Inf Syst Front (2015) 17:1195–1208
DOI 10.1007/s10796-015-9564-3

An intelligent approach to data extraction and task identification for process mining

Jiexun Li[1] · **Harry Jiannan Wang**[2] · **Xue Bai**[3]

Published online: 21 June 2015

Abstract Business process mining has received increasing attention in recent years due to its ability to provide process insights by analyzing event logs generated by various enterprise information systems. A key challenge in business process mining projects is extracting process related data from massive event log databases, which requires rich domain knowledge and advanced database skills and could be very labor-intensive and overwhelming. In this paper, we propose an intelligent approach to data extraction and task identification by leveraging relevant process documents. In particular, we analyze those process documents using text mining techniques and use the results to identify the most relevant database tables for process mining. The novelty of our approach is to formalize data extraction and task identification as a problem of extracting attributes as process components, and relations among process components, using sequence kernel techniques. Our approach can reduce the effort and increase the accuracy of data extraction and task identification for process mining. A business expense imbursement case is used to illustrate our approach.

✉ Harry Jiannan Wang
hjwang@udel.edu

Jiexun Li
Jiexun.Li@oregonstate.edu

Xue Bai
xue.bai@uconn.edu

[1] Department of Business Information Systems, College of Business, Oregon State University, Corvallis, OR 97331, USA

[2] Department of Accounting and Management Information Systems, Lerner College of Business and Economics, University of Delaware, Newark, DE 19716, USA

[3] Department of Operations and Information Management, School of Business, University of Connecticut, Storrs, CT 06269, USA

Keywords Business process management · Computational experiments · Data extraction · Process mining · Task identification · Text mining

1 Introduction

As a key technology of achieving business process intelligence, process mining has received tremendous attention from both industry and academia in recent years, which is indicated by the release of numerous commercial process mining tools and systems and also by newly emerged academic communities such as the IEEE task force on process mining (van der Aalst 2012). Process mining techniques allow for extracting information from event logs, such as audit trails from a workflow system or transaction logs from an ERP (Enterprise Resource Planning) system to discover models describing processes, organizations, and performance. For example, Mans et al. (2009) conducted a case study in a Dutch Hospital, which demonstrated how process mining can be used to provide new insights that facilitate the improvement of workflow in healthcare process.

Process mining techniques can be roughly classified into three categories, i.e., process discovery, process conformance, and process performance analysis (van der Aalst and Weijters 2004). Traditional process analyses are carried out using established methods such as "walk-throughs", interviews, and workshops, which are extremely labor-intensive and time-consuming. Process mining techniques enable the automatic discovery of process models from the data history in various IT systems. Besides discovering process models, useful information on the organizational aspect of business processes can also be mined out, such as the social network structure of participators in the processes, interaction patterns, and network of work transfers. When there is a pre-defined process model, process mining techniques can automatically

check the conformance between the actual events and the existing model. For example, process mining techniques can tell the paths that were not followed and pinpoint the exceptional process instances. Root causes of those identified deviations can be traced down. Process mining can also provide insights on process performance by conducting process simulations. Typical process performance analyses include cycle time analysis, bottleneck analysis, process cost analysis, and etc., which enables process redesign and continuous process improvement.

The input for process mining includes the following important information (van der Aalst and Weijters 2004):

- Process ID: it is necessary to distinguish one process instance from another in business processes. Process ID is domain-specific. In a hospital, the process ID would be a patient ID. In a bank, this would be a customer's account number.
- Task: a well-defined step or status change that is performed in the business process. Typically, a task is represented by a noun phase for status changes, e.g., "registered", "at specialist", "in progress", "completed", or a verb-noun phrase for some actions, such as "submit form" and "approve application".
- Timestamp: a timestamp is needed for each task to bring the events in the right order.
- Originator: the organization resources (Russell et al. 2005) responsible for executing the tasks, such as registrar, administration coordinator, email system, authentication system, etc.
- Data items: the data related to tasks, such as forms, documents, specific data values, etc.

During process mining projects, a significant amount of time is normally spent on extracting the above information from large row-based databases support various enterprise information systems, which is a daunting task (Rodríguez et al. 2012). This is often due to the following reasons based on our experience from several large-scale process-mining projects:

a) *The lack of domain knowledge of the process*. There could be hundreds of processes in large enterprises. The process analysts may not know the target process very well. They often need to read a massive amount of relevant documentations and talk to subject matter experts to learn more about the processes and key tasks in the processes.
b) *The required data exist in different databases and tables*. The required data may not be easily identified by analyzing the database schema alone. For example, many tasks are recorded as database state changes in database columns with very domain-specific names. Extensive domain knowledge is required to identify relevant columns and tables for process mining.
c) *Advanced database skills are needed*. The event log databases could be very large and complex. Data dictionary documentation for a database may comprise more than a thousand pages. To analyze the schema and develop queries to output data in the format aforementioned, advanced database skills are needed. For example, in one of our projects, we developed hundreds of SQL queries to pull data from a large database.

Some important data relevant to the process may not be extracted due to a) and b), resulting in an incomplete dataset problem, while information overload problem may occur due to c) and d). In this paper, we propose a solution to mitigate both incomplete dataset and information overload problems in process mining data extraction by leveraging unstructured process relevant documents and text mining techniques. In particular, we use text mining techniques to parse unstructured process-related documents, such as process maps, process data forms and process policy manuals, to generate a lexicon that contains information about multiple aspects of the targeted process. Such a lexicon is an inventory of words and phrases that describe resources (e.g., manager, CEO, etc.), data items (e.g., portfolio, request, etc.) and process actions (e.g., submit, approve, etc.). Based on this lexicon, we define a process task as a relation in form of<Resource>−<Action>−<DataItem>(e.g., <traveler>−<submit>−<request for reimbursement>) and develop a technique to automatically extract these tasks from documents. Such a collection of tasks can help process analysts get the big picture of the process context. Based on the process component and task extraction results, we rank each attribute in tables in the database according to its relevance to the lexicon, leading to a process-aware database. The ranked attributes and identified task relations in a database environment can be used to help process analysts quickly identify the tables that are most relevant to the targeted process and provides them a much smaller set of tables to focus on. Therefore, our intelligent approach can greatly improve the efficiency and effectiveness of data extraction for process mining.

The remaining of this paper proceeds as follows. We review relevant literature next and introduce our intelligent data extraction framework in Section 3. Section 4 describes our experiments of testing the task identification module. In Section 5, we illustrate the module for database ranking and parsing using a case study. We highlight our contributions and propose some future research directions in Section 6.

2 Literature review

Although business policies have been used in both traditional and analytical process mapping approaches aforementioned, no existing method has (to the best of our knowledge) focused

on deriving process tasks from unstructured business policies using text mining techniques, which is one of our original contributions in this paper. In particular, our study is aimed at utilizing machine learning and natural language processing (NLP) techniques to develop automated tools that can extract process-related information from business documents. Below we briefly review related work on machine learning and NLP for our goals.

Process mining has been extensively discussed in the literature and many tools and techniques have been developed (van der Aalst and Weijters 2004). Process mining aims at the automatic discovering of process, control, data, organizational, and social constructs based on the event logs produced by contemporary transactional information systems, such as ERP, CRM (Customer Relationship Management), and Workflow Management Systems. A comprehensive case is presented in Ingvaldsen (2011) to show how process mining has been applied to real-world ERP transaction data. Research on Business Process Intelligence (BPI) has gained increasing attention, where classical data mining techniques are applied to the event logs to discover knowledge on various performance indicators, such as flow time, resource utilization, and cost (Grigori et al. 2004). Many methods have been developed for process discovery, such as alpha algorithms, heuristic mining algorithm, genetic algorithms, and fuzzy mining algorithms (van der Aalst et al. 2007; Günther and Aalst 2007). Unlike process mining and BPI, our approach leverages unstructured business policy documents rather than structured system event logs as the inputs. Policy documents contain rich information about business processes, including process-related lexicon and relationships. By mining such process-related information from policy documents, our study aims to provide analysts with better guidelines for extracting relevant data from databases for further process mining. Process mapping is a set of methodologies and tools that help organizations identify, understand, and improve their business processes (Hunt 1996). Participative process mapping approaches use interviews, meetings, and workshops as the major instruments to collect process information (Cobb 2004; Madison 2005). Different from a participative approach, an analytical process mapping approach aims to derive process models by using formal theory and techniques, such as linear programming (Aldowaisan and Gaafar 1999), process cost optimization (van der Aalst 2000), computational experiments (Hofacker and Vetschera 2001), and data dependencies analysis (Reijers et al. 2003). One main goal of process mining is automatic process discovery from event logs, which is another approach to analytical process mapping. Although business policies have been used in both the traditional and analytical process mapping approaches mentioned above, no existing method has focused on deriving process tasks from unstructured business policies using text mining techniques, which is a novel approach developed in this paper.

Machine learning, the main vehicle for finding patterns in data, can be categorized into feature methods and kernel methods. For feature methods, each data instance must be represented as a vector of n explicitly defined features to capture the data characteristics, $X=(x_1, x_2, \ldots, x_n)$. Text mining tasks often use words or phrases as features, which can lead to high-dimensionality but sparse feature vectors. Furthermore, for sentences represented in complex structures such as a parse trees, features cannot be easily defined to capture the structural information. Kernel methods are an effective alternative to feature methods for machine learning (Cristianini and Shawe-Taylor 2000). They retain the original representation of objects and use the objects only via computing a kernel function between a pair of objects. Formally, a kernel function is a mapping K: $X\times X\rightarrow[0, \infty)$ from input space X to a similarity score $K(x,y)=\phi(x)\cdot\phi(y)=\sum_i \phi_i(x)\phi_i(y)$, where $\phi_i(x)$ is a function that maps X to a higher dimensional space with no need to know its explicit representation. Such a kernel function makes it possible to compute the similarity between objects without enumerating all the features. Given a kernel matrix of pair-wise similarity values, a kernel machine, such as a support vector machine (SVM), can train a model for making future predictions.

In Natural Language Processing (NLP), various kernels have been applied to information extraction. For simple data representations (e.g., "bag-of-word") in which features can be easily extracted, some basic kernel functions such as a linear kernel, a polynomial kernel, and a Gaussian kernel are often used. For data in structured representations, convolution kernels are frequently used (Dietterich et al. 2002). Convolution kernels are a family of kernel functions, including string/sequence kernels (Lodhi et al. 2002), tree kernels (Zelenko et al. 2003), and so on. They define the similarity between objects as the convolution of "sub-kernels," i.e., the kernels for the decomposition of the objects. String kernels capture the sequence patterns in data instances. Lodhi et al. (2002) defined string kernels on letter or word sequences in sentences for text classification. Tree kernels capture the structure of a syntactic parse tree and have been applied in relation extraction (Zelenko et al. 2003). Some recent studies have revised these tree kernels by incorporating richer semantic information (Bunescu and Mooney 2005; Culotta and Sorensen 2004). Another advantage of kernel methods is that they transform different data representations into kernel matrices of the same format, which enables the integration of heterogeneous information (Cristianini and Shawe-Taylor 2000). Such kernel-based learning methods can also be used to analyze text from business policy documents and extract process-related components and tasks.

3 Framework for intelligent data extraction for process mining

Figure 1 shows the framework of our intelligent data extraction approach for process mining, which is discussed in detail in this section.

3.1 Process lexicon extraction

In order to find all process-mining related attributes from a business database, our proposed approach requires building a process lexicon. We create such a dictionary from two resources: (1) a company's own business process documents, and (2) industry standard process references. Each company establishes and archives various types of documents that describe and record their business processes. Such documents include process maps, business policies, forms, transactional orders, receipts, applications forms, and so on. Content carried in these documents are often in unstructured or semi-structured text format. Given the great amount of embedded information, these documents provide a rich collection for extracting lexicons related to business processes. Li et al. (2010) introduced a text mining approach for process component identification from business policy documents. Three process components, i.e., actions, data items, and organizational resources, are the target of identification. These lexicons are domain-specific and process-specific. Rule-based approaches require a lot of domain expertise and manual efforts. Alternatively, machine learning can be used to automatically learn linguistic patterns from an annotated corpus and identify process components from text documents. We can model this as a sequence labeling problem and use the Conditional Random Field (CRF) approach, based on a discriminative probabilistic model (Lafferty et al. 2001), to identify word subsequences that represent process-related phrases as lexicons.

In one of our previous studies (Li et al. 2010), we used unigrams of words and part-of-speech (POS) tags as features for CRF learning and showed good performance for lexicon extraction (~90 % accuracy). Another resource for creating this process lexicon is based on industry standard process references. There are many business process references available as standards or best practices in the different industries (Wang and Harris 2010). In particular, we have built business process lexicons by manually extract terms and phrases from over 200 process maps from the Oracle process reference model. From each process map, we extract the following: (1) organizational resources: e.g., business planner, finance VP, enterprise managers, etc., (2) actions: e.g., define, initiate, adjust, execute, etc., and (3) data items: e.g., goals, portfolio, budgets, etc. In total, we have extracted 190 resources, 192 actions and 1017 data items from these Oracle models. The three types of process lexicons extracted from the two sources (i.e., client-specific process documents and industry-standard reference models) will be combined and used for the process database parsing and ranking in the next step, described below.

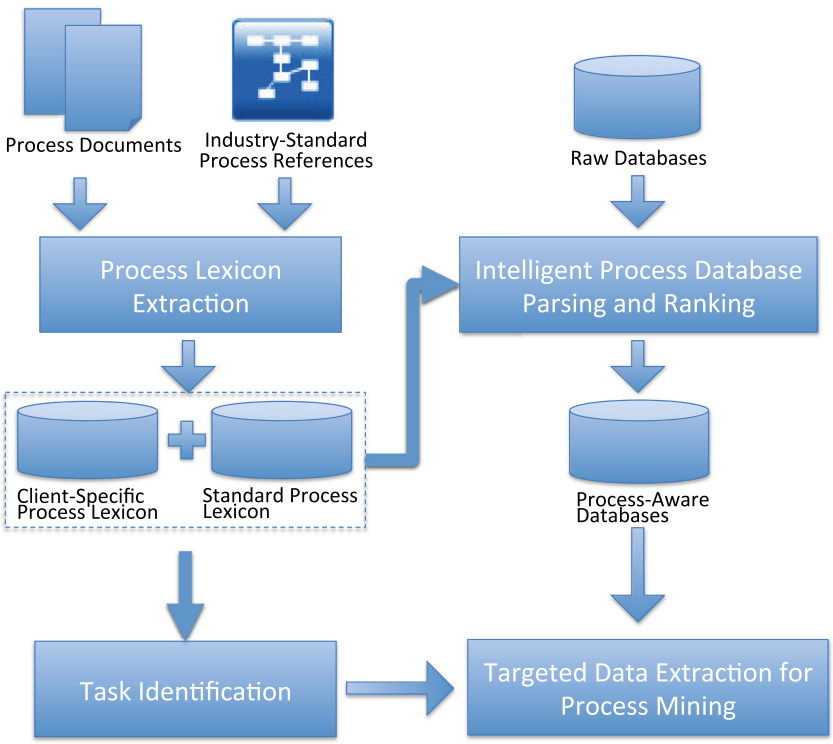


Fig. 1 An intelligent approach to extracting data for process mining

3.2 Intelligent process database parsing and ranking

The objective of this step is to identify all attributes that are relevant to the activities in the relevant business processes. For a small database with limited numbers of tables and attributes, this may be simple and straightforward. However, things can get extremely difficult for a large organization with a huge database of hundreds of tables and thousands of attributes. In order to identify the relevant attributes in a process database for further mining, we propose an intelligent parsing and ranking mechanism.

Given a relational database D with m tables $\{T_1, T_2, \ldots, T_m\}$, each table T_n has a number of attributes, T_n $(A_1, \ldots, A_{n<l>})$, where n is within $\{1,\ldots,m\}$ and $n<l>$is the number of attributes in table T_n. First, we focus on identifying attributes describing activities in business process by filtering attribute data types. The activities must be defined or described in one or multiple attributes in textual format. Therefore, we will use a criteria "data type=text" to filter out all attributes defined in a string type. For instance, in Oracle, commonly used string types include CHAR and VARCHAR2. Then, we score and rank attributes. For a string attribute A_i, the distribution of values is denoted as $\{(v_j, f_j)\}$, where v_j is a value in the column and f_j is the frequency of v_j's occurrence. Since A_i is string type, each of its value v_j is a word sequence $[t_1, t_2, \ldots, t_{m<j>}]$. For each string attribute, we calculate a score to measure its relevance to process by comparing against a process-type specific lexicon $L\{u_1, u_2, \ldots, u_n\}$, which is generated in previous step. Each u_k is a word sequence $[w_1, w_2, \ldots, w_{n<k>}]$.

Next, we need to define a function $sim(v_j, u_k)$ to measure the similarity between a string value v_j of an attribute and u_k in the lexicon. Such a similarity function can be defined in different manners. Each phrase is a word sequence $S=[w_1 \ldots w_n]$, such a sequence can then be further decomposed into n-grams. In the case of unigrams, a cosine similarity function can be used to compare phrases v_j and u_k. Bigrams and trigrams can be included in the similarity functions taking into account the word subsequences in the phrases (Lodhi et al. 2002). That could give a more accurate comparison between the two phrases but would come at a higher computational cost. It is worth noting that stemming may be necessary before computing similarity. Comparative experiments will be necessary to test the effectiveness and efficiency of these feature choices.

For each attribute A_i, we consider each value v_j's frequency as its weight and calculate the weighted average of values for all value $\{v_j\}$ and activity $\{u_k\}$ pairs. This overall similarity score indicates the relevance of attribute A_i to process activities:

$$Score(A_i) = \frac{\sum_j \sum_k f_j sim(A_i.v_j, u_k)}{\sum_j f_j}$$

Now, all string attributes are ranked by the score $Score(A_i)$ in descending order, which can help process analysts quickly identify a set of attributes and tables for targeted process mining data extraction.

To compute the score for each attribute requires an exhaustive traversing through the entire process lexicon. If the number of the words in the lexicon is M, and the number of the words in the database is N, the worse-case scenario for building a ranking order from scratch is $O(M \times N)$. If the size of the database is large, it could be time consuming. However, in reality, the building of the rankings is an ever-going and evolving process, meaning, updates of the ranking is always incremental and based on the last update. Therefore, the search and mapping process only takes place for new record entries since the last update, and ranking updated from the latest ranking orders. Therefore, the computational complexity for an update of a ranking order is $O(M \times \Delta N)$, which is a lot smaller.

3.3 Task identification

As we discussed in Section 1, process analysts may not be familiar with the target processes. The process lexicon discovered from process documents in step 1 can be further analyzed to discover a list of tasks in the processes. However, extracting correct relations among the process components is a difficult problem due to the following reasons:

- Task definition from a text mining point of view. A task or activity in a business process has a very simple definition in the business process management literature. Workflow Management Coalition defines an activity (task) as "a description of a piece of work that forms one logical step within a process" (WFMC 1999). However, in order to automatically identify a task from policy documents, we need a new task definition from a text mining perspective. As we explain in the next section, based on the study of process component identification, we define tasks as relations between different process components to provide a new way of defining tasks in business processes for automatic task extraction.
- *Formulating task identification as a relation extraction problem*. Relational data hidden in other types of text, such as news documents or biomedical literature, are often binary in nature, in the sense that they typically involve a pair of entities, e.g., a person and an organization in the former case, or a pair of proteins in the latter. For task identification in the context of process mining, the relations can be binary or ternary. For example, in the policy statement "the traveler must submit a request for reimbursement to the department within 30 days upon completing the travel", the task "traveler submits request for reimbursement" is a relation among three entities, i.e.,

resource (traveler), action (submit), and data (request for reimbursement).

– *Developing appropriate relation extraction techniques*. Due to the existence of ternary relations in the task identification problem, existing relation extraction techniques may not be directly applicable. In particular, context information for each entity, i.e., process component, must been considered. For instance, in the previously mentioned policy example, there is another resource "department", but "department submits request for reimbursement" is apparently not a valid task for this statement. Furthermore, given that each sentence may contain multiple resources, data items, and actions forming exponential number of resource-action-data combinations as candidate tasks, the question of how to identify valid and meaningful tasks presents a great computational challenge.

Given the fact that task identification requires text mining to discover linguistic patterns for tasks, before we formally model the task relations, we conduct empirical studies using five sets of tagged policies on different universities' travel policies to explore the linguistic patterns for tasks. We find that tasks are always associated with an action and some data items, such as "submits (action) form (data)" and "issue (action) check (data)". We also notice that some tasks explicitly contain resources, such as "department head (resource) approves travel", and "executive director (resource) reviews travel exceptions". Based on the case study of the policies, we define a task as a triple consisting of a resource, an action, and a data item, in the form of "*R——A——D*." A task must contain an action *A*. However, either *R* or *D* (not both) may not be explicitly expressed. Hence, both "*NULL——A——D*" and "*R——A——NULL*" are still considered valid tasks. It is worth mentioning that business policies may contain negative statements (e.g., "Do not submit the request after the due date"). In this study, our main goal is to extract process-related lexicons and tasks. Therefore, we do not consider negations when mining policy documents. If a sentence may contain multiple *R*'s, *A*'s, or *D*'s, multiple potential task instances can be derived. For example, from a sentence "*Employee needs to submit both the travel request form and all receipts*, " we can derive "*employee——submit——travel request form*" and "*employee——submit——receipts*." However, not all of these candidate task instances are meaningful. In order to discriminate meaningful (true) tasks from the others, we can formalize this as an intra-sentence relation extraction problem. For a sentence *S* consisting of resources $\{R_1, \ldots, R_i\}$, actions $\{A_1, \ldots, A_j\}$, and data items $\{D_1, \ldots, D_k\}$, we aim to derive all potential task instances T_{ijk}: $<R_i——A_j——D_k>$ and identify all meaningful tasks among them using binary classification.

Although there are many different types of learning methods, we choose kernel-based learning methods for task extraction for three reasons. First, unlike feature methods, kernel methods do not require explicit elaboration of all features for learning. Second, kernel methods can capture complex structural information in data instances (i.e., sentences in our case). Third, kernel methods also allow defining a composite kernel to combine different information in different formats. Among various popular convolution kernels used in NLP, we choose to use a sequence kernel in this study. The sequence kernel function is defined on an ordered list of tokens in a sentence. Such a sequence can be further decomposed into subsequences of *n*-grams. The string kernel proposed by Lodhi et al. (2002) was first used for text classification by analyzing the subsequences of letters or words. It has been extended by incorporating additional information (e.g., part-of-speech) of tokens and applied to relation extraction from text (Bunescu et al. 2006). Specifically, the simple "bag-of-words" kernel can be regarded as a special case of a sequence kernel when only unigram words are considered. We did not use other more complex kernels such as tree kernels (Zelenko et al. 2003), because policy documents often contain sentences of unusual structures (e.g., long list of terms) which often cause errors for many syntactic tree parsers.

In order to properly extract task relations, we design a procedure for task identification, as shown in Fig. 2. This procedure shows the detailed steps for task identification and the algorithms. We use the following sentence as an example to illustrate the detailed steps of the task identification procedure: "*The traveler must submit a request for reimbursement to the department within 30 days upon completing the travel.*"

Task instance generation Given the process document tagged with resources<R>, actions<A>, and data<D>, we need to first generate all possible<R>−<A>−<D>, <R>−<A>, and<A>−<D>tuples. The words corresponding to "actions" are also stemmed to facilitate annotation. We develop a program to automatically conduct this task. Given two Rs ("traveler" and "department"), one A ("submit"), and one D ("request for reimbursement") in the sentence, five candidate

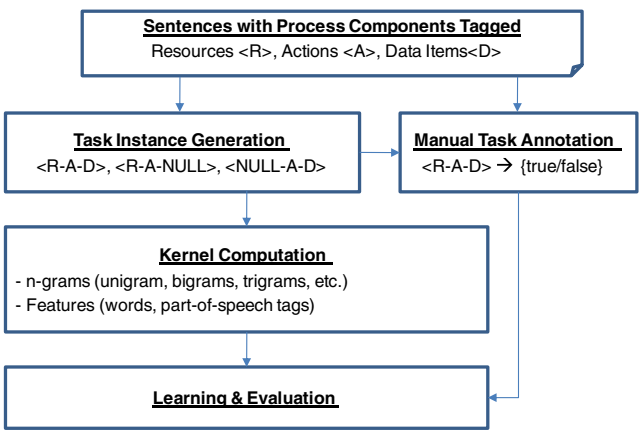


Fig. 2 Task identification procedure

task instances can be derived as follows:

$$< \mathrm{R}|\text{traveler} > - < \mathrm{A}|\text{submit} > - < \mathrm{D}\big|\text{request for reimbursement} >$$
$$< \mathrm{R}|\text{department} > - < \mathrm{A}|\text{submit} > - < \mathrm{D}\big|\text{request for reimbursement} >$$
$$< \mathrm{R}|\text{traveler} > - < \mathrm{A}|\text{submit} >$$
$$< \mathrm{R}|\text{department} > - < \mathrm{A}|\text{submit} >$$
$$< \mathrm{A}|\text{submit} > - < \mathrm{D}|\text{request for reimbursement} >$$

Manual task annotation Although machine learning does not require manual encoding of rules, we need a corpus with all valid tasks annotated by domain experts to train a statistical model. Specifically, the domain expert will go over all possible<R>−<A>−<D>, <R>−<A>, and<A>−<D>tuples and label each tuple as either 1 (valid task) or 0 (invalid task). The five task instances are annotated as follows:

$< \mathrm{R}\mid \text{traveler} > - < \mathrm{A}\mid\text{submit} > - < \mathrm{D} \mid \text{request for reimbursement} >$	1
$< \mathrm{R}\mid \text{department} > - < \mathrm{A}\mid \text{submit} > - < D \mid \text{request for reimbursement} >$	0
$< \mathrm{R}\mid \text{traveler} > - < \mathrm{A}\mid\text{submit} >$	1
$< \mathrm{R}\mid \text{department} > - < \mathrm{A}\mid\text{submit} >$	0
$< \mathrm{A}\mid \text{submit} > - < \mathrm{D}\mid\text{request for reimbursement} >$	1

Kernel computation For each potential task instance T_{ijk}, we need to determine the sequence(s) to be considered as its context in the sentence. However, the definition of a sequence for a task instance is not trivial. First of all, it would not be a good idea to use the whole sentence as the sequence for a task instance. Otherwise, task instances derived from one sentence would be represented in the same sequence and hence the learning method could not learn the discriminant patterns among them.

Based on a basic principle that sequences to represent different task instances from the same sentence should be different, we define two action-centered sequences ($S[R\ldots A]$ and $S[A\ldots D]$) for each task instance. Specifically, $S[R\ldots A]$ represents the word sequence between (and including) R and A, and $S[A\ldots D]$) represents the word sequence between (and including) A and D. It is worth noting that in a sentence R, A, and D, if any, may be in any order, but $S[R\ldots A]$ and $S[A\ldots D]$ should always follow the linear word sequence in the sentence. In this study, we do not consider the before- or after-component-pair sequences so as to simplify the computation and to reduce the similarity between task instances from the same sentences. For example, task instance T_1, $<R \mid \text{traveler} > - < A \mid \text{submit} > - < D \mid \text{request for reimbursement}>$, is represented by the following two sequences.

$S_1[R\ldots A]$: *traveler* must *submit*
$S_1[A\ldots D]$: *submitarequest for reimbursement*

Similarly, $<R \mid \text{department} > - < A \mid \text{submit} > - < D \mid \text{request for reimbursement} >$ is represented by these two:

$S_2[R\ldots A]$: *submita request for reimbursement to thedepartment*
$S_2[A\ldots D]$: *submitarequest for reimbursement*

The mining algorithm follows the kernel-based machine learning framework reviewed before. Given two task instances T_1 and T_2, we calculate the similarity between $S_1[R\ldots A]$ and $S_2[R\ldots A]$ and the similarity between $S_1[A\ldots D]$ and $S_2[A\ldots D]$ using a sequence kernel function K_S, respectively. We implement the sequence kernel function by following the string kernel proposed in (Lodhi et al. 2002). A subsequence is a finite sequence of tokens. For sequence s, we denote by $|s|$ the length of the sequence. $s=s_1\ldots s_{|s|}$. The sequence $s[i:j]$ is the subsequence $s_i\ldots s_j$ of s. We say that u is a subsequence of s, if there exist indices $i=(i_1,\ldots,i_{|u|})$, with $1\le i_1 <\ldots<i_{|u|}\le|s|$, such that $u_j = S_{i_j}$ (the i_jth token in s), for $j=1,\ldots,|u|$, or $u=s[\mathbf{i}]$ for short. The length $l(i)$ of the subsequence in s is $i_{|u|}$-i_1+1. Thus, a sequence s can be mapped into a high-dimensional feature space, in which each dimension $\phi_u(s)$ corresponds to a subsequence u. We define

$$\phi_u(s) = \sum_{\mathbf{i}:u=s[\mathbf{i}]} \lambda^{l(\mathbf{i})}$$

where $0<\lambda\le 1$. These features measure the number of occurrences of subsequences in the s weighting them according to their lengths. λ is the decay factor to penalize subsequences with more interior gaps and therefore longer length. Hence, the inner product of the feature vectors for two sequences s and t gives a sum over all common subsequences weighted according to their frequency of occurrence and length.

$$K_S(s,t) = \sum_u \phi_u(s)\cdot\phi_u(t) = \sum_u \sum_{\mathbf{i}:u=s[\mathbf{i}]} \sum_{\mathbf{j}:u=t[\mathbf{j}]} \lambda^{l(\mathbf{i})+l(\mathbf{j})}$$

By introducing some additional functions, we can compute the kernel function in a recursive and efficient manner. Besides, we may also consider tokens' different attributes such as word and part-of-speech (POS) tag when calculating the sequence similarity. Last, we can combine the two sequence kernels into a composite kernel for two task instances:

$$K(T_1, T_2) = K_s(S_1[R\ldots A], S_2[R\ldots A]) + K_s(S_1[A\ldots D], S_2[A\ldots D]).$$

Learning and evaluation With this composite kernel K, we can transform all data points (i.e., candidate task instances) to a kernel matrix, in which each element is the value of the function K for a pair of task instances. Such a kernel matrix, along with the annotated class labels (1's and 0's), can be supplied as input into a support vector machine (SVM) for training a classification model. Next, we need to evaluate its performance against a test set.

3.4 Targeted data extraction for process mining

The ranked attributes and the list of tasks identified provide a solution to the information overload problem aforementioned. The list of tasks provide the process context information to help process analysts identify process mining related information, which is similar to the common object list approach used in structural modeling phase of object oriented system analysis and design to help identify objects (Dennis et al. 2004). Process analysts can now target some top ranked attributes to start data analysis and extraction. This greatly reduces the amount of effort spent on studying database documents and discussing with domain experts in order to narrow down to the most relevant tables for process mining. In addition, the scores of all attributes provide process analysts a holistic view of the process-aware database in terms of attributes' relevance to the process of interests, which greatly reduces the risk of incomplete data problems.

4 Validation

In this section, we aim to validate our proposed framework by conducting some computational experiments for the task identification and presenting an illustrative case study based on a real business expense imbursement system.

4.1 Test-bed for task identification

We use a set of travel policy documents from a large public university in the US for task identification. These policy documents were publicly available as HTML webpages. We downloaded the documents and segmented them into individual sentences. Some sentences including tables and long lists of objects were regarded as noise and thus removed. In total, our test-bed contains 202 sentences from the travel policy. Our proposed approach requires training a classification model by learning from an annotated dataset. To create such a training corpus, we developed a system with a graphic user interface (GUI) that can display sentences and support annotation of process components in text. One domain expert in business process modeling used this system and manually annotated all process components from the 202 sentences. In total, 530 process components (i.e., 180 resources<R>, 75 actions<A>, 275 data items<D>) appear in these sentences. Since each task is in the form of<R>−<A>−<D>, <R>−<A>, or<A>−<D>, we found 62 sentences containing at least one A and at least one R or one D. Furthermore, 378 candidate task instances were constructed from these 62 sentences. Among them, 193 instances (51.06 %) were identified by the domain expert as meaningful tasks and labeled as 1, whereas the other 185 (48.94 %) were labeled as 0.

4.2 Experimental design

In our computational experiments, we compare sequence kernels under four different settings. The four kernels vary in two aspects: the sequence length (n-grams) and the attributes considered for each token, as summarized in Table 1. In particular, the first kernel K_I is simply a "bag-of-words" kernel in which only unigram words are used in kernel computation. The fourth kernel K_{IV} considers both unigrams and bigrams of word and POS features in the sequence kernel. A popular POS tagging tool, StanfordPOSTagger (http://nlp.stanford.edu/software/tagger.shtml), is used to automatically tag sentences in the test-bed. We choose a widely used SVM package, LibSVM, for kernel learning (www.csie.ntu.edu.tw/~cjlin/libsvm). In our experiments, we conduct a cross-validation to estimate the performances of task identification. Cross-validation is a standard evaluation methodology for classification in machine learning research (Kohavi 1995). Specifically, we perform a leave-one-out cross-validation at the sentence level. Each fold contains task instances derived from one single sentence. For each round, one fold is left out as the testing set to validate the model trained on the instances from the remaining folds. The testing results for all folds are then averaged to get the overall estimate of the classification performance. Standard evaluation metrics for information extraction, i.e., accuracy, precision, recall, and F-measure, are used in our evaluation. Specifically, accuracy evaluates the overall correctness. Precision, recall, and F-measure evaluate the correctness for each class. Precision indicates the correctness of identified tasks and recall indicates the completeness

Table 1 Four sequence kernels for comparison

Kernels	n-grams	Attributes
K_I	unigrams	Word
K_{II}	unigrams	Word+POS
K_{III}	unigrams+bigrams	Word
K_{IV}	unigrams+bigrams	Word+POS

of identified tasks. The F-measure is the harmonic mean of precision and recall. Accuracy, precision, recall and F-measure are formally defined as follows:

$$\text{Accuracy} = (\#\text{ of all correctly classified instances}) \,/\, (\text{total \# of instances})$$
$$\text{Precision} = (\#\text{ of correctly identified tasks}) \,/\, (\text{total \# of instances identified as tasks})$$
$$\text{Recall} = (\#\text{of correctly identified tasks}) \,/\, (\text{total \# of tasks})$$
$$\text{F}-\text{measure} = 2 \times \text{Precision} \times \text{Recall} \,/\, (\text{Precision} + \text{Recall})$$

5 Results and discussion

Figure 3 shows the performances of the four kernels for task extraction. In our data set, 51.06 % out of 378 candidate instances are positive instances, i.e., real tasks. A simplistic baseline can be established as a classifier that classified all instances as positive, which gives: accuracy=51.06 %, precision=51.06 %, recall=100.00 % and F-measure=67.60 %. As compared to this baseline, all four kernel methods achieve over 75 % classification accuracy and over 70 % precision, still at high recall (close to 95 %). This means that our methods only missed very few true tasks (~5 %). Among all instances identified as positive by our methods, users only need to review and screen out a few (less than 30 %) false positives. As an automated tool for extracting process-related information from policy documents, our approach with such a performance can offer significant assistance for task identification and reduce cognitive load of process analysts.

Surprisingly, the simple word kernel K_I outperforms the others in terms of accuracy (79.63 %), precision (74.37 %), and F-measure (82.13 %), except that the kernel K_{III} using unigrams and bigrams of words achieves the highest recall (96.37 %). When taking into account bigrams, sequence kernels, K_{III} and K_{IV}, tend to enlarge the similarity scores between task instances and therefore be less conservative in predicting instances as true tasks, as indicated in the lower precisions and higher recalls. Furthermore, POS tags of words in sequences do not seem to contribute much to task identification in our experiments. However, we should not rule out the possibility that our cross-validation evaluation is conducted on a small corpus with only one policy document. Given the limited vocabulary in the test-bed, word similarity may play a dominant role in kernel-based learning. We expect that, when our approach is used to identify tasks from new documents or new domains, word similarity will be less significant whereas syntactic features such as POS tags will become an important complement for kernel-based learning. A larger test-bed and further experiments will be needed to validate this hypothesis in our future research.

5.1 An illustrative case study

To further validate our proposed framework, we illustrate our approach by using a database related to business expense imbursement. The database schema is simplified based on a real business expense imbursement system of a large public university in the US, which is shown in Fig. 4. The database contents are retrieved from the real system. The sample data for this case study is included in Appendix A.

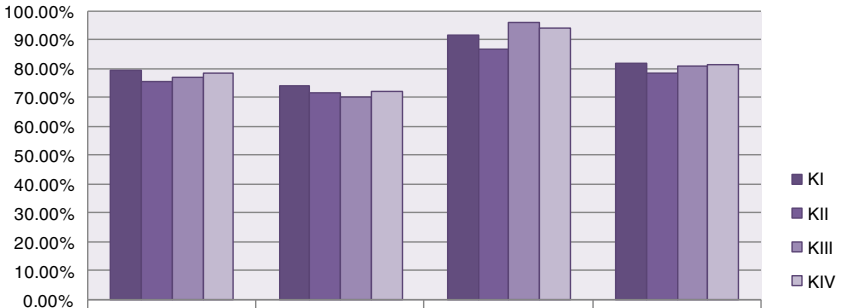
In this simplified database, we focus on four tables related to business expense reimbursement:

- Employee (EID, Name, Department)
- ExpenseForm (FID, AccontNumber, BankAccount)
- Routing (RID, Role, Time, Comments, EID, FID)

  – Foreign key EID references Employee(EID)
  – Foreign key FID references ExpenseForm(FID)

- ExpenseItem(ItemID, Type, Date, Description, Amount, RID)

  – Foreign key RID references Routing(RID)



	Accuracy	Precision	Recall	F-measure
KI	79.63%	74.37%	91.71%	82.13%
KII	75.93%	71.79%	87.05%	78.69%
KIII	77.25%	70.19%	96.37%	81.22%
KIV	78.57%	72.22%	94.30%	81.80%

Fig. 3 Results of kernel-based task extraction

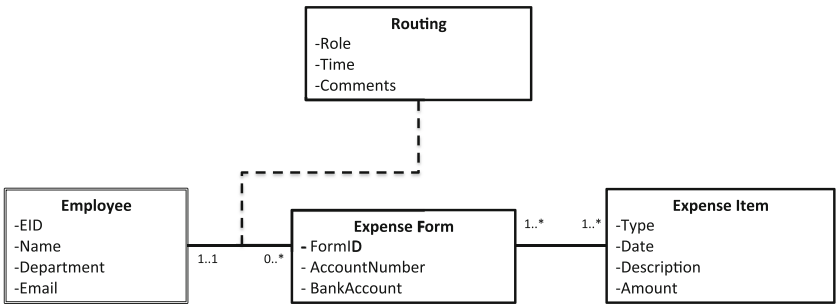


Fig. 4 ERD for the sample database

Note that we assume that we do not understand the semantics of those tables. In this study, we only rank the database attributes of string type, which include "Name", "Department" in the Employee table, "Role" and "Comments" in the Routing table, and "Type", "Description" in the ExpenseItem table. To find attributes most relevant to the expense reimbursement process, we can rank all string-typed attributes by the similarity score as compared to our lexicon extracted from policy documents and industry standards. From the Oracle process reference models, we identify and adopt one specific process model named "Manage Expenses" that is most relevant to expense reimbursement. It has 14 defined tasks as shown in Table 2. Terms/phrases in these tasks can be extracted and used as our lexicon to find attributes related to the business process.

In addition, process-related tasks identified from policy documents can be used as a reference when we review databases and try to find relevant attributes for process mining. In particular, from our experimental study described in the previous section, we identified 193 tasks from a university travel policy. Table 3 shows some sample tasks. This list can further help the process analyst review the business expense imbursement process and identify some key steps and issues for potential process improvements.

Specifically, the attribute Routing(Role) consists of values such as "Originator," "Account Admin," "Supervisor," "Approver," "Reject," "Resubmit," and so on. These terms tend to match some of the "Resources (R)" and "Actions (A)" in the lexicon from Oracle process reference models, such as "Approve", "Credit Card Administrator." In addition, by looking into the set of tasks in Table 3, we can learn that Routing(Role) attribute might represent key tasks in the process, such as "Traveler submit reasonable exceptions", and "Department head approve exceptions." ExpenseItem (Type) consists of values such as "Car Rental," "Registration," "Meal," and so on. These values tend to match our lexicon of "Data Items (D)." Besides, attribute Routing(Comments) contains terms such as "exception," "request," "approve", "exception," "P-card" and so on. These terms match those that occur in tasks identified from business policy documents. Therefore, these attributes will also be identified as process-relevant attributes. These attributes are especially useful in predictive modeling in process mining (Van der Aalst et al. 2011). Since the Routing table includes this attribute "Role" and several attributes necessary for process-mining algorithms, such as case id (FID) and time stamp (Time), this table

Table 2 Lexicon related to "manage expenses" from oracle process reference models

	Resources	Actions	Data items
1	Expense Manager	Maintain	Expense Policies
2	Expense Manager	Maintain	Rules
3	Expense Manager	Process	Expense Reimbursements
4	Expense Manager	Analyze	Expenses
5	Employee	Maintain	Employee Profile
6	Employee	Request	Travel Authorization
7	Employee	Request	Cash Advance
8	Employee	Enter	Expense Report
9	Line Manager	Approve	Travel Authorization
10	Line Manager	Approve	Cash Advance
11	Line Manager	Approve	Expenses Report
12	Credit Card Administrator	Manage	Credit Card Data
13	Expense Auditor	Audit	Expense Report
14	Expense Specialist	Provide	Employee Help Desk Support

Table 3 Example of identified tasks from process policy documents

- Employee reimburse transportation expenses
- Department head designate P-Card holders
- Department head designate transportation purchase authorization
- University Payables submit travel exception request
- Traveler submit reasonable exceptions
- Department head approve exceptions
- Chancellors delegate approval authority
- Employees are informed about direct deposit
- Department head authorizes business travel
- Traveler signs the from

is considered highly relevant to the target process. It is worth noting that this case only illustrates how we can identify attributes related to business processes from a simplified database of four tables. Naturally, more experiments need to be conducted to further test our proposed approach with large enterprise databases.

6 Conclusions

We have proposed a methodology for data extraction and task identification for enterprise process mining. The development of this methodology is motivated by the need in business practice to develop automated and efficient ways to extract relevant data from unstructured business process documentations and poor process-awareness of traditional databases. The status quo approach to data extraction in current practice is manual and ad hoc. The data obtained are largely incomplete and inaccurate. This leads to poor results in the subsequent steps of process mining. To remove the burden of laborious manual work and improve the data quality for process mining, we propose an intelligent data extraction methodology. This methodology first uses text mining techniques to parse the unstructured and client-specific process documents as well as industry-standard reference processes to generate a process lexicon. Then, it develops a scoring criterion to rank all the attributes in tables. Consequently, each table is ranked based on its relevance to the lexicon. Tasks are also automatically identified from the process lexicon to help process analysts get more context information. We apply four different sequence-kernel-based learning methods to task identification and conduct experiments on real-world policy documents to evaluate their performance. An illustrative case study is also presented to further validate the proposed methodology. Our methodology enables fast identification of a set of process components and task relations in large databases, and can improve the data extraction process in terms of both speed and accuracy.

Our key contribution is threefold. Frist, our work is among the first attempts to develop a systematic and automated framework to data extraction and task identification for business process mining. Such a framework is generic and can be applied to different companies, with proper customization of a company's specific policies. Second, we develop a scoring method to rank attributes in databases. Third, we formalize tasks as relations among three different process components, i.e., resource, action, and data, and introduce a new text mining application in the context of automatic task identification for process mining.

As an initial attempt to develop and validate an automated and intelligent approach to data extraction and task identification, our study is prone to several limitations that, in turn, offer opportunities for future research. The data available through public sources are limited. Our experiments were carried out on a small corpus of travel policies. With more data becoming available in the future, there is a potential for performance improvements of our approach in terms of precision and accuracy. It is worth noting that annotation of manual tasks such as sentence-level text analysis is often a costly process and requires significant efforts from annotators. However, once we obtain a model on a reasonably well-annotated training corpus, we may not necessarily need to repeat the annotation process for every new policy document. Nevertheless, the portability of the task identification module across different policy types is still subject to further investigation. Our work can be extended in a number of directions. First, the ranking criterion for the database attributes can be further developed and validated, especially for attributes of non-string type. Second, more datasets can be used to further test the performance of different sequence-kernel-based learning methods. In particular, we plan to validate our hypothesis on why the simple word kernel method outperformed other methods in the experiment presented in this study. Finally, more extensive experiments on process policies from different business domains can be conducted to test the portability of our approach.

Acknowledgments This research was partially supported by a JPMorgan Chase Fellowship from the Institute of Financial Services Analytics at the University of Delaware.

Appendix A: Sample data for the illustrative case study

The example tables with data for the database shown in Figure 4 are included below.

Employee Table:

EID	Name	Department
E001	Joe Wang	MIS
E002	Nina Somers	MIS
E003	Nancy Warren	BS
E004	Jeff Jones	MIS
E005	Linda Proctor	BS
E006	Jennifer Brinkley	Procurement
E007	Debra Berry	Procurement

Expense Form Table:

FID	AccountNumber	BankAccount
F001	BUEC001	BOA001
F002	ACCT001	WSFS001
F003	ACCT002	WSFS001

Expense Item Table:

ItemID	Type	Date	Comments	Amount	FID
I001	Car Rental	8/10/06	Car rental from Arizona to Delaware via United Van Lines	$3,540.00	F001
I002	Hotel	7/21/07	Hotel for CSWIM2007	$350.00	F002
I003	Registration	7/21/07	Registration Fee for CSWIM2007	$150.00	F002
I004	Meal	7/22/07	Dinner	$25.00	F002
I005	Miscellaneous	5/22/07	Chair for faculty member's office	$550.00	F003

Routing Table:

RID	Role	Time	Comments	EID	FID
R001	Originator	9/8/06 10:37	New MIS faculty relocation from Arizona to Delaware per agreement letter.	E002	F001
R002	Supervisor	9/8/06 12:47		E004	F001
R003	Account Administrator	9/11/06 10:36		E003	F001
R004	Approver	9/11/06 11:10		E005	F001
R005	Reject	9/15/06 9:48	The charges for the shipment of a vehicle is not applicable for reimbursement under policy 3–11. A copy of the agreement letter is needed and any exception to policy must be signed by the dean and Provost office.	E006	F001
R006	Resubmit	10/10/06 19:11		E002	F001
R007	Supervisor	10/11/06 8:25		E004	F001
R008	Account Administrator	10/16/06 11:38		E003	F001
R009	Approver	10/16/06 12:28		E005	F001
R010	Procurement	10/16/06 16:36		E006	F001
R011	Originator	8/16/07 11:06		E001	F002
R012	Drafter	8/20/07 10:45		E002	F002
R013	Supervisor	8/20/07 13:12		E004	F002
R014	Account Administrator	8/22/07 10:49		E003	F002
R015	Approver	8/22/07 11:18		E005	F002
R016	Procurement	8/30/07 14:06		E007	F002
R017	Originator	6/15/07 11:44		E002	F003
R018	Supervisor	6/15/07 14:31		E004	F003
R019	Account Admin	6/18/07 14:52		E003	F003
R020	Approver	6/18/07 14:59		E005	F003
R021	Procurement	8/20/07 16:55	This BER is approved as an exception however no further BERs for goods will be approved. Please request a card or have someone with a Pro-card handle these transactions	E006	F003

References

Aldowaisan, T. A., & Gaafar, L. K. (1999). Business process reengineering: an approach for process mapping. *Omega, 27*(5), 515–24.

Bunescu, R., & Mooney R. (2005). A Shortest path dependency kernel for relation extraction. *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. (pp. 724–731) Vancouver, B.C, Canada: Association for Computational Linguistics. http://www.aclweb.org/anthology/H05-1091.

Bunescu, R., Mooney, R., Weiss, Y., Schölkopf, B., & Platt, J. (2006). Subsequence kernels for relation extraction. *Advances in Neural Information Processing Systems, 18*, 171–78.

Cobb, C.G. (2004). *Enterprise process mapping: Integrating systems for compliance and business excellence*. {ASQ} Quality Press.

Cristianini, N., & Shawe-Taylor J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press.

Culotta, A., and J. Sorensen. (2004). Dependency tree kernels for relation extraction. In *42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)* (pp. 423–429). Barcelona, Spain.

Dennis, A., Wixom, B.H., and Tegarden D. (2004). *Systems analysis and design with UML Version 2.0: An Object-Oriented Approach*. Wiley.

Dietterich, T.G., Becker S., Ghahramani Z., Collins M., and Duffy N. (2002). Convolution kernels for natural language. in *Advances in Neural Information Processing Systems 14*. MIT.

Grigori, D., et al. (2004). Business process intelligence. *Computers in Industry, 53*, 321–43.

Günther, C.W., & van der Aalst, W.M.P. (2007). Fuzzy mining: Adaptive process simplification based on multi-perspective metrics. In G. Alonso, P. Dadam, M. Rosemann (Eds.), *Lecture Notes in Computer Science: Vol. 4714. Proceedings of the 5th International Conference on Business Process Management* (pp. 328–343). Berlin, Heidelberg: Springer-Verlag. doi:10.1007/978-3-540-75183-0.

Hofacker, I., & Vetschera, R. (2001). Algorithmical approaches to business process design. *Computers & Operations Research, 28*(13), 1253–75.

Hunt, V. D. (1996). Process Mapping : How to Reengineer Your Business Processes. Wiley. New York

Ingvaldsen, J.E. (2011). Semantic process mining of enterprise transaction data. Norwegian University of Science and Technology.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence 2*, pp. 1137–1143. San Francisco, CA, USA: Morgan Kaufmann.

Lafferty, J., McCallum A., & Pereira F. (2001). Conditional random FIelds: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning* (pp. 282–289). San Francisco. http://portal.acm.org/citation.cfm?id=655813.

Li, J., Wang, H. J., Zhang, Z., & Leon Zhao, J. (2010). A policy-based process mining framework: mining business policy texts for discovering process models. *Journal of Information Systems and E-Business Management, 8*, 169–88.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research, 2*(3), 419–44.

Madison, D. (2005). *Process mapping, process improvement and process management*. Paton Press.

Mans, R. S., Schonenberg, M. H., Song, M., & Bakker, P. J. M. (2009). Application of process mining in healthcare – a case study in a dutch hospital. *Biomedical Engineering Systems and Technologies, 25*, 425–38.

Reijers, H. A., Limam, S., & van der Aalst, W. M. P. (2003). Product-based workflow design. *Journal of Management Information Systems, 20*(1), 229–62.

Rodríguez, C., Engel, R., Kostoska, G., Daniel, F., Casati, F., & Aimar, M. (2012). Eventifier: Extracting process execution logs from operational databases. In *Proceedings of the 10th International Conference on Business Process Management*. Tallinn, Estonia.

Russell, N., van der Aalst W.M.P., ter Hofstede, A. H. M., & Edmond, D. (2005). Workflow resource patterns: Identification, representation and tool support. In *Proceedings of the 17th International Conference on Advanced Information Systems Engineering* (pp. 216–232). Porto: Portugal.

Van der Aalst, W. (2000). Workflow verification: Finding control-flow errors using petri-net-based techniques. *Business Process Management* 19–128.

Van der Aalst, W. M. P. (2012). Process mining: overview and opportunities. *ACM Transactions on Management Information Systems (TMIS), 3*(2), 7.

Van der Aalst, W. M. P., & Weijters, A. (2004). Process mining: a research agenda. *Computers in Industry, 53*(3), 231–44.

Van der Aalst, W. M. P., et al. (2007). Business process mining: an industrial application. *Information Systems, 32*(1), 713–32.

Van der Aalst, W. M. P., Schonenberg, M. H., & Song, M. (2011). Time prediction based on process mining. *Information Systems, 36*(2), 450–75.

Wang, H. J., & Harris, W. (2010). Supporting process design for E-business via an integrated process repository. *Information Technology and Management, 12*(2), 97–109.

WFMC. (1999). Interface 1: Process definition interchange {Q&A} and Examples ({WFMC-TC-1016-X)}, Draft 7.01. *Workflow Management Coalition*.

Zelenko, D., Aone, C., & Richardella, A. (2003). Kernel methods for relation extraction. *Journal of Machine Learning Research, 3*(6), 1083–1106.

Jiexun Li is an Assistant Professor of Business Information Systems in the College of Business at Oregon State University. Prior to OSU, Li was an Assistant Professor at Drexel University and a Visiting Assistant Professor at Fordham University. Li earned his Ph.D. in Management Information Systems from the University of Arizona. Li's research interests include data mining, business analytics, social media analytics, and health informatics. His research has been published in top-tier IS journals including Journal of Management Information Systems, Decision Support Systems, IEEE Transactions, Journal of the American Society for Information Science and Technology, Journal of the Association for Information Systems, Bioinformatics, Communications of the ACM, Expert Systems with Applications, and Information Systems Frontiers.

Harry Jiannan Wang is an Associate Professor of Management Information Systems (MIS) and JPMorgan Chase Fellow in the Lerner College of Business and Economics, University of Delaware. He received Ph.D. in MIS from the Eller College of Management, University of Arizona, USA and B.S. in MIS from Tianjin University, China. His research interests involve business analytics and intelligence, business process management, and enterprise systems. He has published research articles in journals, such as Information Systems Research, Decision Support Systems, ACM Transactions on Management Information Systems, Journal of Database Management, and Information Technology and Management. Dr. Wang has serviced as associate editor, special issue guest editor, and editorial board member for several journals and organized the 2014 Workshop on business Processes and Service and the 2013 China Summer Workshop on Information Management as a conference co-char. He has also been a program co-chair, program committee member, and track co-chair for numerous conferences.

Xue Bai is Associate Professor of Management Information Systems in the School of Business at the University of Connecticut. She received her PhD degree in Management Information Systems from Carnegie Mellon University. Her research interests include data mining, business analytics, mathematical modeling, and online social networks. She is Associate Editor with Information Systems Research. She has published in top management science and information systems journals.