative GPA starting the term <= 13.4371 -> class APROB [96.0%]
Rule 221	Courses enrolled in > 5 Courses enrolled in <= 7 Name of course = DINAMICA DE SISTEMAS Accumulative GPA starting the term > 11.173 Accumulative GPA starting the term <= 11.7333 -> class APROB [95.3%]
Rule 95:	Name of course = INGENIERIA DE SOFTWARE I Accumulative GPA starting the term > 12.8521 Accumulative GPA starting the term <= 13 -> class APROB [94.4%]

7. - Conclusions and future work

In this section some of the main conclusions and contributions of the work are summarized, and some possible future development lines are commented.
As it has been emphasized, the most important point of this research is the acquisition of knowledge from students' academic performance. The main purpose is to provide support

for new students in order they can choose better academic itineraries. Recommendation systems are familiar to this process; this is the technique used in the recommendation engine that stresses accuracy and effectiveness. In this sense, the research has focused in testing, using real data, the application of techniques and tools in data mining to support students when enrolling on a new term. To achieve this goal, we had analyzed real data from students that had taken the same courses in the past, using techniques such as decision trees to discovery trends, patterns and rules that will be used as support for decision taking in the itinerary of a certain university career.

It was found that using data mining, it is possible to develop a model representing the behaviour of students in their way through different academic itineraries. This facilitates a proper vision of the behaviour and performance of the group of students at certain university career and, at the same time, allows feeding the system to offer recommendations for students to increase their effectiveness and relevance at decision taking in relation with the courses to be enrolling on.

We presented four assays by using the same technique with four different configurations, in order to show the advantages and disadvantages of the technique used as well as to detect classifiers that would perform better in real settings. The first trial was a classic 10 fold cross validation over 50488 instances corresponding to enrolments from 2002 to 2007.

However, the same technique was also applied with other setting, penalizing more the error produced by false positives giving more weight to the data corresponding to instances whose class was *failure*. Assay results show that global accuracy was 75.80%. Although global accuracy is lower at this second trial than in the first one, the objective was accomplished, as the number of false positives decreased in relation to false negatives, assuring its effectiveness.

Due to the success of the second assay, a third essay was carried out. The main objective was that the system learns with the record of enrolments made from 2002 to 2007 (training set), considering the set of records representing enrolment in 2008 as the experiment set (unseen set). In this way, we simulated the fact of implementing and testing the recommendation system for students enrolled in 2008.

Finally, a fourth assay was developed, by applying the learning algorithm to all the instances, obtaining 77.3% of global accuracy. The global accuracy of this last trial was greater than the second and the third ones.

It is meaningful to emphasise that assay presented in this work, besides using real data, shows that used tools are very powerful techniques, but also they have weak points when looking for patterns in a domain of knowledge with participation of human factor and has direct relevant consequences in the data.

In addition, pattern detection offers two types of information. On one hand, the student can infer that system’s recommendation is related with his/her global academic performance or to certain courses. Therefore, the student could freely decide, taking into consideration more subjective factors, to enrol or not. For the university the system has relevant information related to students' academic results in one or several courses, information that would not be available applying descriptive statistical techniques. It is worth to mentioning that analyzed information could be used as input in an eventual curriculum modification.

In this way, the main objective of this work is to support students through recommendations, in the complex process of deciding how many and which courses enrol on, taking into consideration academic performance and other similar characteristics. But a second important benefit is that it can offer useful information to meaningfully improve academic performance of students, using as basis a good study plan for the academic period.

Even though the obtained results had been satisfactory, it is necessary to mention that for future works it will be necessary to test the system with data from other faculties in order to know consistency and convergence. We want to establish effectiveness thresholds in the use of these techniques to obtain more and better outcomes in the application of data mining techniques for recommendation systems in this domain of application.

Other important aspect is to obtain the relation between the different courses. Currently, we are working to obtain (from data used in this research) patterns that relate courses with students' academic performance. This type of information will eventually represent an additional element to improve the recommendations offered by the system.

Acknowledgements

This work has been funded by Spanish Ministry of Science and Education through the HADA project TIN2007-64718. César Vialardi is also funded by Fundación Carolina. We would like to thank Eduardo Pérez and Jorge Chue who contributed in this work with their helpful comments.

References

[1] Adomavicius, G., Sankaranarayanan, R., Sen, S., and Tuzhilin, A. Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach. *ACM Transactions on information Systems,* 2005, 23(1), p. 103-145.

[2] Agrawal, R., Imielinski, T., and Swami, A. Mining association rules between sets of items in large databases. *ACM SIGMOD Conference on Management of Data*, 1993, p. 207-216.

[3] Arabie, P., Hubert, J., and De Soete, G. Clustering and Classification. (Eds.) *World Scientific Publishers Company*, 1996. London.

[4] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. Item – Based Collaborative Filtering Recommendation Algorithms, *Proceedings of the 10th international conference on World Wide Web*, 2001, p. 285-295.

[5] Bravo, J., Vialardi, C., and Ortigosa, A. A Problem-Oriented Method for Supporting AEH Authors through Data Mining. *Proc. of International Workshop on Applying Data Mining in E-learning (ADML'07) held at the Second European Conference on Technology Enhanced Learning (EC-TEL 2007)*, 2007, p. 53-62.

[6] Castellano, E. and Martínez, L. ORIEB, A CRS for Academic Orientation Using Qualitative Assessments. *Proceedings of the IADIS International Conference E-Learning*, 2008, p 38-42.

[7] Chen, M., Han. J., and Yu, P. Data mining: an overview from Databases Perpective. *IEEE Transaction on Knowledge and Data Engineering*, 1996, p. 833-866.

[8] Cooley, R., Mobasher, B., and Srivasta, J. Data Preparation For Mining Word Wide Web *Browsing Patterns. Knowledge and Information Systems*, 1999, 1(1), p. 5-32.

[9] Fayyad, U., Piatetsky-Shapiri, G., and Smyth, P. From Data mining to knowledge Discovery in Databases. *AAAI*, 1997, p. 37-54.

[10] Han, J., Pei, J., and Yan, X. Sequential Pattern Mining by Pattern-Growth: Principles and Extensions. *Series in studies in Fuzziness and soft computing*, 2005, 180, p.183-220.

[11] Jain, A.K., Murty, M.N., and Flynn, P.J. Data Clustering. *A Review. ACM Computing Surveys*. 1999, 31(3), p. 264-323.

[12] Mobasher, B. Data Mining for Personalization. The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., Nejdl, W. (Eds.). *Springer-Verlag*, 2007. Berlin Heidelberg, p. 1-46.

[13] Quinlan, J.R. C4.5: Programs for Machine Learning. *Morgan Kaufmann Publishers*, California, USA, 1993.

[14] Romero, C. and Ventura, S. Educational Data Mining: a Survey from 1995 to 2005. *Expert Systems with Applications*, 2007, 33(1), p.135-146.

[15] Schafer, J.B. The application of data-mining to recommender systems. J. Wang (Eds.), *Encyclopedia of data warehousing and mining*, 2005 Hershey, PA. Idea Group, p. 44-48.

[16] Srivastava, J., Cooley, R., Deshpande, M., and Tan, P. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. *SIGKDD Explorations*, 2000, 1(2), p. 12-23.

[17] Superby, J.F., Vandamme, J.P., and Meskens, N. Determination of Factors Influencing the Achievement of the First-year University Students using Data Mining Methods. *Workshop on Educational Data Mining*, 2006, p. 37-44.

Developing an Argument Learning Environment Using Agent-Based ITS (ALES)

Safia Abbas[1] and Hajime Sawamura[2]

[1] Graduate School of Science and Technology, Niigata University
8050, 2-cho, Ikarashi, Niigata, 950-2181 JAPAN
safia@cs.ie.niigata-u.ac.jp
[2] Institute of Natural Science and Technology,
Academic Assembly,Niigata University
8050, 2-cho, Ikarashi, Niigata, 950-2181 JAPAN
sawamura@ie.niigata-u.ac.jp

Abstract. This paper presents an agent-based educational environment to teach argument analysis (ALES). The idea is based on the Argumentation Interchange Format Ontology (AIF) using "Walton Theory". ALES uses different mining techniques to manage a highly structured arguments repertoire. This repertoire was designed, developed and implemented by us. Our aim is to extend our previous framework proposed in [3] in order to i) provide a learning environment that guides student during argument learning, ii) aid in improving the student's argument skills, iii) refine students' ability to debate and negotiate using critical thinking. The paper focuses on the environment development specifying the status of each of the constituent modules.

1 Introduction

Argumentation theory is considered as an interdisciplinary research area. Its techniques and results have found a wide range of applications in both theoretical and practical branches of artificial intelligence and computer science [13, 12, 16]. Recently, AI in education is interested in developing instructional systems that help students hone their argumentation skills [5]. Argumentation is classified by most researchers as demonstrating a point of view (logic argumentation), trying to persuade or convince (rhetoric and dialectic argumentation), and giving reasons (justification argumentation) [12]. Argumentation skill is extremely valuable in the educational field, and it reflects the student's abilities to outline a claim in a logical and convincing way and provides supportable reasons for the claim as well as identifying the often implicit assumptions that underlie the claim. Although argumentation skill is very important in the field of education, students' main barrier is their inability to follow the argument; highlighting the main points of a context [10]. The development of argumentation skills help students to develop their meta-cognitive and higher-order thinking abilities because argumentation requires individuals to externalize and explicitly reflect on their own thinking.

In response to the importance of argumentation skills in education, different argument mapping tools (e.g., Compendium, Araucaria, Rationale, etc.) have been developed [13]. These tools designed to foster students' ability to articulate, comprehend and communicate reasoning by producing diagrams of reasoning and argumentation. They provide a blackboard for students to record a graphical trace of their arguments. The main drawback in these tools is the absence of an administrator to constrict the argument diagram process. In other words, guiding the students to analyze arguments based on scientific theories or evidence [15].

In this paper, we extend our framework proposed in [3] by developing a learning environment(ALES) that uses mining agent-based ITS for teaching argument analysis. ALES uses the highly structured argument repertoire "RADB" to expose expert knowledge. It also models the student's argumentation knowledge and skills then, based on this information, it presents a group of arguments from which the user can choose one to work on. The paper is organized as follows: Section 2 introduces the learning environment (ALES) architecture. Section 3 presents an illustrative example for student-system interaction. Related work is presented in section4. Finally, conclusion and future work are illustrated in section5.

2 ALES Architecture

This section describes the architecture of the proposed learning environment(ALES) as shown in *Fig*.1. The environment consists of four main parts: the domain model represented as a highly structured argument repertoire, the Pedagogical model that contains three components: a parser, a classifier agent, and a teaching model, the student model that keeps track of the student performance and assists the pedagogical model in offering the individualized teaching, and finally the interface model "GUI". Not only does ALES teach argument analysis, but also assesses the student and guides him through personalized feedback. The next subsections illustrate the domain, student and pedagogical models in detail.

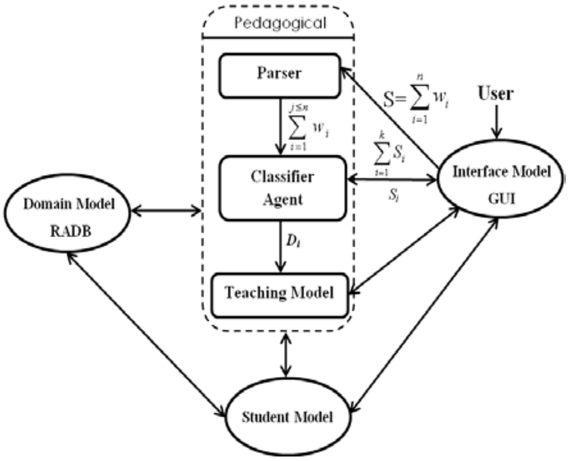


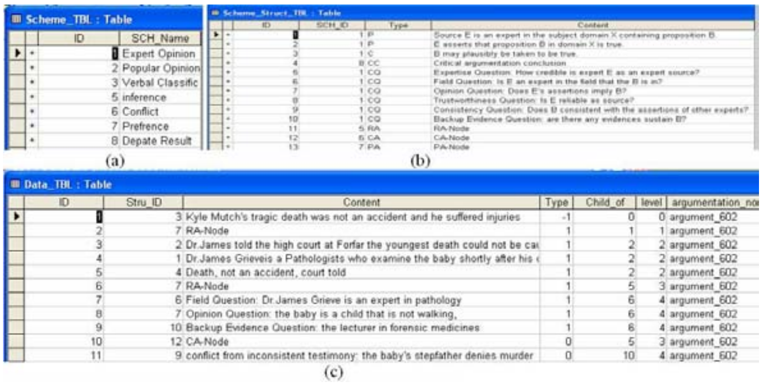
Fig. 1. ALES architecture

2.1 The Domain Model

The domain model is represented in the form of the relational argument database (RADB), it has been developed and implemented by us, see [2, 3] for more details and discussions, which summon a huge number of arguments. These arguments were previously analyzed by experts based on Walton theory using the AIF ontology [6, 11]. The domain model can semantically be represented as a forest of a numerous directed free trees [7]. Each directed tree in the forest lays out a semantic representation for a specific argument analysis. The domain model representation is general enough to encapsulate multiple domains, it also enjoys the extendibility feature, where adding new schemes is permitted. *Fig*.2 describes the various building blocks concerned with the RADB, using screen shots of our implemented system, such that: (a) the table "Scheme_TBL" gathers the name and the index for different schemes, (b) the table "Scheme_Struct_TBL" assembles the details of each scheme in "Scheme_TBL", (c) the "Data_TBL" table contains the analysis of different arguments based on different scheme structure and preserves the constraints of the AIF ontology [6] (s.t. no information node(I-node) refines another I-node). The relation between those different basic tables is shown in *Fig*.3.

2.2 The Student Model

The student model stores details about student's current problem-solving state and long term knowledge progress, that is essential for future student's performance evaluations. The model considers personal information, pre-test evaluation, and performance history. Personal information contains personal data as name, ID, password, ..., etc. The pre-test evaluation permanently assesses the student's argument analysis skills and follows the student progress through learning process. Finally, the performance history implicitly reflects how much the student has done and how well.



Scheme_TBL : Table

ID	SCH_Name
1	Expert Opinion
2	Popular Opinion
3	Verbal Classific
5	inference
6	Conflict
7	Prefrence
8	Depate Result

(a)

Scheme_Struct_TBL : Table

ID	SCH_ID	Type	Content
1	1	P	Source E is an expert in the subject domain X containing proposition B.
2	1	P	E asserts that proposition B in domain X is true.
3	1	C	B may plausibly be taken to be true.
4	8	CC	Critical argumentation conclusion
5	1	CQ	Expertise Question: How credible is expert E as an expert source?
6	1	CQ	Field Question: Is E an expert in the field that the B is in?
7	1	CQ	Opinion Question: Does E's assertions imply B?
8	1	CQ	Trustworthiness Question: Is E reliable as source?
9	1	CQ	Consistency Question: Does B consistent with the assertions of other experts?
10	1	CQ	Backup Evidence Question: are there any evidences sustain B?
11	5	RA	RA-Node
12	6	CA	CA-Node
13	7	PA	PA-Node

(b)

Data_TBL : Table

ID	Stru_ID	Content	Type	Child_of	level	argumentation_no
1	3	Kyle Mutch's tragic death was not an accident and he suffered injuries	-1	0	0	argument_602
2	7	RA-Node	1	1	1	argument_602
3	2	Dr.James told the high court at Forfar the youngest death could not be ca	1	2	2	argument_602
4	1	Dr.James Grieveis a Pathologists who examine the baby shortly after his	1	2	2	argument_602
5	4	Death, not an accident, court told	1	2	2	argument_602
6	7	RA-Node	1	5	3	argument_602
7	6	Field Question: Dr.James Grieve is an expert in pathology	1	6	4	argument_602
8	7	Opinion Question: the baby is a child that is not walking,	1	6	4	argument_602
9	10	Backup Evidence Question: the lecturer in forensic medicines	1	6	4	argument_602
10	12	CA-Node	0	5	3	argument_602
11	9	conflict from inconsistent testimony: the baby's stepfather denies murder	0	10	4	argument_602

(c)

Fig. 2. The main tables in RADB

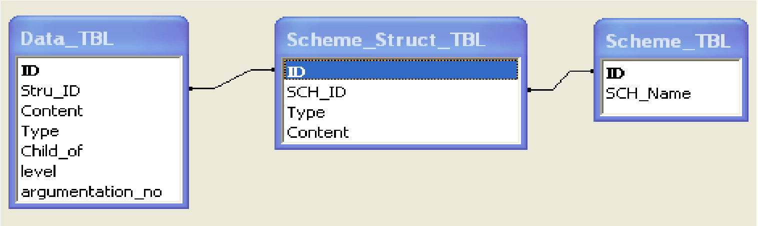


Fig. 3. The relation between the main RADB's tables

2.3 Pedagogical Model

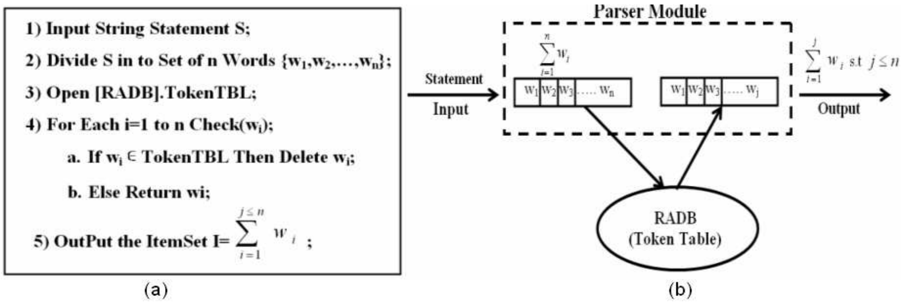
The pedagogical model is responsible for reasoning about the student behavior according to the student model, in order to: i) retrieve the most relevant results to both the subject of search and the students' background, ii) expose the corresponding argument to the selected result, iii) guide the student analysis based on the pre-existing one. The pedagogical model as seen in *Fig.*1 consists of three main components: a parser, a classifier agent, and a teaching model.

2.3.1 Parser

The parser as shown in *Fig.*4(b) receives a statement S from the student. This statement is divided by the parser into tokens, and then the number of tokens is reduced. The tokens are reduced if they belong to a look up table containing a set of all unnecessary words like { a, an, the, he, have, is, him ..., etc }, otherwise it is added to the set of tokens to be sent to the classifier agent. Finally the final crucial set of words { w_1 w_2... w_n } is sent to the classifier agent. The importance of the parser module lies in reducing the set of tokens into a set of significant keywords, which in turn will i) improve the results of the classifier where combinations of unnecessary words vanish, ii)reduce the number of iterations done by the classifier agent. The parser is already implemented as shown in *Fig.*4(a).

2.3.2 Classifier Agent

The classifier agent gathers and controls different mining techniques in order to classify the retrieved contexts based on student's choices. The agent mines the RADB repository aiming to: (i) direct the search process towards hypotheses that are more relevant to student's subject of search; classifying the analogous arguments in different ways based on students' choice, seeking for the most relevant arguments to the subject of search. (ii) add flexibility to the retrieving process by offering different search techniques. The agent offers three search techniques: general search, priority search, and rule extraction search. In the former, the general search classifies and retrieves the arguments based on



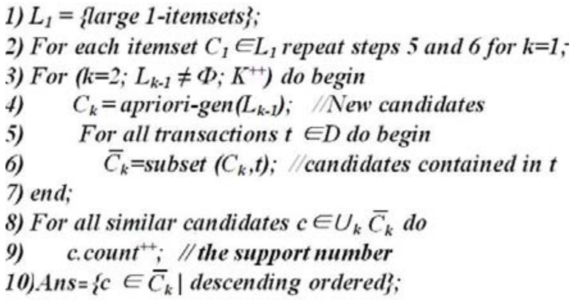
```
1) Input String Statement S;
2) Divide S in to Set of n Words {$w_1, w_2, \ldots, w_n$};
3) Open [RADB].TokenTBL;
4) For Each i=1 to n Check($w_i$);
    a. If $w_i \in$ TokenTBL Then Delete $w_i$;
    b. Else Return wi;
5) OutPut the ItemSet I= $\sum_{i=1}^{j \leq n} w_i$ ;
```

(a)
Parser Module
$\sum_{i=1}^{n} w_i$
Statement
Input
w_1 w_2 w_3 ….. w_n
w_1 w_2 w_3 ….. w_j
$\sum_{i=1}^{j} w_i$ s.t $j \leq n$
Output
RADB
(Token Table)
(b)

Fig. 4. The parser model

the breadth first search technique. The priority search classifies the retrieved contexts based on the maximum support number using an adapted version of the AprioriTid[4] mining technique. In the latter, the rule extraction summarizes the retrieved arguments searching for hidden patterns that are most relevant to the subject of search, then this patterns are exposed in the form of rules. Each rule, for each retrieved argument, contains the affirmative "+" and the negative "-" parts relating to the final conclusion of that argument.

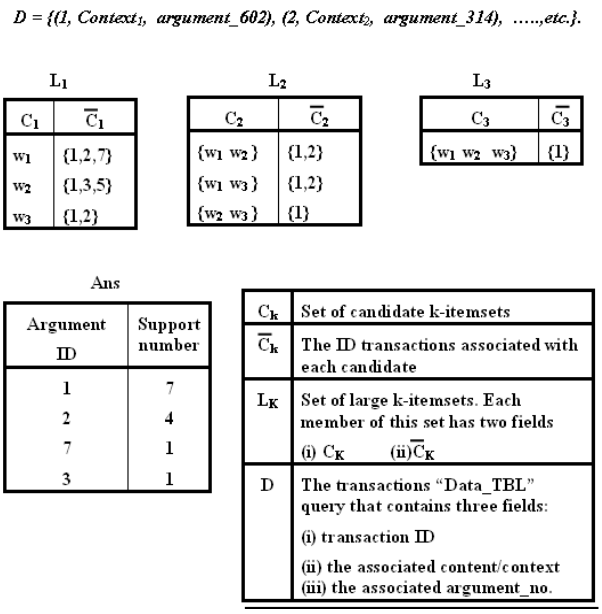
Priority Search: The AprioriTid algorithm [4] has been implemented and embedded to the classifier agent as "Priority Search". The Priority search aims to retrieve the most relevant arguments to the users' subject of search and queuing them based on the maximum support number, such that the first queued argument is the one that has more itemsets[3] related to the subject of search. Although the AprioriTid algorithm has originally been devised to discover all significant association rules between items in large database transactions, the agent employs its mechanism in the priority search to generate different combinations between different itemsets [4, 3]. These combinations will then be used to classify the retrieved contexts and queued them in a descending order based on its support number. As a response to the priority search purpose, an adapted version of the AprioriTid mining algorithm has been applied. This adapted version, as seen in *Fig.* 5, considers the single itemset (1-itemset) size as well as the maximum support number usage, rather than k-itemset for k≥2 and the minimum support number "minsup" mechanism.



```
1) $L_1$ = {large 1-itemsets};
2) For each itemset $C_1 \in L_1$ repeat steps 5 and 6 for k=1;
3) For (k=2; $L_{k-1} \neq \Phi$; $K^{++}$) do begin
4)      $C_k$ = apriori-gen($L_{k-1}$);   //New candidates
5)       For all transactions t $\in D$ do begin
6)          $\overline{C}_k$=subset ($C_k$,t);  //candidates contained in t
7) end;
8) For all similar candidates $c \in U_k \overline{C}_k$ do
9)      c.count$^{++}$;  //the support number
10).Ans={c $\in \overline{C}_k$ | descending ordered};
```

Fig. 5. An enhanced version of AprioriTid

For more clarification, the priority search mines specific parts of the pre-existing arguments based on the users' search criteria. This search criteria enables the student to seek the premises, conclusions or the critical questions lying in the different arguments. For example, suppose the student queries the RADB searching for all information related to "Iraq war". Simply, he may write "the destructive war in Iraq" as the search statement and can choose the conclusion as the search criteria.



D = {(1, Context$_1$, argument_602), (2, Context$_2$, argument_314), …..,etc.}.

L$_1$

C$_1$	$\overline{C}_1$
w$_1$	{1,2,7}
w$_2$	{1,3,5}
w$_3$	{1,2}

L$_2$

C$_2$	$\overline{C}_2$
{w$_1$ w$_2$ }	{1,2}
{w$_1$ w$_3$ }	{1,2}
{w$_2$ w$_3$ }	{1}

L$_3$

C$_3$	$\overline{C}_3$
{w$_1$ w$_2$ w$_3$}	{1}

Ans

Argument ID	Support number
1	7
2	4
7	1
3	1

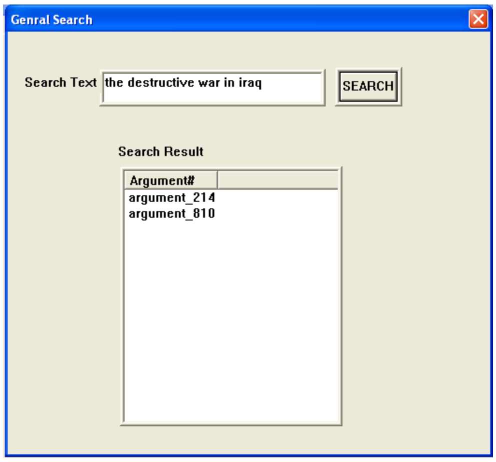
C$_k$	Set of candidate k-itemsets
$\overline{C}_k$	The ID transactions associated with each candidate
L$_K$	Set of large k-itemsets. Each member of this set has two fields (i) C$_K$ (ii)$\overline{C}_K$
D	The transactions “Data_TBL” query that contains three fields: (i) transaction ID (ii) the associated content/context (iii) the associated argument_no.

Fig. 6. The adapted AprioriTid mechanism

In this case, the classifier agent receives the set of significant tokens {destructive, war, iraq} from the parser model. This set is considered as the single size itemset (1-itemset) C$_1$={w$_1$, w$_2$, w$_3$} that contains the most crucial set of words in the search statement. Then, the agent uses the adapted version of the AprioriTid algorithm to generate the different super itemsets $C_{2\leq k\leq 3}$, which are the different combinations between different tokens. So, the generated super itemsets, as seen in *Fig.*6, will be the 2-itemset C$_2$={ w$_1$w$_2$, w$_1$w$_3$, w$_2$w$_3$ }, and the 3-itemset C$_3$={ w$_1$w$_2$w$_3$ }. Afterward, the different conclusions in the different arguments trees will be mined seeking for the most relevant set of arguments Ans={d$_1$, d$_2$, ..., d$_m$ } such that $\forall$ d$_i$$\in$D $\exists$ $C_{k\in\{1,2,..,j\}}\subseteq$ d$_i$. Finally, the results will be queued in a descending order and exposed in a list, where the student can choose the argument name ”Argument_314” from the list to expose the associated context and analysis as in *Fig.* 10.

General Search: The system uses the breadth first [14] search in order to seek the different argument trees and retrieve the most relevant group. The revealed contexts are ordered based on the number of nodes ”nodes cardinality” that contain any keyword, in a way where the first context is the one which has more nodes related to the search statement. For example, suppose the user writes ”the destructive war in Iraq” as a search statement. The revealed contexts, as shown in *Fig.* 7, will be ordered based on the nodes' cardinality. The breadth first search seeks each tree in our RADB, preserving the ancestor-descendant relation [7] by searching first the root, then the children in the same level and so on. Finally, if the user picks one of the resulted search arguments, the associated context and analysis are depicted as shown in *Fig.* 10.

Rule Extraction Search: Rule extraction mining is a search technique in which argument trees are encountered to discover all hidden patterns ”embedded subtrees” [7] that coincide with the relation between some objects. These objects express a set of the most significant tokens of the user's subject of search. Precisely, suppose the student wants to report some information about the relation between the ”USA war” and the ”weapons of mass destruction”. At the beginning, the user's search statements are reduced to the most significant set of tokens by the parser [2][3][1]. Then, the different argument trees, pre-existing in the RADB repository, are mined in order to fetch these different tokens. *Fig.* 8(a) shows the analysis of an argument tree, where some enclosed nodes coincide with the student's search statements, while *Fig.* 8(b) shows the revealed embedded subtree. Finally, each

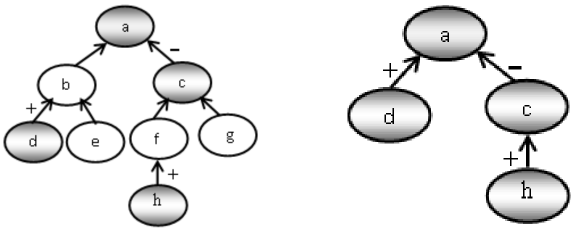


Search Text: the destructive war in iraq

Search Result

Argument#
argument_214
argument_810

Fig. 7. The General search representation form



(a)
The main argument tree

(b)
The resulted embedded subtree

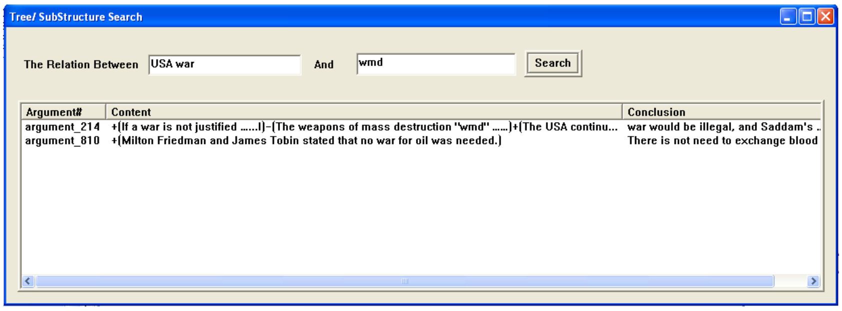
Fig. 8. The tree Rule Extraction search

resulting subtree is expressed in the form of a rule as shown in *Fig.* 9, where ”+” indicates that this node is a support to the final conclusion whereas ”-” is a rebuttal node to the final conclusion.

2.3.3 Teaching Model

The teaching model monitors the student actions, guides the learning process and provides the appropriate feedback. However, In the mean time, it is still in the implementation phase. The model starts its role when the classifier agent sends the document D_i selected by the student. The teaching model checks, according to the current student model, whether the student is in the learning or the assessing phase. If the student is in the learning phase, the document is presented associated with the corresponding analysis as the shown in *Fig.* 10. On the other hand, if the student is in the assessment phase, the student is able to do his own analysis, and the teaching model will guide him during analysis by providing personalized feedback whenever required. The feedback aims to guide the student and refine his analysis and intellectual skills. Two kinds of feedback are provided by the teaching model; partial argument negotiation and total argument negotiation.

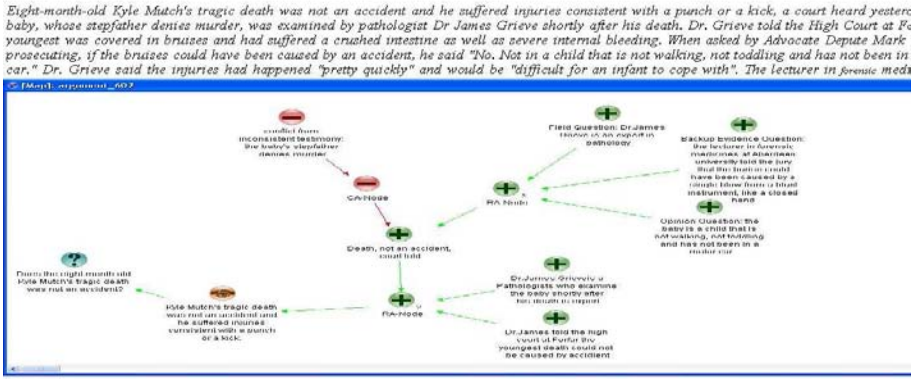
- **Case of partial argument negotiation:** In this case, the student starts analyzing the argument context in the form of a tree in which the root holds the final conclusion of the issue of discussion. The teaching pedagogy used in this case provides partial hints at each node of the analysis tree. They are results of comparing the student’s current node analysis to the original one in the argument database. These hints are provided before allowing the student to proceed further in the analysis process; they aim to minimize the analysis error ratio, as much as possible, for the current analyzed node. Generally, the teaching model guides with the student via the



The Relation Between USA war And wmd Search

Argument#	Content	Conclusion
argument_214	+(If a war is not justified ......l)-(The weapons of mass destruction "wmd" ......)+(The USA continu...	war would be illegal, and Saddam's ..
argument_810	+(Milton Friedman and James Tobin stated that no war for oil was needed.)	There is not need to exchange blood

Fig. 9. The representation form of Rule Extraction search result



Eight-month-old Kyle Mutch's tragic death was not an accident and he suffered injuries consistent with a punch or a kick, a court heard yesterc baby, whose stepfather denies murder, was examined by pathologist Dr James Grieve shortly after his death. Dr. Grieve told the High Court at Fc youngest was covered in bruises and had suffered a crushed intestine as well as severe internal bleeding. When asked by Advocate Depute Mark prosecuting, if the bruises could have been caused by an accident, he said "No. Not in a child that is not walking, not toddling and has not been in car." Dr. Grieve said the injuries had happened "pretty quickly" and would be "difficult for an infant to cope with". The lecturer in forensic medi

[Map]: argument_607
conflict from inconsistent testimony: the baby's stepfather denies murder
CA-Node
Field Question: Dr.James Grieve is an expert in pathology
Backup Evidence Question: the lecturer in forensic medicine at Aberdeen university told the jury that the injuries could have been caused by a single blow from a blunt instrument, like a closed hand
RA Node
Opinion Question: the baby is a child that is not walking, not toddling and has not been in a motor car
Death, not an accident, court told
Does the eight month old Kyle Mutch's tragic death was not an accident?
Kyle Mutch's tragic death was not an accident and he suffered injuries consistent with a punch or a kick.
RA-Node
Dr.James Grieve is a Pathologists who examine the baby shortly after his death to report
Dr.James told the high court at Forfar the youngest death could not be caused by accident

Fig. 10. The representation of the selected argument

partial hints at each node till the error of the current node is minimized to a specific ratio. After then, the student is able to move to the next analysis step (i.e., node).

- **Case of total argument negotiation:** The total argument negotiation is similar to the partial argument negotiation. However, the teaching pedagogy is different in that it provides hints only at the end of the analysis process. In other words, after the student builds the full analysis tree for the selected context, the system interprets and evaluates the student's analysis comparable to the pre-existing one and remarks the errors.

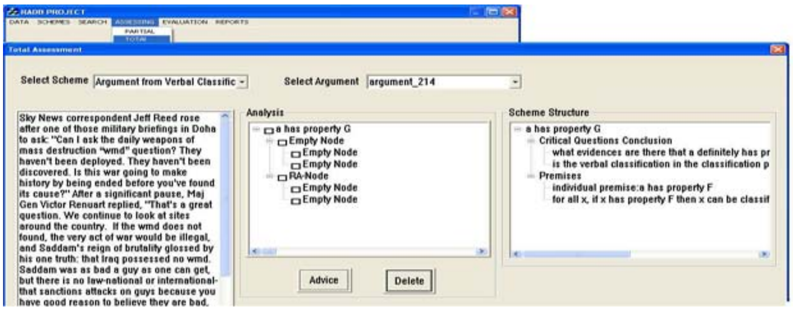
Generally, in the assessing phase, the teaching model presents the transcript of the chosen argument associated with an empty tree skeleton and asks the student to start his own analysis. The student starts the analysis by copy and paste text passages from the transcript or enter free text into the nodes. The teaching model traces each node text and divides it into set of significant tokens, then interprets and evaluates the errors ratios comparable to the pre-existing analysis underlying in the RABD. Finally the model provides the feedback, partially or totally, based on the student choice and records the student's errors for the current transcript, which in turn will be used, by the student model, to evaluate the performance and to follow progress of the student.

3 Illustrative Example

This example shows a complete run for the Total negotiation of the assessing phase. The system interactions are written in normal font. The student's actions are in bold. My illustrations to some actions will be in capital letters.

SUPPOSE THE STUDENT IN THE ASSESSING PHASE CHOOSING THE TOTAL FEEDBACK

PROPERTY. THE SYSTEM WILL GIVE THE STUDENT THE ABILITY TO SELECT SPECIFIC SCHEME TO BE USED IN HIS ANALYSIS, AS SHOWN IN *Fig*. 11.
User≫ "Argument from Verbal Classification Scheme".



Select Scheme: Argument from Verbal Classific
Select Argument: argument_214

Sky News correspondent Jeff Reed rose after one of those military briefings in Doha to ask: "Can I ask the daily weapons of mass destruction "wmd" question? They haven't been deployed. They haven't been discovered. Is this war going to make history by being ended before you've found its cause?" After a significant pause, Maj Gen Victor Renuart replied, "That's a great question. We continue to look at sites around the country. If the wmd does not found, the very act of war would be illegal, and Saddam's reign of brutality glossed by his one truth: that Iraq possessed no wmd. Saddam was as bad a guy as one can get, but there is no law-national or international-that sanctions attacks on guys because you have good reason to believe they are bad.

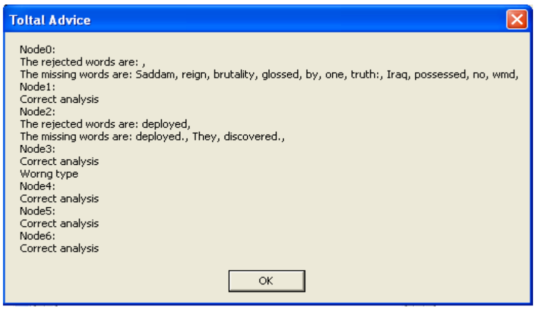
Analysis
- a has property G
  - Empty Node
    - Empty Node
    - Empty Node
  - RA-Node
    - Empty Node
    - Empty Node

Scheme Structure
- a has property G
  - Critical Questions Conclusion
    - what evidences are there that a definitely has pr
    - is the verbal classification in the classification p
  - Premises
    - individual premise:a has property F
    - for all x, if x has property F then x can be classif

Fig. 11. The total negotiation assessment form

THE WHOLE ARGUMENTS, THAT USE THE "ARGUMENT FROM VERBAL CLASSIFICATION SCHEME" IN ITS ANALYSIS, WILL BE LISTED SUCH THAT THE PRIORITY IS TO THE CONTEXTS THAT HAVE NOT BEEN ACCESSED YET BY THE USER DURING THE LEARNING PHASE.
System≫ [argument_214, argument_1, argument_600].
User≫ picks up one of the listed arguments, [argument_214] as example.
System≫ presents the transcript of the chosen argument with an empty tree skeleton as in *Fig*. 11.
User≫ start the analysis by copy and paste text passages from the transcript or enter free text into the nodes then press advice.
System≫ divides each statement in each node into tokens, and compares these tokens with the expert analysis for the same node. Then calculates and records the errors ratio for the whole nodes.
System≫ shows out a declarative report that describes the mistakes of each node separately.

As SEEN IN *Fig*12 THE STUDENT ANALYSIS OF THE FINAL CONCLUSION NODE "NODE0" IS PARTIALLY CORRECT AND THE STUDENT HAS BEEN ADVISED TO USE THESE WORDS IN HIS ANALYSIS {SADDAM, REGION,...}. ALSO IN "NODE3" WHICH IS ON OF THE CRITICAL QUESTIONS, THE ANALYZED STATEMENT IS CORRECT HOWEVER THE TYPE OF THE NODE (SUPPORT OR ATTACH) IS WRONG.



Node0:
The rejected words are: ,
The missing words are: Saddam, reign, brutality, glossed, by, one, truth:, Iraq, possessed, no, wmd,
Node1:
Correct analysis
Node2:
The rejected words are: deployed,
The missing words are: deployed., They, discovered.,
Node3:
Correct analysis
Worng type
Node4:
Correct analysis
Node5:
Correct analysis
Node6:
Correct analysis

Fig. 12. The resulting report

User≫ press OK.
AFTER THE USER FINISHES HIS ANALYSIS TO THE WHOLE CONTEXT, FILLING THE

SUITABLE ANALYSIS FOR EACH NODE, THE SYSTEM WILL CALCULATES AND RECORDS THE WHOLE ARGUMENT ANALYSIS RATIO FOR THAT ARGUMENT. THIS RATIOS RECORDS WILL BE USED LATER TO EVALUATE THE PROGRESS OF THAT STUDENT.

4 Related Work

Early, the field of AI and education was very interesting to most of the researchers, where many instructional systems have been developed to hone students argumentation skills. SCHOLAR and WHY[8] systems are examples for these trials. However, these systems were mainly designed to engage students in a Socratic dialog, which faces significant problems such as knowledge representations to develop a Socratic tutor[8]. This mainly occurred in complex domains like legal reasoning, control or preprocessing, and manipulate the natural language. Later, as a response to these difficulties, a number of argument mapping tools[18, 10, 17, 13] have been developed to foster debate among students about specific argument, using diagrams for argument representation. However, the data mining and artificial intelligence influence, which needed to guide the student to understand the relation between scientific theories and evidence, and refines his argument analysis ability, are missing in these tools.

Recently, a number of mining weblogs[9] and case-based models[5] have been proposed to tackle the mining and the artificial influence problem. The mining weblogs is considered as a classification problem for the legal or informal reasoning considering law. Though, it mines the textual data that is intractable to be processed. On the other hand, the case-based argumentation systems, such as the CATO[5], use the case based reasoning method in order to reify the argument structure through tools for analyzing, retrieving, and comparing cases in terms of factors.

Comparing CATO with our proposed application, both of them provides examples of specific issue to be studied by the different students, as well as evaluates students' arguments comparable to the pre-existing one. Regarding to the search for arguments, both systems support students' search for the existing database, and retrieve the most relevant argument. However, CATO limits the students' search by a boolean combination of factors. Also, in the full-text retrieval search, one can retrieve documents, by matching phrases, which is not relevant to the search subject. On the other hand, ALES provides different search criteria to tackle this problem, as seen in section 2, using different mining techniques in order to: summon and provide a myriad of arguments at the student's fingertips, retrieve the most relevant results to the subject of search, and organize the retrieved result such that the most relevant is the first rowed.

Finally, I. Rahwan presents the ArgDf system [6, 11], through which users can create, manipulate, and query arguments using different argumentation schemes. Comparing ArgDf system to our approach, both of them sustain creating new arguments based on existing argument schemes. In addition, the ArgDf system guides the user during the creation process based on the scheme structure only, the user relies on his efforts and his background to analyze the argument. However, in our approach, the user is not only guided by the scheme structure but also by crucial hints devolved through mining techniques. Accordingly, the creation process is restricted by comparing the contrasting reconstruction of the user's analysis and the pre-existing one. Such restriction helps in refining the user's underlying classification.

In the ArgDf system, searching existing arguments is revealed by specifying text in the premises or the conclusion, as well as the type of relationship between them. Then the user can choose to filter arguments based on a specific scheme. Whereas in our approach, searching the existing arguments is not only done by specifying text in the premises or the conclusion but also by providing different strategies based on different mining techniques in order to: refine the learning environment by adding more flexible interoperability, guarantee the retrieval of the most convenient hypotheses relevant to the subject of search, facilitate the search process by providing a different search criteria.

5 Conclusion and Future Work

In this paper we introduced an agent-based learning environment (ALES) to teach argument analysis. ALES extends the previous work done on building a highly structured argument repertoire (RADB) with managing tool [2, 3]. The main aim of developing this environment is to aid in improving the student's argument skills. ALES serves as a new trend in teaching arguments. The proposed architecture serves the educational process by allowing learning and assessing phases where personalized feedback is provided. ALES guides the student during argument learning, analysis, and preprocessing. In addition, ALES enjoys certain advantage over others, where a relevant and convenient result is assured to be obtained especially when the search statement is in this form: "the destructive war in Iraq". In the future, we intend to (i) integrate an NLP software to aid in polarity classification, in which the underlying RADB arguments are classified into affirmative and rebuttal lists to the issue of discussion, (ii) use the frequent tree mining techniques[7] in order to search for frequent patterns in different arguments, and (iii) consider the interaction between the student model and the pedagogical model, and how this is going to affect the abductive learning phase.

References

1. S. Abbas and H. Sawamura. Argument mining using highly structured argument repertoire. In *The First International Conference on Educational Data Mining (EDM),Montreal, Qubec, Canada*, pages 202–2010, 2008.
2. S. Abbas and H. Sawamura. A first step towards argument mining and its use in arguing agents and its. In *12th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES), Zagreb, Croatia*, pages 149–157. Springer, 2008.
3. S. Abbas and H. Sawamura. Towards argument mining using relational argument database. In *The Second International Workshop on Juris-informatics (JURISIN), Asahikawa Convention Bureau, Hokkaido, Japan.*, pages 22–31, 2008.
4. R. Agrawal and R. Srikant. Fast algorithms for mining rules. In *Proceeding of the 20th VLDB Conference Santiago, Chile*, 1994.
5. V. Aleven and K.D. Ashley. Teaching case-based argumentation through a model and examples empirical evaluation of an intelligent learning environment. In *the Eighth World Conference of the Artificial Intelligence in Education Society*, pages 87–94, 1997.
6. I. Rahawan C. Chesnevar and C. Reed. Towards an argument interchange format. In *The Knowledge Engineering Review*, pages 1–25. Cambridge University Press, 2007.
7. Y. Chi and R. Muntz. Frequent subtree mining-an overview. In *Fundamenta Informaticae*, pages 1001–1038, 2001.
8. Collins and L.Stevens. Goals and strategies of inquiry teachers. In *Advances in Instructional Psychology*, pages 65–119, 1982.
9. J. Conrad and F. Schilder. Opinion mining in legal blogs. In *ICAIL07 Palo Alto, California USA*, pages 231–236, June 4-8, 2007.
10. M. Harrell. Using argument diagramming software in the classroom. In *16th Biennial Workshop-Conference on Teaching Philosophy*, 2006.
11. F. Zablith I. Rahawan and C. Reed. The foundation for a world wide argument web. In *Artificial Intelligence Conference (AAAI).* published in the Artificial Intelligence Journal, April 04, 2007.
12. M. Baker J. Andriessen and D. Suthers. Arguing to learn confronting cognitions in computer-supported collaborative learning environments. In *Kluwer Academic Publishers, Dordrecht/Boston/London*, volume 1, 2003.
13. C. Reed J. Katzav and G. Rowe. Argument research corpus. In *Communication in Multiagent Systems,Lecture Notes in Computer Science,Springer Verlag, Berlin, Germany*, volume 2650, pages 269–283, 2003.
14. S. Nijssen and J. N. Kok. A quickstart in frequent structure mining can make a difference. In *tenth ACM SIGKDD international conference on Knowledge discovery and data mining, USA*, 2004.
15. Suthers Paolucci, A. and A.Weiner. Automated advice-giving strategies for scientific inquiry. In *Intelligent Tutoring Systems, Third International Conference, berlin*, pages 372–381, 1996.
16. I. Rahawan and P.V. Sakeer. Representing and querying arguments on semantic web. In *Computational Models of Argument, P.E. Dunne and T.J.M. Bench-Capon (Eds.), IOS Press*, 2006.
17. C. Reed and G. Rowe. Araucaria: Software for argument analysis, diagramming and representation. In *International Journal on Artificial Intelligence Tools*, volume 13, page 983, 2004.
18. D. Walton and G. Rowe. Araucaria as a tool for diagramming arguments in teaching and studying philosophy. In *Teaching Philosophy*, volume Vol. 29, pages 111–124, 2006.

A Data Mining Approach to Reveal Representative Collaboration Indicators in Open Collaboration Frameworks

Antonio R. Anaya, Jesús G. Boticario
{arodriguez, jgb}@dia.uned.es
E.T.S.I.I. - UNED C/Juan del Rosal, 16, Ciudad Universitaria, 28040 Madrid (Spain)

Abstract. Data mining methods are successful in educational environments to discover new knowledge or learner skills or features. Unfortunately, they have not been used in depth with collaboration. We have developed a scalable data mining method, whose objective is to infer information on the collaboration during the collaboration process in a domain-independent way and to improve collaboration process management and learning in an open collaborative educational web environment. Thus, we used statistical indicators of learner's interactions in forums as the data source and a clustering algorithm to classify the data according to learner's collaboration. We showed the information on learner's collaboration to the tutor and learners to help them with collaboration process management. The experimental results support this method.

1 Introduction

Collaborative learning is a useful strategy to solve the lack of social interaction in most e-learning environments. However, the collaboration process has not been researched in depth [19]. We propose a data mining approach to reveal learner`s collaboration in open collaboration frameworks. We hypothesize that showing information on the collaboration process improves management and teaching in an open collaboration-learning environment.

The educational context of our research is suitable for e-learning and collaborative learning because of the nature of students at UNED (The National Distance Learning University in Spain). These students are mainly adults with responsibilities other than learning. For this reason UNED's students cannot be forced to collaborate in a typical CSCL (Computer Support Collaborative Learning) environment because of the time restrictions of these environments [8]. However, the collaboration learning is very suitable in the UNED's educational context due to the isolation of these students. We solved the problems by providing learners with an open collaborative learning experience, where students could manage their own collaborative learning process. We designed a long-term collaborative learning experience with fourth-year Artificial Intelligence (AI) and Engineering Based Knowledge students. This experience consisted of two main phases within a step-wise approach: the first phase was 3 weeks and the second phase was 10 weeks. It was enough time for students to complete the collaborative work and manage their collaborative process.

Although collaboration environments have been researched, some works have focused on monitoring learner interaction and showing this to learners [10, 5], but they have not used any inferring method to reveal learner collaboration. Others have concentrated on inferring information about the collaboration process [15], but the method used is not domain independent. We argue the need for inferring methods and the domain independence of these methods to be able to transfer the approach to other collaborative learning environments.

Given our educational context and our proposed open collaborative learning environment, we needed to simplify and reduce conceptual problems in order to improve collaboration process management and transfer the ideas to other environments. We achieved this by developing an inferring method that aimed to: 1) reveal learner's collaboration, 2) be domain independent, and 3) offer the information immediately after the process had finished. We applied the proposed approach to help students and tutors to manage the collaboration process by providing information on learner's interaction and learner's collaboration levels.

We covered the objective by using the statistics of interactions in forums as the data source and a clustering algorithm as the inferring method. Forums are a very common service in a collaborative learning environment and the statistics from forums can be obtained just after the interaction has happened. Since the statistics from forums do not give any semantic information, they are domain independent. [6] researched the forums messages in an educational environment and they concluded that the forums analysis can reveal the collaboration of the learners and an analysis by data mining is advisable. The statistical indicators were selected in relation to the learners' activity, initiative, regularity and promoting team-work [17]. That show, the method can be automated and the results are ready before the collaboration experience has finished. Those results provide information on the collaboration process to the tutor of the collaborative environment so that the tutor improved the teaching. Moreover, the same idea could be applied to learners. Thus we showed learner's collaboration levels to both the tutor and learners.

During the academic years 2006-07 and 2007-08 we researched the inferring method to reveal learner's collaboration [1]. During the collaborative learning experience of the year 2008-09 we showed learners the results of the inferring method. We concluded that the inferring method reveals learner's collaboration (more collaborative learners are more active and their activity encourages others to be more active) and the inferred representative collaboration indicators can be measured automatically.

A short overview of methods already used in evaluating the collaboration process is given below. We describe the collaborative learning experience and the inferring method. Next we show the results obtained after applying the inferring method and we explain in detail how the inferring information was shown to learners. Finally, we conclude with the discussion and future works.

2 Related Works

[16] said that the knowledge extraction process is divided into three phases: pre-processing, data mining and post-processing. In this section we describe research works that focus on collaboration process analysis and we explain the data acquisition method, the inferring or data mining method and the use of the results.

There have been various experiments to measure or identify the collaboration that takes place between system users. These experiments can be classified firstly the data acquisition method. We can identify three methods: 1) Qualitative [12]: where participants or experts are asked to evaluate the activities of the participants. 2) Quantitative [20, 15, 9, 3], which collects statistical information on the activities of the participants. 3) Mixed [4, 5, 10, 14]: where both methods are used simultaneously.

When we talk about data mining, these systems can be characterized by the inferring method used to derive the value of certain features, such as the collaboration that has occurred or is occurring. The methods may include: a) analysis by an expert [12], b) comparison with a pre-existing model using machine learning methods [15], c) Different statistical techniques [7] or machine learning, such as clustering [20, 14], fuzzy logic [15], sequential pattern mining [14], and d) the systems can be characterized even by not using any inference system [3, 4, 5, 10].

Finally, the systems can be characterized by the data post-processing method, or by what they do with the results. According to [18], CSCL systems, and in this case the systems that we are analyzing can be characterized by what they do with the results: I) monitoring tools that automatically collect data from students on the interaction, and they show this information [12, 3, 4, 5, 10], II) metacognitive tools that show the information inferred in the mining process, as well as interactions [15, 20, 14], and III) guidance system that proposes remedial measures to help the student, once the right information has been inferred. [6] proved the analysis of the forums by data mining and text mining techniques provide with meaningful feedback about student's performance and a view of the historical progress of a community of learners. We have followed the ideas but in a domain independence way.

We propose the data mining method, whose acquisition method is quantitative because the data source is statistical indicators of learner interaction in forums. As an inferring method we use the clustering algorithm, whose objective is to classify learners according to their collaboration, which is disclosed from learner's interaction. Finally, the proposed data mining method provides learners and tutors with metacognitive information on collaboration. The proposed data mining method is a metacognitive tool, which covers the research objectives.

3 Collaboration experience

We offered a collaborative learning experience to our learners with the objective of solving the common problems of distance learning and e-learning [10]. This research was carried out at UNED, where students are not typical university students. These students are mainly adults with responsibilities other than learning, who fit into the Lifelong Learning Paradigm, which supports the idea that learning should occur throughout a person's lifetime, enabling the integration of education and work in a continuous process. This impacts on the time that students can use to learn, study or take part in a typical CSCL system.

Figure 1 shows the schema of the collaboration learning experience. It was offered at the beginning of the academic years 2006-07, 2007-08 and 2008-09, lasted around 3 months, and was divided into two phases. The 1st phase lasted 3 weeks and learners had to answer an initial questionnaire and do a mandatory task. The 2nd phase grouped learners who had done the mandatory task into three-member teams, and the teams had to follow 6 tasks. Figure 1 shows the number of learners that started the 1st phase and finished it, and finished the 2nd phase. The Figure 1 show the schema of the collaborative learning experience, where there were two phases and some tasks had to be done in a sequential order.

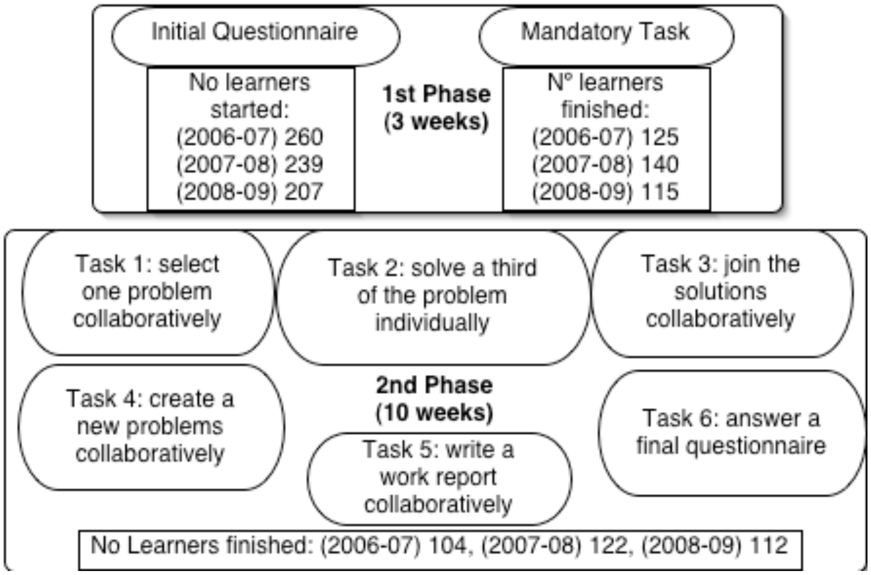

Initial Questionnaire
Mandatory Task
No learners started: (2006-07) 260 (2007-08) 239 (2008-09) 207
1st Phase (3 weeks)
Nº learners finished: (2006-07) 125 (2007-08) 140 (2008-09) 115
Task 1: select one problem collaboratively
Task 2: solve a third of the problem individually
Task 3: join the solutions collaboratively
Task 4: create a new problems collaboratively
2nd Phase (10 weeks)
Task 5: write a work report collaboratively
Task 6: answer a final questionnaire
No Learners finished: (2006-07) 104, (2007-08) 122, (2008-09) 112

Figure 1. Schema of the collaborative learning experience

We provided a learning platform dotLRN (http://dotlrn.org/), which supports all learning experience activities, provides communications services such as forums, and stores all the interactions that take place on the platform in a relation database. During the 1st phase a general virtual environment was opened for all learners of the subject with common services (FAQs, news, surveys, calendar and forums). During the 2nd phase virtual spaces for each three-member team were opened, where the teams could perform the tasks. The specific virtual spaces include documents, surveys, news, a task manager and forums.

4 Method

We developed the inferring method with these objectives in mind: 1) the method should obtain information on learner's collaboration; 2) the method should be domain independent; 3) the method should provide information on collaboration before the collaboration process finished. Thus, it is possible reusing and applying the approach in other e-learning environments. We were looking approaches that could be applied to others at UNED, where there are 4000 curses and over 190000 students enrolled.

The learners of the collaborative learning experience were encouraged to use the forums on the dotLRN platform as the main communication media. The platform stores the forum messages giving information on what thread the messages are in and what message the message has replied to. We focused on forum interactions, because they are a very common service in a collaborative learning environment and the statistics from forums can be obtained just after the interaction has happened and data mining analysis is possible with these indicators [6, 8]. Since the statistics from forums do not give any semantic information, they are domain independent.

In line with the objectives explained above, we used statistical indicators of learner interaction in forums as a data source. According to [17], the features of collaborative learners in these environments are: activity, initiative, regularity and promoting team-work. We proposed these attributes as indicators of the above features: number of threads or conversations that the learner started (num_thrd), and their average, square variance and the number of threads divided by their variance; the number of messages sent (num_msg), and their average, square variance and the number of messages divided by their variance; the number of replies in the thread started by the user (num_reply_thrd), and divided by the number of user threads; the number of replies to messages sent by the user (num_reply_msg), and divided by the number of user messages. The number of threads started and their associated indicators are related to learner initiative. The square variance of the number of threads is related to the regularity of the initiative. The number of messages sent and their associated indicators are related to learner activity and regularity of activity. The number of replies to messages sent and their associated indicators are related to the activity caused by the learner.

We built datasets with the above statistical indicators from every year (2006-07, 2007-08 and 2008-09). The characteristics of the datasets were: Dataset-06-07, 117 instances; Dataset-07-08, 122 instances; Dataset-08-09, 112 instances. Every instance is the statistical indicators of the interactions of one learner. We focused our research on the collaborative period, which started at the end of November and finished at the end of January. We collected the values of these statistical indicators in datasets during the whole collaborative period.

We used a clustering algorithm as the data mining method. We used a clustering method because it classifies data collection without help from any expert, which delays the inferring process. We employed the EM clustering algorithm because of its good results when the method is applied in the learning environment to reveal collaboration. [20, 14, 13].

We obtained a classification of the instances with the EM clustering algorithm. We used the WEKA data mining software [21] and the EM clustering algorithm [7]. We checked the relation of the classification obtained with collaboration.

We needed to know student collaboration from another source to be able to compare their results and validate the approach as a collaborative inferring method. For this reason an expert identified student collaboration in the experiences. The expert read all the forum messages and labeled students according to their collaboration levels. Thus, we obtained a list of most of the students labeled according to their collaboration level. The expert used a scale of 8 values (1, low collaboration level; 9, high collaboration level).

Finally, the method finished by comparing the clustering classification of the learners with the labeled list of learner's collaboration levels. The objective was to measure the average collaboration level of each cluster and to realize that the average collaboration level is different in each cluster.

5 Results

We have conducted this research during the last three years. In 2006-07 and 2007-08 we focused on the aforementioned inferring method in order to prove the usefulness of the method as a collaboration inferring method. During 2008-09 we applied the method to improve collaborative process management and learning. We proved that the clusters obtained from statistical indicators were related to learner collaboration in the last two years [1] and the data for 2008-09 support these conclusions.

We classified the learners into 3 clusters, because the meaning of the classification is easier to understand in relation to collaboration. One cluster represents the low collaboration level, another cluster the medium collaboration level and the third cluster the high collaboration level. Then we run the clustering algorithm EM to obtain 3 cluster and we supplied with the datasets of every year (D-06-07, D-07-08 and D-08-09). These datasets collected the above statistical indicators for every learner.

First of all, we note that the cluster algorithm classifies according to the interaction. One cluster (cluster-0 in the next table) collects learners with low interaction (low values in the statistical indicators), another (cluster-1) collects learners with a medium level of interaction, and the third (cluster-2) collects learners with high interaction (high values of statistical indicators). Then we measured the average collaboration level in each cluster (column "Level" of the next table).

Table 1. Cluster collaboration level average

Dataset	Cluster-0			Cluster-1			Cluster-2		
	N_msg	N_reply_ msg	Level	N_msg	N_reply_ msg	Level	N_msg	N_reply _msg	Level
D-06-07	17.12	10.65	4.38	32.46	26.92	6.15	46.06	38.71	6.74
D-07-08	8.86	6.03	4.79	22.45	17.63	5.61	44.78	39.38	6.11
D-08-09	14.05	11.10	5.14	33.55	26.61	5.75	48.26	44.89	6.74

Table 1 shows the average of the statistical indicator "num_msg" (number of messages sent to the forums), "num_reply_msg" (number of replies to the messages sent to the forums), and the average collaboration level (Level), which was supplied by the expert, in every cluster. The table shows just two statistical indicators because they define the clusters better, although the clustering algorithm EM used datasets with the 12 statistical indicators, which were explained above.

We concluded that the relation between collaboration (collaboration level supplied by the expert) and the clusters, and the statistical indicators is clear. Therefore, the most active learners (cluster-2), i.e., who sent more messages and whose statistical indicator "num_msg" is higher, and who caused more activity (statistical indicator "num_reply_msg" is higher) are the most collaborative learners. From this we can label learners according to their collaboration. Clusters-0 learners are labeled with low collaboration level, cluster-1 learners are labeled with medium collaboration level, and cluster-2 learners are labeled with high collaboration level. Considering the coverage of the evaluations performed over three consecutive academic years and the number of students involved, we can conclude that the relation between the collaboration level and the inferred representative collaboration indicators can be measured automatically, which was done this 2008-09.

6 Result Management

The year 2008-09 we used this method and learner collaboration levels were calculated during the collaborative period. The objective was not to calculate the exactly collaboration level. We argue that calculating the exact value of one variable in an environment, which is in imperfect scientific conditions, is very complicated. The method used offers rough information on the collaboration level, which can be used to improve learning.

We thought that we could show the collaboration level to the tutor of the collaborative environment so that the tutor improved the teaching. The same idea, however, could be applied to learners. Thus we showed learner's collaboration levels to the tutor and learners.

We prepared different ways of showing the information to learners.

- Statistical indicator portlet. We prepared a tool displaying the value of only 4 statistical indicators (num_thrd, num_msg, num_reply_thrd and num_reply_msg) of every week during the collaboration period. The objective was to give information on the interaction during the collaborative process to team-members.

- Collaboration level portlet. We proved that our data mining method reveals the rough learner collaboration level. This tool displays the collaboration level of team-members and the information was updated every week until the end of the collaboration process. The objective was to give information on the collaboration behavior of team-members.

We offered these tools to 2008-09 students. The statistical indicator portlet was offered to 6 teams (18 learners), the collaboration level portlet was offered to 8 teams (24 learners), and both portlets were offered to 6 teams (18 learners). The collaborative learning experience finished, but the academic year has not finished. We are currently analyzing learners' answers to an opinion questionnaire and the collaboration learning experience results to prove the usefulness of the portlets. We offered these questionnaires to teams who had used some tool. The results are explained in the next table.

Table 2. Evaluation of tools

Tools	No. of learners who could use the tool	No. of answers	Average rank [0, 5]
Statistical indicators	18	9	3.33
Collaboration level	24	13	3
Statistical indicators and collaboration level	18	12	3.08

Half of the learners or more, to whom some tool was offered, answered the questionnaire and they had to rank the tools between 5 (highest value) and 0 (lowest). The average rank of every tool is not really high but it is always over half values (2.5). The results are positive but the poor number of answers means that we should be cautions on their analysis. To improve the analysis of the questionnaire we are comparing the above results with the marks and the collaboration period evaluation by the tutor. The aforementioned questionnaire will be contrasted with students' marks from tutors' evaluations and final exams. The latter will be available next June.

7 Conclusion and Future Work

In this paper we have proposed a data mining approach to improve teaching and learning awareness on collaboration features in open collaborative learning frameworks. It infers learner collaboration levels and shows this information to tutors and learners. We thought that the data mining method covers the objective needed to improve the collaboration process. The objectives are: obtaining information on learner collaboration just after collaboration interactions have finished and guarantee domain independency. These objectives guarantee the data mining method can apply to others.

This research focused on obtaining information on the collaboration process using statistical indicators of learner interaction in forums, machine learning technology as the inferring method, and showing the inferred information to tutors as the approach to improve the collaboration process. We have proposed statistical indicators, which are related to the activity: initiative, regularity of the learners and the activity caused by the learners. We think the above features explain the collaborative work [17]. An EM clustering algorithm classified the learner statistical indicators and learner collaboration levels, which were provided by an expert, were used to validate the clustering classification as a collaboration level classification. This research took place over three academic years 2006-07, 2007-08 and 2008-09, and more than 100 students took part in the collaborative learning experience each year (125 in 2006-07, 140 in 2007-08 and 115 in 2008-09). During 2006-07 and 2007-08 the research focused on the inferring method [1] and this 2008-09 the results inferred were shown to learners and their usefulness measured.

The results have proved that the data mining method could reveal representative collaboration indicators and help learners to improve collaboration learning management.

We have proved the clustering approach infers information on the learners' collaboration, but we do not have any empirical conclusion claim that the clustering method is better than other machine learning methods, which can adapt itself to the problem. To clarify this issue we are carrying out parallel research where the inferring method relies on decision tree algorithms [2]. We are currently collecting results from the datasets so that we can subsequently compare the new results from the application of decision tree algorithms with the results reported in this paper. Another open issue is evaluating the tools offered. To date the evaluation has given satisfactions, but the tools could be improved. However, we must be cautions and wait until the results from the opinion questionnaire and the results from the exams and collaboration experience evaluation by the tutor are compared and analyzed.

References

[1] Anaya, A.R., Boticario, J.G. Clustering Learners according to their Collaboration. 13th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2009).

[2] Berikov, V., Litvinenko, A. Methods for statistical data analysis with decision trees. Novosibirsk, Sobolev Institute of Mathematics, (2003)

[3] Bratitis, T., Dimitracopoulou, A., Martínez-Monés, A., Marcos-García, J.A., Dimitriadis, Y. Supporting members of a learning community using interaction analysis tools: the example of the Kaleidoscope NoE scientific network Proceedings of the IEEE International Conference on Advanced Learning Technologies, ICALT 2008, 809-813, Santander, Spain, July 2008.

[4] Collazos, C.A., Guerrero, L.A., Pino, J.A., Renzi, S., Klobas, J., Ortega, M., Redondo, M.A., Bravo, C. Evaluating Collaborative Learning Processes using System-based Measurement. *Educational Technology & Society*, 10 (3), 257-274.

[5] Daradoumis, T., Martínez-Mónes, A., Xhafa, F. A Layered Framework for Evaluating OnLine Collaborative Learning Interactions". International Journal of Human-Computer Studies, Volume 64 , Issue 7 (July 2006), Pages 622-635

[6] Dringus, L.P., Ellis, E. Using data mining as a strategy for assessing asynchronous discussion forums. Computers & Education, 45(2005), 140-160.

[7] Gama, J., Gaber, M.M. (Eds), Learning from Data Streams: Processing Techniques in Sensor Networks, a book published by Springer Verlag, (2007)

[8] Gaudioso, E., Santos, O.C., Rodríguez, A., Boticario, J.G. A Proposal for Modelling a Collaborative Task in a Web-Based Learning Environment. 9th International Conference on User Modeling (UM'03). Workshop 'User and Group models for web-based adaptive collaborative environments'. Johnstown, Pennsylvania (United States)

[9] Hong, W. Spinning Your Course Into A Web Classroom - Advantages And Challenges. International Conference on Engineering Education August 6 – 10, 2001 Oslo, Norway

[10] Martínez, A., Dimitriadis, Y., Gómez, E., Jorrín, I., Rubia, B., Marcos, J.A. Studying participation networks in collaboration using mixed methods. International Journal of Computer-Supported Collaborative Learning. Volume 1, Number 3 / September 2006. 383-408

[11] Martínez, R., Bosch M., Herrero, M.M., Nuño, A.S. Psychopedagogical components and processes in e-learning. Lessons from an unsuccessful on-line course. Computers in Human Behavior 23, 146–161.

[12] Meier, A., Spada, H., Rummel, N. A rating scheme for assessing the quality of computer-supported collaboration processes. Computer-Supported Collaborative Learning (2) (2006) 63–86

[13] Meilâ, M., Heckerman, D. An Experimental Comparison of Model-Based Clustering Methods. Machine Learning, 42, 9–29, 2001

[14] Perera, D., Kay, J., Yacef, K., Koprinska, I. Mining learners' traces from an online collaboration tool. Workshop of Educational Data Mining (AIED'07).

[15] Redondo, M.A., Bravo, C., Bravo, J., Ortega, M. Applying Fuzzy Logic to Analyze Collaborative Learning Experiences in an e-Learning Environment.USDLA Journal. (United States Distance Learning Association).17.2, 19-28.

[16] Romero, C., Ventura. S. Educational data mining: A survey from 1995 to 2005. Expert Systems with Applications 33 (2007) 135–146

[17] Santos, O.C., Rodríguez, A., Gaudioso, E., Boticario, J.G. Helping the tutor to manage a collaborative task in a web-based learning environment. In: AIED 2003: Supplementary Proceedings. (2003) 153–162

[18] Soller, A., Martinez, A., Jermann, P., Muehlenbrock, M. From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. International Journal of Artificial Intelligence in Education, vol. 15, 2005, p. 261-290

[19] Strijbos, J-W., Fischer, F. Methodological challenges for collaborative learning research. Learning and Instruction 17 (2007)

[20] Talavera, L., Gaudioso, E. Mining Student Data To Characterize Similar Behavior Groups In Unstructured Collaboration Spaces. In: Proceedings of the Workshop on Artificial Intelligence in CSCL. 16th European Conference on Artificial Intelligence, (ECAI 2004), Valencia, Spain. 17–23. (2004)

[21] Witten, I. H., Frank, E. Data Mining. Morgan Kaufmann, June (2005).

Dimensions of Difficulty in Translating Natural Language into First Order Logic

Dave Barker-Plummer,[1] Richard Cox[2] and Robert Dale[3]
dbp@csli.stanford.edu, richc@sussex.ac.uk, rdale@science.mq.edu.au
[1] CSLI, Stanford University, CA, USA
[2] University of Sussex, UK
[3] Macquarie University, Australia

Abstract. In this paper, we present a study of a large corpus of student logic exercises in which we explore the relationship between two distinct measures of difficulty: the proportion of students whose initial attempt at a given natural language to first-order logic translation is incorrect, and the average number of attempts that are required in order to resolve the error once it has been made. We demonstrate that these measures are broadly correlated, but that certain circumstances can make a hard problem easy to fix, or an easy problem hard to fix. The analysis also reveals some unexpected results in terms of what students find difficult. This has consequences for the delivery of feedback in the Grade Grinder, our automated logic assessment tool; in particular, it suggests we should provide different kinds of assistance depending upon the specific 'difficulty profile' of the exercise.

1 Introduction

The translation of sentences in natural language (NL) into first-order logic (FOL) is a key part of the logic curriculum; indeed, it can hardly be said that a student understands formal logic if they are not able to carry out this translation task competently. For many students, however, it is a difficult task. The difficulties students face are, at least in part, due to characteristics of the natural language sentences themselves. For example, we would expect it to be relatively easy to translate a natural language sentence when the mapping from natural language into logical connectives is transparent, as in the case of the mapping from *and* to $\wedge$, but more difficult when the natural language surface form is markedly different from the corresponding logical form, as in sentences of the form *A provided that B*.

In this study, we aim to characterize translation tasks based on the student's responses to them. Rather than constructing a model of the student (as for example in [5]), we seek to model the task, and to do so in an empirically grounded way, rather than based on intuitions of the author of the exercise. This is in contrast with our past work, in which we focussed on the nature of the errors that students make when performing translation tasks [1].

1. *If **a** is a tetrahedron then it is in front of **d**.*
2. ***a** is to the left of or right of **d** only if it's a cube.*
3. ***c** is between either **a** and **e** or **a** and **d**.*
4. ***c** is to the right of **a**, provided it (i.e. **c**) is small.*
5. ***c** is to the right of **d** only if **b** is to the right of **c** and left of **e**.*
6. *If **e** is a tetrahedron, then it's to the right of **b** if and only if it is also in front of **b**.*
7. *If **b** is a dodecahedron, then it's to the right of **d** if and only if it is also in front of **d**.*
8. ***c** is in back of **a** but in front of **e**.*
9. ***e** is in front of **d** unless it (i.e., **e**) is a large tetrahedron.*
10. *At least one of **a**, **c**, and **e** is a cube.*
11. ***a** is a tetrahedron only if it is in front of **b**.*
12. ***b** is larger than both **a** and **e**.*
13. ***a** and **e** are both larger than **c**, but neither is large.*
14. ***d** is the same shape as **b** only if they are the same size.*
15. ***a** is large if and only if it's a cube.*
16. ***b** is a cube unless **c** is a tetrahedron.*
17. *If **e** isn't a cube, either **b** or **d** is large.*
18. ***b** or **d** is a cube if either **a** or **c** is a tetrahedron.*
19. ***a** is large just in case **d** is small.*
20. ***a** is large just in case **e** is.*

Figure 1. The 20 sentences in Exercise 7.12.

We are interested in developing a richer understanding of what it is that makes a translation exercise difficult. We identify two different measures of difficulty, and explore the relationship between them; this allows us to characterize each translation problem in terms of its **difficulty profile**, which in turn can be used to vary the kind of feedback provided.

Section 2 provides some background to the present study and introduces the corpus we use. Section 3 introduces our two measures of difficulty; Section 4 then discusses the relationship between these two measures, and what they reveal about the nature of the problems students face. In Sections 5 and 6, we focus on two particular difficulty profiles, exploring what they mean for our assessment tool. Finally, Section 7 provides some conclusions and points to future work.

2 The Data

The corpus consists of student-generated solutions to exercises in *Language, Proof and Logic* (LPL) [3], a courseware package consisting of a textbook together with desktop applications which students use to complete exercises.[1] Students may submit answers to 489 of LPL's 748 exercises online; the other exercises require that students submit their an-

[1] See http://lpl.stanford.edu.

swers on paper to their instructors. The electronic submissions are processed by the Grade Grinder (GG), a robust automated assessment system that has assessed approximately 1.8 million submissions of work by more than 46,000 individual students over the past eight years; this population is drawn from approximately a hundred institutions in more than a dozen countries. These submissions form an extremely large corpus of high ecological validity.

For the work reported here, we focus on a specific exercise we selected from LPL; this is a natural language (NL) to first-order logic (FOL) translation exercise of moderate difficulty, i.e. one that psychometrically discriminates between students. Exercise 7.12 from Chapter 7 (which introduces conditionals) was selected because it has the highest proportion of erroneous submissions of all translation exercises.

This exercise involves translating each of twenty English sentences into FOL. Our desktop applications offer relatively little useful feedback for exercises of this type, so compared to other exercise types, a higher proportion of submissions to the Grade Grinder contain errors. A **submission** for Exercise 7.12 consists of a solution for all twenty sentences, and is considered erroneous if the student makes an error on at least one of the solutions. For this study we focus on the calendar year 2007, during which period a total of 2558 students attempted this exercise. 14% got the exercise right without making a single error. The high proportion of submissions containing errors is in part a reflection of the fact that the exercises contains twenty translation tasks, all of which the student would have to have correct first time to avoid being found erroneous. For this study, we examined the corpus of erroneous submissions of Exercise 7.12, representing the work of a set of 2221 students.

A translation for a sentence (which we refer to here as a **solution**) is considered correct if it is equivalent to a **reference solution**; there are infinitely many correct answers for any sentence, so a theorem prover is employed to determine equivalence. The sentences from Exercise 7.12 are presented in Figure 1. The reference solution for Sentence 1 in Figure 1 is Tet(a) → FrontOf(a,d).

The emphasis in the Grade Grinder is on self-remediation of errors. The Grade Grinder's response to an erroneous submission of the form FrontOf(a, d) → Tet(a), a common error, takes the form:

> *** Your first sentence, "FrontOf(a, d) -> Tet(a)", is not equivalent to any of the expected translations.

This very uninformative, 'canned' feedback is typical of the Grade Grinder's responses. Students using the Grade Grinder may make as many submissions of a given exercise as they need to obtain the correct answer. This means that the corpus contains repeated submissions by the same student of the same task, and enables us to track their path to a solution. A student's work is reported to their instructor only when the student chooses, typically because it is reported as correct or because their deadline has arrived. Further

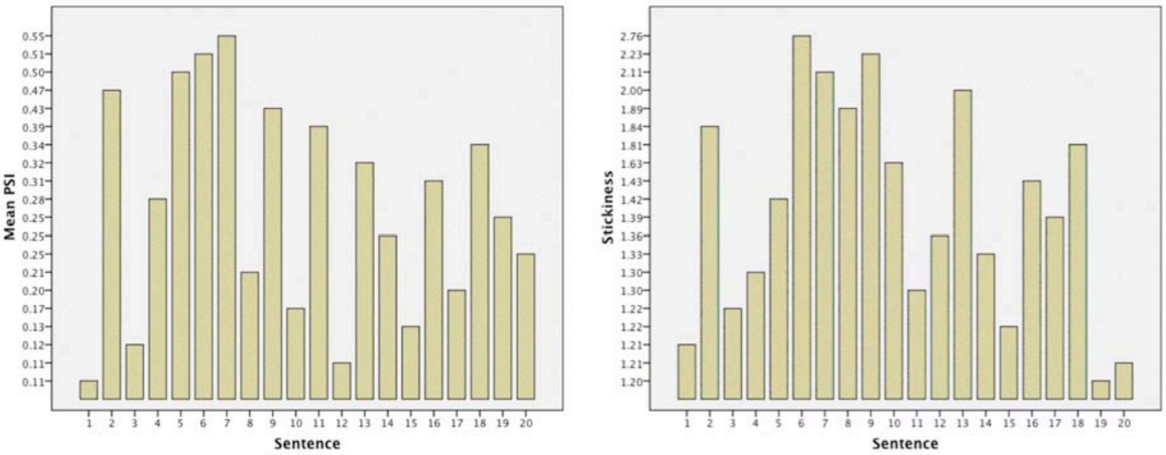


Figure 2. Proportion of students who get each sentence wrong (left), and average stickiness values for the 20 sentences (right).

information on the Grade Grinder, and samples of feedback reports, can be found on the GG website.[2]

3 Measuring Difficulty

How do we measure the difficulty of a translation exercise? Of course, the author of a textbook uses intuitions about difficulty when preparing the exercises used in that textbook. However, even when based on extensive experience, this invariably involves some degree of subjectivity, and it fails to acknowledge that different students may find different problems difficult to different degrees. Ideally we would like to determine the difficulty of an exercise on the basis of empirical data, and further, be able to take account of the fact that different students may face different problems.

We consider here two possible measures of difficulty based on the data we have available.

- First, we can look at the proportion of students who get a particular exercise wrong; the assumption here is that the more students who get an exercise wrong, the more difficult that exercise must be. We refer to this value as the sentence's **PSI** (for 'the *P*roportion of *S*tudents who provide an *I*ncorrect answer'.) Figure 2 (left) shows, for each of the 20 sentences, the proportion of this sample whose initial attempt at that sentence resulted in an incorrect answer. On this basis, we can observe, for example, that Sentences 1 and 12 are relatively easy, whereas Sentences 5, 6 and 7 are relatively difficult.

[2] See http://ggww2.stanford.edu/GUS/gradegrinder.

- A second measure can be obtained by considering how many attempts it takes for a student to determine the correct answer once they have made their initial mistake. We call this the **stickiness** of the error. Thus, every student has a stickiness value for every sentence they get wrong; for any sentence they correct on their second attempt, the stickiness value of that sentence for that student is 1. If we average this value over all students, we obtain a stickiness measure for the sentence. Figure 2 (right) shows the average stickiness values for the 20 sentences in the exercise based, for each sentence, on the subset of the 2221 students in the sample that made an error on it. This illustrates that Sentence 6 is much stickier—which is to say that it is 'harder to fix', even given the Grade Grinder's feedback—than Sentence 5 which with it shares a high PSI.

Since we have both values for each sentence in the data, we can combine these to produce a tuple we refer to as the **difficulty profile** of the sentence; this captures both the likelihood of a student getting the sentence wrong, and the average number of attempts it takes to fix the error.

4 Correlating the Dimensions

As noted above, 2221 students who made submissions to Exercise 7.12 made one or more errors. For each of these students, a co-occurrence matrix was constructed for the 20 sentences. Each cell of an individual student's matrix coded the relationship between one distinct pair of sentences. For example, if the student made an error on Sentence 3 and an error on Sentence 12, then the cell at [3,12] would be coded 1 (co-occurrence of error), otherwise zero. The individual subject matrices were summed to produce a combined matrix for all 2221 subjects.

The summed matrix was input to the SPSS Proximities procedure to produce a similarity matrix. The similarity matrix formed the input to the SPSS Cluster procedure, which was used to compute a multilevel, agglomerative, hierarchical cluster analysis using Ward's method (see, e.g., [4]). The item clusters are arranged hierarchically with individual items at the leaves and a single cluster at the root. Dendrograms provide a graphical display of cluster analysis output. The dendrogram for all subjects' data (Figure 3) shows the sentence clusters. Bifurcations that are more distant from the leaf nodes mean that the clusters are more dissimilar. For example, in Figure 3, the cluster of Sentences 1, 12, 3, and 15 indicates that many students who made errors on, say, Sentence 3 also tended to make errors on Sentences 1, 12 and 15. The primary branch at the root level indicates two quite distinct major clusters and three somewhat less dissimilar subclusters within the upper main branch (labelled 1–4 in Figure 3).

We then looked at where the individual sentences lay on a scatter plot of PSI against Stickiness; we noted that the dendogram clusters correspond to four distinct bands in Figure 4. The figure makes it clear that there is not a direct correspondence between our two mea-

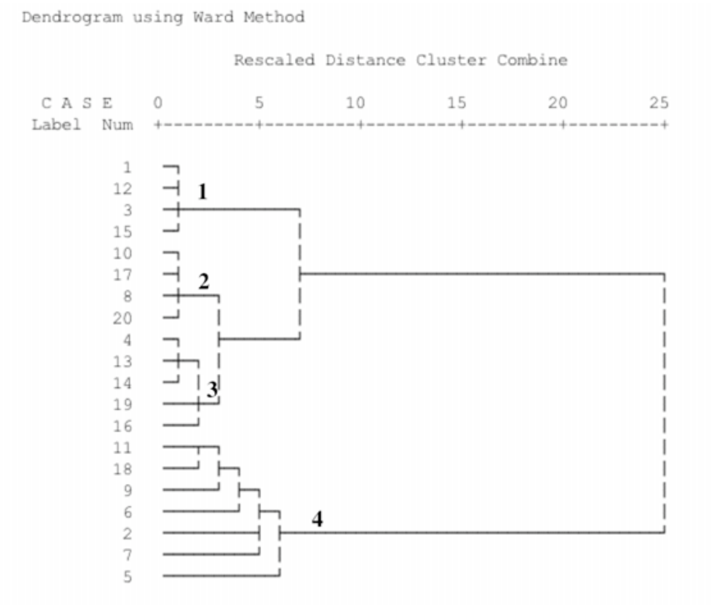

Dendrogram using Ward Method
Rescaled Distance Cluster Combine
C A S E
Label Num
0
5
10
15
20
25
1
12
3
15
10
17
8
20
4
13
14
19
16
11
18
9
6
2
7
5
1
2
3
4

Figure 3. Dendrogram representation of cluster analysis outputs. Leaf node numbers correspond to Exercise 7.12 Sentences 1–20. Clusters labelled 1, 2, 3 and 4 correspond to bands in Figure 4.

sures of difficulty, although there is reasonable correlation in the case of many sentences (Spearman's ρ = .61, p = .004). We single out for particular attention two situations involving outliers (and thus where our two measures of difficulty in some sense 'conflict'): sentences with a high PSI but a low stickiness, and sentences with a low PSI but a high stickiness.

5 Hard to Get Right, But Easy to Fix

We can characterise sentences with a high PSI and a low average stickiness as being hard to get right—many students get them wrong first time—but easy to fix: once you know you've got it wrong, it's easy to work out what the correct answer is.

The most salient examples of this category in our data are Sentences 19 and 20, repeated here for convenience:

19. **a** is large just in case **d** is small.
20. **a** is large just in case **e** is.

Both of these use the natural language expression *just in case*, which translates into FOL as the biconditional, '↔'. As we noted in [1], students find this expression particularly

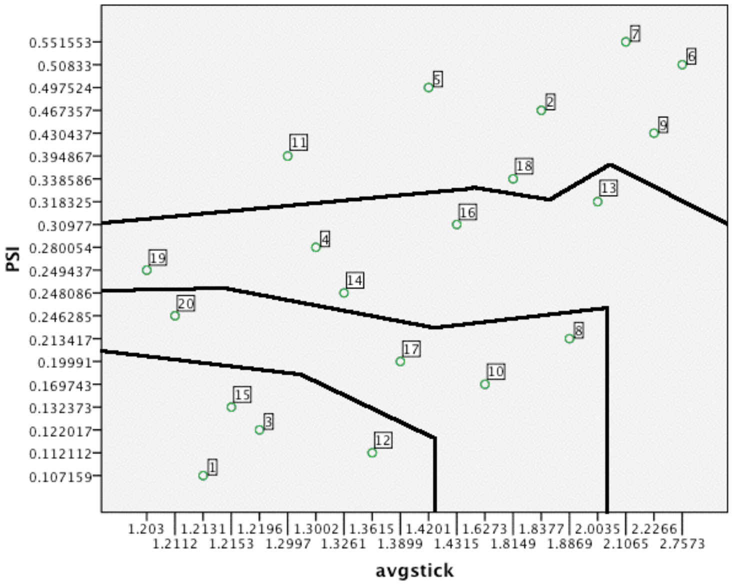


Figure 4. Scatterplot of the difficulty measures PSI and Stickiness. Data point labels indicate Sentences 1–20 of Exercise 7.12 and the ‘bands’ correspond to the four clusters identified in Figure 3.

difficult, perhaps because it is so rarely used (at least with this interpretation) in natural language.[3]

The vast majority of students who get Sentence 19 wrong offer either of the following two translations, reflecting the common misunderstanding of *just in case* as a bare conditional:

- Small(d) → Large(a)
- Large(a) → Small(d)

The analogous errors also occur very frequently for Sentence 20. In each case, we hypothesise that the students realise that the translation involves some kind of conditional; if they don’t get it right the first time, and incorrectly offer a conditional as in the cases above, the most obvious next alternative is the biconditional, which is the correct answer. In effect, there is a very small space of plausible alternative answers given the belief that some flavor of conditional is required, and therefore a relatively low likelihood of making a second error. This space of potential answers is further constrained by the simple nature of the sentences: they are amongst the shortest sentences in the exercise. Sentence 19 mentions only two objects (**a** and **b**) and two unary predicates (Large and Small).

[3] In everyday language, the expression *just in case* is most often used as an approximate synonym for ‘as a precaution’. This is not what is meant when it is used by logicians or mathematicians, as is explicitly taught in the LPL textbook.

This suggests that stickiness is related to the extent to which each submitted (or re-submitted) sentence translation by the student reduces the space of plausible solutions.

6 Easy to Get Right, But Hard to Fix

We can characterise sentences with a low PSI and a high average stickiness as being easy to get right—most students get them right first time—but hard to fix: if you get it wrong, it's hard to work out what the correct answer is.

Sentence 8 represents a reasonable example of this phenomenon:

8. **c** is in back of **a** but in front of **e**.

The implication of the difficulty profile of this example is that most students know how to translate *but* into FOL, but if a student doesn't understand this, being told they are wrong (the current feedback provided by the Grade Grinder) is not of any particular help. The high average stickiness of this sentence suggests that students are at a loss as to what the correct answer might be, perhaps even trying random variations on their initial solution to see what works.

In contrast to Sentence 19, discussed in the previous section, students are less likely to make errors on Sentence 8 and it is much stickier (Figure 2). That the PSI is low is probably explained by the fact that the logical connective that translates *but* is the relatively simple logical *and* ($\wedge$). This translation is introduced in an earlier chapter in LPL and may have been internalized by many students by the time they attempt this exercise.

The surface structure of the NL sentence suggests many possible reasons to suppose that students who err in their first attempt find this sentence sticky. Sentence 8 mentions three objects (**a**, **c**, and **e**) and involves two binary relations (BackOf, FrontOf), and the NL sentence contains an elided reference to **c** which has to be made explicit in the FOL translation. Given the uninformative feedback from the Grade Grinder, we conjecture that students do not know which of these features they have misunderstood. The situation is probably compounded by the fact that this exercise appears in the chapter of the book devoted to conditionals, but does not require a conditional in its translation.

7 Conclusions

What should the implications of the analysis presented here be? We have endeavoured to demonstrate that a unidimensional estimation of difficulty based simply on how many students get an exercise right or wrong masks important variations in the kinds of problems students face. In the analysis presented here, we distinguish the proportion of students who get an exercise wrong as a measure from a measure of how easy it is for a student to *correct* a wrong answer.

More broadly, we can identify four extremes that characterise individual problems:

Hard to Get Wrong, Easy to Resolve: Such problems may be of limited pedagogical value, although they might serve to build a learner's confidence.

Easy to Get Wrong, Easy to Resolve: These might play a role in encouraging care or vigilance, and so might be appropriate for delivery to a careless student.

Hard to Get Wrong, Hard to Resolve: These probably don't belong in the curriculum, since they are likely to engender frustration in the student.

Easy to Get Wrong, Hard to Resolve: These are the most challenging problems, perhaps best reserved only for those students who are on top of the curriculum.

None of these extremes are ideal; what we really want to do is use exercises that are more or less balanced, perhaps with a bias towards difficulty in one or other dimension for particular pedagogical purposes. As a student works through the LPL exercise set, we may be able to incorporate a measure of how they respond to exercises with different difficulty profiles into a student model. Ideally, appropriate examples should be dynamically generated automatically in response to this measure as it is revealed.

We would like to exploit our richer conception of logic exercise difficulty to enhance and enrich Grade Grinder's feedback and to individualise LPL's curriculum. To achieve this the Grade Grinder requires representations of the characteristics of sentences (such as number of predicates, number of constants, and arity). To this end we are currently developing knowledge representations of **stimulus features**: this task involves characterizing each LPL exercise in terms of its **resources**, represented as matrices of properties (for example, number of constants in NL or FOL sentences, number of predicates, types of predicates, their arity, number and types of connectives, and so on). Then we aim to derive, for each sentence, the space of plausible alternative answers based on the number of terms in a sentence, the number of predicates and their arities, *etc*. A large space of plausible alternative answers suggests *prima facie* a more difficult exercise. We will be able to validate the predictive difficulty measures by correlating them with PSI and stickiness, to determine whether these are all of the (and the only) factors playing into students' experiences with the exercise, and which of the surface features are predictive of PSI and which of stickiness.

The sentence characteristics (including the two difficulty measures and the stimulus features) could be encoded in a manner akin to q-matrices and used to represent students' **concept states** at different stages of learning [2]. Stimulus feature analysis of LPL's resources will also inform their decomposition into constituent sub-concepts and skills. Armed with this knowledge we should then be able to track how many times a student has encountered each sub-concept to-date. We may also employ search algorithms and statistical techniques (for example, multivariate logistic regression) to build models of each student's learning [5]. When the Grade Grinder detects that a student is manifesting a more-than-average number of errors, it can generate bespoke exercises for individual students using methods akin to those used in AI-based adaptive psychometric item generation [6]. Such approaches require, *inter alia*, rich representations of stimulus features, empirical data on item difficulty, and the application of item response theory (IRT) [7]. The ultimate aim is to generate exercises, judiciously adjusting the difficulty parameters and concepts they contain for each individual student.

References

[1] D. Barker-Plummer, R. Cox, R. Dale, and J. Etchemendy. An empirical study of errors in translating natural language into logic. In V. Sloutsky, B. Love, and K. McRae, editors, *Proceedings of the 30th Annual Cognitive Science Society Conference*. Lawrence Erlbaum Associates, 2008.

[2] T. Barnes. The q-matrix method: Mining student response data for knowledge. In J. Beck, editor, *Proceedings of the AAAI Workshop on Educational Data Mining*. AAAI Press, Menlo Park, CA., 2005.

[3] J. Barwise, J. Etchemendy, G. Allwein, D. Barker-Plummer, and A. Liu. *Language, Proof and Logic*. CSLI Publications and University of Chicago Press, September 1999.

[4] B.S.Everitt. *Cluster Analysis*. Edward Arnold, third edition, 1993.

[5] Hao Cen, K. Koedinger, and B. Junker. Automating cognitive model improvement by A* search and logistic regression. In J. Beck, editor, *Proceedings of the AAAI Workshop on Educational Data Mining*. AAAI Press, Menlo Park, CA., 2005.

[6] S. E. Embretson. Measuring human intelligence with artficial intelligence. In R. J. Sternberg and J. E. Pretz, editors, *Cognition and Intelligence: Identifying the Mechanisms of the Mind*, chapter 13. Cambridge University Press, Cambridge, UK, 2005.

[7] S. E. Embretson and S. P. Reise. *Item Response Theory for Psychologists*. Lawrence Erlbaum Associates, 2000.

Predicting Correctness of Problem Solving from Low-level Log Data in Intelligent Tutoring Systems

Suleyman Cetintas[1], Luo Si[1], Yan Ping Xin[2] and Casey Hord[2]
[1]{scetinta, lsi}@cs.purdue.edu, [2]{xin, thord}@purdue.edu
[1] Department of Computer Sciences, [2]Department of Educational Studies
Purdue University, West Lafayette, IN, 47906, USA

Abstract. This paper proposes a learning based method that can automatically determine how likely a student is to give a correct answer to a problem in an intelligent tutoring system. Only log files that record students' actions with the system are used to train the model, therefore the modeling process doesn't require expert knowledge for identifying domain specific skills that are needed to solve the problem or students' possible solution methods etc. The model utilizes a set of performance features, problem features, time and mouse movement features and is compared to i) a model that utilizes performance and problem features, ii) a model that uses performance, problem and time features. In order to address data sparseness problem, a robust Ridge Regression algorithm is designed to estimate model parameters. An extensive set of experiment results demonstrate the power of using multiple types of evidence as well as the robust Ridge Regression algorithm.

1 Introduction

Increasing trend in computers' utilization for teaching has led to the development of many intelligent tutoring systems (ITS). It has been shown that ITSs improve students' learning by providing individualized guidance by means of modeling students' cognitive skills while they solve problems [6, 11]. This approach, i.e. building a detailed model of students' domain specific cognitive skills, and updating them accordingly as students proceed is known as "model-tracing" [1] and have been followed by several tutors such as ANDES [5], SlideTutor [7], PAT [10] etc. For instance, the ANDES system utilizes a Bayesian network representation which is constructed from potential solutions of a current problem and is updated after each student action to determine when a student may need help or what method may potentially be used by the student to solve the problem. Building models that require construction of domain specific skills, and following students' progress at the skill level is an accurate way of student modeling; however it is time-consuming and requires experts' knowledge of the domain.

Instead of explicitly modeling students' cognitive skills with the help of a domain expert, it is possible to utilize the detailed low-level log data that ITS collect during the student-tutor interaction, to help the decision making process of a tutor. Robinet et al. have recently proposed a method to discover high-level student behaviors from low-level traces of students in a problem solving environment [14]. Their system uses domain dependent context-action-outcome (CAO) triplets extracted from the low-level log data and clusters them into high-level abilities (HLA) that can later be used for generating or selecting sets of exercises. Although their CAO triplets are domain dependent, their method is at a higher level than the level of the student resolution. Beck et al. note that it is not always obvious how to map the low level cognitive information to higher level teaching actions and propose a machine learning agent that directly learns to predict the probability of whether the student's response will be correct and how long it will take to generate that response [3]. They also note that it's possible to explicitly model how students generate their answers but is time-consuming. Instead, they use 4 groups of model inputs describing the current state of a student (using student, topic, problem, and context related features) and use linear regression to produce their two outcomes: i) whether the student will be able to solve the problem correctly, ii) how much time it will take him to solve it. They claim that any machine learning method doing function approximation will (in theory) work and what is important is the model inputs and outputs.

To the best of our knowledge, there is no prior research on the automatic detection of whether a student will be able to correctly answer a question with a high-level student model (i.e. without using any expert knowledge of the domain), utilizing mouse movement data. Prior work mainly focuses on the model-tracing approach [5, 7, 10] which is a quite different task than high-level student modeling. De Vicente and Pain have human participants use mouse movement data as well as data from student-tutor interaction for motivation diagnosis [8]. In a recent work, Cetintas et al. [4] also utilize mouse movement data along with performance and time features to automatically detect the off-task behaviors of students while they work with the tutoring system. However both of these works focus on tasks that are also very different than predicting the correctness of problem solving via a high-level student model. The works that focus on high-level student modeling utilize only combinations of performance, problem, context, topic, action, outcome related features (that can all be extracted from the low-level logs of user-system interaction) but ignore mouse movement data [3, 14]. Although these features are quite important to improve the effectiveness of student modeling, mouse movement data is another important type of data that can also be incorporated into high-level student modeling to improve the prediction accuracy. Similar to all other features that have been mentioned so far, mouse movement data can also easily be stored and retrieved from low-level user-system logs in every intelligent tutoring system.

This paper proposes a machine learning method that can automatically predict whether a student will be able to correctly answer a problem in a problem solving environment by utilizing multiple types of evidence including performance, problem, time and mouse movement features that are extracted from the log files of students' actions within tutoring software. To address data sparseness problem, the proposed model utilizes a robust Ridge Regression technique to estimate model parameters. The proposed model is compared to i) a model that utilizes performance and problem features; ii) a model that uses performance, problem and time features and iii) all models when data sparseness problem is not addressed (i.e. when model parameters are not learned with Ridge Regression). We show that utilizing multiple types of evidence as well as the robust Ridge Regression technique improves the effectiveness of the student model.

2 Data

Data from a study conducted in 2008 in an elementary school was used in this work. The study was conducted in mathematics classrooms using a math tutoring software (that has

Table 1. Details of the problem solving worksheets. The information of whether problems of a worksheet include i) diagram boxes can be seen under the *Include Diagram Boxes* column; ii) equation boxes can be seen under the *Include Equation Boxes* column; iii) unknown number to be solved for can be seen under the *Include Unknown Number* column. *Shows Correct Answer* column shows whether the correct answer to a question is shown to students after they submit their answer.

Worksheet	Includes Diagram Boxes	Includes Equation Boxes	Includes Unknown Number	Shows Correct Answer
EG/MC Worksheet 1	Yes	No	No	Yes
EG/MC Worksheet 2	Yes	Yes	Yes	Yes
EG/MC Worksheet 3	No	Yes	Yes	Yes
EG/MC Worksheet 4	No	Yes	Yes	No
Mixed Worksheet 1	No	Yes	Yes	Yes
Mixed Worksheet 2	No	Yes	Yes	No
Mixed Worksheet 3	No	Yes	Yes	No

Table 2. Details of the problems collected out of the problem solving sessions. Number of correctly solved problems for training, text splits can be seen under the *# of Correctly Solved Problems* column; number of incorrectly solved problems can be seen under the *# of Incorrectly Solved Problems* column and the total number of problems for training and test splits can be seen under the *Total* column. The percentages in the parenthesis indicate the ratio of positive and negative data for training and test splits as well as the total.

	# of Correctly Solved Problems	# of Incorrectly Solved Problems	Total
Training	713 (65.3%)	379 (34.7%)	1092
Test	578 (66.6%)	290 (33.4%)	868
Total	1291 (65.8%)	669 (34.1%)	1960

been developed by the authors). The tutoring software teaches problem solving skills for Equal Group (EG) and Multiplicative Compare (MC) problems. These two problem types are a subset of the most important mathematical word problem types that represent about 78% of the problems in a fourth grade mathematics textbook [12]. In the tutoring system; first, a conceptual instruction session is studied by a student followed by problem solving sections to test their understanding. Both of conceptual instruction and problem solving parts require students to work one-on-one with the tutoring software and if students fail to pass a problem solving session, they have to repeat the corresponding conceptual instruction and the problem solving session. Space limitations preclude discussing in detail but the tutoring software has a total of 4 conceptual instruction sessions and 11 problem solving worksheets that have 12 questions each (4 for Equal Group worksheets, 4 for Multiplicative Compare worksheets, 3 Mixed worksheets each of which include 6 EG & 6 MC problems) and is supported with animations, audio (with more than 500 audio files), instructional hints, exercises etc.

In a problem solving worksheet, a problem is counted as correctly solved only if all question boxes for the problem are filled correctly. Question boxes for a problem check students' ability to find the correct solution of the problem as well as to fulfill some partial skills that are needed for the solution of a problem when they can't give a full answer. Such partial skills for a problem include answers to i) diagram boxes which check student's mapping of the information given in a problem into an abstract model; ii) equation box which checks whether a student can form a correct equation from the information given in a problem; iii) final answer box which checks whether a student can solve the asked unknown in a problem correctly. Details about which worksheets include which groups of boxes are shown in Table 1. In some of the worksheets, after the student answers the question, the correct answer is also shown to the student after each question in the worksheet (regardless of whether the student's answer is correct or not) to make the student learn from his/her mistake. Details about which worksheets show correct answers are also shown in Table 1.

The study with the tutoring system included 8 students which include 3 students with learning disabilities, 1 student with emotional disorder and 1 student with emotional disorder combined with mental retardation. Students used the tutor for several 30 minute class sessions (on average 18.1255 sessions per student with standard deviation of 3.4408 sessions) during which their interaction with the tutoring system was logged in a centralized database. A total of 1960 problems that were solved, 1291 of which were correctly solved and 669 of which were incorrectly solved. The average number of correctly solved problems per student is 161.37 (with a standard deviation of 42.07) and the average number of incorrectly solved problems per student is 83.62 (with a standard deviation of 27.07). Data from 4 students were used as training data to build the models for making predictions for the other 4 students (who are used as the test data). Details about the training and test splits are given in Table 2.

3 Methods: Least Squares and Ridge Regression

Data sparseness is an important problem which is caused by using limited training data to learn parameters of a model and leads to the common problem of *over-fitting* [9]. Over-fitting as the name implies is the problem of having an excellent fit to the training data which may not be a precise indicator of future test data especially in the case of data sparseness. Regularization is a technique to control the over-fitting problem by setting constraints on model parameters in order to discourage them from reaching large values that lead to over-fitting. We will briefly discuss the Least Squares technique followed by Ridge Regression technique that controls over-fitting [9].

The simplest linear model for regression involves a linear combination of input variables as follows:

$$y(\boldsymbol{x},\boldsymbol{w}) = w_0 + w_1x_1 + \cdots + w_Dx_D = \boldsymbol{w^T x} \tag{1}$$

where $\boldsymbol{x} = (1, x_1, \ldots, x_D)^T$ is an instance of training data of D+1 dimensions and $\boldsymbol{w} = (w_0, w_1, \ldots, w_D)^T$ are model coefficients (w_0 is the bias parameter). For such a model, the sum of squares error between predictions $y(\boldsymbol{x}_n, \boldsymbol{w})$ for each data point $\boldsymbol{x}_n$ and the corresponding target values t_n is as follows:

$$E_D(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - y(\boldsymbol{x}_n, \boldsymbol{w})\}^2 = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \boldsymbol{w^T x}_n\}^2 \tag{2}$$

which can be minimized with a maximum likelihood solution that gives *the Least Squares* solution of the model parameters as follows:

$$\boldsymbol{w}_{ML} = (\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi t} \tag{3}$$

where $\boldsymbol{\Phi}$ is an N*D design matrix whose elements are given by $\boldsymbol{\Phi}_{\boldsymbol{nj}} = \boldsymbol{x}_{\boldsymbol{nj}}$ (i.e. jth dimension of the nth training instance). Ridge Regression adds a quadratic regularization penalty of $E_W(\boldsymbol{w}) = \mathbf{1/2}\,\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w}$ to the data-dependent error (i.e. Eq. (2)) with which the total error function becomes:

$$E_{TOTAL}(\boldsymbol{w}) = E_D(\boldsymbol{w}) + \lambda E_W(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \boldsymbol{w}^{\boldsymbol{T}}x_n\}^2 + \frac{\lambda}{2}\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} \tag{4}$$

where λ is the regularization coefficient that controls the relative importance of data-dependent error $E_D(\boldsymbol{w})$ and the regularization term $E_W(\boldsymbol{w})$. The regularization coefficient in this work is learned with cross validation in the training phase (i.e. splitting the training data into smaller training and test datasets). The exact minimizer of the total error function can be found in closed form as follows:

$$\boldsymbol{w}_{RIDGE} = (\lambda\boldsymbol{I} + \boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi t} \tag{5}$$

which is *the Ridge Regression* solution of the parameters of the model.

4 Modeling Approaches

This section describes the models that are used for evaluation: i) a model that considers performance and problem features, ii) another modeling approach that considers time features as well as performance and problem features, and finally iii) a more advanced model that incorporates mouse movement features with performance, problem and time related features.

4.1 Performance and Problem Based Modeling (PerfProb_Mod)

Using performance and problem based features has been shown to be a useful approach for student modeling in the prior work [3, 14]. The idea of using performance features is quite intuitive since students' performance up to a certain problem is a good indicator of their performance for that problem. Similarly problem related features such as problem difficulty or number of sub skills (types of question boxes in this work) required etc., are very informative to see whether a current student can correctly answer a problem or not.

In this work; 4 performance features are used. The set of 4 performance features are used as a measure of the probability that the student knew the skills asked in a question. The first feature is the *# of correct answers so far* in a problem solving worksheet. Each problem solving worksheet consists of 12 math word problems and a problem is counted as correct only if all question boxes for the problem are filled correctly. The number of correctly solved problems up to a current problem in a worksheet is a good indicator for student's success for the current problem. Second, third and fourth performance features help to assess student's partial skills that are needed for the solution of a problem when they can't give a full answer. Such partial skills for a problem include the abilities to give answers to, as mentioned before, i) diagram boxes which check student's mapping of the information given in a problem into an abstract model; ii) equation box which checks whether a student can form a correct equation from the information given in a problem; iii) final answer box which checks whether a student can solve the asked unknown in a problem correctly. The corresponding features are *percentage of correct diagram answers so far*, *percentage of correct equation box answers so far, percentage of final answers so far* in a problem solving worksheet. They provide the percentage of correct answers given by a student for the associated partial skill boxes of all the solved problems of a current worksheet up to the current problem.

In addition to the 4 performance features, 11 problem features are also used indicating which problem solving worksheet the current problem belongs to. In our model, there are 11 binary variables corresponding to 11 worksheets. If a current problem belongs to 5th worksheet (i.e. MC Worksheet 1), then 5th binary variable will be 1 and all others will be 0. This encoding approach enables the model to associate each problem with the different characteristics of different worksheets. This encoding scheme is also mentioned in Beck's work as "one hot" encoding [3].

Performance and problem based modeling in this work serves as the baseline for all other models and will be referred as PerfProb_Mod.

4.2 Performance, Problem and Time Based Modeling (PerfProbTime_Mod)

Performance and problem based modeling approach is useful in many situations however there are lots of other possible data that can be good indicators of students' success for a current problem such as the time that a student spends while solving a problem. Although not all the prior work used time related features [14], it has been used as a feature by Beck [3].

In addition to the 15 performance and problem features mentioned before, this modeling approach also incorporates the time feature for student modeling. The time feature in this work is defined as the time a student spends while solving a problem.

Performance, problem and time based modeling approach will be referred as PerfProbTime_Mod.

4.3 Performance, Problem, Time and Mouse Tracking Based Modeling (PerfProbTimeMouseT_Mod)

Incorporation of the time feature into the performance and problem based modeling is an effective way of improving student modeling; however there is still more room to improve. Both performance & problem based modeling and performance, problem & time based modeling approaches ignore an important data source with which students are almost always in interaction while they are solving problems in a problem solving environment: the mouse. As far as we know there is no prior research on student modeling that utilize mouse tracking data. More details about the prior work on this

Table 3. Results of PerfProbTimeMouseT_Mod method is shown in comparison to PerfProb_Mod and PerfProbTime_Mod methods for high level student modeling to detect whether a student can correctly solve a given problem. Note that the results for each model for the technique of least squares are shown under the *Least Squares* column, and the results for each model for the technique of Ridge Regression are shown under the *Ridge* Regression column. The percentages in the parenthesis show the relative improvements of each method with respect to the Least Squares version of the PerfProb_Mod model. The performance is evaluated by the F_1 measure.

Methods	**Technique**	
	Least Squares	**Ridge Regression**
PerfProb_Mod	0.4632	0.6749 (+45.70%)
PerfProbTime_Mod	0.5090 (+09.89%)	0.6805 (+46.91%)
PerfProbTimeMouseT_Mod	0.5249 (+13.32%)	0.6894 (+48.85%)

modeling approach as well as utilizing mouse movement data can be found in the Introduction section.

In addition to the 4 performance related features, 11 problem related features and 1 time feature that have been mentioned; this modeling approach incorporates 3 more features as mouse tracking data. The first feature is the *maximum mouse off time* in a problem which provides the knowledge of the biggest time interval (in seconds) in which mouse is not used for a current problem. Second and third mouse tracking features are the *average x movement* and *average y movement* respectively. They basically assess average number of pixels the mouse is moved along the x and y axes in 0.2 second intervals.

Performance, problem, time and mouse tracking based modeling that we propose will be referred as PerfProbTimeMouseT_Mod.

5 Experimental Methodology: Evaluation Metric

To evaluate the effectiveness of the off-task behavior detection task, we use the common F_1 measure, which is the harmonic mean of precision and recall [2,13]. Precision (p) is the ratio of the correct categorizations by a model divided by all the categorizations of that model. Recall (r) is the ratio of correct categorizations by a model divided by the total number of correct categorizations.

$$F_1 = \frac{2pr}{p+r} \qquad (6)$$

6 Experiment Results

This section presents the experimental results of the methods that are presented in Methods section. All the methods were evaluated on the dataset described in Data section.

An extensive set of experiments are conducted to address the following questions:

- How effective is the PerfProbTime_Mod method that utilizes performance, problem and time features with respect to PerfProb_Mod method that utilizes performance and problem features?
- How effective is the PerfProbTimeMouseT_Mod method that utilizes mouse tracking data as well as performance, problem and time features with respect to PerfProb_Mod and PerfProbTime_Mod methods?
- How effective is the approach of utilizing the Ridge Regression technique to estimate the model parameters?

6.1 The Performance of Performance, Problem and Time Based Modeling (PerfProbTime_Mod)

The first set of experiments was conducted to measure the effect of including the time feature in the PerfProb_Mod model. The details about this approach are given in detail in Section 4.1.

More specifically, PerfProbTime_Mod model is compared with PerfProb_Mod and their performances are shown in comparison to each other in Table 3. It can be seen that the PerfProbTime_Mod model outperforms PerfProb_Mod model. The lesson to learn from this set of experiments is that time feature is helpful when it is combined with performance and problem related features for the task of predicting whether a student will be able to correctly answer a current problem. This explicitly demonstrates the power of incorporating the time feature into the performance and problem related based modeling.

6.2 The Performance of Performance, Problem, Time and Mouse Tracking Based Modeling (PerfProbTimeMouseT_Mod)

The second set of experiments was conducted to measure the effect of including the mouse tracking data in the PerfProbTime_Mod model. The details about this approach are given in detail in Section 4.2.

More specifically, PerfProbTimeMouseT_Mod method is compared with the other two models and its performance is shown in comparison to the other two models in Table 3. It can be seen that the PerfProbTimeMouseT_Mod model outperforms both PerfProbTime_Mod and PerfProb_Mod models. This set of experiments show that mouse movement features are helpful when they are combined with performance and problem related features along with the time feature for high level student modeling. This explicitly demonstrates the power of incorporating the mouse tracking features into performance, problem and time based modeling.

6.3 The Performance of Utilizing the Robust Ridge Regression Technique

The last set of experiments was conducted to measure the effect of utilizing the technique of Ridge Regression for learning the model parameters for each of the models. The details about this approach are given in detail in Section 3.

More specifically, Ridge Regression learned models are compared to Least Squares learned models. The performance of Ridge Regression versions of each model is shown in comparison to Least Squares versions in Table 3. It can be seen that the Ridge Regression version of each model outperforms Least Squares versions with its regularization framework. This confirms that Ridge Regression models better solve the data sparseness problems in this application.

7 Conclusion and Future Work

This paper proposes a novel machine learning method for high-level student modeling (that doesn't require any expert knowledge of the domain to extract skills, or possible solutions that students may follow) to detect if a student can correctly solve a current problem in a problem solving environment while using an intelligent tutoring system. This model relies only on the low-level log data that is available from the log files from students' actions within the software. The proposed model makes use of a set of evidence such as performance, problem, time and mouse movement features and is compared to i) a model that utilizes performance and problem related features, ii) a model that uses performance, problem and time features together. To address data sparseness problem, the proposed model utilizes a robust Ridge Regression technique to estimate model parameters.

An extensive set of empirical results show that the proposed method that automatically detects whether a student will be able to correctly answer a problem substantially outperforms the model that uses performance and problem related features as well as the model that utilizes performance, problem and time features together. Furthermore empirical results show that the proposed model attains a better performance by utilizing the technique of Ridge Regression over the standard Least Squares Regression technique.

There are several possibilities to extend the research. For example, different students have different types of characteristics for solving problems (e.g. using more or less time to solve the problems; having difficulties with particular types of questions and/or problems or different mouse usage types etc.). Therefore, personalized models tend to provide more accurate detection results than a single model for all students. Future research work will be conducted mainly in this direction.

Acknowledgements

This research was partially supported by the NSF grants IIS-0749462 and IIS-0746830. Any opinions, findings, conclusions, or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsor.

References

[1] Anderson, J.R., Boyle, C.F., Corbett, A., Lewis, M. Cognitive Modeling and Intelligent Tutoring. *Artificial Intelligence*, 1990, 42, p. 7-49.

[2] Baeza-Yates, R. & Ribeiro-Neto, B. Modern Information Retrieval. *Addison Wesley*, 1999.

[3] Beck, J.E., Woolf, B.P. High Level Student Modeling with Machine Learning. *Proceedings of the Intelligent Tutoring Systems Conference,* 2000, p. 584-593.

[4] Cetintas, S., Si, L., Xin, Y. P., Hord, C. & Zhang, D. Learning to Identify Students' Off-task Behavior in Intelligent Tutoring Systems. *Proceedings of the 14th International Conference on Artificial Intelligence in Education,* 2009, to appear.

[5] Conati, C., Gertner, A.S., VanLehn, K. Using Bayesian Networks to Manage Uncertainty in Student Modeling. *User Model, User Adapt, Interact,* 2002, 12(4), p. 371-417.

[6] Corbett, A.T. and Anderson, J.R. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User Adapted Interaction,* 1995, 4, p. 253-278.

[7] Crowley, R., Medvedeva, O. and Jukic, D. SlideTutor: A model-tracing Intelligent Tutoring System for teaching microscopic diagnosis. *Proceedings of the 11th International Conference on Artificial Intelligence in Education,* 2003.

[8] De Vicente, A. and Pain, H. Informing the detection of the students' motivational state: an empirical study. *Proceedings of the Intelligent Tutoring Systems Conference,* 2002, p. 933-943.

[9] Hastie, T., Tibshirani, R. & Friedman, J. The Elements of Statistical Learning. *Springer,* 2001.

[10] Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A. Intelligent Tutoring Goes to School in the Big City. *Intelligent Journal of Artificial Intelligence in Education,* 1997, 8.

[11] Koedinger, K.R., Corbett, A.T., Ritter, S., Shapiro, L.J. Carnegie Learning's Cognitive Tutor: Summary of Research Results. *White Paper, Carnegie Learning,* 2000, Pittsburg, PA.

[12] Maletsky, E. M. et al. Harcourt Math, *Indiana Edition*, *2004,* Chicago: Harcourt.

[13] Rijsbergen, C. J. v. Information Retrieval*, 2nd edition,* 1979*,* University of Glasgow.

[14] Robinet, V., Bisson, G., Gordon, M. B., Lemaire, B. Inducing High-Level Behaviors from Problem-Solving Traces Using Machine-Learning Tools. *IEEE Intelligent Systems,* 2007, 22, 4, p. 22-30.

Back to the future: a non-automated method of constructing transfer models

Mingyu Feng and Joseph Beck
{mfeng, josephbeck}@wpi.edu
Computer Science Department, Worcester Polytechnic Institute

Abstract. Representing domain knowledge is important for constructing educational software, and automated approaches have been proposed to construct and refine such models. In this paper, instead of applying automated and computationally intensive approaches, we simply start with existing hand-constructed transfer models at various levels of granularity and use them as a lens to examine student learning. Specifically, we are interested in seeing whether we can evaluate schools by examining the grain-size at which its students are best represented. Also, we are curious about whether different types of students are best represented by different transfer models. We found that better schools and stronger students are best represented by models with a fewer number of skills. Weaker students and schools are best represented, for our data, by models that allow no transfer of knowledge in between skills. Perhaps surprisingly, to accurately predict the level at which a student represents knowledge it is sufficient to know his standardized test score rather than indicators of socio economic status or his school.

1 Introduction

The topic of representing domain knowledge is fundamental in the construction of intelligent tutoring systems (ITS). This representation is important not only because it denotes the language used in constructing the tutor (e.g. the level at which to construct hints), but also because it makes claims about the level at which students represent knowledge and transfer it between problems. For this reason, such models are sometimes called transfer models [7].

Given the importance of transfer models, it is not surprising that their construction has been a major focus in the educational data mining (EDM) community. For example, Barnes has done considerable work with trying to induce transfer models, in this work called q-matrices [4], from data [1, 2]. Winters [16] has compared a variety of statistical approaches for constructing transfer models, including cluster methods such as k-means and dimensionality reduction such as non-negative matrix factorization. One common thread of this work is that it produces models that are typically more compact than those created by experts. This difference is both a source of strength (perhaps students learn differently than experts believe?) and a source of weakness (if the models are less understandable or make it harder to represent pedagogical knowledge why should we use them?). Although it would be an expensive undertaking, we are unaware of a controlled study showing that a tutor using automatically constructed model provides superior teaching compared to a tutor using to hand-constructed transfer model (or vice versa). Rather than inferring a transfer model from scratch, there is a hybrid approach called learning factors analysis (LFA) [9]. This technique starts with a transfer model, typically built by hand, and computationally tries various modifications to the model to better align it with student performance data (e.g. see [5, 6]). Although LFA is intuitively appealing, it has not demonstrated any dramatic improvements in model fit. Although its potential to shed light on scientific questions, such as the level of knowledge that learners use to represent written words [11], is a great benefit, it is unfortunate that modifying hand-created models does not result in substantially stronger[1] models.

Rather than trying to invent another complex and computationally intensive technique, we take an alternate view of the problem. We know from prior research that students of differing proficiency have somewhat different representations of the domain, with more skilled learners having a more compact (i.e. coarser) representation [11]. We also know that different tutorial interventions influence the representation that learners acquire, with better interventions causing learners to develop a more compact representation [10]. A common fallacy is the belief that finer-grain models will fit learner data better, or at least will fit better given sufficient training data, since they are able to represent subtler distinctions in the domain. This belief is incorrect since fine-grained models not only make subtle distinctions in skills, they (typically) also assert that skills are independent of each other. So practice in one skill does not help with another. If learners are able to transfer knowledge amongst skills, a coarser-grain model will better fit performance data. Given these results, perhaps it makes more sense to skip over automated techniques and simply start with transfer models at various levels of granularity and use them as a lens to examine student learning. In this way, we can still do interesting science with our large datasets but do not have to focus on complex machinery that might not be that helpful.

The goal of this paper is a case study in hand-constructed models of various grain sizes in interpreting data collected from an ITS. Specifically, we are interested in whether models of different granularity better fit distinct subgroups, and, consequently, whether we can use this approach to evaluate schools by examining the grain-size at which their students are best represented. Given two schools where one is better predicted by a coarser transfer model, that school is probably the better one. This approach is different than simply looking at which school has the highest test score performance. If a weaker school changes its curriculum and its students have a better mental model of the domain and are transferring better, they might still lag a stronger school in raw knowledge and consequently in test scores. This approach can potentially detect such schools. We can validate this hypothesis in two ways. First, we have an idea of the quality of the schools we are evaluating (although the person interpreting the data did not). Second, instead of partitioning students by school, we can use their state assessment test score and partition them by math proficiency. If we see a trend for stronger students, it is reasonable to believe it applies to stronger schools.

The advantage of this approach is that it is easy. Also, if one transfer model does a better job at a particular school, since that model is expert-constructed it should not be (any more) difficult (than usual) to construct tutorial content for the model; whereas automated models might not fit educators' understanding of the domain. Also, since there are a

[1] By "substantially stronger" we do not mean statistically reliably different. We acknowledge there have been changes in Bayesian Information Criterion (BIC) scores that correspond to reliable improvements, but it would be difficult to distinguish such a tutor built with a revised transfer model from the one built with the original.

limited number of grain sizes for our model, there is a definite limit on the amount of content creation that is required. For this research, we use data collected as part of the ASSISTment project (www.assistment.org) as our testbed.

2 The ASSISTment system

The ASSISTment system [13] is a web-based system that presents math problems to students who range from approximately 12 to 16 year-olds. When a student has trouble solving a problem, the system usually provides instructional assistance to lead the student through by breaking the problem into scaffolding steps, or displaying hint messages on the screen, upon student request. Each ASSISTment question consists of an *original question* and a list of *scaffolding questions*. The original question usually has the same text as found in the Massachusetts Comprehensive Assessment System (MCAS) test while the scaffolding questions were created through breaking the original question down to the individual steps by our content experts. A student is initially presented a question that usually has several skills needed to solve it correctly. If the student gets the question correct he can move on to next question, otherwise he is forced to go through a sequence of scaffolding questions (or scaffolds). Students work through the scaffolding questions, possibly with hints and buggy messages, until they eventually get the problem solved. Student actions and tutorial responses are time-stamped and logged into our database.

3 Methods

3.1 Construction of different grain sized transfer models

A fine grained model was constructed during a seven hour long “coding session” in 2005 at WPI where our subject-matter expert and the ASSISTment project director created a set of skills and used those skills to tag all of the existing 8th grade MCAS items. They imposed the limit that no one item would be tagged with more than three skills. Thus, many of our ASSISTment System questions had three scaffolding questions; we wanted the fine grainedness of the modeling to match the fine grainedness of the scaffolding. During the “coding session”, the subject-matter expert reviewed the problems and conducted a cognitive task analysis to identify what knowledge was needed to perform each task. When the coding session was over, we wound up with about a model of 106 skills, called the WPI-106 model. To create the coarse-grained models, we used the fine-grained model to guide us. We decided to use the same five broad strands that were used by the Massachusetts Department of Education to tag each MCAS item with exactly one strand. Since our mapping was inferred from the WPI-106, it was not the same as the state’s mapping. Therefore, it was named the WPI-5. Furthermore, we allowed multi-mapping, i.e., allowing an item to be tagged with more than one skill. Similarly, we adopted the name of the 39 learning standards (nested inside the five strands) in the Massachusetts Curriculum Framework, associated each skill in the WPI-106 to one of the learning standards, and thus we created the model WPI-39. This process is illustrated in Table 1. After the students had taken the state tests, the state released the items in that test, and our subject-matter expert tagged up these items in all the transfer models.

The first column in Table 1 lists eight of the 106 skills in the WPI-106 model. For instance, *equation-solving* is associated with problems involving setting up an equation and solving it; while *equation-concept* is related to problems that have to do with equations in which students do not actually have to solve them. The two skills are nested inside of “Patterns, Relations and Algebra” in the third column which itself is one piece of the five skills that comprises the WPI-5 transfer model. The value of the fine grained model was shown in [14] by analyzing of data from over 1000 students’ two years usage of ASSISTment system. In [14], we presented evidence that, in general, the WPI-106 model did a better job at tracking students’ knowledge and, thus, made a more accurate prediction of their end-of-year exam scores than the coarser grained models.

Table 1. Hierarchical relationship among transfer models.

WPI-106	WPI-39	WPI-5	WPI-1
Inequality-solving	Setting-up-and-solving-equations	Patterns, Relations, and Algebra	Math
Equation-solving			
Equation-concept			
X-Y-graph	Understand-line-slope-concept		
Congruence	Understand-and-applying-congruence-and-similarity	Geometry	
Similar-triangles			
Perimeter	Using-measurement-formulas-and-techniques	Measurement	
Area			

3.2 Approach

We have explained the nested hierarchical structure of our transfer models, and shown that the fine-grained model did the best *overall* at predicting student performance. Now we will examine our results more closely to see how different transfer models fit *different groups* of students.

3.2.1 Data

The dataset we use was collected during 2004-2005 school year. It involves 495 8th-grade students (approximately 13 years old) from two middle schools who have used the ASSISTment system on at least 6 days, with an average of 9 days. The item-level MCAS test report is available for all students so that we are able to evaluate accuracy of our models at state test score prediction. Since the scaffolding questions show up only if the students answer the original question incorrectly, students who answer the original question correctly do not have a chance at scaffolding questions, and would only be credited for the original question in the data. In order to avoid this selection effect, we preprocess the data using a compensation strategy to mark all scaffolding questions correct if a student gets an original question correct. Also, because our transfer models allow multi-mapping (one question associated with multiple skills), we choose to use a simple credit-blame strategy where if a student succeeds in answering a question, we mark all associated skills as being correctly applied, while when a student answers a question incorrectly, we only blame the weakest skill of the student, i.e. the skill on which the student has shown worst performance. After preprocessing, the data set contains 147,624 data points, among which 45,135 come from original questions. On average, each student answers 91 original questions. It is worth pointing out that during our modeling process, student response on original questions and scaffolding questions are used in an equal manner and they have the same weight in evolution.

The first portion of this research involves partitioning students into groups to determine if different groups of students have different patterns for learning math skills. Naturally, the 495 students can be separated by the schools they were in, with 312 from school F and 183 from school W. We also try to separate them by their performance level at the 2005 MCAS test. The high performing group includes the 128 students whose performance level is assessed by the state as “Advanced” or “Proficient”; the medium group includes the 154 students whose performance level is “Needs Improvement”, and the low performing group has the rest 213 students at performance level “Warning”. While these performance levels are somewhat specific to Massachusetts, they are at least criterion-referenced and much more general than numbers extracted from a student model or raw scores on a test (what qualifies as “Proficient” in Massachusetts is probably similar to “Proficient” in Macedonia). Our hypothesis is that students from a stronger school, or higher performing group, would show more transfer in their knowledge acquisition than those from a weaker school, or lower performing groups. Therefore, for the stronger students and schools the coarser grained model will better describe their learning and provide more accurate prediction of their MCAS test scores.

3.2.2 Modeling

In order to track individual student’s development of skills over time and make predictions, we choose to fit mixed-effects logistic regression models [8]. A mixed-effects model consists of both *fixed effects*, parameters corresponding to an entire population or repeatable levels of factors, and *random effects*, parameters corresponding to individual subject drawn randomly from a population. This approach takes into account the fact that responses of a student on multiple items are correlated. Moreover, the random effects allow the model to learn parameters for individual students separately. We use a logistic model because our dependent measure is dichotomous (0/1 for incorrect/correct). Regarding to the independent variables, for the fixed effects, we used a timing variable to represent the amount of time elapsed since the beginning of the school year, so that the model tracks the knowledge acquisition process longitudinally over time. Skills are included in the model as a factor to identify the skills associated with each response. Both the main effects of skills and an interaction term between the timing variable and skills are included in the model. Therefore, the model will learn an intercept (representing initial knowledge) and a slope (representing learning rate) for each skill separately. The timing variable is introduced as a random effect as well, in order to account for the learning rate variation of each individual student. The model is illustrated as below. To simplify the illustration, suppose TIME is the only covariate we care about in the model (*skill* can be introduced in a similar way). Thus, a 2-level representation of the model in terms of *logit* can be written as

Level-1 model: $$\log[\frac{p_{ij}}{1-p_{ij}}] = b_{0i} + b_{1i} * TIME_{ij}$$

Level-2 model: $$b_{0i} = \beta_0 + v_{0i}$$
$$b_{1i} = \beta_1 + v_{1i}$$

Where p_{ij} is the probability that student *i* gives a correct answer at the *j*th opportunity of answering a question;

TIME

$TIME_{ij}$ refers the *j*th opportunity when student *i* answered a question. In our data, it is a continuous value representing the number of months (assuming 30 days in a month) elapsed since the beginning of the school year.

b_{0i}, b_{1i} denote the two learning parameters for student *i*. b_{0i} represents the “intercept” or how good is the student’s initial knowledge; b_{1i} represents the “slope” that describes the change (i.e., learning) rate of student *i*.

β_0, β_1 are the fixed-effects and represent the “intercept” and “slope” of the whole population average change trajectory.

v_{0i}, v_{1i} are the random effects and represent the student-specific variance from the population mean.

We fit the mixed-effects logistic regression models with R (http://www.r-project.org/) using the glmer() function in the *lme4* package [3], using “logit” as a link function. For simplicity, assuming knowledge was changing linearly (in logistic space) over time. One model is fit for each school and each performing group separately. Given a student’s learning parameters on different skills, the skill-tagging of each MCAS question, and the exact test date of MCAS, we can calculate the probability of positive response from the student to each MCAS test question. Then we sum the probabilities up as the prediction of students’ MCAS scores. Two prediction evaluating functions are chosen, mean absolute difference (MAD), and mean difference (MD), as below.

$$MAD = \frac{1}{n}\sum_{i=1}^{n}\left|MCAS_i - prediction_i\right| \qquad MD = \frac{1}{n}\sum_{i=1}^{n}(MCAS_i - prediction_i)$$

where $MCAS_i$ is the actual MCAS score of the ith student, and $prediction_i$ is the predicted score from our model. Both measures are used since MAD gives a good estimate the closeness of the prediction to actual scores while MD allows us to see if a certain model has been overestimating or underestimating.

3.3 Results and discussion

The results for both school F and school W are summarized in Table 2. As shown in Table 2, school F has a flat error line across all four different transfer models. The MAD for the WPI-39 model is the lowest, and yet a paired t-test that compares the absolute pair-wise differences of individual students among all models suggested that there is no reliable difference. However, for school W, the line tilts: the MAD of the WPI-39 model is reliably lower than those of the WPI-1 and WPI-5 models, indicating school W is better predicted by a finer grained model than by coarser grained models. Note that we are not able to fit the statistical model for school W with the WPI-106 transfer model (there is a technical glitch we do not understand and are investigating). We encounter the same problem later in the paper, which admittedly bring up some caveats in interpreting our results. The second part of Table 2 shows the values of MD for each model. The results indicate that both schools are optimized at the WPI-39 model. In general, student performance on the state test is overestimated by our models except that the WPI-106 model underestimates school F; and school W is even more overestimated than school F across known results from all the three models. As we know that, theoretically a one-skill model assumes perfect transfer. Since that is unlikely to happen, it would tend to overestimate student performance. And for a weaker school, perfect transfer is even more improbable. Thus, the overestimation would be greater since students are probably learning a collection of 106 unrelated skills. The tendency of overestimate decreases as the granularity of transfer models increases, and a very fine grained model such as the WPI-106 model that assumes no transfer or very low transfer may even underestimate when there is actually some level of knowledge transfer. We can see that in Table 2, the MD goes from negative to positive when we use the WPI-106 model for School F. Given these results, based on our hypothesis we would predict school F is the stronger school. An examination of both schools' MCAS performance reports (for current achievement) and information on their Annual Year Progress (AYP, for changes in performance) confirms our prediction.

Table 2. Results for students grouped by schools

Results	School	WPI-1	WPI-5	WPI-39	WPI-106
MAD	School F	4.188	4.168	4.124	4.175
	School W	4.669	4.601	4.329	N/A
MD	School F	1.362	0.932	0.477	-1.000
	School W	3.043	2.867	2.012	N/A

Table 3. Results for students grouped by performance levels

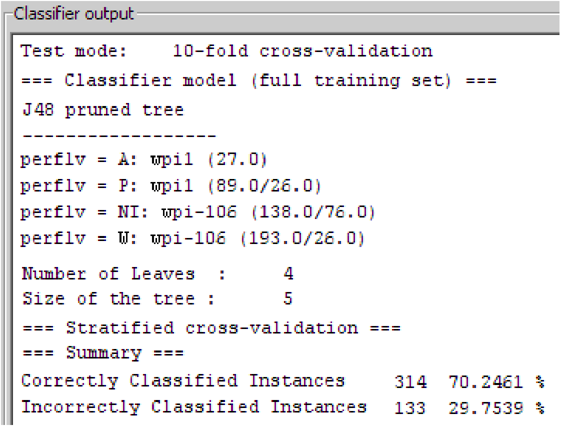
Results	Performance Level	WPI-1	WPI-5	WPI-39	WPI-106
MAD	Advanced/Proficient	2.673	2.834	2.489	3.249
	Needs improvement	3.180	3.243	2.900	N/A
	Warning	4.027	4.092	3.518	N/A
MD	Advanced/Proficient	-1.726	-2.034	-1.210	-2.715
	Needs improvement	1.534	1.744	0.893	N/A
	Warning	3.023	3.136	2.212	N/A

As mentioned in section 1, a second validation approach is that instead of partitioning students by school, we can use their state assessment test score and partition them by math proficiency. If we see a trend for stronger students, it is reasonable to believe it applies to stronger schools. Therefore, as reported in section 3, we split all the 495 students into 3 groups based on their state test performance level, and fit a mixed-effects logistic regression model to each group separately for different transfer models. The values of MAD and MD are summarized in Table 3. We see a slight support with MAD: for the students at the high end, the WPI-39 does the best job at predicting their state test scores, reliably better than the other three models, while the WPI-106 model does reliably worse than the WPI-1 and WPI-5 models, suggesting there is certain amount of knowledge transfer happening with the high performing students. However, since we do not obtain results of the WPI-106 model for the other two groups, it is hard to draw a conclusion there. When it comes to the MD measure, we notice some support as well. Obviously, the advanced and proficient students have been underestimated by all models, and the amount of underestimation goes worst when the finest grained model, the WPI-106 model, is applied. On the contrary, the medium and low performing students are all overestimated under all the models. Just as we hypothesize, the finer grained models overestimate less than the coarser grained models, and the better performing, stronger groups are less overestimated than the weaker groups. Therefore, weaker students are better represented by transfer models that are finer-grained.

3.4 A bottom-up aggregation approach

Rather than starting with an *a* priori disaggregation, we now focus on treating students as individuals and discovering commonalities among students who are best-fit with a particular transfer model. We have collected demographic data about several properties of a student, such as which school he/she goes to, ethnicity, gender, etc. Finding out the relation among these properties and which transfer model best fits this student is our goal. Our plan is to bring together model-fitting information and student characteristics, and then use a machine learning classifier to determine the best-fit model. This bottom-up aggregation is a strong alternative to proposing and testing disaggregation, and will scale nicely as we get more descriptors for each student.

For this purpose, we first re-fit models for all the students as one group[2] and identify which model best fits each individual student. The best-fit model information is then combined with other properties of the student in a new data set. Specifically, the properties we use are: gender, free-lunch status (indicative of family income), special education status, ethnicity, and state test performance level. These properties are picked because they are easy to access, and all of them have meanings to researchers working with other populations in other locations. In comparison, properties such as the school a student attends are much less useful to those in other locations. Given the new data set, we built a J48 (C4.5 revision 8) decision tree in Weka 3.6 [15]. The constructed J48 pruned tree is show in Figure 1 that tells how the classifier uses the attributes to make a decision. The constructed tree is extremely simple with just 5 nodes. The WPI-1 model is overall the best fitting model for Advanced (A) and Proficient (P) students, and the WPI-106 is for "Needs improvement" (NI) or Warning (W) level students. The numbers in brackets after the leaf nodes indicate the number of instances assigned to that node, followed by how many of those instances is incorrectly classified as a result. In our case, the correct classification rates are relatively good for students at performance level of A, P, and W. Yet, for students at performance level of NI, even though the WPI-106 model is the best fit, it is not dominant with 76 out of 138 instances misclassified. It is encouraging that this simple decision tree can achieve a predictive accuracy of over 70% during stratified cross-validation. Although the decision tree only uses MCAS performance, it was provided with the variables described above but was unable to find a use for them. This result suggests the appropriate level of transfer model granularity really seems to depend on student knowledge, rather than on variables that may correlate with knowledge such as family wealth. Therefore, if tutor designers have students with rather different levels of knowledge, they might wish to use different levels of their skill hierarchy. This point



```
Test mode:    10-fold cross-validation
=== Classifier model (full training set) ===
J48 pruned tree
------------------
perflv = A: wpi1 (27.0)
perflv = P: wpi1 (89.0/26.0)
perflv = NI: wpi-106 (138.0/76.0)
perflv = W: wpi-106 (193.0/26.0)

Number of Leaves  :     4
Size of the tree :      5

=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances     314   70.2461 %
Incorrectly Classified Instances   133   29.7539 %
```

Figure 1. Result of classifying in Weka

[2] We had to reduce the number of students to 447 from 495 because of a memory limit of R.

does not contradict the use of evaluating interventions [10] and schools by model granularity: other properties certainly matter in how well knowledge transfers, but for our dataset they are not as predictive as the student's knowledge.

4 Contributions, Future work, and Conclusions

The contribution of this paper lies in several aspects. First, automated techniques for revising transfer models for better knowledge representation have shown no huge improvements in accuracy but have addressed interesting scientific questions. Is there a way we can do interesting science on educational data sets and avoid the "irritating" automation step? Our answer is "yes," if it is possible to build a hierarchy of transfer models with different granularity. Previous experience tells us that this is not a rare thing to have, and not very hard to think about. The hierarchy can be used for runtime benefit of intelligent tutoring systems such as the control of mastery learning or generation of feedback messages for students of various proficiency levels. It can also be used to evaluate schools and be validated via high stake test performance. Second, through the usage of a bottom-up aggregation approach, the problem is changed. Rather than trying to automate the model search, why don't we automate seeing which student best fits which model? Third, we argue that hand-created transfer models and a bottom-up approach to aggregating students is a better use of human brains and computational power than approaches that focus search efforts on revising the domain model. Better understanding what parts of the scientific enterprise can be best done by people and which are better done computationally is a major issue in EDM.

A major open question of this work is whether just because a student is best modeled at a coarser grain size, shall we use such a model to drive tutorial instruction? For example, even though strong students are best modeled by a single skill "Math," it is not obvious how one would design hint messages in a system that only recognized one skill. A hybrid approach would be to track student knowledge and drive mastery learning at a coarser grain size, but provide feedback using a finer-grained model. A second question is that, since student knowledge is changing over time, perhaps we should use different level models to represent a student at different points in his learning?

In this paper, we start with existing hand-constructed transfer models at various levels of granularity, and use them as a lens to examine student learning. Specifically, we start by examining whether we can evaluate schools by determining the grain-size at which its students are best represented. We also examined what models best fit students at different levels of proficiency, and found some support for the idea of stronger students being better fit with coarser transfer models. The most interesting analysis was the bottom-up aggregation and using classification to find clusters of students who learn similarly. This analysis suggests transfer model granularity really seems to be about student knowledge. Finally, we argue that it is more productive to focus analytical effort on which students should use which transfer models rather than on automatically refining those models.

Acknowledgements

We thank Dr. Neil Heffernan for his help and insightful comments on this work. This research was made possible by the U.S. Department of Education, Institute of Education Science (IES) grants #R305K03140 and #R305A070440, the Office of Naval Research grant # N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. All the opinions, findings, and conclusions expressed in this article are those of the authors, and do not reflect the views of any of the funders.

References

[1] Barnes, T. (2005). Q-matrix Method: Mining Student Response Data for Knowledge. In Beck. J (Eds). *Educational Data Mining: Papers from the 2005 AAAI Workshop*.

[2] Barnes, T. (2006). Evaluation of the q-matrix method in understanding student logic proofs. *Proceedings of the 19th International Conference of the Florida Artificial Intelligence Research Society* (FLAIRS 2006), Melbourne Beach, FL, May 11-13, 2006.

[3] Bates, D. (2007). Linear mixed model implementation in *lme4*. University of Wisconsin, 15 May 2007.

[4] Birenbaum, M., Kelly, A., & Tatsuoka, K. (1993). Diagnosing knowledge states in algebra using the rule-space model. *Journal for Research in Mathematics Education*, 24(5), 442-459.

[5] Cen, H., Koedinger, K, & Junker, B. (2005). Automating cognitive model improvement by A* search and logistic regression. In Beck. J (Eds). *Educational Data Mining: Papers from the 2005 AAAI Workshop*. Technical Report WS-05-02. Menlo Park, California: AAAI Press. pp. 47-53.

[6] Cen, H., Koedinger, K., & Junker, B. (2006). Learning factor analysis – A general method for cognitive model evaluation and improvement. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*. Springer-Verlag: Berlin. pp. 164-175.

[7] Croteau, E., Heffernan, N. T. & Koedinger, K. R. (2004). Why are Algebra word problems difficult? Using tutorial log files and the power law of learning to select the best fitting cognitive model. In J.C. Lester, R.M. Vicari, & F. Parguacu (Eds.) *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*. Berlin: Springer-Verlag. pp. 240-250.

[8] Hedeker, D. & Gibbons, R. D. (2006). Longitudinal Data Analysis. Hoboken, NJ: John Wiley & Sons.

[9] Koedinger, K. & Junker, B. (1999). Learning factor analysis: Mining student-tutor interactions to optimize instruction. Presented at Social Science Data Infrastructure Conference. New York University. November, 12-13, 1999.

[10] Koedinger, K. R., & Mathan, S. (2004). Distinguishing qualitatively different kinds of learning using log files and learning curves. In the Working Notes of the ITS2004 Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes.

[11] Leszczenski, J. M., & Beck, J. E. (2007, July 9). What's in a word? Extending learning factors analysis to modeling reading transfer. *Proceedings of the AIED2007 Workshop on Educational Data Mining*, Marina del Rey, CA, 31-39.

[12] Newell, A, & Rosenbloom, P.S. (1993). Mechanisms of skill acquisition and the law of practice. In P. S. Rosenbloom, J. E. Laird, & A. Newell (Eds.), *The Soar Papers: Research on integrated intelligence*. Cambridge, MA: MIT Press.

[13] Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R, Walonoski, J.A., Macasek. M.A., Rasmussen, K.P. (2005). The Assistment Project: Blending Assessment and Assisting. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th International Conference on Artificial Intelligence in Education*, pp. 555-562. Amsterdam: ISO Press.

[14] Feng, M, Heffernan, N., Heffernan, C. & Mani, M. (in press). Using mixed-effects modeling to analyze different grain-sized skill models. To appear in *the IEEE Transactions on Learning Technologies Special Issue on Real-World Applications of Intelligent Tutoring Systems*.

[15] Weka 3: Data Mining Software in Java. http://www.cs.waikato.ac.nz/ml/weka/

[16] Winters, T., Shelton, C., Payne, T., & Mei, G. (2005). Topic Extraction from Item-Level Grades. In Beck. J. (Eds). *Educational Data Mining: Papers from the 2005 AAAI Workshop*. Menlo Park, California: AAAI Press. pp. 7-14.

How do Students Organize Personal Information Spaces?

Sharon Hardof-Jaffe[1], Arnon Hershkovitz[1], Hama Abu-Kishk[2], Ofer Bergman[3], Rafi Nachmias[1]
{sharonh2, arnonher, nachmias}@post.tau.ac.il, hama@bgu.ac.il, o.bergman@sheffield.ac.uk
[1] Knowledge Technology Lab, School of Education, Tel Aviv University, Israel
[2] Department of Communication Studies, Ben-Gurion University, Israel
[3] Information Studies Department, Sheffield University, UK

Abstract. The purpose of this study is to empirically reveal strategies of students' organization of learning-related digital materials within an online personal information archive. Research population included 518 students who utilized the personal Web space allocated to them on the university servers for archiving information items, and data describing their directory hierarchies. Several variables for measuring folders size and depth were defined, and four of them were chosen as best representing different aspects of the user's archive structure. Then, as a result of cluster analysis of the students, four organization strategies emerged, refining the classical piling/filing classification: piling, one-folder filing, small-folders filing, and big-folder filing. Also, associations were found between the organization strategies and archive size, students' studies degree. A discussion of this study and further research is provided.

1 Introduction

Personal information management (PIM) is an emerging research field focusing on the activities by which a person keeps, saves and organizes information items in order for her or him to later retrieve them [4]. In the current knowledge age, PIM has a central role in learning processes, as students create and collect many information items, and organize them into personal information archives. During PIM activities, students construct knowledge regarding the subject matter as they collect, evaluate, choose, tag, sort, classify and name information items. The purpose of this study is to investigate students' organization strategies of personal archives using data mining techniques.

Previous research have identified two main organizational strategies for PIM: Piling and Filing [14]. The pilers are those who tend to gather many items in the main documents directory (e.g., "My Documents" for files, "Inbox" for e-mails). The filers, by contrary, tend to sort the items into labeled folders, according to some categorization. The resulted structure of the personal information space reflects the user's organization strategy, hence examining students' archives might shed light on how they deal with PIM activities [2, 8].

Over the years, PIM studies have heavily relied on traditional data-collection methodologies which usually allow only a small number of participants, thus their external validity is limited. Recently, data mining methods have been suggested as enabling identification and measurement of PIM activities and personal information space structures for large populations [9, 11]. During this study, we have investigated online storage space used by university students using data mining techniques, in order to identify students' personal information space organization strategies. Applying data mining techniques on data drawn from online storage spaces presents PIM-related research with new and fascinating opportunities, and is the core of this research.

2 Background

2.1 PIM and Learning

The nature of information has dramatically changed in the digital era, as information is easily accessible, mostly distributed, presented in multiple formats, and hypertext-oriented. While learning, students create personal information spaces, negotiating between the huge amount of available information - from various resources and environments - and their limited processing abilities at any given time. Students therefore need to acquire Personal Information Management (PIM) literacy in order to efficiently manage their own learning environment, which is normally associated with the nature of the subject matter and the assignment requirements [16].

PIM literacy [16] is not just a set of practical actions of saving and retrieving information items; it is an integral and a centric part of the learning process, as through it, and by constructing an information archive, students construct knowledge. The constructive approach to learning emphasizes the fact that knowledge is constructed through a process in which learners actively integrate new knowledge with previous knowledge [7]. During the process of information seeking, students organize collected items into an information construction, by using cognitive skills, such as naming, sorting and categorizing [13].

Bloom's Taxonomy of Educational Objectives [5] presents six levels of cognitive skills: knowledge, comprehension, application, analysis, synthesis, and evaluation. The three main filing skills (i.e., naming, sorting, categorizing) might be related to different levels in Bloom's taxonomy: a) *Knowledge* "includes those behaviors […] which emphasize the remembering, either by recognition or recall of ideas, material or phenomena" (p. 62). By *naming* a folder, the student has to recall some basic knowledge about the files within it, or to recognize their main theme, in order to define and label it; b) *Analysis* is the separation of materials or concepts into component parts, during which the student "is required to determine their connections and interactions" (p. 145) and to recognize their organizational principles. When *sorting* materials, students select the related folder(s) for each of the new information items, hence explicitly identify the relationships among the items using the hierarchy; c) *Synthesis* is defined as "putting together elements and parts so as to form a whole" (p. 162). In order to construct a personal information space, many items are being combined together to form a hierarchical structure – a process which requires *categorization* skills.

These PIM activities are part of a process of integrating new knowledge into previous constructed knowledge as any information item the student adds to her or his personal information space, is being connected to the other items by its location in the hierarchy. While information items are being connected, knowledge, analysis and synthesis skills are constantly being applied in a spiral process during which the personal information space is being formed and is continually evolving. Therefore, we believe that PIM activities have an inherent learning component.

2.2 PIM Organization Strategies

Malone [14] was the first to classify Personal Information Management (in the context of office organization) into two types of strategies: Piling and Filing. The Piling style is characterized by papers being heaped on top of each other (latest papers are on the top of the heap), with the pile carries no label. Filing is characterized by papers being distributed into physical files, labeled according to a certain categorization (determined by the filer). Malone found that piles were useful for small collections, where the users could still remember the location of each paper within the pile, however as piles grew users could not keep track of their papers.

The folder hierarchy is the standard mechanism for organizing personal information in digital environments. This mechanism allows users to create a personal classification scheme, based on categories and dimensions they see as relevant (e.g., role, project, time). In today's offices, papers are replaced by digital information items (e.g., files, e-mails), filing is done into directories (folders) with labels referring to their category, and piling is typically done by heaping the information items in a root directory, such as "My Documents" for files and "Inbox" for e-mails. Previous research has shown that most of the users tend to employ a mixture of Piling and Filing [19].

The binary classification of Piling/Filing was refined by many other PIM classifications, and was extended mainly to describe different filing activities over time (i.e., *when* do users file their files?) [1, 6, 20]. In the context of learning, strategies were defined regarding the creation time of new folders: a) *Pre builders* - students who create new folders before they produce any items to put in them; b) *Post builders*, who prefer to create new folders after a set of new items is collected [8]. Our study is aiming on refining the Piling/Filing classification, based on empirical data describing personal online archives.

2.3 Data Mining Methods in PIM Research

Data describing how users organize their personal information space had been usually collected by means of traditional research methodologies, e.g., in-depth interviews, semi-structured interviews, screen captures, and questionnaires [3, 6]. Over the last few years, data mining has been suggested as a promising methodology for PIM research, and several PIM studies have already demonstrated the strength of this approach [11, 18]. For example, Clustering algorithms were used for identifying groups of files (on desktop) having the same context, and for grouping together email messages according to their content [10, 15], demonstrating the collection and analysis of large datasets, which would not have been possible using traditional methods.

The main purpose of this study is to empirically examine personal information space organization strategies in the context of learning processes on a large population of students, in order to refine the traditional piling/filing classification.

3 Methodology

3.1 Research Field

Tel Aviv University enables each of its students to keep and manage personal information items on the Web, within the university's Learning Content Management System (*HighLearn* by Britannica Knowledge Systems Inc.), which serves about 26,000 students and comprises of over 4,300 courses [17]. Users of this environment can upload files, create folders, and retrieve files by navigating or searching.

3.2 Research Population and Data File

The study was conducted on data describing online archives of 2,081 undergraduate students, graduate students and staff who kept information items in their virtual personal directory. The data included the list of files and folders (full paths) for all the users, where each personal information space had a unique random identification. The raw data included more than 70,000 rows, each of which refers to one file or folder. Data were collected on August 2008. After excluding students with less than 10 files in their archive, a new data file for analysis was created, holding 48,744 rows of 518 students.

3.3 Procedure

In order to examine different strategies for personal information space organization, four variables describing the organization were chosen and computed for each student: 1) *Files per folder* – average folder size; 2) *Largest folder* – number of files in the largest folder, including root directory; 3) *Pile rate* – ratio between pile size (root directory) and archive size (total number of files); and 4) *Inner-pile rate* - ratio between the largest folder size (not including root directory) and archive size (total number of files). *Files per folder* and *largest folder* were transformed for having a maximum value of 30, and 100 accordingly, in order to normalize their distribution. Then, Two-step Cluster Analysis of the students into k disjoint groups was applied (using SPSS), in order to classify students according to their personal information space organization strategy by the four variables. After several iterations, k=4 was chosen as resulting in the best fitting clustering.

4 Results

A short descriptive statistics of the data file is given in Table 1. On average, each student has 80.52 (SD=170.17) files and 13.58 (SD=45.33) directories.

Table 1. Descriptive statistics for the four describing variables

Variable	Minimum	Median	Maximum	Mean (SD)
Files per folder	0.34	10.55	235	16.16 (23.06)
Largest folder	1	16	339	27.15 (35.24)
Pile rate	0	0.17	1	0.38 (0.40)
Inner-pile rate	0	0.20	1	0.28 (0.28)

After clustering the students according to the four variables, we have calculated means and SD for each variable within each cluster; results are given in Table 2, where maximum and minimum values for each variable are **bolded** and *italicized*, accordingly.

Table 2. Means (SD) of the four variables by which the clusters were formed

Cluster	N	Files per folder	Largest folder [# files]	Pile rate	Inner-pile rate
1	141	17.78 (7.33)	22.71 (16.60)	**.97** (.08)	*.02 (.06)*
2	49	14.70 (7.49)	18.77 (8.53)	*.09 (.11)*	**.86** (.13)
3	262	*6.10 (4.49)*	*14.52 (11.55)*	.18 (.20)	.26 (.16)
4	66	**23.10** (7.67)	**71.62** (28.00)	.13 (.19)	.48 (.27)
All	518	12.26 (8.97)	24.42 (24.19)	.38 (.40)	.28 (.28)

As might be seen from the table, Cluster 1 (n=141) is characterized by extreme values of two variables' means among clusters: *Pile rate* gets a maximum (0.97), and *inner-pile rate* gets a minimum (0.02). These results imply that in this cluster, most of the students' files are stored in the root directory (hence it is not surprising that the second largest folder is extremely small). These two extreme values of variables are typical for Piling organization strategy.

In Cluster 2 (n=49), again the means of the same two variables as in Cluster 1 get to their extreme values, however in different direction. In this cluster, the mean of *pile rate* is minimal (0.09), and we may think that this is a non-piling strategy. However, the mean of *inner-pile rate* is relatively high (0.86), which indicates on the existence of a folder holding a large share of the archive. That means that the files were saved in one main folder out of the root directory – a strategy that we may call One-folder Filing.

Cluster 3 (n=262) has minimum mean values for two variables: *Files per folder* and *Largest folder*, i.e., students in it have small folders on average (6.1), and their largest folder is also relatively small (14.52). This suggests that the cluster represents a Small-folders Filing organization strategy.

In Cluster 4 (n=66), the means of the same two variables as in Cluster 3 take their extreme values: Both *files per folder* (23.1) and *largest folder* (71.62) are maximal. By examining the mean value of *pile rate* (0.13), it might be concluded that about 87% of their files are filed, with one folder containing about half of their files (0.48). Therefore, this cluster, which we call Big-folder Filing, describes a mixture of filing and piling.

According to this analysis of the clusters, we present the following classification of personal information space organization strategies: Piling, One-folder Filing, Small-folders Filing, and Big-folder Filing. Table 3 shows the distribution of the four types in the research population.

Table 3. Personal Information Space Organization Strategies distribution

Personal information space organization strategies (cluster number)	N	% of students
Piling (1)	141	27
One Folder Filing (2)	49	9
Small Folders Filing (3)	262	51
Big Folder Filing (4)	66	13

For examining the association between the archive size and its organization strategy, mean values for archive size (total number of files) were compared between the clusters. Using Univariate ANOVA test, it was shown that the means are significantly different. As may be seen from Table 4, two strategies (Piling, One-folder Filing) have a small archive size on average (24.4 and 22.31, respectively), while the largest mean value for archive size (284.73) was found in the Big-folder Filing cluster. This indicates that larger archives are associated with strategies of filing into more than one directory.

Table 4. Archive size in the different clusters

Personal information space organization strategies (cluster number)	N	Archive size statistics			
		Minimum	Median	Maximum	Mean *(SD)*
Piling (1)	141	10.00	17	155.00	24.40 (20.30)
One-folder Filing (2)	49	10.00	18	52.00	22.31 (10.85)
Small-folders Filing (3)	262	10.00	38	967.00	70.18 (101.04)
Big-folder Filing (4)	66	42.00	144	2170.00	284.73 (369.04)

5 Discussion

The main purpose of this study was to empirically identify different types of personal information organization strategies, which are part of Personal Information Management (PIM), and to do so for a large population, using data mining methodologies. PIM is not only a coherent and integral part of the learning process in the digital era - it is a process through which students learn. Therefore, researching PIM in the context of learning is very important for having a broader understanding of the learning process. Applying data mining techniques for PIM research brings new and fascinating opportunities to this field, as was demonstrated in this study.

Focusing on users' management of online personal archives, we were able to empirically identify four types of archiving strategies: a) Piling – most of the files are in the root directory; b) One-folder Filing – most of the files are located in one folder, under the root directory; c) Small Folders Filing – items are being divided into many relatively small folders (about 6 files per folder on average); d) Big-folder Filing – items are being divided into folders (about 23 files per folder on average) with about a half of them located in one big folder. These four types refine the classical Filing/Piling binary classification [14]. As the results suggest, students who tend to be Big-folder Filers, manage the largest archives and have relatively many files per folder on average. In order to construct a hierarchy of large coherent folders of different items related to a certain context (represented by each folder's name), students are required to a meaningful integration and generalization processes regarding the subject matter.

Our analysis showed that more than half of the participating students were categorized as Small-folders Filers. As this strategy is characterized by the use of small folders, this might imply that there are relatively many near-empty folders. Empty folders might indicate on a pre-building strategy, as was previously observed in the context of students' PIM [8]. Having many empty folders might increase PIM complexity, as well as having big folders. The strategy of Big-folder Filing was found in this study as associated with large archives, supporting previous findings [11].

In the context of learning, increasing PIM complexity is of special interest as PIM activities require cognitive skills. Bloom's cognitive taxonomy for learning objectives [5] enables us to analyze the three main PIM activities – i.e., naming, sorting, and categorization – in the light of three different levels of the taxonomy's cognitive skills: knowledge, analysis, and synthesis, accordingly. Regarding the four personal organization strategies found in this study, we might suggest different levels of reflected activities. In Piling strategy, the students neither name, sort nor categorize any information items. In One Folder Filing strategy, the students name only few folders and don't sort or categorize at all. In Small Folders Filing, the students name folder and sort information items into them, however they only do little categorization (since they join only few items into each folder). Only in Big-folder Filing strategy, students name, sort and categorize many items into folders. As the results suggest, managing bigger archives requires a wider range of cognitive skills. Replicating the process described in this article over several points in time might enlighten issues regarding changes over time of PIM strategies and their related cognitive activities.

PIM is subjective and idiosyncratic, and because PIM research mostly uses qualitative data collection from relatively small populations, it might seem that there are as many PIM variations as there are researched users [12]. However, using a large research population and data mining techniques, unexpected patterns might arise, suggesting similarities between groups of users, as was shown in this study. To promote the creation of large datasets, Chernov et al. [9] have suggested building a repository of PIM activity log files; this then would serve the PIM research community. Since it is likely that there will be problems obtaining participants' consent to trace their PIM activity over time, it might be easier to collect structural data reflecting accumulating activity.

References

1. Abrams, D., Baecker, R., and Chignell, M. (1998). Information archiving with bookmarks: personal Web space construction and organization. *Proceedings of the SIGCHI conference on Human factors in computing systems*. Los Angeles, California, United States: ACM Press/Addison-Wesley Publishing Co.

2. Barreau, D. (2008). From Novice to Expert: Personal Information Management Behaviors in Learning Contexts. *CHI 2008 Workshop*. Florence, Italy.

3. Bergman, O., Beyth-Marom, R., and Nachmias, R. (2008). The user-subjective approach to personal information management systems design: Evidence and implementations. *The American Society for Information Science and Technology*, 59(2), 235-246.

4. Bergman, O., R. Beyth-Marom, and R.Nachmias. (2003). The User Subjective Approach to Personal Information Management Systems. *Journal of the American Society for Information Science*, 54(9), 872-878.

5. Bloom, B.S. (1956). *Taxonomy of Educational Objectives: The Classification of Educational Goals*, McKay, New York.

6. Boardman, R. and Sasse, M.A. (2004). "Stuff goes into the computer and doesn't come out": a cross-tool study of personal information management. *Proceedings of the SIGCHI conference on Human factors in computing systems*. Vienna, Austria: ACM.

7. Brooks, J.G. and Brooks, M.G. (1993). *In Search of Understanding: The Case for Constructivist Classrooms*, Association for Supervision and Curriculum Development, Alexandria, VA.

8. Chang, S.-J. and Ko, M.-H. (2008). Behaviors of PIM in Context of Thesis and Dissertation Research. *CHI 2008 workshop*. Florence Italy.

9. Chernov, S., Demartini, G., Herder, E., Kopycki, M., and Nejdl., W. (2008). Evaluating Personal Information Management Using an Activity Logs Enriched Desktop Dataset *CHI 2008 Workshop*. Florence, Italy

10. Chirita, P.A., Gaugaz, J., S.Costache, and W.Nejdl. (2006). Desktop context detection using implicit feedback. *SIGIR 2006 Workshop on Personal Information Management*. Seattle WA, USA.

11. Fisher, D., Brush, A.J., Gleave, E., and Smith, M.A. (2006). Revisiting Whittaker & Sidner's "email overload" ten years later. *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*. Banff, Alberta, Canada: ACM.

12. Kelly, D. (2006). Evaluating personal information management behaviors and tools. *Communications of the ACM*, 49(1), 84-86.

13. Lansdale, M.W. (1988). The psychology of personal information management. *Applied Ergonomics*, 19(1), 55-66.

14. Malone, T.W. (1982). How do people organize their desks? (Extended Abstract): Implications for the design of office information systems. *Proceedings of the SIGOA conference on Office information systems*. Philadelphia, Pennsylvannia, United States: ACM.

15. Manco, G., Masciari, E., and Tagarelli, A. (2008). Mining categories for emails via clustering and pattern discovery. *Journal of Intelligent Information Systems*, 30(2), 153-181.

16. Mioduser, D., Nachmias, R., and Forkosh-Baruch, A. (2009). New Literacies for the Knowledge Society, in *International Handbook of Information Technology in Primary and Secondary Education*, J.M. Voogt and G.A. Knezek, Editors. Springer. p. 23-41.

17. Nachmias, R. and Ram, J. (2009). Insights from a Decade of Campus-wide Implementation of Blended Learning in Tel Aviv University. *The International Review of Research in Open and Distance Learning*, 10(2).

18. Teevan, J., Dumais, S.T., and Horvitz, E. (2005). Personalizing search via automated analysis of interests and activities. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. Salvador, Brazil: ACM.

19. Whittaker, S. and Hirschberg, J. (2001). The character, value, and management of personal paper archives. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 8(2), 150-170.

20. Whittaker, S. and Sidner, C. (1996). Email overload: exploring personal information management of email. *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*. Vancouver, British Columbia, Canada: ACM.

Improving Student Question Classification

Cecily Heiner and Joseph L. Zachary
{cecily,zachary}@cs.utah.edu
School of Computing, University of Utah

Abstract. Students in introductory programming classes often articulate their questions and information needs incompletely. Consequently, the automatic classification of student questions to provide automated tutorial responses is a challenging problem. This paper analyzes 411 questions from an introductory Java programming course by reducing the natural language of the questions to a vector space, and then utilizing cosine similarity to identify similar previous questions. We report classification accuracies between 23% and 55%, obtaining substantial improvements by exploiting domain knowledge (compiler error messages) and educational context (assignment name). Our mean reciprocal rank scores are comparable to and arguably better than most scores reported in a major information retrieval competition, even though our dataset consists of questions asked by students that are difficult to classify. Our results are especially timely and relevant for online courses where students are completing the same set of assignments asynchronously and access to staff is limited.

1 Introduction

Students often ask their questions and express their information needs incompletely. Consequently, the automatic classification of student questions, with the goal of ultimately providing automated tutorial responses, is a challenging problem. For example, the following are information requests from novice programming students:

- "How do i [*sic*] return the file extension only?"
- "I need help extracting a file extension from a filename."

Although phrased differently, both sentences indicate the same need, namely help with the file extension extraction problem; therefore, they should be classified the same way.

This paper classifies student questions by matching them to previous questions with similar meanings but different phrasings. We deployed a software system in an introductory computer science course for approximately one semester to collect ecologically valid data. The system mediated and logged help requests between students and teaching assistants(TAs), capturing both the students' natural language and the associated Java files. The goal of this phase of the research is to quantitatively compare various approaches of classifying the questions that novice programming students ask. The ultimate goal is to be able to provide automated answers to free form student questions by recycling answers to similar previous questions.

2 Prior Work

The AutoTutor project has researched a number of different analytical approaches for processing student language in response to tutorial prompts. They demonstrated that Latent Semantic Analysis(LSA)[8] and cosine similarity with natural language were viable approaches to selecting text for intelligent tutoring dialog with human raters as the gold standard[13, 14]. The PedaBot project followed a similar line of research with a few fundamental differences. First, the PedaBot project matched student discussions to similar previous student discussion[7]. Because students are notoriously bad at articulating their discussion points, matching student input to student input is a more difficult problem than matching student input to expert-provided input. Second, although the PedaBot approach did not require expert-provided answers, it did require a list of expert-provided technical terms. The PedaBot project avoided generating these manually by automatically extracting them from a textbook or other authoritative, expert provided resource[7]. Like the AutoTutor group, the PedaBot group examined various techniques for calculating similarity of the discussions in the system, with the focus on LSA and cosine similarity[7].

Together, these groups have demonstrated convincingly that LSA and cosine similarity are a promising direction for processing tutorial dialogue, but the general approach still has a number of serious weaknesses. First, the research results are not as compelling as they could be. The AutoTutor group reports correlations with $r < 0.5$[13], and the PedaBot group reports finding discussions of “moderate relevance” or discussions that rank three on a four point Likert scale[7]. Second, the approaches outlined require significant expert-authored resources, either in the form of a list of ideal answers in the case of AutoTutor or in the form of a list of technical terms for PedaBot, and matching these technical terms is critical to both approaches. However, students (especially novice programming students), often do not use technical vocabulary in articulating questions. Third, the approaches seem to rely on students being unrealistically verbose in their interactions with the system. In the AutoTutor dataset, the average length of student responses was 18 words[13], and in our dataset, after stop words are removed, the median length of a student question is six terms. Literature in the information retrieval community has shown that longer queries are often more effective and robust[2], and LSA is most effective with between 300 and 500 terms in the final matrix[3, 13].

By contrast, work in the information retrieval community has generally focused on the query or perhaps a question as the articulation of a user’s information needs. A typical web query is between two and three words in length (e.g. [2]) which is quite a bit shorter than a discussion. Although a typical factoid question is longer than two or three words, it is also quite short compared to a discussion. Providing automated answers to factoid questions extracted from community question answering services has been extensively studied as part of the Question Answering Track at Text Retrieval Conference (TREC) (e.g. [12]). Later versions of the TREC competition utilized more difficult datasets and more difficult tasks. Consequently, the scores in later years of the competition were often lower (e.g. [4]), and comparing TREC results across years is like comparing apples and oranges. The relatively low TREC competition scores suggest that answering questions is a difficult task, even without the extra complications from student generated data.

One of the best systems submitted to TREC-9, LCC-SMU, specifically mentions exploiting a technique called “answer caching” to provide answers to some questions that utilize different wordings to express the same information need[9]. Answer caching is a technique that matches an incoming question to a similar previous question (or group of questions with the same answer) in order to recycle an answer. The original paper on answer caching reports a 1-3% improvement on a dataset with well-formed, grammatical, well-spelled questions. This paper reports a similar result on a more difficult dataset.

Classifying the questions that students ask is a more challenging task than automatically answering factoid questions for several reasons. First, most questions that students ask tend to be about assignments and exams(e.g. [6]) instead of factoids. Factoid questions can usually be answered with a word or a phrase, while questions about assignments and exams generally require answers with one or more sentences. Thus, the space of correct answers for questions that students ask is much larger than the space of correct answers to a factoid question, and the larger space makes question classification more difficult. Second, questions asked by students in a class exist in an extensive educational context, so the question "How do I draw a pyramid?" has a very different meaning in an introductory programming class than it would in an introductory art class. Third, questions asked in class are often of a more subjective nature, such as coding style, but factoid questions are often of a more objective nature. Fourth, the questions students ask tend to be ungrammatical and contain typos and spelling errors.

3 Data

Questions asked in Introduction to Computer Science 1 (CS1410) at the University of Utah form the dataset for this paper. Most students in Computer Science 1 are age 18-22. Computer Science 1 is the first required computer science course for computer science majors, with a strong emphasis on the Java programming language. The course has long hours for novice programmers and typically high dropout, fail, and withdrawal rates. The majority of students who take Computer Science 1 hope to major in computer science or a related field, but they must pass that class along with three others with sufficiently high grades to attain official status as a computer science major. Although approximately 233 students were active in the course during the study period, only 63 of them asked questions while using the study's logging software during the study period.

The goal of this research is to classify the questions that students ask by automatically identifying similar previous questions. To facilitate analysis of student questions, a proprietary software system logged the questions that the students asked and the accompanying source code during the study period. The long term goal is to complete the analysis for a question in real time and exploit it for an instant tutorial intervention. In the interim, when a student asks a question, the system logs the student's question and source code and passes it to a teaching assistant (TA) who can answer the question in person or remotely. The TA then tags the question to indicate an answer category.

We tagged all of the data by associating all questions that could be answered with the same response to the same, unique tag. Then an undergraduate TA tagged approximately 15% of the data, assigning tags from a set devised for that assignment. The TA did not recode the other 85% of the data, but because the inter-rater reliability for the questions we sampled was better than 95%, we included all of the data in the final dataset. This left a dataset of 411 questions from 13 different assignments covering a total of 136 answer categories or information needs. Of the 411 questions, 275 of the questions (136 subtracted from 411) were repetitive in nature, and had a similar previous question. That means that 66% of the questions were repetitive. Excluding stop words, length of student questions ranged from 0 to 93 terms, with a median of six words and a mode of four words. Approximately 2% of the questions had no terms after stop words were excluded. More than 90% of the questions had 16 terms or fewer.

4 Similarity Scoring

We follow typical practice for processing the natural language in the questions that the students asked. The sentences were tokenized based on spaces and other special characters. Stop words (e.g. “me”, “you”, and “the”) were excluded. Then, a Porter stemmer[10] removed the word endings leaving just the word stem (e.g. the word “extension” became “extens”). The word stems from each question then form a vector. Table 1 shows some sample student questions and the corresponding vectors of stems.

Table 1: Sample Questions, Vector Stems, and Answer Categories

	Natural Language	Vector Stems	Answer Category
Q1	How do i return the file extension only?	return file extens	File extension extraction
Q2	my variable for rectSideOne is suppose to be 1/9, the program is returning a 0 for this calculation. I have no idea why.	Variabl rectsideon suppos 1/9 program return calcul idea	Integer division
Q3	I need help extracting a file extension from a filename.	need help extract file extens filename	File extension extraction
Q4	program is not computing volume correctly	Program comput volum correctly	Integer division
Q5	Im having trouble understanding why (1/9) equals 0.0 instead of 0.111111	trouble understand 1/9 equal	Integer division

The word stems that remained for each question populated a frequency matrix f_{ij}, which gives the number of times word stem j appears in question i. This matrix has 411 rows (one per question) and one column per unique word stem. The questions were compared in the order they were originally asked by the students to all previously asked questions. Specifically, we used cosine similarity (Equation 1) to compare question i against each of questions 1 through i-1, using weights w_{ij} computed from the frequencies f_{ij} with standard term frequency, inverse document frequency (tfidf) weighting. The tfidf weights were recomputed each time the algorithm advanced to the next question, which effectively enlarged the model by one question.

The remainder of the analysis utilizes an online learning framework to identify similar previous questions. Each question is compared to all previous questions, and the previous question with the highest cosine similarity score (as shown in Equation 1) when compared to this question is considered the most similar. If the current question and the most similar question have the same answer category, the system earns a point for accuracy. For example, in Table 1, Q2 would only be compared to Q1, and the system would not earn a point for accuracy. However, Q5 would be compared to Q1, Q2, Q3, and Q4. Of these, Q2 would be the most similar, and since Q2 and Q5 share an answer category, the system would earn one point for accuracy.

$$\frac{\sum_{j=1}^{t} w_{cj} w_{pj}}{\sqrt{\sum_{j=1}^{t} (w_{cj})^2}\sqrt{\sum_{j=1}^{t} (w_{pj})^2}}$$

Equation 1: Cosine Similarity

5 Analysis and Results

5.1 Baseline Cosine Similarity

As shown in Figure 1 and Table 2, we report accuracy scores with three different denominators. In total questions, 411 is always the denominator. In repetitive questions, either 275 or 204 is the denominator depending whether or not the data is disaggregated. Of these, only the repetitive questions bar could theoretically reach 100%. In both cases, the numerator is the number of correct similar questions found(93). As a baseline, cosine similarity is applied to the natural language of the students' questions. With that baseline, the algorithm can classify approximately 35% of the repetitive questions or 23% the total questions. For those questions, an answer to a previous question could theoretically be recycled to answer that question.

Table 2: Classification Accuracy Counts and Percentages

	Aggregated		Disaggregated	
	Total Questions	Repetitive Questions	Total Questions	Repetitive Questions
Baseline	93/411 (23%)	93/275 (35%)	113/411 (27%)	113/204 (55%)
With Error Msgs and Answer Cache	104/411 (25%)	104/275 (39%)	111/411 (27%)	111/204 (54%)

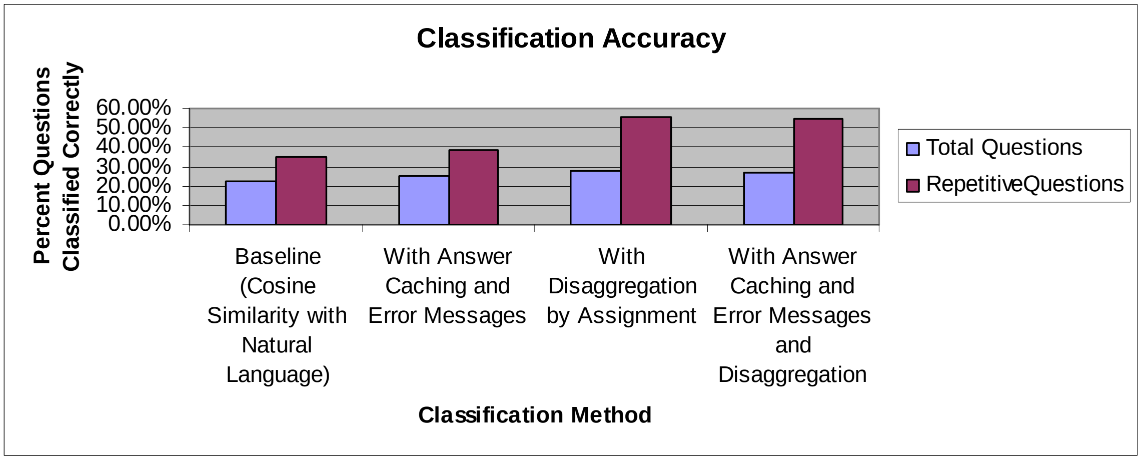


Figure 1 Classification Accuracy

5.2 *With Error Messages and Answer Caching*

The classification techniques described so far are inherently domain independent. However, the low accuracy of question classification suggests room for substantial improvement. One possible way to improve classification is to leverage some domain specific knowledge, specifically the error messages from the compiler. Since more than 40% of the questions were submitted with code that did not compile, the compiler error messages represent a source of substantial unused data.

A naïve approach to incorporating compiler output would be to simply tokenize the errors and include them just as the natural language was included. The problem is that errors such as missing import and capitalization will appear to be very similar because they contain four similar tokens (“cannot”, “find”, “symbol”, and “class”), and the algorithm will be unable to distinguish between them. To remediate this problem, some of the most common compiler errors and code snapshots are processed by Java code that generates a brief description of the underlying error based on the code snapshot, and then the underlying error is incorporated into the model. For example, the common error message “cannot find symbol- class Scanner” is processed and becomes “missingImport”, and the common error message “cannot find symbol –class string” becomes “capitalization”.

Previous work has exploited a technique called “answer caching” to provide answers to some questions that utilize different wordings to express the same information need[10]. Answer caching matches an incoming question to a similar previous question in order to recycle an answer. The answer caching technique then leverages the additional language in the similar question to build a more robust language model of that information need. Specifically, answer caching merges the data from vectors that indicate a similar information need to form a single vector. Without answer caching, the five questions in Table 1 are modeled with five vectors. With answer caching, they are represented with two vectors, one for “File extension extraction”(the sum of the vectors for Q1 and Q3) and one for “Integer division”(the sum of the vectors for Q2, Q4, and Q5).

The original paper on answer caching reports a 1-3% improvement on a dataset with well-formed, grammatical, well-spelled questions. Figure 1 demonstrates a similar improvement when incorporating both answer caching and the processed error messages, even though our data consists of student questions with typos and other complications. Interestingly, the processed error messages alone do not improve classification, and answer caching alone produces a minor improvement of less than 1%, but the combination of the techniques improves accuracy by 3% of the total questions. The numerator in for the “With Answer Caching and Error Messages” method is 104, and the denominators are the same as they were in the baseline conditions, 411 for total questions and 275 for repetitive questions.

5.3 *Disaggregating by Assignment*

For a final improvement in classification accuracy, the data was disaggregated by assignment. For example, assignment1 questions were compared only to other assignment1 questions and assignment5 questions were compared only other assignment5 questions. As shown in Figure 1, this technique improved the numerator to 113 questions classified correctly or 27% of total questions and 55% of repetitive questions

With the data disaggregated by assignment, incorporating answer caching and error messages reduced accuracy slightly(111 questions classified correctly). The lack of sufficient data to model different kinds of compiler errors is probably the cause of a drop in accuracy when answer caching and error messages are incorporated. Because compiler errors are being reduced to a single term, several of them are necessary to boost the compiler error terms to a heavy enough weight to influence the similarity algorithm. Excluding error messages and answer caching returns the classification algorithm to a domain independent state. Compiler error messages are a source of data that are only relevant in the computer science domain. By contrast, natural language and assignment numbers are a data source that is available in virtually every educational domain.

5.4 Discussion

We have shown that the baseline algorithm can be improved by incorporating an answer caching/compiler error extension (an additional 4% of repetitive questions classified.) This improves on the earlier results on answer caching[9], and is especially noteworthy given that we obtained our results on a student-generated corpus.

Most importantly, we have shown that the baseline algorithm can be improved by disaggregating by assignment (an additional 7% of repetitive questions classified.) In fact, this percentage substantially understates the actual improvement that we observed. To facilitate comparison in the bar charts, we use the same denominators throughout. However, when comparing questions only within the same assignment, the number of repetitive questions is actually smaller (204). Using that as the denominator yields a classification accuracy of 55% of repetitive questions.

Classification accuracies of 55% are neither great nor terrible. They are good enough that a desperate student who is working on an assignment at midnight might actually be able to find a useful bit of information when a human TA is not on duty. In such a situation, a bad answer may be better than no answer. However, they are low enough to raise concern that the system may not answer student questions correctly, and worse, the system might lead the student astray. At least two alternatives are possible intermediate steps to deploying this in a real classroom. First, the existing corpus could be leveraged as a starting point for designing common error detectors and appropriate interventions. Second, a human TA could supervise the classification algorithm, and override any incorrect decisions that it makes, until the number of incorrect decisions decreases.

6 Limitations and Future Work

6.1 Classification Schemes for Questions that Novice Programmers Ask

Given a set of categories, classifying questions appears to be relatively straightforward for humans. However, no widely accepted set of categories or taxonomy exists for the questions that novice programmers ask. Previous work has suggested either 42, 88, or

226 different categories for compiler errors [1, 5, 11], and compiler errors only account for half of the questions in the data set presented in this paper. Those papers are simply trying to classify compiler errors based on the compiler error message, not the underlying misconception the student has expressed. Furthermore, a single piece of code may have multiple issues. Ideally, students would request help at the point of a partial impasse, but students appear to frequently wait until they have reached a full impasse before requesting help. The resulting code often has many problems. In this study, such a question would have probably been assigned a label that encompasses a broad range of problems. Work on classification schemes that allow free-response student-input to be assigned multiple, more-fine-grained designations would be applicable for question classification as well as other problems. That research will probably also require work on partial parsing, and other approaches for handling poorly formed student input that cannot be parsed with readily available tools.

6.2 Usability issues

A number of usability issues on both the teacher and the student side must be resolved before an automatic question answering system can be deployed in a classroom setting. On the teacher side, training may be necessary to classify student questions correctly. Once automated interventions are added, the teacher will need to determine if the student still needs human help because the system classified the question incorrectly or because the automated intervention was ineffective. On the student side, studies should investigate whether or not a drop-down menu of frequently asked questions can help students articulate their questions, and whether or not students accept automated answers to their questions, especially if they know that a human TA is on duty and available.

6.3 Model of time spent

The data collected for this study could be reanalyzed to build a model of how long it takes to answer a particular question taking into account factors such as the student asking the question, the question that was asked, and the teacher answering the question. Such a model could help answer questions about which factors are most important in predicting the amount of time it will take to answer a question. Such a model may also allow the system to automatically perform triage, determining which questions are the quickest and most urgent to answer and suggesting that a teacher answer those first, thus reducing total student wait time. However, some students who are accustomed to first in first out service might complain that such an approach is unfair.

6.4 Improving the feature set and data set

A larger dataset may support stronger claims and possibly allow interesting disaggregations in different dimensions. For example, with more data, the combination of disaggregating by assignment and including error messages and answer caching may be more accurate than disaggregating by assignment alone. The feature set could be better. For example, compound words are not well analyzed, so a question containing the words “monthly” and “payment” may not have any terms in common with a question containing the word “monthlyPayment” even though they are clearly semantically similar.

Additionally, many compiler errors and features that could be extracted from student source code, such as extra semicolons before the body of a loop, are ignored.

7 Contributions and Conclusions

We show that cosine similarity is a non-trivial baseline for experiments to improve accuracy in question classification. We discuss previous results showing that answer caching can improve accuracy by 1-3% and extend previous work on answer caching by achieving similar improvement on a more difficult dataset and demonstrating that it is a valid approach for tutorial dialogue by utilizing an ecologically valid dataset. We demonstrate that additional improvements in accuracy are possible by exploiting other sources of data beyond the natural language of student questions, and we demonstrate additional modest gains in accuracy using some compiler errors. These techniques produce a 4-7% improvement in question classification accuracy, bringing total question classification accuracy to 28% of all questions or 42% of repetitive questions, or 56% of repetitive questions when disaggregated by assignment.

Our classification methods work over half the time for student-generated questions, assuming that the questions can be separated by assignment. Thus, our methods would work particularly well in a course in which the same assignments are used over and over, and our long-term goal of using a classification-based approach to automatically answer questions would be especially valuable in a course in which students have low access to course staff. These two conditions are typical of online classes, which represent a fast growing segment of courses in higher education.

Acknowledgements

This work was partially supported by the National Science Foundation, under REESE Grant No. DRL-0735264. Thank you to the students and instructors who used the data collection system. Peter Hastings provided an excellent stopword list. Thank you to reviewers who have provided useful feedback to improve this research.

References

[1] Ahmadzadeh, M., Elliman, D., Higgins, C., An analysis of patterns of debugging among novice computer science students, in Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education. 2005, ACM: Caparica, Portugal.

[2] Belkin, N. J., Kelly, D., Kim, G., Kim, J. Y., Lee, H. J., Muresan, G., Tang, M. C., Yuan, X. J., Cool, C., Query length in interactive information retrieval, in Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. 2003, ACM: Toronto, Canada.

[3] Bradford, R. B., An empirical study of required dimensionality for large-scale latent semantic indexing applications, in Proceeding of the 17th ACM conference on Information and knowledge management. 2008, ACM: Napa Valley, California, USA.

[4] Dang, H. T., Lin, J., Kelly, D. Overview of the TREC 2006 Question Answering Track. *Text REtrieval Conference*, 2006.

[5] Jadud, M. C., A First Look at Novice Compilation Behaviour Using BlueJ. Computer Science Education, 2005. 15(1): p. 25 - 40.

[6] Kim, J., Shaw, E., Chern, G., Herbert, R. Novel tools for assessing student discussions: Modeling threads and participant roles using speech act and course topic analysis. *Artificial Intelligence in Education*, 2007. IOS Press.

[7] Kim, J., Shaw, E., Ravi, S., Tavano, E., Arromratana, A., Sarda, P., Scaffolding On-Line Discussions with Past Discussions: An Analysis and Pilot Study of PedaBot, in Intelligent Tutoring Systems. 2008. p. 343-352.

[8] Landauer, T. K., Foltz, P. W., Laham, D., An Introduction to Latent Semantic Analysis. Discourse Processes, 1998. 25(2/3): p. 259-284.

[9] Pasca, M. A., Harabagiu, S. M., High performance question/answering, in Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. 2001, ACM: New Orleans, Louisiana, United States.

[10] Porter, M. F., An algorithm for suffix stripping. Program, 1980. 14: p. 130-137.

[11] Thompson, S. M., An Exploratory Study of Novice Programming Experiences and Errors. 2006, University of Victoria. p. 153.

[12] Voorhees, E. M. Overview of the TREC 2001 Question Answering Track. *Text REtrieval Conference (TREC)*, 2001.

[13] Wiemer-Hastings, P., Wiemer-Hastings, K., Graesser, A. How Latent is Latent Semantic Analysis? *Proceedings of the 16th International Joint Congress on Artificial Intelligence*, 1999. Morgan Kaufmann.

[14] Wiemer-Hastings, P., Wiemer-Hastings, K., Graesser, A. C. Approximate natural language understanding for an intelligent tutor. *12th International Florida Artificial Intelligence Research Conference*, 1999. AAAI Press.

Why, What, and How to Log? Lessons from LISTEN

Jack Mostow[1] and Joseph E. Beck[2]
mostow@cs.cmu.edu, joseph.beck@EducationalDataMining.org
[1]Project LISTEN, Robotics Institute, Carnegie Mellon University
[2]Computer Science Department, Worcester Polytechnic Institute

The ability to log tutorial interactions in comprehensive, longitudinal, fine-grained detail offers great potential for educational data mining – but what data is logged, and how, can facilitate or impede the realization of that potential. We propose guidelines gleaned over 15 years of logging, exploring, and analyzing millions of events from Project LISTEN's Reading Tutor and its predecessors.

1 Introduction

There is presumably unanimous consensus in the EDM community that it is useful to log tutor data. However, there is far from a consensus about which data to log, or how. As a step toward this goal, this paper attempts to distill lessons from over fifteen years of experience in logging and mining data from Project LISTEN's Reading Tutor [1] and its predecessor [2]. We cite relevant publications about that work, but work on logging tutor data in other projects is outside the scope of this paper and hence not cited here. Rather we describe some guidelines we have developed along the way. We do not claim they are the best possible way to log tutor data, only that we have found them sufficiently helpful in our project to recommend them in considering why, what, and how to log.

The Reading Tutor uses speech recognition to listen to children read aloud, and helps them learn to read [3-8]. It logs its interactions with students throughout the school year in fine-grained, comprehensive detail, logging – and timestamping – every launch of the tutor, session with a student, story read in whole or part, text sentence displayed, multiple choice question and response, utterance by student or tutor, word recognized, keyboard input, and mouse click. The Reading Tutor logs this data directly to a relational database. The school server then forwards the data overnight to an aggregated database in our lab.

2 Why to log

Project LISTEN illustrates multiple purposes that logging tutor data can serve.

2.1 Tutoring

Besides logging data for later analysis, a tutor can query logged data itself at runtime. The Reading Tutor queries 12 of the 32 database tables that it logs:

- Enrollment and login use tables of classes and students.
- Usage tracking queries tables of launches and sessions, e.g. to track how long the student used the Reading Tutor so far today, which it displays on the screen.
- The database keeps the student's place even if the student logs out or the tutor exits or crashes [9]. The story choice mechanism uses the table of story

encounters to bookmark the student's position in a story, and to remember whose turn it is to pick a story next.

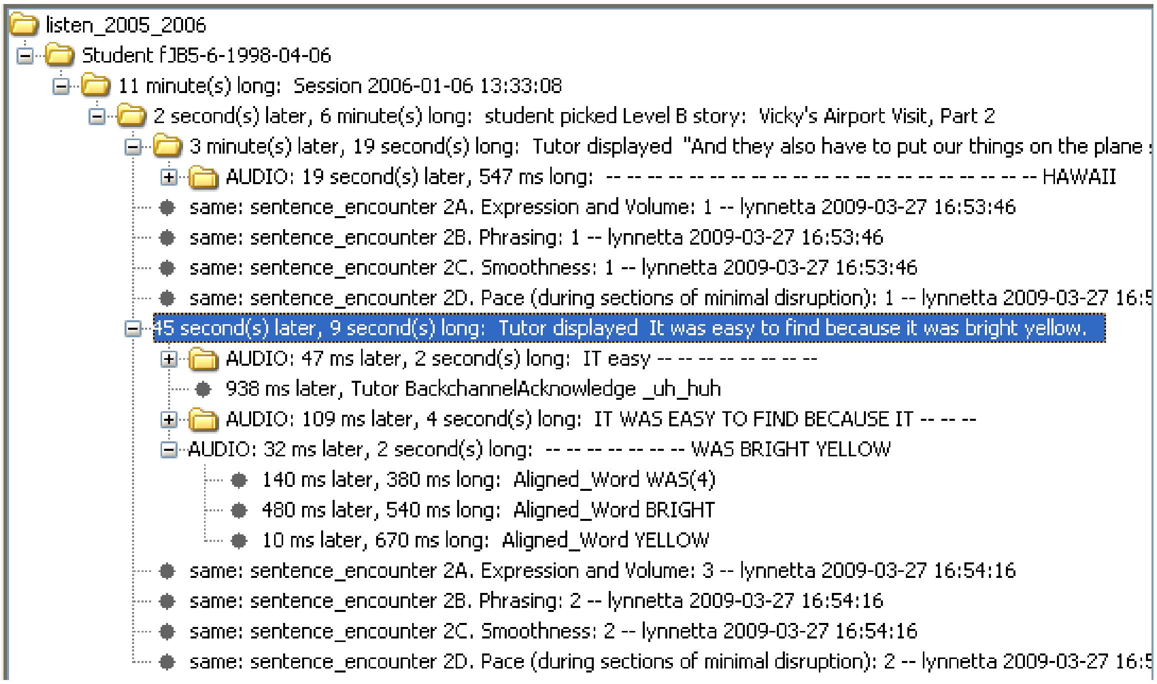
- The tables for the student model determine which level of stories to pick, when to promote or demote the student based on oral reading fluency, which letter-sound mappings to teach, and which words to practice.

2.2 *Reporting*

A separate program used the database to generate various types of password-protected, web-accessible reports on demand. A report for teachers summarized students' usage and progress [10]. A report for technical support staff computed usage and reliability at each school, and flagged possible technical problems indicated by falloff in usage.

2.3 *Browsing*

Project LISTEN's Session Browser [11, 12] enables researchers to find examples of tutorial interactions with specified characteristics, display them in human-understandable form, explore them in dynamically variable detail, and annotate them, as Figure 1 shows.



```
listen_2005_2006
  Student fJB5-6-1998-04-06
    11 minute(s) long:  Session 2006-01-06 13:33:08
      2 second(s) later, 6 minute(s) long:  student picked Level B story:  Vicky's Airport Visit, Part 2
        3 minute(s) later, 19 second(s) long:  Tutor displayed  "And they also have to put our things on the plane :
          AUDIO: 19 second(s) later, 547 ms long:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- HAWAII
          same: sentence_encounter 2A. Expression and Volume: 1 -- lynnetta 2009-03-27 16:53:46
          same: sentence_encounter 2B. Phrasing: 1 -- lynnetta 2009-03-27 16:53:46
          same: sentence_encounter 2C. Smoothness: 1 -- lynnetta 2009-03-27 16:53:46
          same: sentence_encounter 2D. Pace (during sections of minimal disruption): 1 -- lynnetta 2009-03-27 16:5
        45 second(s) later, 9 second(s) long:  Tutor displayed  It was easy to find because it was bright yellow.
          AUDIO: 47 ms later, 2 second(s) long:  IT easy -- -- -- -- -- -- -- --
            938 ms later, Tutor BackchannelAcknowledge _uh_huh
          AUDIO: 109 ms later, 4 second(s) long:  IT WAS EASY TO FIND BECAUSE IT -- -- --
          AUDIO: 32 ms later, 2 second(s) long:  -- -- -- -- -- -- -- -- WAS BRIGHT YELLOW
            140 ms later, 380 ms long:  Aligned_Word WAS(4)
            480 ms later, 540 ms long:  Aligned_Word BRIGHT
            10 ms later, 670 ms long:  Aligned_Word YELLOW
          same: sentence_encounter 2A. Expression and Volume: 3 -- lynnetta 2009-03-27 16:54:16
          same: sentence_encounter 2B. Phrasing: 2 -- lynnetta 2009-03-27 16:54:16
          same: sentence_encounter 2C. Smoothness: 2 -- lynnetta 2009-03-27 16:54:16
          same: sentence_encounter 2D. Pace (during sections of minimal disruption): 2 -- lynnetta 2009-03-27 16:5
```

Figure 1: Event tree displayed by Project LISTEN's Session Browser

The event highlighted in the tree is a 9-second-long encounter of the sentence *It was easy to find because it was bright yellow*, 45 seconds after the previous sentence encounter, which lasted 19 seconds and occurred 3 minutes into a 6-minute encounter of the Level B (grade 2) story "Vicky's Airport Visit, Part 2," which the student picked 2 seconds after starting an 11-minute-long session at 1:33pm on January 1, 2006. A human annotator later rated both sentences on a scale from 1 to 4 for four items of a fluency rubric [13]. The second sentence encounter has been expanded here to reveal three utterances, the last of which is expanded to show the text words *was bright yellow* aligned against the speech recognizer output. The Session Browser derives the tree structure from the temporal relationships among the events stored in the database.

2.4 Mining

Researchers query Reading Tutor databases from different school years for educational data to mine. These queries are shorter, faster to write, easier to understand, and less error-prone than the scripts used to analyze tutor log files [1]: "For example, analysis of the preemptive assistance experiment was hampered by the absence of a single uniform log entry for the generic event "give help on a word." Instead, an *ad hoc* script had to recover this implicit event by aggregating over the more specific events logged for each specific type of help, and for the Reading Tutor's spoken outputs. This script is sufficiently complex to doubt its correctness."

3 What to log

To put tutor data logging in perspective, it is helpful to think of the tutor and student as each sensing, thinking, and acting within the wider environment surrounding them – physically, socially, institutionally, culturally, and so forth. This environment includes the machine on which the tutor runs, the classroom and school where it's located, the student's classmates and teacher(s), and so on.

We can use this framework to characterize logged information. To illustrate with authentic concrete examples, we use it to categorize database tables logged by the 2007-2008 version of the Reading Tutor – as well as additional information that might be useful to log, but that we currently do not.

3.1 Environmental data

A "Candid Camera" analysis of video recorded by a camera mounted on a Reading Tutor monitor [14] showed that a tutor equipped with computer vision could potentially capture useful information about its physical and social context, such as whether a student is present or absent, looking at the screen or off-task, alone or with classmates, and working independently or interacting with a teacher. However, at present a typical tutor has limited if any ability to observe its environment directly.

In the absence of such observations, a tutor may input information about the student and context directly from the student, teacher, or other sources. Information logged about the environment may be a static fact (seldom or never updated), a maintained value (updated more or less frequently), or a timestamped event (appended to the ongoing historical record of the interaction). The 2007-2008 Reading Tutor's static environmental data included listed 11 schools, 37 classes, and 540 users (446 with at least one session). Its maintained data included 960 stories, updated when students added stories.

3.2 *Raw input*

A tutor's sensors are typically limited to keyboard, mouse, and microphone, enabling it to observe a subset of the student's actions. 2007-2008 raw input included 458,984 mouse clicks, 491,095 oral reading utterances, and 9,188 free-form spoken responses.

3.3 *Interpreted input*

A tutor's input observations go through layers of interpretation. For instance, the 2007-2008 Reading Tutor asked 105,223 multiple choice questions, of which 5,158 had a designated right answer. It interpreted 3,988 of these 5,158 responses as correct, 1,009 as incorrect, and 161 as null due to timing out, logging out, or crashing.

Similarly, the speech recognizer interpreted the 476,713 oral reading utterances as 1,747,259 spoken words, against which the Reading Tutor aligned 3,550,641 text words to interpret them as read, misread, or omitted.

Since the Reading Tutor's speech recognizer is imperfect, we rely on later manual transcription when we want to know what the student actually said. However, due to its cost, we transcribe only selectively. For our research on teaching reading strategies, we transcribe only students' free-form spoken responses to comprehension prompts [15].

Human interpretation of recorded speech is not restricted to transcribing it. For instance, an expert reading teacher annotated the transcribed student answers with how she would have responded to each answer, and why. Likewise, for an analysis of students' oral reading prosody, two annotators independently rated a sample of 200 recorded oral reading utterances on a 4-point fluency rubric [13].

3.4 *Student model*

A tutor cannot observe students' thinking directly, but may infer student models from observed student behavior. Fine-grained models represent estimates of individual skills. The 2007-2008 Reading Tutor tracked exposure to vocabulary by maintaining 265,579 counts of how many times 480 students saw or typed 16,932 distinct words. It tracked individual phonics skills with knowledge tracing of students' performance in word-building activities where they clicked on letter tiles to spell out a word spoken by the Reading Tutor. During 4,190 such activities, students built 26,562 words. The Reading Tutor interpreted the student's first attempt at each letter in a word as correct or incorrect. It accordingly made 78,069 updates to its estimates of students' phonic skills.

3.5 *Tutor decisions*

A tutor's observations guide layers of tutorial decision. For instance, after classifying words in the current utterance (if any) as read, misread, or omitted, the Reading Tutor decides whether to wait, backchannel, prompt the student, give praise, go on, read the sentence aloud, or give help on an individual word, and if so, what type of help. The 2007-2008 Reading Tutor logged 181,788 such decisions.

In analyzing such decisions, it is necessary not only to know the tutor's observable actions and the decisions that led to them, but the alternative choices for each decision. The 2007-2008 Reading Tutor logged this information for some types of decisions. For example, it logged not only its randomized decisions to backchannel or prompt the student, but also its decisions not to do so. Logging this additional information is crucial for subsequent analyses of the randomized controlled experiments embedded in the Reading Tutor.

Reading Tutor activities and interventions can set variables to record randomized decisions about which word to give which if any treatment, or outcomes in the form of what responses the student chose or typed. It logged 175,978 such timestamped variable assignment events. For example, it used this mechanism to log its randomized decisions about whether to explain a vocabulary word in depth, briefly, or not at all [16].

Without such logging the alternatives for each decision, it is necessary to reconstruct them. For example, the Reading Tutor's decisions about what type of decoding assistance to give on words are randomized, but are constrained by the types of help feasible and appropriate for a given word. The Reading Tutor cannot give a rhyming hint for the word *orange* because no words rhyme with it. It could sound words longer than four phonemes but refrains from doing so on grounds of inadvisability. In analyzing the relative efficacy of different types of help, a "level playing field" comparison requires knowing which types were possible on a given word. Otherwise, comparing the success of help based on the student's later performance on the word will be biased in favor of help such as rhyming hints, which tend to be available only on easier words. Proper analysis of help efficacy involved approximating this information based on lexical properties of the word and detailed knowledge of the Reading Tutor implementation [17]. In retrospect, it would have been better to log the actual set of choices for each decision.

Tutorial decisions go through layers of implementation to transform them into external behavior (acting). For instance, the 2007-2008 Reading Tutor logged 41,177 decisions about how to intervene before a story, before a sentence, or after a story. These interventions are specified in the same activity language used to represent stories as sequences of different types of steps, such as reading, listening, editing, and choosing.

3.6 Tutor actions

A tutor's external actions are typically limited to graphical display and audio output. The 2007-2008 Reading Tutor logged 1,964,620 records of .wav files played in full or in part.

We refrain from logging every individual graphical action because there are too many, nor does it seem useful to do so. However, the text displayed by the Reading Tutor is generally logged as part of the action that displays it. For instance, logging a sentence encounter includes the text of the sentence. Likewise, logging a multiple choice question includes the displayed prompt and menu of choices.

3.7 Events

Tutorial interaction can be described at different temporal *grain sizes*. The 2007-2008 Reading Tutor logged 3,184 launches, of which 71 ended in crashes with logged stack dumps, and 10,462 sessions, defined as starting when a student logged in, and ending by logging out, timing out, or crashing. It logged 193,101 executions of scripts for logging in or out, picking a story or other activity, performing an activity, editing a story, and updating information about a class, student, or story. The 21,580 story encounters logged included 357,630 sentences displayed. The 310,375 logged steps included, among others, 120,506 reading, 120,506 picking, 18,412 editing, 17,8891 timing out, 9,253 free-form speaking, 4,191 word building, 2,240 entering a password, and 861 "WordSwap" games comprising 11,728 sentences, each with one misread word to click on, and 19,581 clicks.

3.8 Observing the tutor

Reconstructing the exact appearance of the screen from logged tutor data is problematic. One reason is that the actual screen appearance at a given moment depends not only on what the tutor intends to display, but on what the operating system draws, and how promptly. One approach to this problem is the ability to replay logged tutor sessions. This feature can be very useful, but is infeasible to add after the fact to the Reading Tutor, and might be impractical anyway due to its multimodal, mixed-initiative, overlapping, stochastic, time-sensitive nature. Replay is much simpler for tutors limited to mouse and keyboard input and strict turn-taking, especially if their responses are deterministic and independent of student response time.

Another problem is that graphical design evolves across successive tutor versions. Old versions may not run as before (if at all), due to environmental changes such as updates to external content, upgrades to the operating system, and porting to faster computers.

We have found over the years that the most reliable way to document the appearance and behavior of a given version is to videotape children using it. However, this method is cumbersome and expensive to do well, and the result is inconvenient to use without the additional expense of indexing it to internally logged tutor data. Programmable screen capture software, fast PCs, and cheap disk space enable tutors to record video or at least occasional screenshots of their interactions, which we plan to do in the future.

We generally assume that tutor actions are observable by students. This assumption can be false. For instance, the display monitor may be turned off, broken, or maladjusted. Even likelier, the headset may be broken, disconnected, or plugged in to the wrong socket. Unlike humans, whose motor actions are accompanied by sensory feedback, computers do not actually see and hear the same displays and audio as their users. Screen capture is only a partial solution to this problem. A full solution will require additional devices to sense how tutor output actually looks and sounds in the environment.

4 How to log

Our experience has led us to develop the following 10 guidelines for logging interactions.

4.1 *Log directly to a database.*

Log files are easy to record, flexible in what information they can capture, (sometimes) human-understandable to read, and useful in debugging. However, we have found them unwieldy to aggregate across multiple sessions and computers, and difficult to parse and analyze in ways not anticipated when they were designed [1].

Logging tutorial interactions directly to a suitably designed and indexed database instead of to log files eliminates the need to parse them – a time sink and error source in our past. Starting with the 2003-2004 school year, the Reading Tutor has logged its interactions to a database, making their analysis easier, faster, more flexible, less bug-prone, and more powerful than analyzing conventional log files. This practice has enabled or facilitated nearly all the dozens of subsequent papers listed on Project LISTEN's Publications page.

Moreover, logging straight to a database supports immediate efficient access. This capability is essential for real-time use of logged information – in particular, by the tutor itself. The Reading Tutor's student model relies on information logged to its database. In contrast, although the Pittsburgh Science of Learning Center's DataShop (located at pslcdatashop.web.cmu.edu) uses a database, it generates the database only after the fact by parsing tutor logs in the form of XML files.

Each year's version of the Reading Tutor adds, drops, or modifies tables or fields in the database schema, and therefore has its own archival database. This practice facilitates running a query on one school year's data at a time. Also, each Project LISTEN member has his or her own database to modify freely without fear of altering archival data.

4.2 *Include computer, student ID, and start time as standard fields.*

A key insight is that student, computer, and time typically suffice to identify a unique tutorial interaction of a given type. Together they distinguish the interaction from those of another type, computer, or student. (We include computer ID in case the student ID is not unique, and also because some events, such as launching the Reading Tutor, do not involve a student ID.) There are two reasons this idea is powerful. First, these fields serve as a primary key for every table in the database, simplifying both access and the learning curve for working with the data. Second, nearly every tutor makes use of the concepts of students, computers, and time, so this recommendation is broadly applicable.

4.3 *Index event tables by computer, student ID, and start time.*

Database indices enable very fast retrieval even from tables of millions of events.

4.4 *Log end time as well as start time.*

This additional information makes it possible to compute the duration of a non-instantaneous event, measure the hiatus between the end of one event and the start of another, and determine if one event starts before, during, or after another event – capabilities essential for the Session Browser [11, 12] described in Section 2.3 above, in particular to infer hierarchies of events based on temporal relations among them. Logging the start and end time of an event in the same record requires the tutor to remember until the end of the event what information to log about it, such as when it began. Logging start and end times to different records merely replaces this requirement with the messier task of matching them up after the fact. Some tutors log start times but not end times. For instance, logging the end time of an event may be problematic for a web-based tutor that doesn't know when the student leaves a page or window.

4.5 Include a field for the parent event start time.

A field for an event's parent makes joins easier to write and faster to execute. For example, the sentence_encounter_start_time field of the record for a read word makes it easier and faster to find the sentence encounter containing the word than by querying for the sentence encounter that started before the word and ended after it.

4.6 Log each school year's data to a different database.

Each year's version of the Reading Tutor adds, drops, or modifies tables or fields in the database schema, and therefore has its own archival database. This practice also facilitates running a query on one school year's data at a time. Also, each team member has his or her own database to modify freely without fear of altering archival data. Making it easy for researchers to create new tables and views that are readily accessible to each other is key. This step enables "best practices" to propagate quickly, whereas if separate copies of the data are kept, "version skew" can become a problem.

4.7 Design databases to support aggregation across sites.

To merge separate databases from multiple schools into a single database, the MySQL server at each site sends each day's transactions to our lab server, which simply re-executes them on the aggregated database. To make this solution possible, each table must be able to combine records from different sites. Therefore the Reading Tutor uses student IDs unlikely to recur across different classes or schools, so as to distinguish data from different students. Similarly, it uses computer, ID, and start time to identify events, instead of numbering them from 1 separately at each site.

4.8 Name standard fields consistently within and across databases.

Naming fields consistently in successive versions of a tutor makes them easier for the Session Browser to extract. Thus most of the tables in a Reading Tutor database have fields named machine_name, user_id, start_time, and (for non-instantaneous events) end_time. Timestamps in MySQL (at least the version we used) are limited to 1-second resolution, so tables that require higher resolution encode the milliseconds portion of start and end times in fields named sms and ems, respectively. Adding new tables and fields is fine, but keeping the names of the old ones facilitates reuse of code to analyze tutor data.

4.9 Use a separate table for each type of tutorial event.

Using a separate table for each event type (e.g. story, sentence, utterance, click) lets us include fields for features specific to that event type, such as the title of a story, the text of a sentence, the audio recording of an utterance, or the word the student clicked on.

4.10 Logging the non-occurrence of an event is tricky.

"Subjunctive logging," or recording what could have happened but did not, is still an active topic of discussion in the EDM community. But tutor designers and data miners should be aware that recording some non-events can greatly simplify later analyses. For example, the Reading Tutor logs each word it hears the student read. The time at which a skipped word *wasn't* read is undefined, so the Reading Tutor logs its start and end time as null. However, logging its (non-null) sentence_encounter_start_time, and the position in the text sentence where it should have been read, lets analysis queries and the Session Browser retrieve words as ordered in the sentence, whether or not they were read.

5 Conclusion

The goal of this paper is to provide useful advice for logging tutor data. We discussed why to log tutor data, giving four examples from Project LISTEN – tutoring, reporting, browsing, and mining. We discussed what to log, with examples from the Reading Tutor. Finally, we proposed ten concrete guidelines for how to log tutor data.

Our guidelines exploit three simple but powerful ideas. First, logging tutorial interaction directly to a suitably designed and indexed database instead of to log files eliminates the need to parse them, and supports immediate efficient access. Second, student, computer, and time interval suffice to identify a tutorial event of a given type. Third, temporal relations among time intervals define a useful hierarchical structure of tutorial events.

Evidence for our guidelines includes the dozens of publications they have enabled. They helped us implement a flexible, efficient tool to browse tutor data in understandable form yet with minimal dependency on tutor-specific details. A short vignette may best illustrate their power. In a research talk, Dr. Vincent Aleven reported a partial correlation of students' learning gains in his tutor with the rate at which they asked for hints, controlling for pretest scores. The second author of this paper used the Reading Tutor database to replicate Dr. Aleven's result – before he finished his talk.

Acknowledgements

This work was supported in part by the National Science Foundation under ITR/IERI Grant No. REC-0326153 and in part by the Institute of Education Sciences, U.S. Department of Education, through Grants R305B070458, R305A080157, and R305A080628 to Carnegie Mellon University. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute, the U.S. Department of Education, or the National Science Foundation. We thank the educators, students, and members of Project LISTEN who have helped generate, log, and analyze our data.

References (Project LISTEN publications at www.cs.cmu.edu/~listen)

[1] Mostow, J. and G. Aist. Evaluating tutors that listen: An overview of Project LISTEN. In K. Forbus and P. Feltovich, Editors, *Smart Machines in Education*, 169-234. MIT/AAAI Press: Menlo Park, CA, 2001.

[2] Mostow, J., S.F. Roth, A.G. Hauptmann, and M. Kane. A prototype reading coach that listens [AAAI-94 Outstanding Paper]. *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 785-792. 1994. Seattle, WA: American Association for Artificial Intelligence.

[3] Mostow, J. and J. Beck. When the Rubber Meets the Road: Lessons from the In-School Adventures of an Automated Reading Tutor that Listens. In B. Schneider and S.-K. McDonald, Editors, *Scale-Up in Education*, 183-200. Rowman & Littlefield Publishers: Lanham, MD, 2007.

[4] Mostow, J. Experience from a Reading Tutor that listens: Evaluation purposes, excuses, and methods. In C.K. Kinzer and L. Verhoeven, Editors, *Interactive literacy education: facilitating literacy environments through technology*, 117-148. Lawrence Erlbaum Associates, Taylor & Francis Group: New York, 2008.

[5] Mostow, J., G. Aist, C. Huang, B. Junker, R. Kennedy, H. Lan, D. Latimer, R. O'Connor, R. Tassone, B. Tobin, and A. Wierman. 4-Month evaluation of a learner-controlled Reading Tutor that listens. In V.M. Holland and F.P. Fisher, Editors, *The Path of Speech Technologies in Computer Assisted Language Learning: From Research Toward Practice*, 201-219. Routledge: New York, 2008.

[6] Mostow, J., G. Aist, P. Burkhead, A. Corbett, A. Cuneo, S. Eitelman, C. Huang, B. Junker, M.B. Sklar, and B. Tobin. Evaluation of an automated Reading Tutor that listens: Comparison to human tutoring and classroom instruction. *Journal of Educational Computing Research*, 2003. *29*(1): p. 61-117.

[7] Mostow, J., G. Aist, J. Bey, P. Burkhead, A. Cuneo, B. Junker, S. Rossbach, B. Tobin, J. Valeri, and S. Wilson. Independent practice versus computer-guided oral reading: Equal-time comparison of sustained silent reading to an automated reading tutor that listens. *Ninth Annual Meeting of the Society for the Scientific Study of Reading* 2002. Chicago, Illinois.

[8] Beck, J.E., K.-m. Chang, J. Mostow, and A. Corbett. Does help help? Introducing the Bayesian Evaluation and Assessment methodology. *9th International Conference on Intelligent Tutoring Systems*, 383-394. ITS2008 Best Paper Award. 2008. Montreal.

[9] Aist, G. and J. Mostow. Faster, better task choice in a reading tutor that listens. In V.M. Holland and F.P. Fisher, Editors, *The Path of Speech Technologies in Computer Assisted Language Learning: From Research Toward Practice*, 220-240. Routledge: New York, 2008.

[10] Alpern, M., K. Minardo, M. O'Toole, A. Quinn, and S. Ritzie. Project LISTEN: Design Recommendations and Teacher Tool Prototype. In *Human Computer Interaction Institute*. 2001, Carnegie Mellon University: Pittsburgh, p. 85.

[11] Mostow, J., J. Beck, A. Cuneo, E. Gouvea, and C. Heiner. A generic tool to browse tutor-student interactions: Time will tell! *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, 884-886. 2005. Amsterdam.

[12] Mostow, J., J. Beck, and others. Lessons from Project LISTEN's Session Browser. In C.R. Morales, et al., Editors, *Handbook of Educational Data Mining*. Taylor & Francis Group: London, under review.

[13] Mostow, J. and M. Duong. Automated Assessment of Oral Reading Prosody. *Proceedings of the 14th International Conference on Artificial Intelligence in Education (AIED2009)* 2009. Brighton, UK.

[14] Kominek, J., G. Aist, and J. Mostow. When listening is not enough: potential uses of vision for a reading tutor that listens. *Working Notes of the AAAI Spring Symposium on Intelligent Environments*, 161-167. 1998. Stanford, CA.

[15] Zhang, X., J. Mostow, N.K. Duke, C. Trotochaud, J. Valeri, and A. Corbett. Mining Free-form Spoken Responses to Tutor Prompts. *Proceedings of the First International Conference on Educational Data Mining*, 234-241. 2008. Montreal.

[16] Heiner, C., J. Beck, and J. Mostow. Automated Vocabulary Instruction in a Reading Tutor. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 741-743. 2006. Jhongli, Taiwan.

[17] Heiner, C., J.E. Beck, and J. Mostow. Improving the help selection policy in a Reading Tutor that listens. *Proceedings of the InSTIL/ICALL Symposium on NLP and Speech Technologies in Advanced Language Learning Systems*, 195-198. 2004. Venice, Italy.

Process Mining Online Assessment Data

Mykola Pechenizkiy, Nikola Trčka, Ekaterina Vasilyeva, Wil van der Aalst, Paul De Bra
{m.pechenizkiy, e.vasilyeva, n.trcka, w.m.p.v.d.aalst}@tue.nl, debra@win.tue.nl
Department of Computer Science, Eindhoven University of Technology, the Netherlands

Abstract. Traditional data mining techniques have been extensively applied to find interesting patterns, build descriptive and predictive models from large volumes of data accumulated through the use of different information systems. The results of data mining can be used for getting a better understanding of the underlying educational processes, for generating recommendations and advice to students, for improving management of learning objects, etc. However, most of the traditional data mining techniques focus on data dependencies or simple patterns and do not provide a visual representation of the complete educational (assessment) process ready to be analyzed. To allow for these types of analysis (in which the process plays the central role), a new line of data-mining research, called *process mining*, has been initiated. Process mining focuses on the development of a set of intelligent tools and techniques aimed at extracting process-related knowledge from event logs recorded by an information system. In this paper we demonstrate the applicability of process mining, and the ProM framework in particular, to educational data mining context. We analyze assessment data from recently organized online multiple choice tests and demonstrate the use of process discovery, conformance checking and performance analysis techniques.

1 Introduction

Online assessment becomes an important component of modern education. It is used not only in e-learning, but also within blended learning, as part of the learning process. Online assessment is utilized both for self-evaluation and for “real” exams as it tends to complement or in some cases even replace traditional methods for evaluating the performance of students.

Intelligent analysis of assessment data assists in achieving a better understanding of student performance, the quality of the test and individual questions, etc. Besides, there are still a number of open issues related to authoring and organization of different assessment procedures. In Multiple-Choice Questions (MCQ) testing it might be important to consider how students are supposed to navigate from one question to another, i.e. should the students be able to go back and forward and also change their answers (if they like) before they commit the whole test, or should the order be fixed so that students have to answer the questions one after another? Is it not necessarily a trivial question since either of two options may allow or disallow the use of certain pedagogical strategies. Especially in the context of personalized adaptive assessment it is not immediately clear whether an implied strict order of navigation results in certain advantages or inconveniences for the students. In general, the navigation of students in e-Learning systems has been actively studied in recent years. Here, researchers try to discover individual navigational styles of the students in order to reduce cognitive load of the students, to improve usability and learning efficiency of e-Learning systems and support personalization of navigation [2]. Some recent empirical studies demonstrated the feasibility and benefits of feedback personalization during online assessment, i.e. the type of immediately presented feedback and the way of its presentation may significantly influence the general performance of the students [9][10]. However, some students may prefer to have less personalization and more flexibility of navigation if there is such a trade-off. Overall, there seem to be no “best” approach applicable for every situation and educators need to decide whether current practices are effective.

Traditional data mining techniques including classification, association analysis and clustering have been successfully applied to different types of educational data [4], also including assessment data, e.g. from intelligent tutoring systems or learning management systems (LMS) [3]. Data mining can help to identify group of (cor)related questions, subgroups (e.g. subsets of students performing similarly of a subset of questions), emerging patterns (e.g. discovering a set of patterns describing how the performance in a test of one group of students, i.e. following a particular study program, differs from the performance of another group), estimate the predictive or discriminative power of questions in the test, etc. However, most of the traditional data mining techniques do not focus on the process perspective and therefore do not tell much about the assessment process as a whole. *Process mining on the contrary focuses on the development of a set of intelligent tools and techniques aimed at extracting process-related knowledge from event logs recorded by an information system.*

In this paper we briefly introduce process mining [7] and our ProM tool [8] for the EDM community and demonstrate the use of a few ProM plug-ins for the analysis of assessment data coming from two recent studies. In one of the studies the students had to answer to the tests’ questions in a strict order and had a possibility to request immediate feedback (knowledge of correct response and elaborated feedback) after each question. During the second tests student had a possibility to answer the questions in a flexible order, to revisit and earlier answers and revise them as well.

The remainder of the paper is organized as follows. In Section 2 we explain the basic process mining concepts and present the ProM framework. In Section 3 we consider the use of ProM plug-ins on real assessment data, establishing some useful results. Finaly, Section 4 is for discussions.

2 Process Mining Framework

Process mining has emerged from the field of Business Process Management (BPM). It focuses on extracting process-related knowledge from event logs[1] recorded by an information system. It aims particularly at discovering or analyzing the complete (business, or in our case educational) process and is supported by powerful tools that allow getting a clear visual representation of the whole process. The three major types of process mining applications are (Figure 1):

1) *conformance checking* - reflecting on the observed reality, i.e. checking whether the

[1] Typical examples of event logs may include resource usage and activity logs in an e-learning environment, an intelligent tutoring system, an educational adaptive hypermedia system.

modeled behavior matches the observed behavior;

2) *process model discovery* - constructing complete and compact process models able to reproduce the observed behavior, and
3) *process model extension* - projection of information extracted from the logs onto the model, to make the tacit knowledge explicit and facilitate better understanding of the process model.

Process mining is supported by the powerful open-source framework ProM. This framework includes a vast number of different techniques for process discovery, conformance analysis and model extension, as well as many other tools like convertors, visualizers, etc. The ProM tool is frequently used in process mining projects in industry. Moreover, some of the ideas and algorithms have been incorporated in commercial BPM tools like BPM|one (Pallas Athena), Futura Reflect (Futura Process Intelligence), ARIS PPM (IDS Scheer), etc.

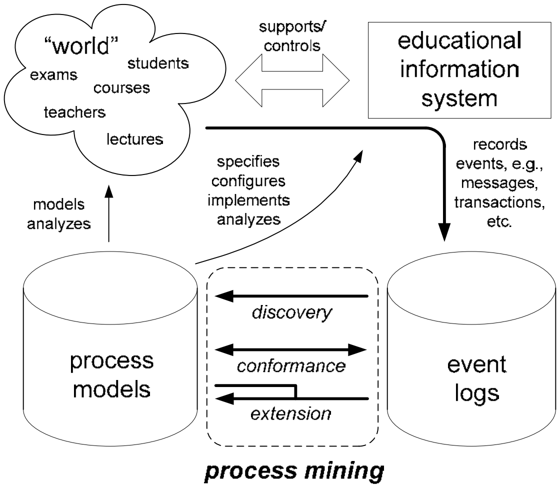


Figure 1. **The process mining spectrum supported by ProM**

3 Case Studies

We studied different issues related to authoring and personalization of online assessment procedures within the series of the MCQ tests organized during the mid-term exams at Eindhoven University of Technology using Moodle[2] (Quize module tools) and Sakai[3] (Mneme testing component) open source LMSs.

To demonstrate the applicability of process mining we use data collected during two exams: one for the Data Modeling and Databases (DB) course and one for the Human-Computer Interaction (HCI) course. In the first (DB) test students (30 in total) answered to the MCQs (15 in total) in a strict order, in which questions appeared one by one. Students after answering each question were able proceed directly to the next question

[2] http://www.moodle.org
[3] http://www.sakai.org

(clicking “Go to the next question”), or first get knowledge of correct response (clicking the “Check the answer”) and after that either go the next question (“Go to the next question”) or, before that, request a detailed explanation about their response (“Get Explanations”). In the second (HCI) test students (65 in total) had the possibility to answer the MCQs (10 in total) in a flexible order, to revisit (and revise if necessary) the earlier questions and answers. Flexible navigation was facilitated by a menu page for quick jumps from one question to any other question, as well as by “next” and “previous” buttons.

In the MCQ tests we asked students to also include the confidence level of each answer. Our studies demonstrated that knowledge of the response certitude (specifying the student’s certainty or confidence of the correctness of the answer) together with response correctness helps in understanding the learning behavior and allows for determining what kind of feedback is more preferable and more effective for the students thus facilitating personalization in assessment [3].

For every student and for each question in the test we collected all the possible information, including correctness, certitude, grade (determined by correctness and certitude), time spent for answering the question, and for the DB test whether an answer was checked for correctness or not, whether detailed explanation was requested on not, and how much time was spent reading it, and for the HCI test whether a question was skipped, revisited, whether answer was revised or the certitude changed.[4]

In the remainder of this section we demonstrate how various ProM plug-ins supporting dotted chart analysis, process discovery (Heuristic Miner and Fuzzy Miner), conformance checking, and performance analysis [1][6] allow to get a significant better understanding of the assessment processes.

3.1 Dotted Chart Analysis

The dotted chart is a chart similar to a Gantt chart. It shows the spread of events over time by plotting a dot for each event in the log thus allowing to gain some insight in the complete set of data. The chart has three (orthogonal) dimensions: one showing the time of the event, and the other two showing (possibly different) components (such as instance ID, originator or task ID) of the event. Time is measured along the horizontal axis. The first component considered is shown along the vertical axis, in boxes. The second component of the event is given by the color of the dot.

Figure 2 illustrates the output of the dot chart analysis of the flexible-order online assessment. All the instances (one per student) are sorted by the duration of the online assessment (reading and answering the question and navigation to the list of questions). In the figure on the left, points in the ochre and green/red color denote the start and the

[4] Further details regarding the organization of the test (including an illustrative example of the questions and the EF) and the data collection, preprocessing and transformation from LMS databases to ProM MXML format are beyond the scope of this paper, but interested readers can find this information in an online appendix at http://www.win.tue.nl/~mpechen/research/edu.html.

end (passed/failed) of the test. Triangles denote the moment when the student submits an answer or just navigates to another question. Green triangles denote correct responses with low (LCCR – light green) and high (HCCR – dark green) certainty, red triangles correspondingly – wrong responses (light red – LCWR, dark red – HCWR), white triangles – the cases when the student navigated to the next question without providing any response. The blue squares show the moments when the students navigated from the list of the questions (menu) to a question of the quiz (or just submitted the whole test).

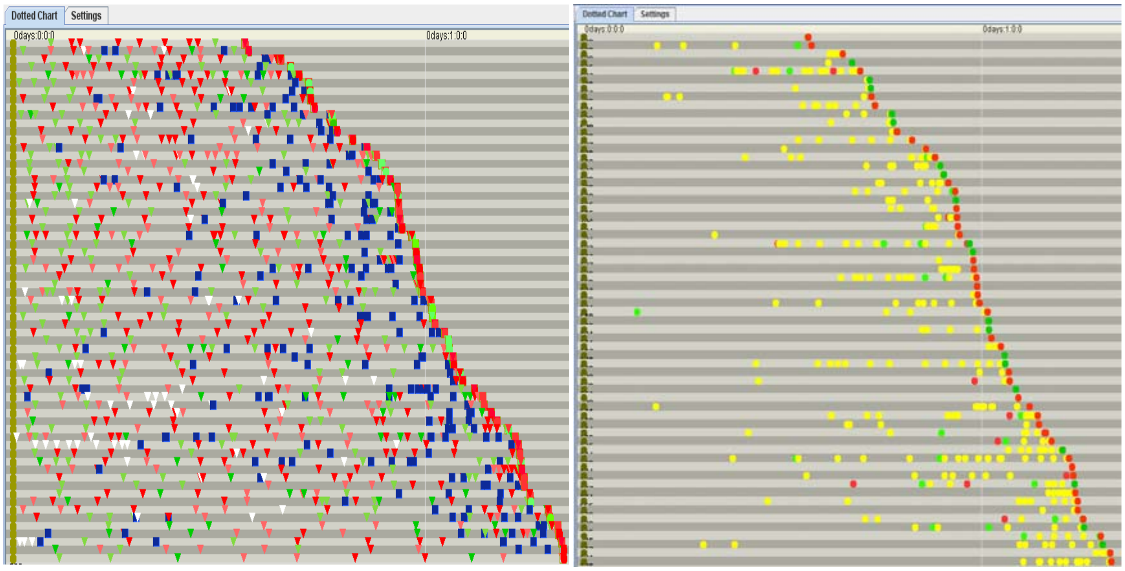


Figure 2. Two dotted charts extracted from the test with flexible order navigation; (1) the overall navigation and answering of questions (left chart), and (2) the effects of changes (right chart)

We can clearly see from the figure that most of the students answered the questions one by one, and provided more correct answers for the first questions of the test than for the last questions. They used the possibility to flexibly navigate mainly at the end of the test: students navigating to the list of the questions and then to the different questions from the list. It can be also clearly seen that only few students read and skipped some questions, not providing their answers first, and then returning to those questions back to provide an answer.

In the figure on the right, we can see the when students revisited the questions. Points in yellow correspond to the situations when correctness of the answers did not change, and points in red and green correspond accordingly to changes to wrong and correct answers. We can see that in a very few cases the correctness was changed, most changes do not affect correctness (e.g., a wrong answer was changed to another wrong answer). Moreover, changes from right to wrong or from wrong to write had similar frequencies, thus not significantly changing the end results.

3.2 Process discovery

In some cases, given a usage log we may have limited knowledge about the exact procedure of the assessment but want to discover it based on the data from the log. There exist several algorithms that can automatically construct a depiction of a process. This process representation typically comes in form of a (formal) mathematical model supporting concurrency, sequential and alternative behavior (like, e.g., the model of Petri nets, Heuristic or Fuzzy miner).

Figure 3 illustrates for the DB test a part (for the first 3 questions) of the discovered process (left) as a Heuristic net, and animation of the same part after conversion to the Fuzzy model (middle), and for the HCI test the complete Heuristic net (right), abstracted from the type of the answer, but from which it is clear which jumps between the questions were popular. From the visualization of the DB test process we can see what possibilities students had, and what the main "*flows*" were globally or at a particular time.

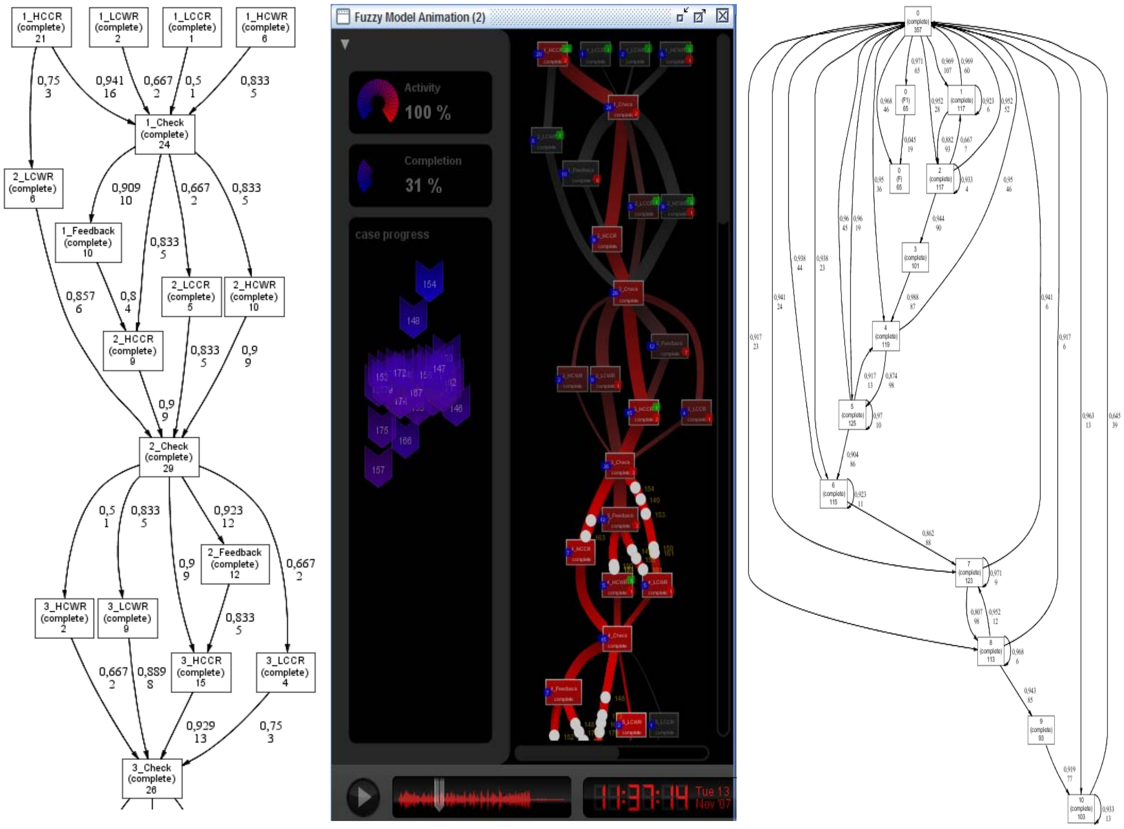


Figure 3. Heuristic nets of strict order (left) and flexible order tests (right)

3.3 Process analysis

In some cases, the goal is not to discover the real learning process but to analyze some normative or descriptive model that is given a-priori. For example, the Petri net shown in Figure 4 (formally) describes the generic pattern of answering questions in the DB test allowing for answer-checks and feedbacks. Now it is interesting to see whether this model conforms to reality (and vice versa) and augment it with additional information learned from the event logs. The advantage of having the answering pattern represented as a Petri net is that this allows for many different analysis techniques. ProM offers various plug-ins to analyze Petri nets (verification, performance analysis, conformance, etc.). Models like the one in Figure 4 can be discovered or made by hand. It is also possible to first discover a model and then refine it using the tool Yasper (incorporated into ProM). Figure 4 was constructed using Yasper and this was a one-time task for this test-type and in principle an authoring tool can be developed to facilitate an automatic translation of the multiple-choice tests with varying properties to Petri nets.

As every question can be answered correctly or wrongly, and with either high or low confidence, there are four possibilities for the first step in the net from Figure 4. The transition HCCR, for example, denotes that the answer is given with high confidence and that it was correct; the other three starting transitions are similar. After answering the question the student can check his answer or just go the next question. The latter decision is modeled by an internal transition (painted in black) that goes to the final place of the net. In case the student has decided to check the answer, he can also ask for some feedback afterwards.

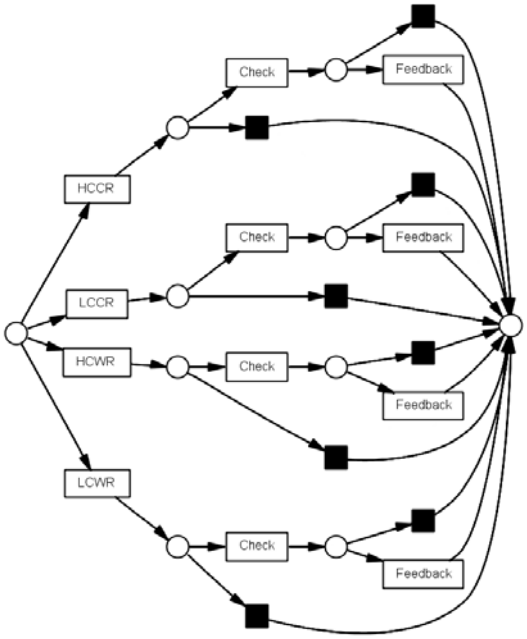
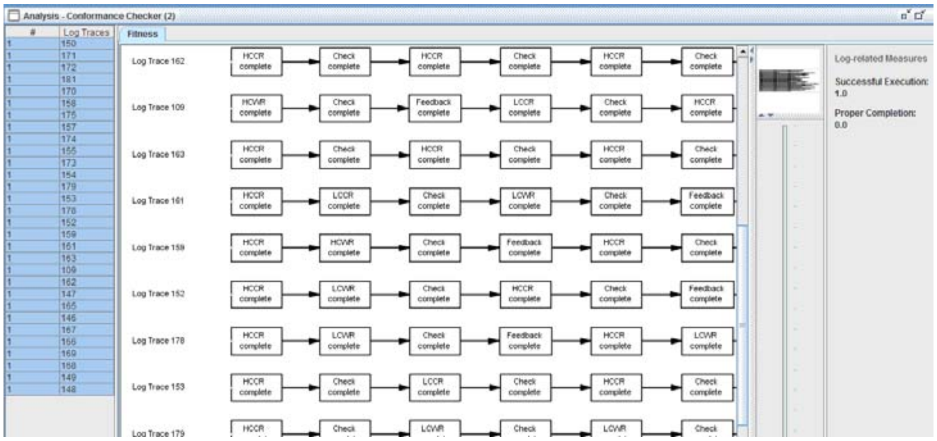


Figure 4. **A Petri net representing the question pattern**

To illustrate the many analysis possibilities of ProM, we show some results obtained using the *Conformance checker* and the *Performance Analysis with Petri net* plugin.

The purpose of conformance analysis is to find out whether the information in the log is as specified. This analysis may be used to detect deviations, to locate and explain these deviations, and to measure the severity of these deviations. We are mostly interested in the notion of *fitness* which is concerned with the investigation whether a process model is able to reproduce all execution sequences that are in the log, or, viewed from another angle, whether the log traces comply with the description in the model (the fitness is 100% if every trace in the log corresponds to a possible execution of the model). This notion is particularly useful for finding out whether (or how often) the students respected the specified order for answering questions (to discover frauds, for example).

Figure 5 shows the result of conformance checking when applied on our log and the Petri net from Figure 4. In this, so-called *log perspective* of the result, each trace from the log has all its mismatched events colored in orange. In our case, however, there are no orange events, therefore there are no mismatches between the specified answering pattern and the actual exam data.



Successful Execution:
1.0

Proper Completion:
0.0

Log Trace 162: HCCR complete → Check complete → HCCR complete → Check complete → HCCR complete → Check complete

Log Trace 109: HCWR complete → Check complete → Feedback complete → LCCR complete → Check complete → HCCR complete

Log Trace 163: HCCR complete → Check complete → HCCR complete → Check complete → HCCR complete → Check complete

Log Trace 161: HCCR complete → LCCR complete → Check complete → LCWR complete → Check complete → Feedback complete

Log Trace 159: HCCR complete → HCWR complete → Check complete → Feedback complete → HCCR complete → Check complete

Log Trace 152: HCCR complete → LCWR complete → Check complete → HCCR complete → Check complete → Feedback complete

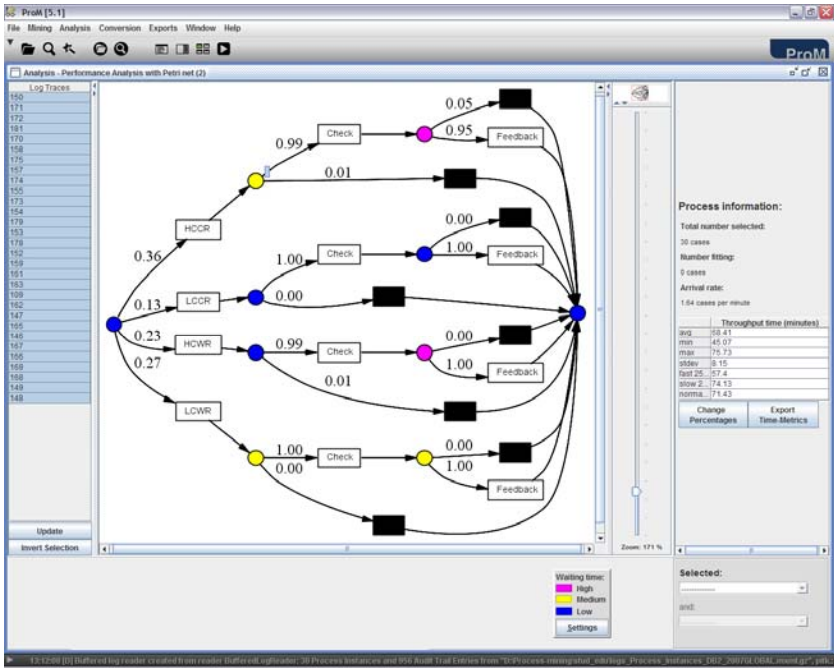
Log Trace 178: HCCR complete → LCWR complete → Check complete → Feedback complete → HCCR complete → LCWR complete

Log Trace 153: HCCR complete → Check complete → LCCR complete → Check complete → HCCR complete → Check complete

Log Trace 179: HCCR → Check → LCWR → Check → LCWR → Check

Figure 5. Result of conformance checking showing a 100% fitness

Our next analysis is of a different kind. Instead of checking for the correctnes of the exam behavior, we provide a means to assess the performance of the answering process. The *Performance analysis with Petri net* plugin can extract the Key Performance Indicators from the log, summarizing them in an intuitive way, and graphically present them on a Petri net describing the process under consideration. For our purpose we apply the plugin with the exam data log and the answering pattern from Figure 6 (only for the first question of the test).



Process information:

Total number selected:
30 cases

Number fitting:
0 cases

Arrival rate:
1.64 cases per minute

	Throughput time (minutes)
avg	58.41
min	45.07
max	75.73
stdev	8.15
fast 25...	57.4
slow 2...	74.13
norma...	71.43

Change Percentages | Export Time-Metrics

Figure 6. **Results of applying the *Performance analysis with Petri net* plug-in**

The result of the analysis is shown in Figure 6. In the right panel different throughput-type metrics are displayed; from there we, e.g., see that the average duration of the test was 64.41 minute. The central panel shows the answering pattern, colored and annotated with performance information. The numbers on the arcs represent probabilities. As shown, 35% percent of the students answered the first question right and had high confidence. We could also see that almost all students checked their answers and asked for feedback afterwards. Places are colored with respect to their soujourn time, i.e., with respect to the time the process spends in this place. From the picture we can thus see that the answering time was short (the first question was easy), and that the students who answered with high confidence spent more time on the feedback (regardless on the correctness of the answer).

4 Conclusions and Future Work

Data mining techniques have been successfully applied to different types of educational data and have helped to address many issues by using traditional classification, clustering and association analysis techniques. Although the process perspective in educational domains has received some attention, most of the traditional intelligent data analysis approaches applied in the context of educational data mining do not consider the process as a whole (i.e., the focus is no data or simple sequential structures rather than full-fledged process models).

In this paper, we illustrated some of the potential of process mining techniques applied to online assessment data where students in one of the tests were able to receive tailored immediate EF after answering each of the questions in the test one by one in a strict order, and in the other test – to receive no feedback but to answer question in a flexible order. This data was of a sequential nature, i.e. it did not include concurrency. However, other educational processes have lots of concurrency and this can be discovered by ProM. Applying process mining techniques for other types of assessment data, e.g. grades for traditional examinations is therefore an interesting possibility.

ProM 5.0 provides a plugable environment for process mining offering a wide variety of plug-ins for process discovery, conformance checking, model extension, model transformation. Our further work includes the development of EDM tailored ProM plug-ins. On the one hand, this would help bringing process mining tools closer to the domain experts (i.e. educational specialists and researchers), who not necessarily have all the technical background. On the other hand, this will help to better address some of the EDM specific challenges related to data preprocessing and mining. Besides this, the development of the authoring tools for assessment modules with specialized ProM plug-ins would allow to significantly simplify some of the processes for conformance analysis as e.g. a Petri net representing certain assessment procedure can be generated completely automatically.

Acknowledgements

This work is supported by NWO (the Dutch Science Foundation). We would like to thank the many people involved in the development of ProM.

References

[1] Günther, C.W., van der Aalst, W.M.P. Fuzzy Mining: Adaptive Process Simplification Based on Multi-perspective Metrics. In: G. Alonso et al. (eds), *Proc. of Int. Conf. on Business Process Management*, LNCS 4714, p. 328-343. Springer-Verlag, 2007.

[2] Makany, T., Engelbrecht, P.C., Meadmore, K., Dudley, R., Redhead, E.S., & Dror, I.E.: Giving the learners control of navigation: Cognitive gains and losses. In L. Gomez et al. (Eds.), *Proceedings of INTED'07*, 2007.

[3] Romero, C., Ventura, S., García, E. Data Mining in Course Management Systems: MOODLE Case Study and Tutorial. *Computers and Education*, 51. pp. 368-384, 2007.

[4] Romero, C., Ventura, S. Educational Data Mining: A Survey from 1995 to 2005. *Expert Systems with Applications*, 33(1), p. 135-146, 2007.

[5] Rozinat, A., van der Aalst, W.M.P. Conformance Checking of Processes Based on Monitoring Real Behavior. *Information Systems* 33(1), p. 64-95.

[6] Song, M., van der Aalst, W.M.P. Supporting Process Mining by Showing Events at a Glance. In K. Chari, A. Kumar (eds), *7th Annual Workshop on Information Technologies and Systems* (WITS'07), p. 139–145, 2007.

[7] van der Aalst, W.M.P., Weijters, A.J.M.M., Maruster, L. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data. Engineering* 16(9), p. 1128–1142, 2004.

[8] van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P. The ProM framework: A New Era in Process Mining Tool Support. In: Ciardo, G., Darondeau, P. (eds.) *Application and Theory of Petri Nets*, LNCS 3536, p. 444–454. Springer, Heidelberg, 2005.

[9] Vasilyeva, E., De Bra, P., Pechenizkiy, M., Puuronen, S. Tailoring feedback in online assessment: influence of learning styles on the feedback preferences and elaborated feedback effectiveness. In: *Proc. of 8th Int. Conf. on Advance Learning Technologies* (ICALT 2008), IEEE CS Press, p. 834-838, 2008.

[10] Vasilyeva, E., Pechenizkiy, M., and De Bra, P.: Adaptation of Elaborated Feedback in e-Learning, In: W. Nejdl et al. (Eds.), *Proc. of Int. Conf. on Adaptive Hypermedia* (AH'08), LNCS 5149, Springer-Verlag, Berlin, Heidelberg, p. 235-244 (2008)

Obtaining Rubric Weights For Assessments By More Than One Lecturer Using A Pairwise Learning Model

J. R. Quevedo[1] and E. Montañés[2]
[1]Artificial Intelligent Center, Oviedo University
(quevedo@aic.uniovi.es)
[2]Computer Science Department, Oviedo University
(montaneselena@uniovi.es)

Abstract. Specifying the criteria of a rubric to assess an activity, establishing the different quality levels of proficiency of development and defining weights for every criterion is not as easy as one a priori might think. Besides, the complexity of these tasks increases when they involve more than one lecturer. Reaching an agreement about the criteria and the levels of proficiency might be easier taking into account the abilities students must achieve according to the purpose of the subject. However, the disagreement about the weights of every criterion in an assessment rubric might easily appear. This paper focuses on the automatic weight adjustment for the criteria of a rubric. This fitting can be considered as a global perception that the whole group of lecturers have about the accuracy of solving an activity. Firstly, each lecturer makes a proposal of weights and then, from a set of pairs of students he/she globally expresses who of each pair has solved better the activity for which the rubric was designed. Secondly, an approach based on the pairwise learning is proposed in this work to obtain adequate weights for the criteria of a rubric. The system commits fewer errors than the lecturers and makes them improve and reconsider some aspects of the rubric.

1 Introduction

Lots of changes are involved within the new process of convergence to a European Space of Higher Education [12]. The subjects are designed as a set of abilities students must reach. The process encourages a careful inclusion and integration of several methodologies which must point in the right direction in order to guarantee students reach such abilities. Especial emphasis has been made over transversal abilities, such as group working highly demanded by companies from university graduates [1]. Finally, this new paradigm makes new assessment strategies arise to provide reliable information about the skills students in terms of the abilities must reach.

This paper focuses on developing a reliable mechanism to evaluate to what extent students acquire the abilities of a course when more than one lecturer with different perspective is involved in the assessment. Firstly, lecturers have to reach an agreement about what criteria are adequate to consider and then they have to establish minimum thresholds for them. A common strategy for this purpose consists in using evaluation rubrics, which have been increasingly gaining relevance in the last years. They are scoring guides that describe the requirements for various levels of proficiency in the development of certain activity. The purpose is to find out the conditions of success and their degree of fulfillment [11].

The contribution of this paper is a method which engages the variety of preferences every lecturer expresses to contrast and adjust weights to the criteria of a rubric. The challenge consists of including, reproducing and summing up as clearly as possible the global and different perception lecturers have about the achievement of the activities. The hypothesis of departure is that this method leads to reconsider some aspects in the design of a rubric. The approach is based on a pairwise learning system [7] based on Support Vector Machines (SVM). It feeds on information provided by making every lecturer decide between pairs of students who have solved an activity more satisfactorily. The fact that a system uses information of more than one expert (lecturer in this case) was successfully adopted before [2], [3].

2 Evaluation rubrics

Evaluation rubrics [9] can be defined as explicit summaries of the criteria for assessing a particular activity and the different levels of potential achievement for each criterion. There are many features a rubric should fulfill in order to be effective and helpful both for students and lecturers. Then, although it is not an easy task, lecturers must make an effort to carefully design them. One of the requirements must be to reflect the most significant elements related to success in a learning task. Commonly, this expresses the basic skills a student must reach. But they have to provide more information than just a list of goals. It must also enable both students and lecturers to accurately and consistently identify the level of competency of development. This makes students know beforehand what lecturers expect from them and what the characteristics a quality work is required to have. In relation to these conditions, rubrics have to encourage self-assessment of students to become more aware of their own learning process. In this sense, students do not have to wait for the correction of the lecturer to know if the development of the activity is adequate or not [10]. Besides, it helps lecturers to adjust the marks for the activities of the students more accurately and fairly, avoiding discriminatory treatment and adding transparency to the assessment process.

In any case, once the evaluation criteria have been established on the vertical axis of the rubric and the quality levels on the horizontal axis, the rubric must be completed in the sense referred to the weights of every criterion included in the final mark. Table 1 shows a general diagram of a rubric where Q means the quality levels, C corresponds to the criteria, W defines the weight, $N_{i,j}$ contains a description of what a student has been able to do in relation to the i criterion to reach the j level and w_i is the weight of the i criterion.

Table 1 General diagram of a rubric

C	*1*	*2*	…	*Q*	*W*
1st	$N_{1,1}$	$N_{1,2}$	…	$N_{1,q}$	$w_1(\%)$
…	…	…	…	…	…
pth	$N_{p,1}$	$N_{p,2}$	…	$N_{p,q}$	$w_q(\%)$

The weights in the rubric must be comparative. The assignment of a weight to each criterion to obtain the final mark shows the relevance a lecturer grants to it, to which one is assigned more and less relevance and their effect on the final mark.

3 Pairwise learning

In the design of a rubric the marks lecturers grant to two students according to his/her global and comparative perception between both are likely to disagree with the marks resulting from the predefined weights of the rubric. This could happen, for instance, when both students reach a certain level in some criteria but one of them is better in one criterion, for example, in originality or in the degree of knowledge. Moreover, the rank could be reversed. Besides, if lecturers were perfect experts, then it would be possible to assume that a mark in an evaluation scale with regard to certain criterion has the same meaning for all lecturers when evaluating all the students, although not for all lecturers [5]. Hence, this means that there are different scales with different levels. But, in any case a perfect expert would be coherent and would have great ability to discriminate, even the degree of the uniformity of the teaching team. Unfortunately, lecturers are not perfect experts. A common effect is known as the batch effect, which often modifies the marks so that a student obtains a higher/lower mark when he/she is assessed together with other students who are clearly worse/better.

Despite those discrepancies, the hypothesis that lecturers are able to decide who is better from two students in the execution of an activity can be regarded as reliable. Taking into account this principle, the goal is to obtain a global ranking of the resolutions of an activity from the partial ranking between members of a pair. If this is possible, and in fact it is, then it is not necessary for lecturers to agree on the marks, just to order them in pairs.

The students' marks in each criterion according to the weights previously defined by lecturers are available and that information will be expressed by judgement preferences. Fig. 1 shows a diagram of the whole process.

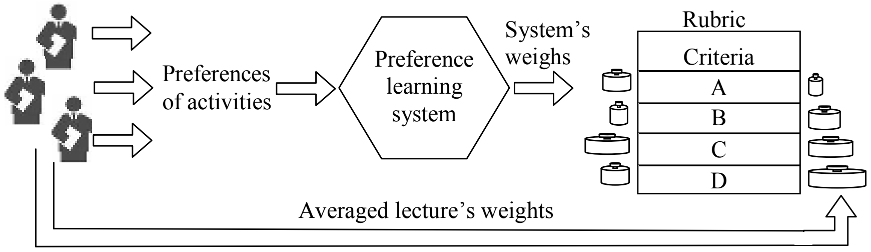

Preferences of activities
Preference learning system
System's weighs
Rubric
Criteria
A
B
C
D
Averaged lecture's weights

Fig. 1 Process of adjusting weights of the criteria of a rubric from preferences

A preference judgment [4] is an ordered pair whose first element has been preferred to the second. Then, a set of positive preferences (the first element is preferred to the second) and a set of negative ones (the second element is preferred to the first) are built. The challenge is to obtain a ranking as coherent as possible to the preferences expressed by the lecturers about the students. This ranking must be able to establish a correct order to other unknown items which could be introduced to the system in the future. Then, the target is to obtain a function from the set of solutions of an activity proposed by the students so that it satisfies the following rule for the majority of unknown students.

$$u \text{ is preferred to } v \leftrightarrow f(u) > f(v) \tag{1}$$

For this purpose, one could think of applying a traditional regression method. However, these methods aim to find a function that minimizes the loss between the real and forecasted values, whereas in the preference learning model the target is to minimize the times the value that defines the ranking of a student is lower than the second one in each preference, taking into account that such student has been preferred to the other. Then, this guarantees that the marks are coherent with the preferences rather than produces an exact mark.

The problem could also be reduced to find just a linear function, since the weights of the criteria are the same, regardless of the marks the students obtain in such criteria. For instance, if a student has the mark 7 in a criterion, the weight of this criterion is the same if such student had had the mark 9. An advantage of the linear case is the intuitive interpretation of the function, since each criterion will be weighted using its corresponding value. For instance, a null value means that this criterion does not have any effect over the final mark. Hence, the rule shown in (1) is expressed as the rules shown in (2) for positive and negative preferences respectively:

$$\begin{array}{l} u \text{ is preferred to } v \leftrightarrow f(u) > f(v) \leftrightarrow f(u-v) > 0 \\ v \text{ is preferred to } u \leftrightarrow f(u) < f(v) \leftrightarrow f(u-v) < 0 \end{array} \tag{2}$$

Therefore, the function will be positive if the first item is preferred to the second one and negative otherwise. This allows defining a training example as *u-v* with class +1 and *u-v* with class -1.

Once the data set is defined, it is possible to learn a ranking function from the binary classification yield by a binary classifier that separate the classes depending on the sign returned. Among the different methods available to obtain such function, one might a priori think of using a machine learning system based on a decision tree. But even if the performance of that kind of models is high, it may not be as useful as expected, since such methods tend to use the minimum possible number of features. From the point of view of the weights of the criteria, this means that many criteria will have weights equal to zero. However, SVM maximize the margin between the model and the nearest examples, called support vectors and this makes them use all relevant features [8]. Imagine that most of the students obtain the same degree of fulfillment in two criteria. These criteria are redundant since it will be easy to obtain the degree of one of them in function of the other. In this case, a decision tree would choose one of the criteria since both have the same relevance, whereas SVM would give similar weights, which is the desirable situation in this case. Figure 2 compares a possible model obtained by the two approaches, where a positive example is labeled with class +1 and a negative one with class -.

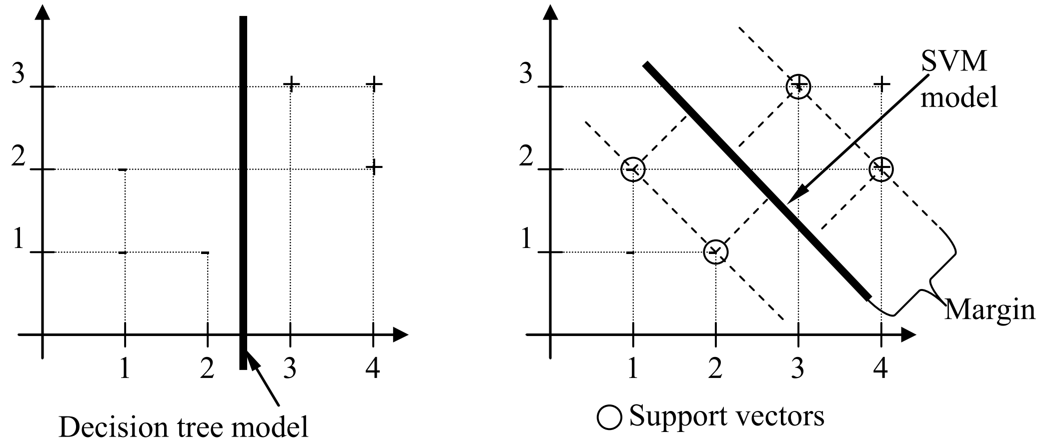


Figure 2. Representation of the model generated by two different binary separators. The decision tree model tries to use the minimum number of features, whereas SVM tries to generate a model that uses all relevant features.

Particularly, SVM [13], [14], [15] is a good choice for performing pairwise learning to aggregate the knowledge extracted from a set of different experts (lectures in this case) [2]. The linear learned function passes through the origin of coordinates and it is then expressed by

$$f(u) = \langle w, u \rangle = \sum_{j=1}^{p} w_j u_j \qquad (3)$$

where w is the vector that will defined the separation hyperplane, u is the representation of a student that will be the vector of marks in the criteria, $\langle w, u \rangle$ denotes the scalar product of vectors w and z and p is the number of criteria. From a student u the function provides a value that quantifies the preference of such student against other students. The weights of the criteria are obtained from the vector that defines the separation hyperplane. In fact, it is only necessary to normalize and multiply it by the maximum mark a student could reach in the activity. Unlike SVM, a complex process would have been necessary to obtain the weights from the rules produced by decision tree method

4 Experiments

This section describes the data set and some experimental settings. It also includes a discussion of the results.

4.1 Data set and experimental settings

The data set comes from a core and compulsory course of second year of the Computer Science degree related to database design. Several activities were proposed during the year to perform continuous evaluation. Particularly, the lecturers of the subject agree in defining 10 criteria for Activity 1, 9 criteria for Activities 2 and 4 and 8 criteria for

Activity 9. The number of students for the experiments was 44 and the number of lecturers was 3.

Two kind of judgement preferences were taken into account. The first group includes those preferences where one of the students of the pair has better mark in all criteria. In this case, the value of the preference is obvious, that is, it will be positive if such student is the first student of the pair and negative otherwise. All the preferences satisfying such condition were included in the data set. The preferences of the second kind are those whose students have better marks in some criteria and worse in others. In this case, just a large enough random sample of pairs of students (60 preferences) satisfying such condition was considered. Then, the sample was equally split in as many subsets as lecturers of the subject. Each sample set was presented to a lecturer who has had a look at the resolutions of every pair and has expressed his/her opinion about who solved the activity better within each pair. Obviously, lecturers have not take into account any weight of the criteria; they just express their own general impression about which one is better. Notice that taking into account all possible pairs means to compare about $(n^2-n)/2$ where n is the number of students (not all of them since the preferences of the first kind were removed from this group), what would be unapproachable.

The experiments were performed using LibSVM [6] with default parameters together with the Spider Matlab toolbox [16]. The default parameters consist of choosing a linear kernel and of setting the trade-off between training error and margin to be 1.

4.2 Discussion of results

Five different experiments were compared for each activity. The first three consisted of checking in what extent the preferences of each of the three lecturers are coherent with the marks computed according to their own weights previously fixed. This is shown in the first three rows of Tables 2-5. The fourth experiment consisted of checking in what extent the preferences of all the lecturers together are coherent with the marks computed according to the weights obtained as the average of the weights of the three lecturers. This is shown in the fourth row of Tables 2-5. Finally, the fifth experiment consisted of checking in what extent the preferences of all the lecturers together are coherent with the marks computed according to the weights the learning process produces from the preference data. This is shown in the last row of Tables 2-5. Notice that the errors committed when the averages of the weights among the lecturers are considered are not necessary the averaged errors committed by each lecturer on their own. Besides, the number of preferences considered when the averages of the weights are computed and when the learning process is applied is the sum of the preferences of all the lecturers.

Table 2 shows that lecturers commit some errors when they express their global impression with regard to their own weights of the criteria in Activity 1. Particularly, they disagree between 5% and 15% of the preferences, whereas the system is able to accurately reproduce a summary of all them (0% of error). Notice that using the averaged weights does not lead to an improvement. It seems that this activity presents great difficulties when defining a set of weights, since the weights of the system in general are quite different from those previously defined by the lecturers.

Table 2. Weights of the lecturers, weights averaged and weights of the system for Activity 1

	Criteria weights										Pairwise	
Lectures	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Nº**	**Errors**
1	2.5	2.5	1	1	1	1	0.25	0.25	0.25	0.25	20	5%
2	2	2	1	1	1	1	1	0.5	0.25	0.25	20	15%
3	2	3	1	1	1	0.5	0.5	0.5	0.25	0.25	20	15%
Average	2.17	2.5	1	1	1	0.83	0.58	0.42	0.25	0.25	60	11.66%
System	1.28	1.90	1.28	0.48	0.19	0.95	0.48	0.96	1.52	0.95	60	0%

In case of Activity 2 presented in Table 3, lecturers seem to agree among them about the weights, but there are slightly high differences between the marks of these weights and their own preferences, since they commit between 20% and 35% of errors. In this case the proposed system produces 8.33% of error against 20% if the average of weights is used. The differences between the weights produced by the system and those of the lecturer are useful for the lecturers as a feedback to make them think about the relevance of the criteria.

Table 3. Weights of the lecturers, weights averaged and weights of the system for Activity 2

	Criteria weights									Pairwise	
Lectures	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**Nº**	**Errors**
1	0.5	0.5	0.5	0.75	0.5	0.75	0.5	0.5	0.5	20	35%
2	0.5	1	0.5	0.5	0.5	0.75	0.5	0.25	0.5	20	20%
3	1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	20	20%
Average	0.67	0.67	0.5	0.58	0.5	0.67	0.5	0.42	0.5	60	20%
System	0.85	0.85	0.8	0.45	0.23	0.28	1.1	0.23	0.23	60	8.33%

The results of Activity 3 shown in Table 4 are quite similar to those of Activity 2. Again the system is able to engage the information of the lecturer team to reduce the error.

Table 4. Weights of the lecturers, weights averaged and weights of the system for Activity 3

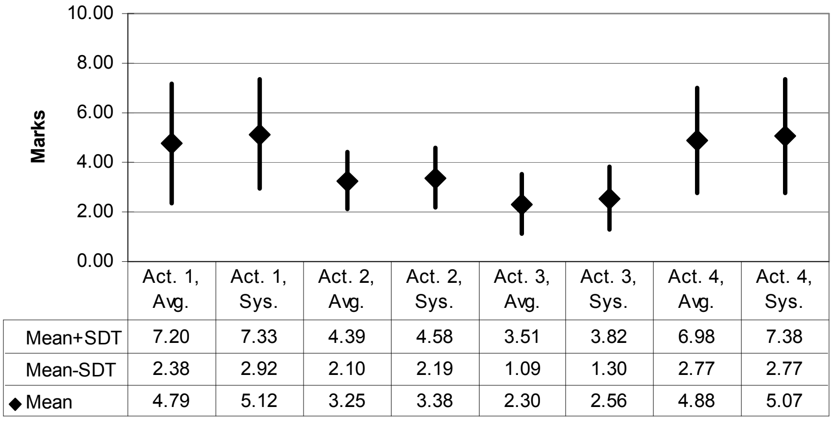
	Criteria weights								Pairwise	
Lectures	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**Nº**	**Errors**
1	0.75	0.75	0.5	0.75	0.75	0.5	0.5	0.5	20	15%
2	0.25	0.5	0.5	0.75	0.5	0.75	1	0.75	20	30%
3	1	0.5	0.25	1	0.25	0.75	1	0.25	20	25%
Average	0.67	0.58	0.42	0.83	0.5	0.67	0.83	0.5	60	21.66%
System	1.1	1.2	0.44	0.44	0.37	0.73	0.22	0.48	60	5%

Looking at Table 5 for Activity 4, criteria 8 and 9 is quite interesting. In this case lecturer 1 does not take into account criterion 8 and lecturer 2 does not take into account criterion 9, but lecturer 3 grants equal weight to both criteria. This is a conflictive case and the system according to the preferences of the lecturers agrees with lecturer 1 about the criteria 8 and with lecturer 3 about criteria 9. This proves that the system try to sum up the preferences of all lecturers, although it produces a bit more error than lecturer 1.

Table 5. Weights of the lecturers, weight averaged and weights of the system for Activity 4

	Criteria weights									Pairwise	
Lectures	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**Nº**	**Errors**
1	1.5	1	2	1	0.5	1	2	0	1	20	0%
2	0.5	2	2	1	0.5	1	2	1	0	20	25%
3	1	2	1.5	1	0.5	1	2	0.5	0.5	20	25%
Average	1	1.67	1.83	1	0.5	1	2	0.5	0.5	60	16,66%
System	1.94	1.40	1.71	1.29	0.89	0.63	1.65	0	0.49	60	5%

In general, the weights produced by the system differ from those granted by the lecturer before. Let us notice that the percentage of errors committed by the learning system are considerable lower that the rest ways of considering the weights. This means that this system is able to quite accurately reproduce the whole preferences of the lecturers. This also means that lecturers are not perfect experts because their own way of setting weights are not so coherent with their own preferences. Hence, the weights produced by the system make lecturers check their own incoherencies in order to change all or some weights which leads to establish a more accurate rubric. In fact, it helps to reach a consensus of the assessment process to encourage transparency and avoiding discriminatory treatment.



	Act. 1, Avg.	Act. 1, Sys.	Act. 2, Avg.	Act. 2, Sys.	Act. 3, Avg.	Act. 3, Sys.	Act. 4, Avg.	Act. 4, Sys.
Mean+SDT	7.20	7.33	4.39	4.58	3.51	3.82	6.98	7.38
Mean-SDT	2.38	2.92	2.10	2.19	1.09	1.30	2.77	2.77
◆ Mean	4.79	5.12	3.25	3.38	2.30	2.56	4.88	5.07

Figure 3. The averaged marks over the students when they are obtained from the weights averaged over the lecturers and from the weights the system grants

Applying the weighs the system produces would benefit some students and damages others. But, the question is that if there would be a global benefit or damage. Figure 3 shows the averaged marks of each activity together with the dispersion with regard to the use of averaged weights and to the use of the weights yield by the system.

At sight of Figure 3, one can observe that the mean and the deviation hardly vary between using average weights and the weights of the system. This allows concluding that the global benefit or damage will be the same. The advantage is that the marks of the students will be more accurately with regard to the global impression of the lecturer team. Notice that this process is internal among the lecturers and can be transparent for the students. Hence, it is not necessary to provide information to the student about the way of defining the weights.

5 Conclusions and Future Work

This work proposes a method based on preference learning to improve and adjust the weights granted to the criteria of an evaluation rubric according to the global impression of lecturers about pairs of activities solved by students when more than a lecturer is involved in the assessment process. The system proposed allows summing up the preferences of all the lecturers at the same time, and in fact, it reduces the errors between their own preferences and the original weights granted by every lecturer alone. Initially, lecturers give higher weight than the system yields from their preferences or vice versa. The tendency, unconsciously or not, of mixing criteria or taking into account other abilities such as transversal ones or those related to the attitude may be the cause of these disagreements. The results suggest lecturers must think about going more in depth into the design of the rubrics and about establishing more accurately the criteria and their relevance alone and together with their colleagues. Also, the weights the system grants benefit or damage the students the same with regard to consider the averages of the weights of all lecturers.

A proposal for future work is to find out if either grouping or breaking down the criteria makes lecturers improve the design of the rubrics.

Acknowledgements This research has been partially supported by the MICINN grants TIN2008-06247 and TIN2007-61273. The support of the University of Oviedo to the project entitled *La minería de datos como mecanismo de ayuda para la toma de decisiones en la actividad docente dentro del marco del Espacio Europeo de Educación Superior* is also gratefully acknowledged.

References

[1] Alan, J. Learning Outcomes in Higher Education. *Students in Higher Education,* 1996, 21(1), p. 93-108.

[2] Bahamonde, A., Bayón, G. F., Díez, J., Quevedo, J. R., Luaces, O., del Coz, J. J., Alonso, J., Goyache, F. Feature Subset Selection for Learning Preferences: A Case Study.

Proceedings of the 21st International Conference on Machine Learning, ICML 2004, Banff, Canada, 2004, p. 49-56.

[3] Bahamonde, A., Díez, J., Quevedo, J.R., Luaces, O., Del Coz, J. J. How To Learn Consumer Preferences From The Analysis Of Sensory Data By Means Of Support Vector Machines (SVM). Trends in Food Science & Technology, 18 (1), 2007, pp. 20-28.

[4] Branting, K., Broos, P. Automated Acquisition Of User Preferences. International Journal of Human-Computer Studies, 1997, p. 55–77.

[5] Cohen, W.W., Schapire, R.E., Singer, Y. Learning To Order Things. En Michael I. Jordan, Michael J. Kearns, y Sara A. Solla, editors, Advances in Neural Information Processing Systems, The MIT Press, 1998, 10.

[6] Fan R.E., Chen P.H., Lin. C.J. Working Set Selection Using Second Order Information For Training SVM. Journal of Machine Learning Research, 2005, 6, p. 1889-1918.

[7] Joachims, T. Optimizing Search Engines Using Clickthrough Data. In Proceedings of the ACM Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, July 23-26, 2002.

[8] Joachims, T. Text Categorization With Support Vector Machines: Learning With Many Relevant Features Proceedings of ECML-98, 10th European Conference on Machine Learning, Springer Verlag, Heidelberg, DE, 1998, 1398, p. 137-142.

[9] Moskal, B. M. Scoring Rubrics: What, When, And How? Practical Assessment, Research, and Evaluation, 2000, 7, 3.

[10] National Research Council. National Science Education Standards. Washington (DC): National Academy Press, 1996.

[11] Nitko, A.J. Educational Assessment Of Students. Upper Saddle River, NJ: Merrill, 2001.

[12] Realising the European Higher Education Area, Conference of European Ministers responsible for Higher Education, Berlín 2003. http://www.bologna-berlin2003.de/

[13] Schölkopf, B., Smola, A. J. Learning With Kernels. MIT Press, 2002.

[14] Shawe-Taylor, J. Cristianini, N. Kernel Methods For Pattern Analysis. Cambridge

University Press, 2004.

[15] Vapnik, V. Statistical Learning Theory. John Wiley, 1998.

[16] Weston J., Elisseeff A., BakIr G., Sinz F. Spider: Library Of Objects In Matlab. Software available at: http://www.kyb.tuebingen.mpg.de/bs/people/spider/

Collaborative Data Mining Tool for Education

Enrique García[1], Cristobal Romero[1], Sebastian Ventura[1], Miguel Gea[2], Carlos de Castro[1]

{egsalcines, cromero, sventura, cdecastro}@uco.es, mgea@ugr.es
[1]Department of Computer Science, University of Cordoba
[2]Department of Computer Languages and Systems, University of Granada

Abstract. This paper describes a collaborative educational data mining tool based on association rule mining for the continuous improvement of e-learning courses allowing teachers with similar course's profile sharing and scoring the discovered information. This mining tool is oriented to be used by instructors non experts in data mining such that, its internal operation is transparent to the user and the instructor can be focused in to the analysis of the results and make decisions about how to improve e-learning courses.

Introduction

Nowadays, there are a variety of general data mining tools and frameworks. Some examples of commercial mining tools are DBMiner [1], SPSS Clementine [2], DB2 Intelligent Miner [3], etc. And some examples of public domain mining tools are Weka [4], RapidMiner [5], Keel [6], etc. All these tools are not specifically designed for pedagogical/educational purposes and it is cumbersome for an educator to use these tools which are normally designed more for power and flexibility than for simplicity. However, there are also an increasing number of mining tools specifically oriented to educational data such as: Mining tool [7] for association and pattern mining, MultiStar [8] for association and classification, EPRules [9] for association, KAON [10] for clustering and text mining, Synergo/ColAT [11] for statistics and visualization, GISMO [12] for visualization, Listen tool [13] for visualization and browsing, TADA-Ed [14] for visualizing and mining, O3R [15] for sequential pattern mining, Sequential Mining tool [16] for pattern mining, MINEL [17] for mining learning paths, Simulog [18] for looking for unexpected behavioral pattern. Moodle mining tool [19] for classification, clustering and association rule mining. All these tools are oriented to be used by a single instructor or course administrator in order to discover useful knowledge from their own courses. So, they don't allow a collaborative usage in order to share all the discovered information between other instructors of similar courses (contents, subjects, educational type: elementary and primary education, adult education, higher, tertiary and academic education, special education, etc.). In this way, the information discovered locally by teachers could be joined and stored in a common repository of knowledge available for all instructors for solving similar detected problems.

In this paper, we describe an educational data mining tool based on association rule mining and collaborative filtering for the continuous improvement of e-learning courses and it directed to teachers non experts in data mining. The main objective is to make a mining tool in which the information discovered can be shared and scored between different instructors and experts in education.

Implementation of the collaborative data mining tool

We have developed a data mining tool with two subsystems: client and server application (Figure 1). The client application uses an association rule mining tool for discovering interesting relationships through student's usage data in the form of IF-THEN recommendation rules. The server application uses a collaborative recommender system to share and score the previously obtained rules by instructors of similar courses with other instructors and experts in education.

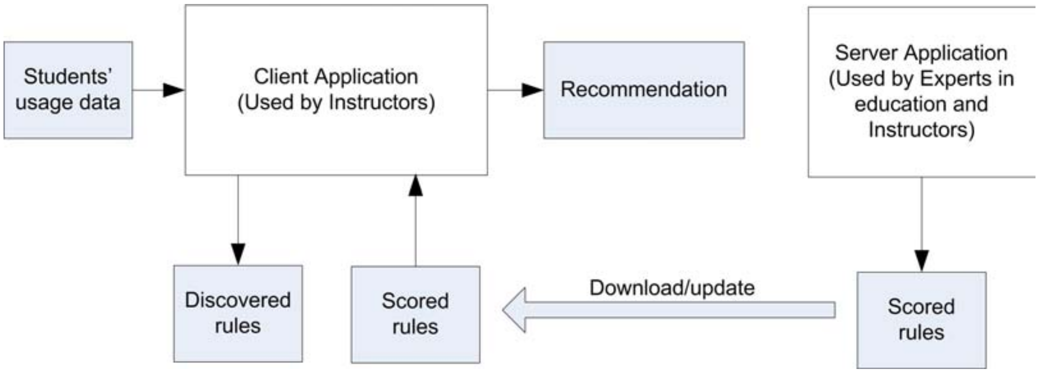

Students' usage data
Client Application (Used by Instructors)
Recommendation
Server Application (Used by Experts in education and Instructors)
Discovered rules
Scored rules
Download/update
Scored rules

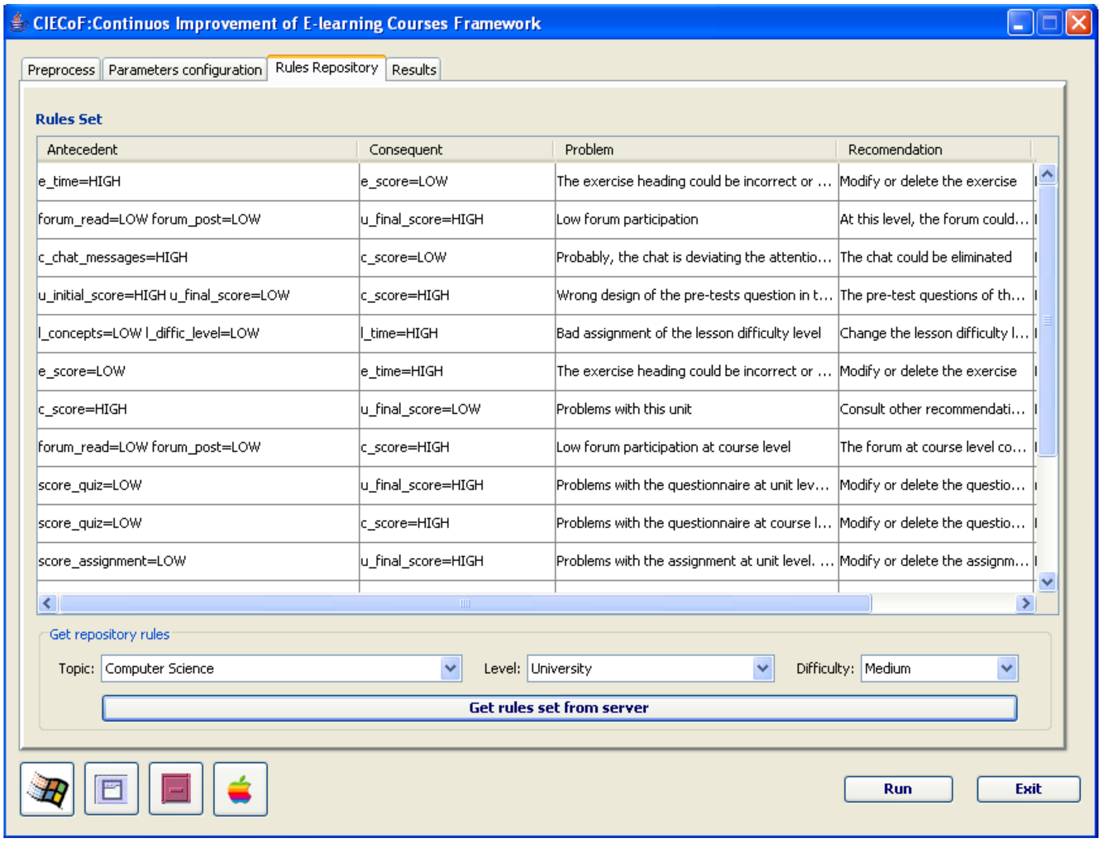
Figure 1. Collaborative data mining tool

As we can see in Figure 1, the system is based on client-server architecture with N clients, which applies an association rule mining algorithm locally on students' usage data. In fact, the client application uses the Predictive Apriori algorithm [20], because it does not require the user to specify parameters such as the minimum support threshold or confidence values. The only parameter is the number of rules to be discovered, which is a more intuitive parameter for a teacher non expert in data mining. The association rules discovered by the client application must be evaluated to decide if they are relevant or not, therefore the client application uses an evaluation measure [21] to classify the rules as being expected or unexpected, comparing them with the scored rules stored in a collaborative rules repository maintained on server side. Also, the expected rules found are then expressed in a more comprehensible format of recommendation about possible solutions to problems detected in the course. The teacher sees the recommendation and can determine if it is relevant or not for him/her in order to apply/use the recommendation. On the other side, the server application allows managing the rules repository using collaborative filtering techniques with knowledge-based techniques [21]. The information in the knowledge base is stored in form of tuples (rule-problem-recommendation-relevance) which are classified according to a specific course profile. The course profile is represented as a three-dimensional vector related with the following characteristic of his/her course: Topic (the area of knowledge, e.g. Computer Science or Biology); Level (level of the course, e.g. Universitary, High School, Elementary or Special Education); and Difficulty (the difficulty of the course, e.g., Low or High). These similarities between courses are available to other teachers to assess in terms of applicability and relevance. A group of experts in online education from University of Córdoba, Spain, propose the first tuples of the rule repository and also vote on those tuples proposed by other experts. On the other hand, teachers could discover new tuples (in the client application) but these must be validated by the experts (in the sever application) before being inserted in the rule repository.

1.1 *Client application*

As we mentioned before, the client application is used by instructors in order to find association rules. The main feature of the client application is its specialization in educational environments. Before applying our mining algorithm, the data have to be pre-processed in order to adapt them to our specific data model. First, the teacher has to select the origin of the data to be mined. We have two different formats available for input data: 1) the Moodle relational database, for teachers that work with Moodle as well as the INDESAHC authoring tool [22], so all our attributes are used directly; or 2) a Weka [4] ARFF text file, for teachers that use other LMSs and, therefore, other attributes. Also, the teacher can restrict the search field, we have also added a few parameters related with the analysis depth. Firstly, the teacher must select the level of granularity to carry out the analysis: course, unit, lesson or a specific table of the data base such as course-unit, course-lesson, course-exercise, course-forum, unit-exercise, unit-lesson, lesson-exercise among others.

The rules repository (see Figure 2) is the knowledge database upon which the analysis of the discovered rules is based. Before running the algorithm, the teacher downloads from the server, the current knowledge database, according to his/her course profile.



CIECoF:Continuos Improvement of E-learning Courses Framework
Preprocess
Parameters configuration
Rules Repository
Results

Rules Set

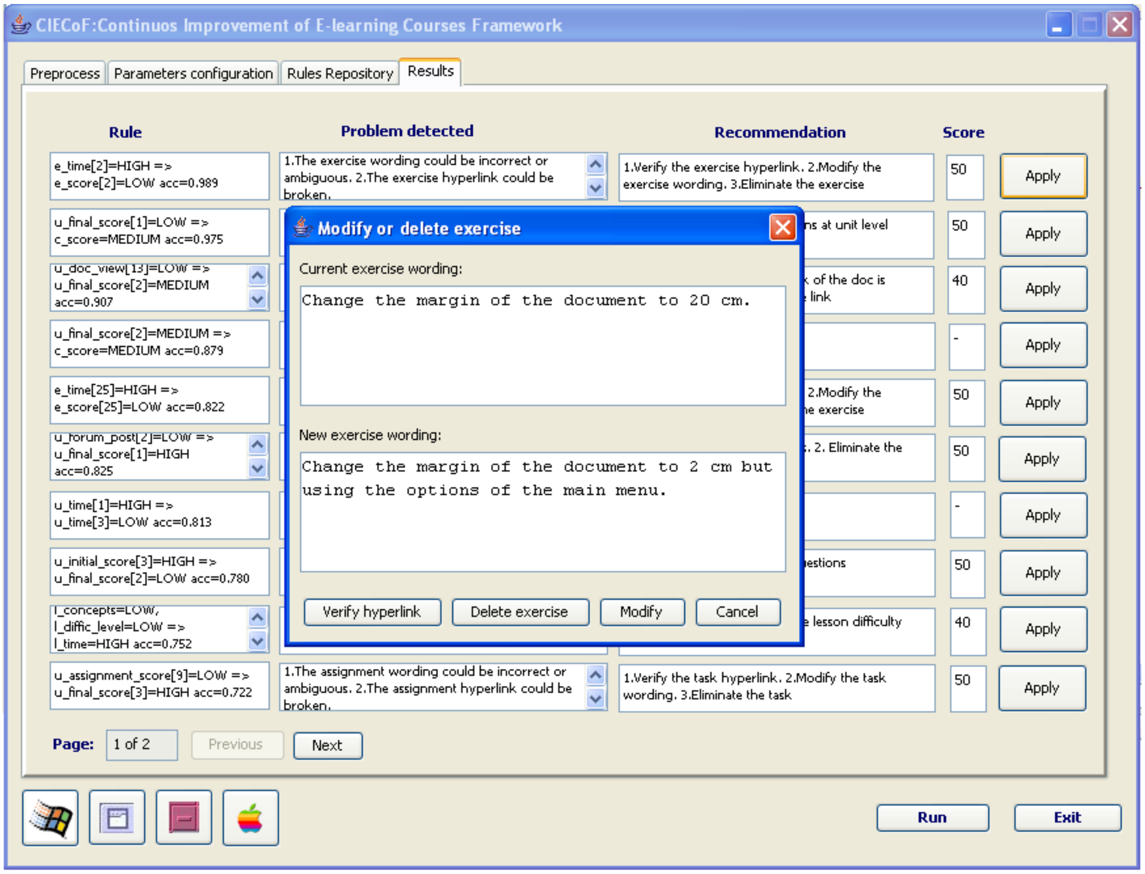
Antecedent	Consequent	Problem	Recomendation
e_time=HIGH	e_score=LOW	The exercise heading could be incorrect or ...	Modify or delete the exercise
forum_read=LOW forum_post=LOW	u_final_score=HIGH	Low forum participation	At this level, the forum could...
c_chat_messages=HIGH	c_score=LOW	Probably, the chat is deviating the attentio...	The chat could be eliminated
u_initial_score=HIGH u_final_score=LOW	c_score=HIGH	Wrong design of the pre-tests question in t...	The pre-test questions of th...
l_concepts=LOW l_diffic_level=LOW	l_time=HIGH	Bad assignment of the lesson difficulty level	Change the lesson difficulty l...
e_score=LOW	e_time=HIGH	The exercise heading could be incorrect or ...	Modify or delete the exercise
c_score=HIGH	u_final_score=LOW	Problems with this unit	Consult other recommendati...
forum_read=LOW forum_post=LOW	c_score=HIGH	Low forum participation at course level	The forum at course level co...
score_quiz=LOW	u_final_score=HIGH	Problems with the questionnaire at unit lev...	Modify or delete the questio...
score_quiz=LOW	c_score=HIGH	Problems with the questionnaire at course l...	Modify or delete the questio...
score_assignment=LOW	u_final_score=HIGH	Problems with the assignment at unit level. ...	Modify or delete the assignm...

Get repository rules

Topic: Computer Science Level: University Difficulty: Medium

Figure 2. Rules repository panel

Finally, after downloading the rule repository and configuring the application parameters or using default values, the teacher executes the association rule algorithm. Then, client application shows the results obtained in a table (see Figure 3), with the following fields: rule (discovered IF-THEN rule), problem (detected by the rule), recommendation (about how to solve the problem), score (of experts and others instructors have set to the rule) and apply button (to use/apply the recommendation in his/her course).



Rule	Problem detected	Recommendation	Score	
e_time[2]=HIGH => e_score[2]=LOW acc=0.989	1.The exercise wording could be incorrect or ambiguous. 2.The exercise hyperlink could be broken.	1.Verify the exercise hyperlink. 2.Modify the exercise wording. 3.Eliminate the exercise	50	Apply
u_final_score[1]=LOW => c_score=MEDIUM acc=0.975		ns at unit level	50	Apply
u_doc_view[13]=LOW => u_final_score[2]=MEDIUM acc=0.907		k of the doc is link	40	Apply
u_final_score[2]=MEDIUM => c_score=MEDIUM acc=0.879			-	Apply
e_time[25]=HIGH => e_score[25]=LOW acc=0.822		2.Modify the he exercise	50	Apply
u_forum_post[2]=LOW => u_final_score[1]=HIGH acc=0.825		. 2. Eliminate the	50	Apply
u_time[1]=HIGH => u_time[3]=LOW acc=0.813			-	Apply
u_initial_score[3]=HIGH => u_final_score[2]=LOW acc=0.780		estions	50	Apply
l_concepts=LOW, l_diffic_level=LOW => l_time=HIGH acc=0.752		lesson difficulty	40	Apply
u_assignment_score[9]=LOW => u_final_score[3]=HIGH acc=0.722	1.The assignment wording could be incorrect or ambiguous. 2.The assignment hyperlink could be broken.	1.Verify the task hyperlink. 2.Modify the task wording. 3.Eliminate the task	50	Apply

Modify or delete exercise

Current exercise wording:

```
Change the margin of the document to 20 cm.
```

New exercise wording:

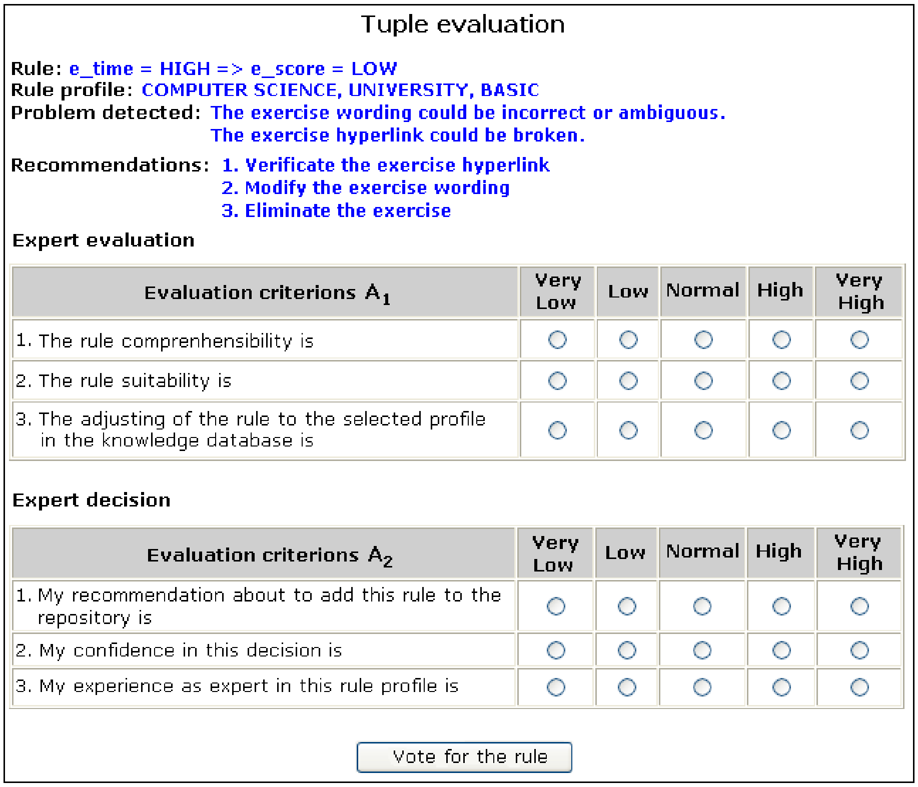
```
Change the margin of the document to 2 cm but
using the options of the main menu.
```

Figure 3. Results panel

We have distinguished between two types of recommendations: 1) Active, if it implies a direct modification of the course content or structure; or 2) Passive, if it detects a more general problem in the course or unit and it advices the teacher to consult more specific recommendations related with these didactic resources. Active recommendations can be linked to: modifications in the formulation of the questions (see Figure 3) or the practical exercises/tasks assigned to the students; changes in previously assigned parameters such as course duration or the level of lesson difficulty; or the elimination of a resource such as a forum or a chat room.

1.2 Server application

The server application is used by experts and instructors. The experts in education insert the tuples and they explicitly vote for them by indicating degrees of preference (see Figure 4). The teachers vote implicitly when they push the “Apply” button, in order to side-step one of the main problems for collaborative filtering systems, that is how to encourage teachers to vote or evaluate. In this case, if teachers apply one of the recommendations to their course, they are implicitly voting for this specific tuple.



Tuple evaluation

Rule: e_time = HIGH => e_score = LOW
Rule profile: COMPUTER SCIENCE, UNIVERSITY, BASIC
Problem detected: The exercise wording could be incorrect or ambiguous.
The exercise hyperlink could be broken.

Recommendations: 1. Verificate the exercise hyperlink
2. Modify the exercise wording
3. Eliminate the exercise

Expert evaluation

Evaluation criterions A_1	Very Low	Low	Normal	High	Very High
1. The rule comprenhensibility is	○	○	○	○	○
2. The rule suitability is	○	○	○	○	○
3. The adjusting of the rule to the selected profile in the knowledge database is	○	○	○	○	○

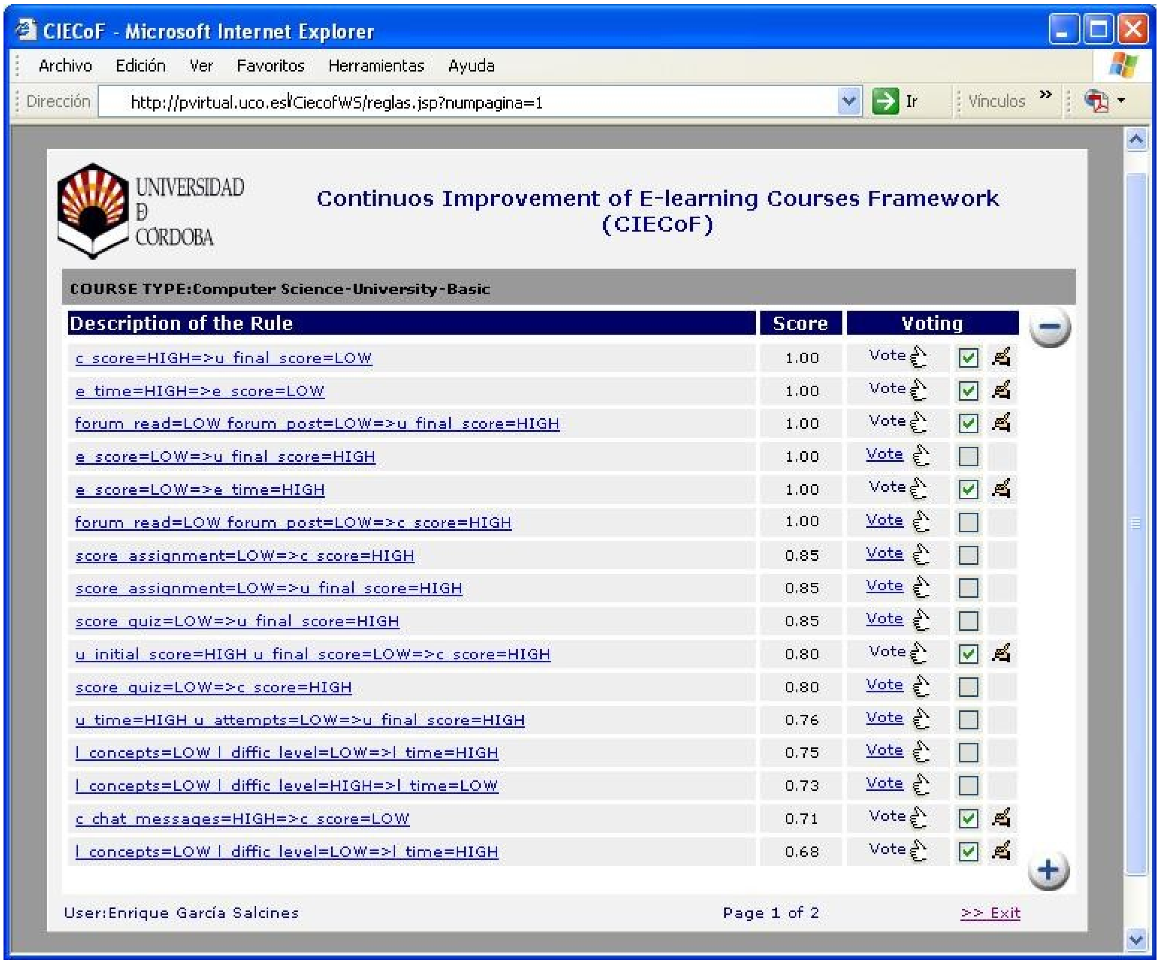
Expert decision

Evaluation criterions A_2	Very Low	Low	Normal	High	Very High
1. My recommendation about to add this rule to the repository is	○	○	○	○	○
2. My confidence in this decision is	○	○	○	○	○
3. My experience as expert in this rule profile is	○	○	○	○	○

Figure 4. Vote panel

The server application is a web-based application for managing the knowledge database or tuple repository (see Figure 5). In order to access easily to all the editing options for the repository, a general course profile was created which is the profile used by the experts in educational domain. These experts have permission to introduce new tuples into the rule repository and vote explicitly for existing ones (see Figure 4). In order to allow information exchange (tuples) between client and server, we have developed a web service for downloading/uploading the repository. Each time a client application updates its repository, all the tuples are reordered in the repository.

Finally, we must mention that an evaluation of this collaborative data mining tool can be found in [21].


CIECoF - Microsoft Internet Explorer
Archivo
Edición
Ver
Favoritos
Herramientas
Ayuda
Dirección
http://pvirtual.uco.es/CiecofWS/reglas.jsp?numpagina=1
Ir
Vínculos
UNIVERSIDAD
D
CÓRDOBA

Continuos Improvement of E-learning Courses Framework (CIECoF)

COURSE TYPE:Computer Science-University-Basic

Description of the Rule	Score	Voting		
c_score=HIGH=>u_final_score=LOW	1.00	Vote	☑	
e_time=HIGH=>e_score=LOW	1.00	Vote	☑	
forum_read=LOW forum_post=LOW=>u_final_score=HIGH	1.00	Vote	☑	
e_score=LOW=>u_final_score=HIGH	1.00	Vote	□	
e_score=LOW=>e_time=HIGH	1.00	Vote	☑	
forum_read=LOW forum_post=LOW=>c_score=HIGH	1.00	Vote	□	
score_assignment=LOW=>c_score=HIGH	0.85	Vote	□	
score_assignment=LOW=>u_final_score=HIGH	0.85	Vote	□	
score_quiz=LOW=>u_final_score=HIGH	0.85	Vote	□	
u_initial_score=HIGH u_final_score=LOW=>c_score=HIGH	0.80	Vote	☑	
score_quiz=LOW=>c_score=HIGH	0.80	Vote	□	
u_time=HIGH u_attempts=LOW=>u_final_score=HIGH	0.76	Vote	□	
l_concepts=LOW l_diffic_level=LOW=>l_time=HIGH	0.75	Vote	□	
l_concepts=LOW l_diffic_level=HIGH=>l_time=LOW	0.73	Vote	□	
c_chat_messages=HIGH=>c_score=LOW	0.71	Vote	☑	
l_concepts=LOW l_diffic_level=LOW=>l_time=HIGH	0.68	Vote	☑	

User:Enrique García Salcines   Page 1 of 2   >> Exit

Figure 5. Server application interface

2 Conclusions

In this paper we have shown a data mining tool that uses association rule mining and collaborative filtering in order to make recommendation to instructors about how to improve e-learning courses. This tool enables to share and score the discovered rules by other teachers of similar courses. Currently, the mining tool has been only used by a group of instructors and expert involved in the development of the own tool. So, in the future we want to test the tool with several groups of external instructors and experts in order to can test the usability of the tool with external users.

Acknowledgments

The authors gratefully acknowledge the financial support provided by the Spanish department of Research under TIN2008-06681-C06-03 and P08-TIC-3720 Projects.

References

[1] DBMiner. Available at http://www.dbminer.com. 2009.

[2] Clementine. Available at http://www.spss.com/clementine/. 2009.

[3] Miner. Available at http://www-306.ibm.com/software/data/iminer/. 2009

[4] Weka. Available at http://www.cs.waikato.ac.nz/ml/weka/. 2009

[5] RapidMiner. Available at http://rapid-i.com/. 2009.

[6] Keel. Available at http://www.keel.es/. 2009.

[7] Zaïane, O., Luo, J. Web usage mining for a better web-based learning environment. *In Proceedings of conference on advanced technology for education*, 2001. Banff, Alberta , pp. 60–64.

[8] Silva, D., Vieira, M., 2002. Using data warehouse and data mining resources for ongoing assessment in distance learning. *In IEEE International Conference on Advanced Learning Technologies*, 2002. Kazan, Russia, pp. 40–45.

[9] Romero, C., Ventura, S., and Bra, P. D. Knowledge discovery with genetic programming for providing feedback to courseware author. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, 2004, 14(5), pp. 425–464.

[10]Tane, J., Schmitz, C., Stumme, G. Semantic resource management for the web: An elearning application. *In Proceedings of the WWW Conference*, 2004 New York, USA, pp. 1–10.

[11]Avouris, N., Komis, V., Fiotakis, G., Margaritis, M., Voyiatzaki, E. Why logging of fingertip actions is not enough for analysis of learning activities. *In Workshop on Usage analysis in learning systems at the 12th International Conference on Artificial Intelligence in Education*, 2005. Amsterdam, Netherland, pp. 1-8.

[12]Mazza, R., Milani, C. Exploring usage analysis in learning systems: Gaining insights from visualisations. *In Workshop on Usage analysis in learning systems at 12th International Conference on Artificial Intelligence in Education*, 2005. New York, USA, pp. 1-6.

[13]Mostow, J., Beck, J., Cen, H., Cuneo, A., Gouvea, E., Heiner, C. An educational data mining tool to browse tutor-student interactions: Time will tell! In *Proceedings of the Workshop on Educational Data Mining*, 2005. Pittsburgh, USA, pp. 15–22.

[14]Merceron, A., Yacef, K. TADA-Ed for educational data mining. *Interactive multimedia electronic journal of computer-enhanced learning*, 2005, 7(1), pp. 267-287.

[15]Becker, K., Vanzin, M., Ruiz, D. A. Ontology-based filtering mechanisms for web usage patterns retrieval. *In International Conference on E-Commerce and Web Technologies*, 2005. München, Germany, pp. 267–277.

[16]Romero, C., Porras, A.R., Ventura, S., Hervás, C., Zafra, A. Using sequential pattern mining for links recommendation in adaptive hipermedia educational systems. *In International Conference Current Developments in Technology-Assisted Education*, 2006. Sevilla, Spain, pp.1016-1020.

[17]Bellaachia, A., Vommina, E., Berrada, B. Minel: a framework for mining e-learning logs. *In Proceedings of the 5th IASTED international conference on Web-based education*, 2006. Mexico, pp. 259-263.

[18]Bravo, J., Ortigosa, A. Validating the Evaluation of Adaptive Systems by User Profile Simulation. *In Proc. Workshop on User-Centred Design and Evaluation of Adaptive Systems*, 2006, Dublin, Germany, pp. 52-56.

[19]Data mining algorithms to classify students. C. Romero, S. Ventura, P. Espejo, C. Hervas. *Educational Data Mining Conference EDM 2008*, 2008, 20-21 Junio, Montreal, pp. 8-17.

[20]Scheffer, T. Finding Association Rules That Trade Support Optimally against Confidence. *Intelligent Data Analysis*, 2005, 9(4), pp. 381-395.

[21]García, E., Romero, C., Ventura, S., de Castro, C. An architecture for making recommendations to courseware authors through association rule mining and collaborative filtering. *UMUAI: User Modelling and User Adapted Interaction*, 2009, 19(1-2), pp. 99-132.

[22]De Castro, C., García, E., Romero, C., and Ventura, S. Herramienta autor INDESAHC para la creación de cursos hipermedia adaptativos. *Revista latinoamericana de tecnología educativa* 3, 2004, pp. 349-367.

Predicting Student Grades in Learning Management Systems with Multiple Instance Genetic Programming

Amelia Zafra and Sebastián Ventura
{azafra,sventura}@uco.es
Computer Science and Numerical Analysis Department, University of Cordoba

Abstract. The ability to predict a student's performance could be useful in a great number of different ways associated with university-level learning. In this paper, a grammar guided genetic programming algorithm, G3P-MI, has been applied to predict if the student will fail or pass a certain course and identifies activities to promote learning in a positive or negative way from the perspective of Multiple Instance Learning (MIL). Computational experiments compare our proposal with the most popular techniques of MIL. Results show that G3P-MI achieves better performance with more accurate models and a better trade-off between such contradictory metrics as sensitivity and specificity. Moreover, it adds comprehensibility to the knowledge discovered and finds interesting relationships that correlate certain tasks and the time devoted to solving exercises with the final marks obtained in the course.

1 Introduction

The design and implementation of the virtual learning environment (VLE) or e-learning platforms have grown exponentially in the last years, spurred by the fact that neither students nor teachers are bound to a specific location and that this form of computer-based education is virtually independent of any specific hardware platforms [1]. These systems can potentially eliminate barriers and provide: flexibility, constantly updated material, student memory retention, individualized learning, and feedback superior to the traditional classroom, thus becoming an essential accessory to support both the face-to-face classroom and distance learning.

The use of these applications accumulates a great amount of information because they can record all the information about students' actions and interactions in log files and data sets. Nowadays, there has been a growing interest in analyzing this valuable information to detect possible errors, shortcomings and improvements in student performance and discover how the student's motivation affects the way he or she interacts with the software [2-4]. All previous studies have used traditional supervised learning to represent the problem. However, such representation generates instances with many missing values because the information about the problem is incomplete. Each course has different types and numbers of activities and each student carries out the number of activities considered most interesting, dedicating more or less time to resolve them. In this context, the Multiple Instance Learning (MIL) representation makes possible a more appropriate representation of available information. MIL stores the general information of each pattern by means of bag attributes and specific information about the student's work on each pattern by means of a variable number of instances. This paper tackles the problem from a MIL perspective and presents a grammar guided genetic programming (G3P) algorithm, G3P-MI, to solve it. The most representative paradigms in MIL are compared to our proposal. Experimental results show that G3P-MI is more effective in obtaining a more accurate model as well as in finding a trade-off between contradictory measurements like sensitivity and specificity. Moreover, it adds comprehensibility to the knowledge discovered, allowing interesting relationships between activities, resources and results to be obtained.

The paper is organized as follows. Section 2 introduces multi-instance learning and section 3 presents the problem of classifying students' performance from a multi-instance perspective. Section 4 reports on experiment results which compare our proposal to the most representative multiple instance learning paradigms. Finally, Section 5 summarizes the main contributions of this paper and suggests some future research directions.

2 Multiple Instance Learning

Multiple Instance Learning (MIL) introduced by Dietterich et al. [5] consists of generating a classifier that will correctly classify unseen patterns. The main characteristic of this learning is that the patterns are bags of instances where each bag can contain different numbers of instances. There is information about the bags because a bag receives a special label, but the labels of instances are unknown. According to the standard learning hypothesis proposed by Dietterich et al. [6] a bag is positive if and only if at least one of its instances is positive, and it is negative if none of its instances produce a positive result. The key challenge in MIL is to cope with the ambiguity of not knowing which of the instances in a positive bag is really a positive example and which is not. In this sense, this learning problem can be regarded as a special kind of supervised learning problem where the labeling information is incomplete.

This learning framework is receiving growing attention in the machine learning community because numerous real-world tasks can be very naturally represented as multiple instance problems. If we go through them, we can find specifically developed algorithms for solving MIL problems [5,6,7] or, on the other hand, contributions which adapt popular machine learning paradigms to the MIL context, such as multi-instance lazy learning algorithms [8], multi-instance tree learners and multi-instance rule inducers [9], multi-instance neural networks [10], multi-instance kernel methods [11], multi-instance ensembles [12] and finally, a multi-instance evolutionary algorithm [13].

3 Predicting Students' performance based on the e-learning Platform

Predicting student's performance based on work they have done on the Virtual Learning Platform is an issue under much research. This problem shows interesting relationships that can suggest activities and resources to students and educators that can favour and improve both their learning and effective learning process. Thus, it can be determined if all the additional material provided to the students (web-based homework) helps them to assimilate the concepts and subjects developed in the classroom or if some activities are not useful to improve the final results.

The problem could be formulated as follows. A student could do different activities in a course to enable him to acquire and strengthen the concepts acquired in class. Later, at the end of the course, there is a final exam. A student with a final score higher or equal than a minimum required passes a module, while a student with a mark lower than that minimum fails that lesson or module. With this premise, the problem consists of predicting if the student will pass or fail the module considering the time dedicated, the number and type of activities done for the student during the course.

The types of activities considered in this study are quizzes, assignments and forums. They have shown its effectiveness to strengthen the learning in a lot of studies. A summary of the information available for each activity in our study is shown in Table1.

Table1. Information summary considered in our study

ACTIVITY	ATTRIBUTE NAME	ATTRIBUTE DESCRIPTION
Assignment	numberAssignment	Number of practices/tasks done by the user in the course.
	timeAssignment	Total time in seconds that the user has been in the assignment.
Forum	numberPosts	Number of messages sent by the user forum.
	numberRead	Number of messages read by the user forum.
	timeForum	Total time in seconds that the user has been in the forum.
Quiz	numberQuiz	Number of quizzes seen by the user.
	numberQuiz_a	Number of quizzes passed by the user.
	numberQuiz_s	Number of quizzes failed by the user.
	timeQuiz	Total time in seconds that the user has been in the quiz.

3.1 MIL representation of the problem

In this problem, each student can execute a different number of activities: a hard-working student may do all the activities available but, on the other hand, there can be students who have not done any activities. Moreover, there are some courses with only a few activities along with others with an enormous variety and number of them. MIL allows a representation that adapts itself perfectly to the concrete information available for each student, eliminating the missing values that abound when traditional representation is used. In MIL representation, each pattern represents a student registered in a course. Each student is regarded as a bag which represents the work carried out. Each bag is composed of one or several instances. Each instance represents the different types of work that the student has done. Therefore, each pattern/bag will have as many instances as the different types of activities done by the student. This representation fits the problem completely because general information about the student and course is stored as bag attributes, and variable information is stored as instance attributes.

Each instance is divided into 3 attributes: type of Activity, number of exercises in that activity and the time devoted to completing it. Eight activity types are considered which are *ASSIGNMENT_S*, number of assignments that the student has submitted, *ASSIGNMENT* referring to the number of times the student has visited the activity without submitting finally any file. *QUIZ_P*, number of quizzes passed by the student, *QUIZ_F* number of quizzes failed by the student, *QUIZ* referring to the times the student has visited a survey without actually answering it, *FORUM_POST* number of messages that the student has submitted, *FORUM_READ* number of messages that the student has read and *FORUM* that refers to the times the student has seen different forums without entering them. In addition, the bag contains three attributes, student identification, course identification and the final mark obtained by the student in that course. A summary of the attributes that belong to the bag and to the instances is presented in Table2.

Table2. Information about bags and information about instances

BAG		INSTANCE	
User-Id	Student identifier.	*TypeActivity*	Type of activity which represents the instance. The type of activities considered are eight: FORUM read, written or consulted, QUIZ passed or failed and ASSIGNMENT submitted or consulted.
Course	Course identifier.	*timeActivity*	Time spent to complete the tasks of this type of activity.
FinalMark	Final mark obtained by the student in this course.	*numberActivity*	Number of activities of this type completed by the student.

4 Experimentation and Results

Experiments compare the performance of G3P-MI to other MIL techniques. All experiments are carried out using 10-fold stratified cross validation and 10 different runs for each partition are executed to measure the performance of evolutionary algorithm. First, the problem domain is described briefly. Then, the results are shown and discussed. Finally, the comprehensibility of the rules generated by G3P-MI will be shown.

4.1 Problem domain used in Experimentation

This study employs the students' usage data from the virtual learning environment at Cordoba University that makes use of Moodle platform[14]. The research includes the information for 7 courses with 419 students. The details about the 7 e-Learning courses are given in Table 3. For the purpose of our study, the collection of data was carried out during an academic year from September to June, just before the Final Examinations. All information about each student for both representations is exported to a text file using Weka ARFF format [15].

Table3. General information about the courses

COURSE IDENTIFIERS	ICT-29	ICT-46	ICT-88	ICT-94	ICT-110	ICT-111	ICT-218
Number of Students	118	9	72	66	62	13	79
Number of Assignments	11	0	12	2	7	19	4
Number of Forums	2	3	2	3	9	4	5
Number of quizzes	0	6	0	31	12	0	30

4.2 Multi-Instance Grammar Guided Genetic Programming

G3P-MI is an extension of traditional GP systems, called grammar-guided genetic programming G3P [16]. G3P facilitates the efficient automatic discovery of empirical laws providing a more systematic way to handle typing by using a context-free grammar which establishes a formal definition of syntactical restrictions. The motivation to include this paradigm is that it retains a significant position due to a flexible representation using solutions of variable length and the low error rates that it achieves both in obtaining classification rules, and in other tasks related to prediction, such as feature selection and the generation of discriminant functions.

We follow an approach where an individual represents IF-THEN rules that add comprehensibility to the discovered knowledge and the fitness function to evaluate the rules obtained will be *sensitivity *specificity*. These measurements allow us to consider both successes in the positive and negative class assigning a value of 0 when no example of one class is classified and value of 1 when both classes are full classified.

The main steps of our algorithm are based on a classical generational and elitist evolutionary algorithm. Initially, a population of classification rules is generated. Once the individuals are evaluated with respect to their ability to solve the problem, the main loop of the algorithm is composed of the parent selection using a binary tournament selector, then recombination and mutation processes [16] are carried out with a probability of 90% and 10% respectively, and finally, the population is updated by direct replacement with elitism, that is, the offspring replace the present population and the best individual in the population is included. The procedure is repeated until de algorithm reaches a maximum number of one hundred generations or the best individual in the population achieves a full classification (a value of 1 in fitness function).

4.3 Comparison with Multiple Instance Learning techniques

The most relevant proposals based on MIL presented to date are considered to solve this problem and compared to our proposal designed in JCLEC framework [17]. The different paradigms compared included, *Methods based on Diverse Density*: MIDD, MIEMDD and MDD; *Methods based on Logistic Regression*: MILR; *Methods based on Support Vector Machines*: MISMO uses the SMO algorithm for SVM learning in conjunction with an MI kernel; *Distance-based Approaches*: CitationKNN and MIOptimalBall; *Methods based on Supervised Learning Algorithms*: MIWrapper using different learners, such as Bagging, PART, SMO, AdaBoost and NaiveBayes; MISimple using PART and AdaBoost as learners and MIBoost. More information about the algorithms considered could be consulted at the WEKA workbench [15] where these techniques are designed. The average results of accuracy, sensitivity and specificity are reported in Table 4.

G3P-MI obtains the most accurate models. Also, this approach achieves a trade-off between the contradictory measurements of sensitivity and specificity. If we observe the results of the different paradigms, it can be seen how they optimise the sensibility measurement in general at the cost of a decrease in the specificity value. This leads to an incorrect prediction of which students will pass the course. This classification problem has an added difficulty since we are dealing with a variety of courses with different numbers and types of exercises which make it more complicate to establish general relationships among them. Nonetheless, G3P-MI in this sense is the one that obtains the best trade-off between the two measurements, obtaining the highest values for sensitivity. Moreover, G3P-MI obtains interpretable rules to find pertinent relationships that could determine if certain activities influence the student's ability to pass, if spending a certain amount of time on the platform is an important contribution or if there is any other interesting link between the work done and the final results obtained.

Table 4. Results for multiple instance learning algorithms

	Algorithm	Accuracy	Sensitivity	Specificity
Methods Based on supervised learning (Simple)	PART	0.7357	0.8387	0.5920
	AdaBoostM1&PART	0.7262	0.8187	0.5992
Methods Based on supervised learning (Wrapper)	Bagging&PART	0.7167	0.7733	0.6361
	AdaBoostM1&PART	0.7071	0.7735	0.6136
	PART	0.7024	0.7857	0.5842
	SMO	0.6810	0.8644	0.4270
	NaiveBayes	0.6786	0.8515	0.4371
Methods Based on Distance	MIOptimalBall	0.7071	0.7218	0.6877
	CitationKNN	0.7000	0.7977	0.5631
Methods Based on Boost	DecisionStump	0.6762	0.7820	0.5277
	RepTree	0.6595	0.7127	0.5866
logistic regression	MILR	0.6952	0.8183	0.5218
Methods Based on Diverse Density	MIDD	0.6976	0.8552	0.4783
	MIEMDD	0.6762	0.8549	0.4250
	MDD	0.6571	0.7864	0.4757
Evolutionary algorithm	**G3P-MI**	**0.7429**	**0.7020**	**0.7750**

4.4 *Comprehensibility in the knowledge discovery process*

Our system has the advantage of adding comprehensibility and clarity to the knowledge discovery process. G3P-MI generates a learner based on IF-THEN prediction rules. These rules are simple, intuitive, easy to understand and provide representative information. In continuation, we show an example of the rule generated:

```
IF ( (NumberOfActivities ≥ 3) AND (TypeOfActivity EQ QUIZ_P) )  OR
   ( (NumberOfActivities IN [3-8]) AND (TimeOfActivity IN [2554. 11602]) )  OR
   ( NumberOfActivities [6-8]) )
THEN
         The student passes the course
ELSE
         The student fails the course
```

According to this rule, we can determine that passing the course requires at least three passed quizzes, or doing between three and eight activities dedicating between 2554 and

11602 seconds to solve them, or finishing from six to eight activities of any type. We can conclude that the most relevant activity is the quizzes that do not require dedicating a certain time and require completing less number of tasks. On the contrary, the rest of the activities imply handing in more tasks and spending more time to get similar results.

5 Conclusions and Future Works

This paper describes the use of G3P-MI to solve the problem of predicting a student's final performance based on his/her work in VLE from MIL perspective. To check effectiveness, the most representative paradigm of multiple instance learning is applied to solve this problem, and the results are compared. Experiments show that G3P-MI has better performance than the other techniques at an accuracy of 0.743 and achieves a trade-off between sensitivity and specificity at values of 0.702 and 0.775. Moreover it obtains representative information about the problem that is very useful to determine if all the additional material provided to the students (web-based homework) helps them to better assimilate the concepts and subjects developed in the classroom or what activities are more effective to improve the final results.

The results obtained are very interesting. However, there are still a few considerations to improve them. For example, the work only considers if a student passes a course or not. It is would be interesting to expand the problem to predict students' grades (classified in different classes) in an e-learning system. Thus, more interesting relationships could be found between the work done by the student and the precise mark obtained. Another interesting issue consists of determining how soon before the final exam a student's marks can be predicted. If we could predict a student's performance in advance, a feedback process could help to improve the learning process during the course.

References

[1] Chou, S and Liu, S. Learning Effectiveness in Web-based Technology-mediated Virtual Learning Environment. HICSS'05*:* Proceedings of the 38th Hawaii International Conference on System Sciences, Washington, USA, *2005*.

[2] Finaei-Bidgoli, B. and Punch, W. Using Genetic Algorithms for Data Mining Optimization in an Educational Web-based System. Genetic and Evolutionary Computation, 2, 2252–2263, 2003.

[3] Superby, J.F., Vandamme, J.P., Meskens, N. Determination of Factors Influencing the Achievement of the First-year University Students using Data Mining Methods. EDM'06: Workshop on Educational Data Mining, 37-44, 2006.

[4] Kotsiantis, S.B., Pintelas, P.E. Predicting Students Marks in Hellenic OpenUniversity. ICALT'05: The 5th International Conference on Advanced Learning Technologies, 664-668, 2005.

[5] Dietterich, T.G., Lathrop R. H., Lozano-Perez, T., Solving the multiple instance problem with axis-parallel rectangles, Artificial Intelligence, 89 (1-2), 31–71, 1997.

[6] Maron, O., Lozano-Pérez, T. A framework for multiple-instance learning. NIPS'97: Proceedings of Neural Information Processing System 10, Denver, Colorado, USA, 570–576, 1997.

[7] Zhang, Q., Goldman, S. EM-DD: An improved multiple-instance learning technique, in: NIPS'01: Proceedings of Neural Information Processing System 14, Vancouver, Canada, 1073–1080, 2001.

[8] Wang, J., Zucker, J.-D. Solving the multiple-instance problem: A lazy learning approach, in: ICML'00: Proceedings of the 17th International Conference on Machine Learning, Stanford, CA, USA, 1119–1126, 2000.

[9] Chevaleyre, Y.-Z., Zucker, J.-D. Solving multiple-instance and multiple-part learning problems with decision trees and decision rules. Application to the mutagenesis problem. AI'01: Proceedings of the 14th of the Canadian Society for Computational Studies of Intelligence,LNCS 2056, Ottawa, Canada, 204–214, 2001.

[10] Chai, Y.M., Yang, Z.-W. A multi-instance learning algorithm based on normalized radial basis function network. ISSN'07: Proceedings of the 4th International Symposium on Neural Networks. LNCS 4491, Nanjing, China, 1162–1172, 2007.

[11] Han Q.Y., Incorporating multiple SVMs for automatic image annotation, Pattern Recognition, 40(2), 728–741, 2007.

[12] Zhou, Z.-H., Zhang, M.-L. Solving multi-instance problems with classifier ensemble based on constructive clustering, Knowledge and Information Systems 11(2), 155–170, 2007.

[13] Zafra, A., Ventura, S., Romero C., Herrera-Viedma, E. Multi-Instance Genetic Programming for Web Index Recommendation, 2009, doi:10.1016/j.eswa.2009.03.059.

[14] Rice, W. H. Moodle e-learning course development. A complete guide to successful learning using Moodle. Pack Publishing, 2006.

[15] Witten, I.H., Frank, E. Data Mining: Practical machine learning tools and techniques, 2nd Edition. Morgan Kaufmann, San Francisco, 2005.

[16] Whigham, P. A. Grammatical bias for evolutionary learning, Ph.D. thesis, School of Computer Science, University College, University of New South Wales, Australian Defence Force Academy, Canberra, Australia, 1996.

[17] Ventura, S., Romero, C., Zafra, A., Delgado, J.A., Hervás, C.: JCLEC: A java framework for evolutionary computation soft computing. Soft Computing, 12(4), 381-392, 2008.

Visualization of Differences in Data Measuring Mathematical Skills

Lukáš Zoubek, Michal Burda
{Lukas.Zoubek, Michal.Burda}@osu.cz
Department of Information and Communication Technologies, Pedagogical Faculty, University of Ostrava, Českobratrská 16, 701 03 Ostrava, Czech Republic

Abstract. Identification of significant differences in sets of data is a common task of data mining. This paper describes a novel visualization technique that allows the user to interactively explore and analyze differences in mean values of analyzed attributes. Statistical tests of hypotheses are used to identify the significant differences and the results are then presented using Hasse diagrams. The presented technique has been tested on real data coming from pedagogical tests focused on evaluation of mathematical skills of secondary school students in Czech Republic. The results show that the proposed tool provides comprehensible representation of the data.

1 Introduction

Knowledge discovery from databases (also known as Data Mining) is a methodology for extraction of non-trivial, previously unknown, and potentially useful knowledge from data [4]. It is broadly used in a commercial sector, research and other domains.
A characteristic feature of Data Mining methods is an intensive utilization of computers for difficult computations and testing of large amount of combinations.

The objective of this paper is to present the results of application of a data mining method on data coming from educational tests of secondary school students. In the concrete, a technique for identification of statistically significant differences among mean values is described.

Such method together with the novel visualization technique described here allows the analyst to explore data and view significant differences among mean values of groups of students. The process is on-line: the attributes used to partition the data into groups are set interactively by the user. The results are immediately presented in a graphical form and the user is allowed to change settings in order to allow him or her to iteratively explore the data and find some useful knowledge.

1.1. Related work

An extensive amount of research has been done on data exploration and data mining. Let us focus on visualization techniques related to the main objective of this paper only.

Eick in [3] presents three interesting techniques, where 3D bar chart, scatterplot and a combination of para-boxes, bubble plots and box plots allow to visually analyze values of quantitative attributes.

Authors of [5] describe a visualization of hypothesis tests in multivariate linear models by representing hypothesis and error matrices of sums of squares and cross-products as ellipses, implemented for R, an open-source statistical software [10].

Two prevailing approaches to visualize association rules [1] are compared in [11]. First approach uses two-dimensional matrix to view support and confidence of the rules. Another approach is to use directed graph. The nodes of the graph represent items, and the edges represent the associations. Paper [6] experiments further with animation of the edges to depict the associations.

The co-author of this paper has discussed concept lattices and the approach that utilizes Hasse diagram with negative edges. In [2], these two techniques are compared.

2 Original data

To evaluate performance of the presented analytic tool, a database consisting of educational data has been used. The database comes from research realized at more than 90 secondary schools in the Czech Republic. All the schools are located in Moravia-Silesian region. During the original research, about 8000 students were tested in mathematics, native language (Czech), foreign language (English or German) and general study pre-requisites [7].

The secondary schools engaged in the research can be split into nine categories depending on their orientation and specialization. The categories are as follows:

- Economic *(ECO)*,
- Grammar school - gymnasium *(GRA)*,
- Lyceum *(LYC)*,
- Social and health studies *(SAH)*,
- Natural science *(NAT)*,
- Trade and service *(TAS)*,
- Social science *(SOC)*,
- Technical *(TEC)*,
- Art studies *(ART)*.

Another data attributes about the students are sex, age, and city. After cleanup, data about 7 906 students (males and females together) have been obtained. Table 1 shows distribution of students depending on the type of the school.

For the need of our actual research presented in the article, only the mathematical skills have been analyzed. During realization of the original research, each student had to answer 61 mathematical questions. The correctness of each answer has been then encoded into a binary value. The correct answer is represented by value 1, while the wrong answer is represented by value 0.

Table 1. Number of students depending on the type of school and sex

Type of school	Number of males	Number of females
ECO	212	522
GRA	807	1 279
LYC	309	491
SAH	47	589
NAT	102	143
TAS	224	713
SOC	8	101
TEC	1 965	319
ART	18	60
TOTAL	**3 692**	**4 214**

The test questions have been specially prepared in cooperation with pedagogical experts so as to cover eight important mathematical skills. They can be characterized as follows:

- Understanding of the number as a concept expressing quantity (*skill1*);
- Numerical skills (*skill2*);
- Understanding of mathematical symbols and signs (*skill3*);
- Orientation and work with table (*skill4*);
- Graphical reception and work with graph (*skill5*);
- Understanding of plane figures and work with them, spatial imagination (*skill6*);
- Function as a relation between quantities (*skill7*);
- Logical reasoning (*skill8*).

In the next step of data preparation, each of the eight mathematical skills presented above has been evaluated depending on the corresponding answers. For each student, the skills have been evaluated separately. Each of the skills has been characterized by a percentage (0-100) representing the level of the skill. The evaluation strategy has been prepared again in cooperation with pedagogical experts. So, at the end, each student has been represented by a vector of eight values corresponding to eight skills (attributes).

3 The method

On the above described data, a method for searching statistically significant differences among mean values has been applied. We have been searching for significant differences among the means (averages) of mathematical skills.

To identify significant differences, a statistical test of hypotheses could be used. For our purpose, a two sample Student's t-test for testing the equality of means has been used [9]. The test statistic is:

$$t = \frac{\overline{X} - \overline{Y}}{S}, \quad \text{where} \quad S = \sqrt{\frac{s_X^2}{m} + \frac{s_Y^2}{n}},$$

and where $\overline{X}$ and $\overline{Y}$ are the means of the two samples, s^2_X and s^2_Y are the sample variances

$$s_X^2 = \frac{1}{m-1}\sum_{i=1}^{m}(\overline{X} - x_i)^2 \quad \text{and} \quad s_Y^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\overline{Y} - y_i)^2 \; .$$

The test statistic *t* has Student’s distribution with

$$f = \frac{(s_X^2 / m + s_Y^2 / n)^2}{(s_X^2 / m)^2 / (m-1) + (s_Y^2 / n)^2 / (n-1)}$$

degrees of freedom. Thus, for sufficiently high $|t|$, say $|t| > T_f(1\text{-}0.05)$, where T_f is a cumulative distribution function of the Student’s distribution with *f* degrees of freedom, we can reject the hypothesis of equal means, that is, we can consider $\overline{X}$ and $\overline{Y}$ to be statistically significantly different.

This way we can test each combination of mean values. Consider e.g. data in the following table:

Table 2. Table shows aggregated data representing *skill1*. (Variance is a square of stdev)

	ART	ECO	GRA	LYC	NAT	SAH	SOC	TAS	TEC
average	66.02	66.5	77.24	70.37	62.32	62.66	65.83	63.03	69.59
stdev	16.52	16.55	14.16	15.96	15.59	16.5	15.83	17.1	16.52
count	78	734	2086	800	245	633	109	937	2284

By testing each pair of the mean values, we can obtain the following inequalities that represent statistically significant differences:

ART < GRA; ART < LYC; NAT < ART; SAH < ART; ART < TEC; ECO < GRA; ECO < LYC;
NAT < ECO; SAH < ECO; TAS < ECO; ECO < TAC; LYC < GRA; NAT < GRA; SAH < GRA;
SAH < GRA; SOC < GRA; TAS < GRA; TEC < GRA; NAT < LYC; SAH < LYC; SOC < LYC;
TAS < LYC; NAT < SOC; NAT < TEC; SAH < SOC; SAH < TEC; SOC < TEC; TAS < TEC.

Generally, the described technique proceeds as follows:

1. A test characteristic *c* is selected, i.e. the attribute whose average differences we would like to explore (e.g. some mathematical skill, in our case).

2. Optionally, a selection condition is defined. Selection condition determines, which data rows will be processed only (e.g. grammar schools only).

3. A partitioning attribute is selected (e.g. sex). The partitioning attribute is a categorical attribute that is used to partition the data into groups G_1, G_2, …, G_n, among which the differences of means would be analyzed.

4. A statistical testing of differences among *c*'s mean values of groups G_1, G_2, …, G_n is performed. That is, the difference of mean values among all combinations of groups G_i and G_j are tested. We have used two-sample Student's t-test with level of significance $\alpha = 0.05$.

5. As the result, a relation describing statistically significant inequalities among the groups is obtained: $G_i > G_j$ with respect to *c*.

Thus, the obtained inequalities are based on statistical testing of hypotheses. The results may be very interesting to the analyst. Unfortunately, plain textual representation of the obtained relationships seems not to be very synoptic. *Is there any way of representing them graphically?*

The obtained inequalities may be visualized using a Hasse diagram. Hasse diagram is a graph with each group G_i being represented with a vertex. A downward line is drawn from G_i to G_j, if the statistical test has indicated that the mean value computed for group G_i is significantly greater than mean value computed for G_j (i.e. $G_i > G_j$) and there is no such G_k that $G_i > G_k$ and $G_k > G_j$.

Generally, the Hasse diagram should be understood as follows: a node X is significantly greater than Y, if there exists a downward path from X to Y. The path from X to Y may lead through other nodes – however, it must be always downward. Thickness of the line represents intensity of the difference.

For instance, see Figure 2 depicting inequalities extracted from example data characterized in Table 2. From Figure 2 can be for example seen, that grammar schools (GRA) have the greatest average skill level, whereas natural science (NAT) and social and health studies (SAH) are the worst, but there is not a significant difference among them. Similarly, lyceum (LYC) and technical schools (TEC) are not different either.

Please note that accordingly to the theory of statistics, performing large amount of simultaneous statistical tests increases the test error far beyond the level of significance α [8]. Therefore, the obtained inequalities should be considered only as hypotheses indicating some interesting relationship within data – we can never treat the results as a sure and proven knowledge, if obtained that way.

4 Results

This section presents the results of the proposed tool when applied to a set of real data. The data characterizing mathematical skills of secondary school students have been analyzed from three points of view.

4.1. Male or female

The aim of the first test is to analyze difference between male and female students over the eight mathematical skills analyzed. In the first part, the type of secondary school has not been considered for the test. The results show significant differences in average values of levels for all analyzed skills. For all skills, the average values computed for male students are significantly higher. Hasse diagram characterizing this situation is shown in Figure 1. The lowest difference (2.79%) is obtained for *skill4* (males 71.88%; females 69.09%). On the contrary, the maximal difference (5.57%) between males and females is in the case of *skill6* (males 53.49%; females 47.92%).

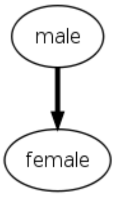


Figure 1. Hasse diagram representing the situation, when the average level computed for male students is significantly different compared to female students

The results of the detailed analysis, when the different types of secondary school have been separated, show, that the secondary schools could be sorted into three groups. Grammar schools (GRA), lyceums (LYC) and economic schools (ECO) can be characterized by the fact that the average skill level characterizing all analyzed skills is significantly higher for male students. In the case of natural science (NAT), trade and service (TAS), social and health studies (SAH), and technical schools (TEC), only for some skills is the average level computed for males significantly higher than for females. The concrete skills and types of school are summarized in the Table 3. The results for remaining schools (art studies (ART), social science (SOC)) do not show significant difference of average skill level for any skill. Unfortunately, relevancy of the data characterizing male students at social science secondary school is low because of very small number of recordings (only eight male students).

Table 3. In the case of four schools, only several skills show significant difference of average skill levels

Type of school	Significantly different skills
NAT	*skill1, skill2, skill3, skill6, skill7, skill8*
TAS	*skill1, skill2, skill4, skill6, skill8*
SAH	*skill1, skill2, skill8*
TEC	*skill3, skill5*

4.2. Difference of the skills

In the second part, individual skills have been evaluated and compared. For this analysis, male and female students are not separated into two groups. From the eight skills to be analyzed, two skills (*skill1* and *skill4*) are characterized by the highest average level. Both *skill1* and *skill4* are significantly different from the remaining six skills, while not being significantly different each other. On the other hand, the students have reached the lowest average level for the *skill5*. The mean value is again significantly different from all the other analyzed skills. Table 4 shows order of the skills depending on the average skill level. When two or more skills are not significantly different, they are presented on the same line. As it can be seen, the difference between *skill1* and *skill4*, and *skill2* is only 2%. Due to the fact, that the number of items is high (N = 7 906), this difference is evaluated by the statistic test as significantly different.

Table 4. Average skill levels computed for the skills analyzed in the research.

Skill	Average skill level
skill1, skill4	70%
skill2	68%
skill3, skill8	57%
skill7	54%
skill6	50%
skill5	42%

There are only slight differences in the order of the individual skills when the type of school or the sex is considered as an attribute. As we can expect, the values of average level vary for different types of school engaged in the research. This effect is analyzed in the next section.

4.3. Effect of the secondary school

To provide complete analysis of the data, the effect of the school type on the skills has been also evaluated using the presented tool. The average level of the grammar school (GRA) students (both male and female students) is the highest for all the analyzed skills. It is significantly different compared to the other schools. Then, it could be said, that technical schools (TEC) and lyceums (LYC) are characterized with the second highest average level for most of the skills. The values are again significantly different from the remaining schools. The order of the other types of school depends on the concrete skill and no general rule can be derived from the data. Figure 2 shows the Hasse diagram prepared from the data characterizing *skill1*. Grammar schools (GRA) are placed alone on the top of the diagram, which represents the highest average level obtained for the skill. Lyceums (LYC) and technical schools (TEC) are placed together on the same level just below the grammar schools (GRA). Absence of a path between them corresponds to the fact, that there is no significant difference between them for *skill1*.

For *skill2* and *skill7*, the average level obtained for lyceum (LYC) students is significantly different (higher) compared to the average value obtained for technical school (TEC) students.

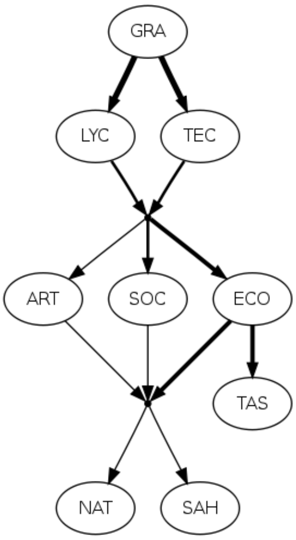


Figure 2. Hasse diagram created for *skill1* (understanding of the number as a concept expressing the quantity)

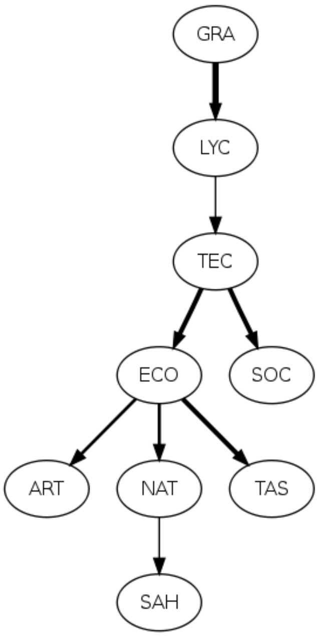


Figure 3. Hasse diagram created for *skill7* (function as relation between quantities)

Only in the case of *skill5*, the result is markedly different. Figure 4 shows the Hasse diagram obtained.

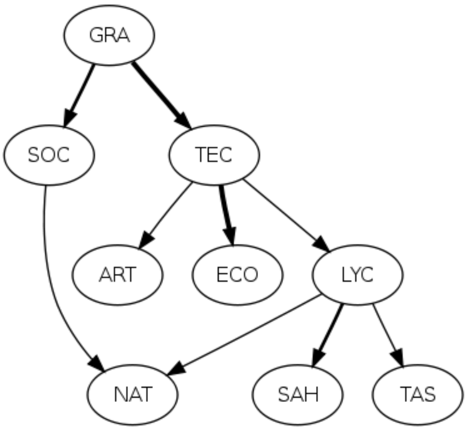


Figure 4. Hasse diagram created for *skill5* (graphical reception and work with graph)

In the next step of the analysis, we focused on evaluation of absolute differences between various types of schools. This analysis shows another two interesting facts. In the case of *skill5*, the difference between the highest average level (grammar school (GRA)) and the lowest average level is only about 8.5%. It represents the smallest difference among the analyzed skills. For grammar schools (GRA), the average skill level reached 46.5%. On the contrary, the worst average level has been obtained for art (ART) and natural science (NAT) and social and health studies (SAH) students (about 38%). This fact strongly corresponds to the results presented in the previous parts, where the average level representing the *skill5* has been determined as very poor compared to the other skills and also the Hasse diagram (Figure 4) representing order of the schools is slightly different.

The greatest difference (over 21%) has been reached for *skill3* and *skill7*. For both the skills, the maximal average level characterizes grammar schools (65% and 64% respectively) and the minimal average level reached art schools (about 43%). In the case of *skill3*, the average level reached for art school is significantly different from the values obtained for other types of school. For the other skills, the difference varies between 14% and 17%.

The variety of absolute difference between types of school is also evident from the diagrams obtained. When the absolute difference is minimal (*skill5*, Figure 4), the structure of the diagram is much wider compared to the skills characterized with maximal absolute difference (e.g. *skill7*, Figure 3). The *skill7* is represented with very narrow structure of the diagram representing the significant differences among averages of the skill levels.

5 Conclusion

We have introduced a new tool for visualization of statistically significant differences among the mean values of quantitative attributes. The method is based on statistical tests of hypotheses of equal means. Firstly, a set of tests is performed in order to determine significant differences among all combinations of tested mean values. The results are then visualized in the Hasse diagram which represents the extracted information in easily understandable format. The proposed technique has been applied on data characterizing mathematical skills of secondary school students. From the results obtained, we can pick up very poor work with graphs (*skill5*) typical for all types of secondary schools.

In the future, the authors of this paper plan to utilize Hasse diagrams to visualize other types of knowledge (e.g. impact rules).

References

[1] Agrawal, R. Fast discovery of association rules. In: Advances in knowledge discovery and data mining. AAAI Press/MIT Press, 1996, 307-328.

[2] Burda, M. Visualization of cosymmetric association rules using Hasse diagrams and concept lattices. In: Znalosti, Hradec Králové, Czech Republic, 2006, ISBN 80-248-1001-8.

[3] Eick, S.G. Visualizing multi-dimensional data. SIGGRAPH Comput. Graph. **34**(1), 2000, 61-67.

[4] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthursamy, R., eds. Advances in Knowledge Discovery and Data Mining. AAAI Press/MIT Press, 1996.

[5] Fox, J., Friendly, M., Monette, G. Visualizing hypothesis tests in multivariate linear models: the heplots package for R. In: Directions in Statistical Computing, Springer-Verlag, 2008.

[6] Hetzler, B., Harris, W., Havre, S. Visualizing the Full Spectrum of Document Relationships. 1998. [online] http://citeseer.ist.psu.edu/ hetzler98visualizing.html

[7] Kubincová, L., Malčík, M. Trstiny of skills of the 1st year secondary schools pupils. In: Information and Communication Technology in Education, Rožnov pod Radhoštěm, Czech Republic, 2008.

[8] Miller, R.G. Simultaneous statistical inference, 2nd edition. Springer, 1981. ISBN 978-0387905488.

[9] NIST/SEMATECH. E-handbook of statistical methods. [online] http://www.itl.nist.gov/div898/handbook/index.htm.

[10] R Development Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2008. [online] http://www.r-project.org.

[11] Wong, P.C., Whitney, P., Thomas, J. Visualizing Association Rules for Text Mining. In: INFOVIS, 1999, 120-123.

Author Index

Keyword Index