2008-05-09

Induction-Based Approach to Personalized Search Engines

Wadee Saleh Alhalabi
University of Miami, wadee_h@hotmail.com

UNIVERSITY OF MIAMI

INDUCTION-BASED APPROACH TO PERSONALIZED SEARCH ENGINES

By

Wadee S. Alhalabi

A DISSERTATION

Submitted to the Faculty
of the University of Miami
in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

Coral Gables, Florida

May 2008

UNIVERSITY OF MIAMI

A dissertation submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

INDUCTION-BASED APPROACH TO PERSONALIZED SEARCH ENGINES

Wadee S. Alhalabi

Approved:

____
Dr. Miroslav Kubat
Associate Professor of Electrical
& Computer Engineering

____
Dr. Terri A. Scandura
Dean of the Graduate School

____
Dr. Moiez A. Tapia
Professor of Electrical
& Computer Engineering

____
Dr. Shihab Asfour
Professor of Industrial Engineering,
& Associate Dean of Engineering

____
Dr. Abdel-Mottaleb
Associate Professor of Electrical
& Computer Engineering.

____
Dr. Weizhao Zhao
Associate Professor of
Biomedical Engineering

____
Dr. Xiaodong Cai
Assistant Professor of Electrical
& Computer Engineering

ALHALABI, WADEE S. (Ph.D., Electrical and Computer Engineering)

Induction-Based Approach to Personalized Search Engines (May 2008)

Abstract of a dissertation at the University of Miami.

Dissertation supervised by Professor Miroslav Kubat and Professor Moiez Tapia
No. of pages in text. (90)

In a document retrieval system where data is stored and compared with a specific query and then compared with other documents, we need to find the document that is most similar to the query. The most similar document will have the weight higher than other documents. When more than one document are proposed to the user, these documents have to be sorted according to their weights. Once the result is presented to the user by a recommender system, the user may check any document of interest. If there are two different documents' lists, as two proposed results presented by different recommender systems, then, there is a need to find which list is more efficient. To do so, the measuring tool "Search Engine Ranking Efficiency Evaluation Tool [SEREET]" came to existence. This tool assesses the efficiency of each documents list and assigns a numerical value to the list. The value will be closer to 100% if the ranking list efficiency is high which means more relevance documents exist in the list and documents are sorted according to their relevance to the user. The value will be closer to 0% when the ranking list efficiency is poor and all of the presented documents are uninteresting documents to

the user. A model to evaluate ranking efficiency is proposed in the dissertation, then it is proved it mathematically.

Many mechanisms of search engine have been proposed in order to assess the relevance of a web page. They have focused on keyword frequency, page usage, link analysis and various combinations of them. These methods have been tested and used to provide the user with the most interesting web pages, according to his or her preferences. The collaborative filtering is a new approach, which was developed in this dissertation to retrieve the most interesting documents to the user according to his or her interests. Building a user profile is a very important issue in finding the user interest and categorizes each user in a suitable category. This is a requirement in collaborative filtering implementation. The inference tools such as time spent in a web page, mouse movement, page scrolling, mouse clicks and other tools were investigated. Then the dissertation shows that the most efficient and sufficient tool is the time a user spent on a web page. To eliminate errors, the system introduces a low threshold and high threshold for each user. Once the time spent on a web page breaks this threshold, an error is reported.

SEREET tool is one of the contributions to the scientific society, which measures the efficiency of a search engine ranking list. Considerable work were carried, then the conclusion was that the amount of time spent on a web page is the most important factor in determining a user interest of a web page and also it is a sufficient tool which does not require collaborations from other tools such as mouse movements or a page scrolling. The results show that implicit rating is a satisfactory measure and can replace explicit rating. New filtering technique was introduced to design a fully functional recommender system. The linear vector algorithm which was introduced improves the vector space algorithm (VSA) in time complexity and efficiency. The use of machine learning enhances the retrieved list efficiency. Machine learning algorithm uses positive and negative examples for the training, these examples are mandatory to improve the error rate of the system. The result shows that the amount of these examples increases proportionally with the error rate of the system.

Acknowledgements

In writing this dissertation, I would like to express thanks to God who guided me to choose the right way toward success and who bestowed me with great parents and a fabulous family, as well as wonderful advisors and committee members.

I would like to dedicate this work to my father, who rests in peace after providing us with all the necessities of life and gracing us with his wisdom. I am sure he would be proud of this work if he was still around. I can imagine the wide, wonderful smile on the face of my mother, who is in good health. She never gave up on me and gave me the power to continue my education despite great personal challenges. My appreciation is extended to my family, wife, and children who empower me and provide me with a pleasant environment and wonderful times, and have always been satisfied and patient with what little time my studies sometimes has left for them.

I am grateful to my advisors, who enlightened my path and expanded my knowledge and to whom I owe a great debt for my success. Dr. Kubat and Dr. Tapia were the best advisors and friends I have ever had. They showed me how to identify a problem, how to conduct research, how to collect data, how to gather evidence, and how to draw conclusions. They have guided me toward professorship.

I have great respect for my committee members whose time and insights helped me write a solid dissertation. My appreciation is extended to Dr. Asfour, who I heavily relied upon. I am so grateful to Dr.Abdel-Mottaleb, who gave me the best advice and helped me to come up with a dissertation that I am proud of and also to Dr. Zhao, who is one of the best instructors I have ever met. He shared with me, as he does with all of his students, his complete and outstanding knowledge and experience. My appreciation goes to Dr.

Cai, who consented to be a committee member at a later stage in my dissertation. I am so proud to be around all of the people in the ECE department and my own department in Saudi Arabia. My appreciation is extended to all my friends who are there when I am in need.

Thank you all.

TABLE OF CONTENTS

Page

LIST OF EXAMPLES vii

LIST OF FIGURES viii

LIST OF TABLES x

Chapter

1 INTRODUCTION 1

2 REVIEW OF RELEVANT LITERATURE 9

3 SEARCH ENGINE PERSONALIZATION TOOL BASED ON THE LINEAR VECTOR ALGORITHM 22
3.1 System design 24
3.2 Experimental results 35

4 TIME SPENT ON A WEB PAGE IS SUFFICIENT TO INFER A USER'S INTEREST 45
4.1 Problem and performance criteria 45
4.2 Methodology of the study 46
4.3 Results 50

5 AN ALGORITHM TO EVALUATE THE EFFICIENCY OF A SEARCH ENGINE RANKING 54
5.1 Proposed mechanism for ranking........ 55
5.2 The Average normalized modified retrieval rank........ 67

6 A FULLY AUTOMATED RECOMMENDER SYSTEM USING A FILTERING TECHNIQUE 70
6.1 Problem and performance criteria 72
6.2 Methodology 72
6.3 Results 74
6.4 Discussion 77

7 CONCLUSION AND SUGGESTIONS FOR FUTURE WORK 80
7.1 Linear vector algorithm 81
7.2 Time spent on a web page is sufficient to infer a user's interest 82
7.3 Algorithm to Numerically Evaluate the Efficiency of a Search Engine Rank Methodology 82
7.4 A Fully Automated Recommender System Using a Filtering Technique . 83

Bibliography 85

LIST OF EXAMPLES

Chapter Page

3 Example 3.1 28

4 Example 4.1 47

4 Example 4.2 49

5 Example 5.1 55

5 Example 5.2 56

5 Example 5.3 57

5 Example 5.4 59

5 Example 5.5 60

5 Example 5.6 61

5 Example 5.7 67

LIST OF FIGURES

Chapter Page

3 Figure (3-1): SEPT System 24

3 Figure (3-2): LVA vs. vector space behavior 28

3 Figure (3-3): Downloading time per thread 34

3 Figure (3-4): Linear vector responses to the training process........ 38

3 Figure (3-5): LVA vs. VSA using the keyword “Research Fund” 39

3 Figure (3-6): LVA vs. VSA using the keyword “Jaguar” (animal) 40

3 Figure (3-7): LVA vs. VSA using the keyword “Beetle” (car) 41

3 Figure (3-8): result.htm file........ 42

3 Figure (3-9): Linear vector with minimum amount of knowledge 43

3 Figure (3-10): LVA with minimum and maximum knowledge vs. VSA 43

4 Figure (4-1): Average time spent on a Web page 51

4 Figure (4.2): Maximum time spent on a Web page 51

4 Figure (4.3): Minimum time spent on a web page........ 52

5 Figure (5-1): The retrieved web pages........ 56

5 Figure (5-2): Relevant to the query for example 5.1 56

5 Figure (5-3): Relevant to the query for example 5.2 57

5 Figure (5-4): Relevant to the query for example 5.3 57

6 Figure (6-1): Collaborative filtering 71

6 Figure (6-2): Experiment #1 75

6 Figure (6-3): Experiment #2 75

Chapter Page

6 Figure (6-4): Experiment #3 77

6 Figure (6-5): Collaborative filtering using implicit rating................................ 78

LIST OF TABLES

Chapter Page

3 Table (3-1) Word frequency 30

4 Table (4-1): Average time spent in seconds on a web page for user #1 48

4 Table (4-2): Maximum time spent on a web page for a particular user ……. 49

4 Table (4-3): Minimum time spent on different web pages for a particular user 50

6 Table (6-1): User/web page rate 73

6 Table (6-2): Collaborative filtering data 74

Chapter 1

Introduction

The Internet revolution gave rise to the search engine - the tool capable of identifying among the billions of web sites those that are relevant to the user's needs. Starting in the mid-1990s, hundreds of companies specializing in these tools have appeared. Many of them have gone out of business, others have merged, and yet others have joined this thriving market only recently, seeking either to outperform their predecessors, or to fill previously unexplored niches.

The main task for a search engine is to choose, from among the billions of web sites, those that best reflect the topic characterized by a set of keywords. Several solutions have been proposed and are now habitually exploited by existing search engines. Of particular interest are the methods of personalizing the search engine to a wide spectrum of a specific user.

In this study, we attempt to determine if a search engine’s result can be enhanced by exploring the web page’s contents. Presuppositions in this work appear to lend weight to our hypothesis that a new algorithm could be developed to help inexperienced users utilize search engines more efficiently. It is our belief that if the web page text can be obtained, then the rank given to it according to each user’s interest could be improved. Search engines are the only available search tools on the Web today. Although they are widely used, search engine results are not efficient to satisfy the user. By exploring the text content of a web page with a personalized search engine using new tools such as a linear algorithm, in addition to the use of learning ability, a much greater ranking efficiency would result. Our study is centered on the hypothesis that we can design an algorithm that explores the search engine result, and downloads the text content of each URL [full name] appearing in the search result. After that, the algorithm stores the text of each URL in a distinct memory location. Then, it compares the content of each URL with a reference document provided by the user. Finally, the algorithm assigns a weight to each document and ranks them accordingly. If the suggested algorithm is intelligent enough to learn, we can apply its knowledge to the ranking task. This hypothesis implies that the new rank is much better than the original search's result. It also implies that the algorithm achieves the new rank using its knowledge. We designed an algorithm to embark on this quest. To show that this algorithm improved the search engine rank result, we conducted several experiments. Finally, we compared our algorithm, the "linear vector algorithm", with a widely used algorithm, the "vector space algorithm", and the findings were spectacular. The principle and design of this algorithm, along with the experiment and results, are provided in chapter three.

Perhaps the most natural approach relies on a knowledge base that, for each user, contains a personal profile that defines his or her preferences. Our brief overview of previous literature, presented in chapter two, indicates that several mechanisms for building this profile by generalized observations of the user's behavior have already been suggested by our predecessors. What interests us in the research reported here is to what extent the time spent by a user on a web page can be used to measure his or her interest in the page. Chapter four discusses the methodology and the result we found in using time to infer user interest of a web page. The user interest profile is employed to help search engines to work efficiently as a tool.

Giving a closer look at search engines as a tool to retrieve information from the Web, we can say that the principle of this tool is simple. Upon entry of the user's query, the search engine analyzes its repository of stored web sites and returns a list of relevant hyperlinks ordered by the relevance of the web sites to the user's needs. Many mechanisms to assess this relevance have been exploited, among them keyword frequency, page usage, link analysis, and various combinations of these three. Each of the multitude of alternative ranking algorithms leads to a different hyperlink ordering. Hence it becomes necessary to determine which of these algorithms yields the best results in terms of offering the most realistic set of hyperlinks to an average user query. A two-pronged strategy is necessary if the question is to be answered in a satisfactory manner. First, we need appropriate experimental procedures that submit to the machine well-selected testing queries to which the relevant answers are known. Second, we need performance criteria to evaluate the quality of the search engine responses to the testing queries.

In our work, we focus on the latter aspect. As presented in chapter two, the previous research has predominantly used the current classical performance metrics of precision and recall that are commonly used in the field of information retrieval. However, the utility of these metrics for search engine evaluation is limited: precision and recall establish whether the returned list contains the predominantly relevant links, and how many relevant links are missing. What they ignore is whether more relevant links find themselves high up on the list. We address our method accompanied by examples and comparison with existing algorithms in chapter five.

Collaborative filtering builds a database of preferences. It relates items to users to provide automated predictions for new items. When a company releases a product such as music, a movie, or a book, it tests the product with consumer data previously stored in its database. The rate that was provided by the users usually reflects other users' opinions. For instance, if a company (X) is going to release a product (P), and the company (X) wants to know in advance which customer segment would prefer the product (P), then the company (X) will check its database and present the product (P) to some customers, such as (C1, C2 and C3). (C1, C2 and C3) were selected from one cluster (S). This means that customers (C1, C2 and C3) have the same preferences (please refer to figure 5.1). If (C1, C2 and C3) have rated the product (P) 5 out of 5, then all other customers in cluster (S) would most likely give the same or a very close rating to the product (P). If the company (X) releases its product (P) to the market, then it can confidently recommend it to all its customers in cluster (S).

Chapter six presents the results of several experiments that we have conducted in a filtering system. We collected an enormous amount of data from more than one hundred users working at ten distinct and distant locations. This data was compiled using the filtering technique to find out whether a fast recommender system could be implemented to improve the recommendation result or not. The conclusion and discussion of the dissertation are presented in chapter seven.

In summary, this dissertation investigates ranking algorithms. We design a linear algorithm to rank the result and compare it with an existing and widely used algorithm.

Then, we investigate a way to personalize the search engine so that we can use a filtering system as a web page recommender system. To personalize the system, we need to build a user profile. This is achieved in chapter four where we found the best and least expensive way to infer user interest. Then, we found a method to evaluate search engine efficiency other than precision and recall. We proof our method mathematically and drive a formula with three claims to show how our method stays valid in different scenarios.

Problem definitions and performance criteria:

In this section, we present the significance of the research and its contribution, along with the problem definition and performance criteria.

To examine the new linear ranking algorithm, we will test its performance against a widely used algorithm, and its learning ability and error rate in recommending new web pages to users.

Many factors can be interpreted as indicating a specific user's interest in a given web page: time spent on a web page, mouse movements, page scrolling. What we want to find out in this context is the importance of the amount of time a user spends viewing a web page. The answer to this question will tell us how crucial – as compared to other criteria – it is to include this information in the user's profile. We have conducted several experiments using the inputs of dozens of volunteers experimenting with a transparent experimental setup (please refer to chapter four). When visiting a web site, the user provides the explicit rating of the site as perceived by the user. At the same time, the system implicitly rates the site based on the log of the user's behavior.

To respect their privacy, the volunteers could switch off the monitoring system any time they wanted to work unsupervised (this might have modestly corrupted the results). In order to obtain a sufficient amount of information, the measurements were taken over three months.

In order to measure the performance precisely and draw a conclusion from this part of the experiment, we had to define the performance criteria, which is the amount of time a user spent on a web page. To measure the time precisely, we had to implement a computer program to record the time when a user logged in and out of each web page visited. An explanation of the method and measures taken are provided in chapter four.

In the third part of our work, we designed a performance evaluation tool, as shown in chapter five. To ensure precise measurement, we had to experiment with the ranking result provided by the search engine and record the list of a rank. We extracted the relevant web page, found its exact position in the list and applied the formula to find the overall system performance.

Finally, we had to find a particular web page related to the topic that the user was looking for and match it with his or her interest. We used a filtering technique to recommend a particular web page that had been strongly recommended by a user in the same cluster. The performance criteria that we measured were the appreciation and satisfaction the user reported. We can infer appreciation and satisfaction by measuring the rating given by our user to the proposed web page.

Problem statement:

Search engines have become one of the most widely used tools online. Most users use search engines before visiting a web page to find out the web page that best meets their needs. Efficiency of search engine retrieval and its performance is the major problem addressed by this research. The dissertation fulfills the need of an efficient search engine that returns the result of a query with optimum performance. The performance is then measured on a user-satisfaction scale. The second problem addressed by the dissertation is the numerical evaluation of the retrieval system. At present, there is no tool that can numerically evaluate the order of the retrieved links. We investigated this problem.

Research questions:

In this dissertation, we investigated many unsolved problems in the search engine field. We addressed the problems in the following questions:

1- Can we design a linear algorithm to rank a search engine result and then test it against an existing algorithm?

2- Can this algorithm be intelligent enough to learn and, consequently, improve its performance?

3- How can we build a user profile efficiently?

4- Do we need all factors, such as mouse movement, page scrolling, and commands, such as print, save, etc., to infer a user interest?

5- Can we use time only to infer a user interest?

6- Can we design an algorithm to numerically evaluate the efficiency of a search engine ranking?

7- Can a filtering technique be implemented successfully on a web page recommender system, as it was for music, movies, and books?

Research goal:

The main goal of this work is to find the answers to the research questions and contribute their solutions to the problems facing search engine research fields.

The specific research goals are as follows:

1- Design a linear algorithm to assign weight to the retrieved documents.

2- Use time only as a factor to evaluate user interest in a web page.

3- Design an algorithm to numerically evaluate the search engine ranking efficiency.

4- Use a filtering technique to recommend a web page to a user.

5- Design a fully automated recommender system based on a filtering technique and time inference engine.

Chapter 2

Review of Relevant Literature

Earlier literature has reported several alternative approaches to personalized search engines [1, 2, 5 and 12]. For instance, in the paper by Fan and others [1], the authors demonstrate how their approach improves the search's efficiency and utility. The authors argue that the term weighting strategy should be context specific, where different term weighting strategies are applied to different contexts. In our earlier work [33, 38], we implemented a search engine personalization tool and reported experimental results showing how personalization managed to improve the quality of the ranking of the search engine output. Many approaches can be used to personalize search engines, the most important among them perhaps the use of machine learning techniques. In our work [33, 38], we demonstrated the advantages of the use of induction in this context, and we reported significant improvement of the quality of the search engine's ranking. Also Agichtein and others [3] emphasize the importance of feedback that, in their experiments, brought about an improvement of ranking efficiency by 31%.

Sufficient amount of recent activities in machine learning contributed to information retrieval and web search. Boyan and others [2] introduced a heuristic method to optimize the search's result and to improve the overall system performance. In that particular work, the authors showed how they successfully implemented a machine learning approach to improve the search's retrieval efficiency. They designed their system "Learning Architecture for Search Engine Retrieval" within whose framework they implemented a learning algorithm to assign different weights according to the word location in a text. The authors rely on content-based approach to improve the searching result. Their method relies on the work done by Fan and others [1]. This approach, as we suggested, would improve the result efficiency in quality but harm the time complexity as shown in [33, 38].

Poincot and others [4] introduced a new approach to compare documents and calculate their similarities using machine learning. The authors showed how documents' similarities could be calculated using neural network (Kohonen maps). They used keyword association with a document. Each document is retrieved into a particular cluster. Similar documents are retrieved into a specific cluster. The authors compare the performance of their system "CDS document map" with another system "ADS" and conclude that the two systems are complementary. CDS is a self-organizing map, where documents are gradually clustered by subject themes. The tool is based on keywords associated with the documents. For one selected document, the system locates it on the CDS document map and retrieves articles clustered in the same area. It was developed at the Centre de Données astronomiques de Strasbourg, France, thus called "CDS," ADS stands for "(NASA) Astrophysics Data System," This system has the capability to find all similar abstracts in the ADS database, with "keyword request," Chakrabarti and others [5] presented a web mining algorithm using hub and authority's techniques to discover relevant Web pages. Kleinberg [57] proposed a notion for the importance of web pages. He suggested that a web page's magnitude and importance should depend on the performed search query and each page should have a separate "authority" rating. The authority rating is based on the links coming to this particular page. It should also have a separate "hub" rating which is based on the links going out from this particular page. So the hub refers to the links from the page and authority refers to the links to the page. Ahonen and others [7] experimented with the co-occurring ‘text phrase.’ The ‘text phrase’ has a number of words in a particular sequence. The longest sequence is calculated from number of documents. The author reported that many textual characteristics were revealed.

Liu and others [10] proposed a personalized web search for improving retrieval effectiveness. They implemented a machine learning algorithm to capture the user interests. Every time the user connects to a given web address, the system logs its URL and classifies it into a personalized cluster. This process improves the overall system performance as every URL is reflected on the subsequent query. The authors use the learning algorithm as mandatory and without the user awareness, which means the user does not know that the system is monitoring all his or her activities. The authors conclude that their method is effective and efficient. Müller [11] provided another implementation of Web Information Retrieval, using machine learning algorithm to track users’ interests. Their algorithm attracts relevant Web pages and filters out irrelevant ones. They implemented their system in a multi-agent approach. The authors claim that their system offers a highly open architecture which can be easily expanded with less maintenance. The system is transparent and robust as the authors reported. Lin and others [15] showed how machine learning algorithms could be used to classify web documents. This method explores the documents before the searching process starts. Then, the algorithm categorizes the documents according to the knowledge acquisition process. The authors reported that the knowledge acquisition process of the system learns classification from classified Internet documents. Therefore, it classifies new documents accordingly.

Kim and Chan [8] advocated the use of user profile for search engine personalization and showed that building a hierarchical user's profile by monitoring his or her browsing activities is possible. Sometimes, explicit rating (where a user has submitted his or her rating for a particular web page) is available. For those cases where this is impossible, Kim and Chan [8] showed how the system can infer implicit rating from the user’s earlier behavior. Stevens [21] described the distinction between explicit and implicit rating and discussed their advantages and disadvantages, without recommending any concrete model. White and others [27] carried out experiments whose goal was to compare implicit and explicit ratings. They showed that implicit rating could replace explicit rating with high accuracy. This is important because it removes the problem of recording explicit ratings. Sarwar and others [32] conducted a study that showed that even though explicit rating is precise, many users would rate fewer Web pages than they actually visited. The authors concluded that explicit rating was not as efficient as previously thought. Grudin [30] explained that users tend to stop rating Web pages if there are no direct benefits for them. Middleton and others [26] argued that explicit rating interferes with the user’s behavior. They believe that no one can convince a user to rate a Web page explicitly even when benefits are obvious. Also Shapira and others in their paper [9] reason that explicit indicators disturb user browsing. They explore many implicit indicators (other than just time) and reported that they found promising results, but they still observed that combinations of implicit and explicit indicators are much more useful. Likewise, Oard and Kim [13] argue that implicit feedback can substitute for explicit rating because it avoids the difficulties associated with gathering the information from users. The authors propose three different categories for implicit feedback: examination, retention and reference. Nichols [14] indicates that the difficulties of collecting explicit rating from users can be avoided by using implicit ratings. However, he found privacy violation the only obstacle in implementing implicit rating. Hill and Terveen implemented the PHOAKS system [19] and reported that more than 90% accuracy of correct recommendation can be obtained when using implicit rating. Jung [23] explores different factors related to implicit rating. He collected data from such indicators as the numbers of mouse clicks, mouse movement, copying, rollover, duration, select all, print, forward, and some others. Then, he compares this data with explicit rating and reports that the number of mouse clicks is best suited to serve as an implicit indicator. Claypool and others [24] conducted a study in which they compared implicit rating with explicit rating and showed that the time spent on a give page and the amount of page scrolling provide high accuracy correlation with explicit rating. They suggest that mouse click or page scrolling alone do not lead to reliable predications of user's preferences. Goecks and Shavlik [25] have investigated mouse movement, scrolling and browsing activities and suggest that a machine learning system might induce user's interests depending on these factors. Experimenting with his system, Letizia Lieberman [28] explores the possibility that a link-based approach might indicate implicit rating. He argues that it is in an indication of interest if a user follows a link, and then dwells on that link for some time. However, if the user bounces back instantaneously, the Web page is most likely regarded as uninteresting.

Badi and others [29] explore the reading activities and organizing activities. Reading activities could be the time spent in reading a document, the number of mouse clicks, the number of text selections, the number of scrolls, the time spent in scrolling, the number of documents access, etc., whereas organizing activities could be the number of object creations, the number of object moves, the number of object resizing, the number of object deletions, the number of background color changes, etc. Measurements of these activities were recorded and analyzed. The authors reported that they have found a great correlation between explicit rating and their reading and organizing activities combined. However, they suggest that using each model of activities such as reading activities or organizing activities by itself has much less accuracy.

In our earlier work [33, 38], we built a user profile for each user. Each profile is stored in a distinct file. From those files, the system extracts keywords as positive and negative examples. Positive examples are those web pages with high degree of similarity to the user interest. For instance, the user who is interested in biology and looking for the keyword "jaguar" would have web pages related to biology as positive examples, and web pages related to car are considered as negative examples. Negative and positive examples are extracted from positive and negative web pages. We reported that the use of positive and negative examples in a machine learning system improves retrieval efficiency.

Oard and Kim [35] explored four different categories of user behavior. They experimented with a user who wants to examine (view, listen and select) a document, retains (prints, bookmarks, saves and purchases), references (copies, pastes, forwards, replies, cites and links) and annotates (mark ups, rates and publishes) a document. The authors reported that a combination of this behavior has a positive impact in determining users' interests.

Konstan and others [18] reported that their work shows that implicit rating predications based on the time spent in reading are very close in accuracy with predictions based on the explicit ratings. Morita and Shinoda [34] hypothesized that a user would spend a longer time in reading an interesting article than an uninteresting one. The time spent depends on article length and readability. The authors concluded that the time spent is proportional to the level of interest. However, it is not related to the article length or readability. Kim and Chan [22] examined closely the time spent factor as an implicit indicator. They divided the time into three duration categories: complete duration, active window duration and ‘look at’ it duration. After their close investigation, the authors reported that the time spent on a Web page is strongly related to the user's interest. Hill and others [20] monitor user's actions while the user is in editing activity or reading activity. The time that a user spends in editing or reading reflects the amount of interest in a Web page. Chan [31] conducted an experiment involving a machine learning algorithm to personalize the search engine. His classifier uses frequency and dwelling time as well as the recently visited Web page. The author suggests that the more recently visited Web page is more interesting to the user than the older one.

Precision and Recall is the most widely used tool to evaluate an information retrieval system. Precision and Recall are used to evaluate the efficiency of information retrieval systems. Precision is defined as the ratio of the relevant documents retrieved to the total number of retrieved documents. Recall is defined as the ratio of the relevant documents retrieved to the relevant documents in the database of the system. It is used by scientists to evaluate retrieval information systems. Zhang and Dong [44] present a review of many ranking algorithms and discuss the deficiencies in the existing techniques. The authors propose an algorithm with a multidimensional technique and claim an improvement in the ranking result. Their algorithm produces more relevant documents and better precision. Shafi and Rather [45] use precision and recall to evaluate the performance of five different search engines. Chu and Rosenthal [46] present the same evaluation criteria for retrieval performance as the work proposed by Shafi and Rather [45]. However, they use precision and response time instead of precision and recall. Li and Danzig [47] introduce a new ranking algorithm. They argue that their technique is much better in space and time complexity. The authors claim that their system has a better precision and recall than the existing algorithms.

New approaches evolved in ranking algorithms with new ideas, but precision and recall was used to evaluate the retrieval system. Eastman and Jansen [49] explore the impact of query operators on web search engine results. The authors use coverage, relative precision and ranking as questions trying to answer in their research. Goncalves and others [50] present an algorithm to measure the effectiveness of a retrieval system as an overall. It measures how much a document is relevant to the query, but it does not compare two retrieval systems. It does not show if a rank of a retrieval system is efficient enough. The authors use precision and recall as an evaluation tool. Yuwono and others [51] explore the relevance feedback in affecting the retrieved documents. The authors use precision and recall as a tool to evaluate the ranking efficiency. Hawking and others [52] tried to answer the question "Can link information result in better PageRank?" The authors discuss the effectiveness of a search engine and its performance by measuring its precision and recall. Yuwono and Lee [54] provide four different ranking algorithms: Boolean Spread Activation, Most-cited, TFxIDF and Vector Spread Activation. The authors use different queries to compare these four algorithms with each other. Their ranking evaluation was based on precision and recall. The hypertext algorithm was a new approach proposed by Brin and Page [53] to improve the ranking of retrieved web pages. The authors claim that this approach would improve the search result by having high precision rank. Baeza-Yates and Davis [55] show that link attribute of a Web Page can improve the ranking by improving the precision of the system. Trotman and O'Keefe [56] use precision to evaluate the ranking algorithm. They depict how a weight is awarded to each document.

Pay per performance (PPP) search engine is a different approach in search engine ranking. Goh and Ang [16] discuss this approach and use precision and recall to evaluate the ranking performance. Ljosland [58] presents a comparison between three search engines: Atavista, Google and Alltheweb. The author uses precision to evaluate the performance of each engine. Bifet and Catillo [60] explore the top web pages appearing in the rank. They also explore the shifted ones. The authors use precision to calculate the efficiency of the rank.

Precision and recall was used in most ranking evaluation as we saw in previous works. However, many scientists use different evaluation tools. Precision and recall can evaluate the retrieval system, but they cannot precisely evaluate the efficiency of the rank. Any change in the order of the retrieved documents does not necessarily affect the precision and the recall. This variable (the order of the retrieved documents) cannot be measured using precision and recall method.

Clarke and Cormack [48] introduced a new approach toward ranking evaluation. Their work was to evaluate each document and give a specific weight to the document according to all other retrieved documents. They are interested in documents' weight according to other documents. Their method would change the order of the retrieved documents. But it does not evaluate the rank itself. Algorithms for ranking retrieved documents such as these introduced in [43, 53 and 57] were used to rank web pages, however, they still do not measure the ranking algorithm and its efficiency. Kamvar and others [62] explore many PageRank scheme and provide two algorithms. They present the Adaptive PageRank and the Modified Adaptive PageRank. The authors have not discussed the ranking evaluation in their work. White and others [59] present an evaluation to encourage user to interact with the search result, they showed how their approach improves the PageRank. However, their paper does not show any tool to numerically evaluate a PageRank.

Some more evaluation tools other than precision and recall were introduced. Losee and Paris [61] oppose the use of precision and recall as a measure to evaluate search engine ranking performance. The authors suggest a probability method and proved that their proposed solution would result in a much better evaluation. The authors present the Average Search Length (ASL). ASL finds the average position of the retrieved document. This method is much better than precision in evaluating the ranking performance. However, as the authors mention, a small number of relevant documents in the top of the rank may represent a superior performance. They present the Expected Search Length (ESL) as an alternative approach. This method counts only the non-relevant documents. In this evaluation the system must minimize the ESL value for better performance. The authors [61] advocate our approach in finding an alternative method to measure the performance of a ranking system. Haveliwala [17] compares two different ranking methods by measuring the degree of similarity. He calculates the degree of overlap between the top URLs of the two ranking lists. Our approach is to find a numerical evaluation for each ranking list rather than comparing the two different ranks.

Pazzani [36] explores collaborative filtering and its impact on a recommender system. The author combines collaborative filtering with content-based and demographic filtering to improve recommender systems' result. The availability of information in the system does not guarantee efficiency. Thus, using collaborative filtering in this regard boosts the recommender system tremendously. Prem and others [40] propose a similar approach as Pazzani [36]. They suggested that incorporating collaborative filtering with content-based approach enhances the recommender system. Miha [37] builds a collaborative filtering and a user profile. He states that collaborative filtering has the advantage of creating relationships between non-textual items, because it ignores the content of the documents and depends only on preference engine. In other words, the author concludes that using collaborative filters has the advantage of recommending a web page based on its rating by other users, not by its textual contents. This is a great advantage for non-textual documents because they do not have textual content such as video or audio. Therefore, using collaborative filters ignores document contents and calculates weights according to users' recommendation and does not look into the document contents. Collaborative filtering was first used in the Tapestry project at Xerox PARC [47]. The system allowed users to find documents based on recommendations and comments by other users. Few problems were found with the system due to the fact that a rather small group of people were contributing to the knowledge-based system.

Oka and others [39] introduced an approach in which they unified collaborative filtering domains. They argued that the data collected for books' preferences is stored in a specific domain and the data collected for the movie preferences is stored in another domain not accessible to the former. The authors introduced their algorithm which they claim should unify all data. The data collection contains different domains such as books, movies and music. These domains are unified into a single platform accessible to all domains' members.

Resnick and others [41] reported that evaluation of different collaborative filtering algorithms is always a controversial subject, which comes to undecided conclusion among scientists. Some researchers use a specific evaluation technique; other use totally different ones. This is because different collaborative filtering algorithms need different matrices. We sometimes have the number of items much larger than the number of users, such as researchers and papers, whereas, sometimes, we have the number of users much greater than the number of items, such as users and movies. Many different scenarios require different mathematical formulas. Many other factors affect the selection of mathematical formula depending on scenario, the amount of data and the selection of data set. Nakamura and Abe [42] proposed their new algorithm for collaborative filtering-weighted majority prediction. This is a prediction algorithm where a learner would have a sequence of trials with the goal of the learner to make as few mistakes as possible when a prediction is made. The authors report that the new method performs better than previous algorithms.

In summary, the personalized search engine was addressed by many investigators as an approach to resolve search engines' difficulties. The machine learning method was investigated as a possibility for major contribution to search engine enhancement. Researchers in machine learning provided fabulous results related to search engines' applications. Building a user profile is a crucial approach toward search personalization. A lot of research has been published in this field to determine the appropriate tool in building a satisfactory user profile. This profile should reflect the user interest in a web page. Collaborative filtering was addressed in earlier literature as a pioneer method for recommender system. In our research, we deployed these principles to investigate the basic building blocks for constructing a fully automated recommender system for search engine using collaborative filtering.

Chapter 3

Search Engine Personalization Tool

Based on the Linear-Vector Algorithm

The study of Internet search engines represents a dynamic field, full of challenging research issues and questions that still wait for their systematic investigation. This chapter introduces Linear Vector Algorithm (LVA), proposed as an efficient mechanism to rank the hyperlinks returned by a search engine [33, 38]. The method can be categorized as a knowledge-based system where the term “knowledge” refers to a user profile as induced from the logs of the user’s previous behavior (converted into positive and negative examples of his or her previous preferences).

This chapter presents the Linear Vector Algorithm (LVA) and discusses its use for improving the ranking quality, as well as its mode of operation, and some considerations related to the design and implementation. The primary objective is to improve upon an earlier approach from the field of information retrieval (in terms of time complexity and ranking performance). This earlier approach is known as Vector Space Algorithm (VSA).

When a typical Internet search engine receives a user’s query, mostly expressed in terms of a list of keywords, its first task is to identify those web pages (documents) that contain the keywords. This is followed by a second task, which is to estimate the relevance of these documents to the user’s query. After that the user is provided with a list of hyperlinks for these documents, ordered by their relevance. Historically, a document’s relevance to the user’s needs has been calculated from

1- The frequency of the keywords in different parts of the documents, 2- The popularity of these documents (measured by the time spent on them by an average user), where the system keeps history of a user profile for a specific period of time which last for few months.

3- The structure of the links to and from related Web pages.

This basic tenet of this dissertation is that each user has somewhat different interests and preferences. Moreover, these interests and preferences come to a certain degree of prediction based on the contents of the documents that this particular user has visited in the past. Alternatively, the user may want to indicate his or her preferences by providing examples of relevant documents (the documents proposed by the user as interesting web pages or interesting documents). Suppose a user submitted the word ‘puma’ to the search engine. He or she is required to submit some relevant documents. If the user is interested in sport, then a few documents regarding sport will be submitted as interesting documents. However, if the user is interested in wildlife, then some documents regarding animals will be submitted as relevant documents. The dissertation presents (and experimentally evaluates) a *search-engine personalization tool* that uses this personal information to rank the hyperlinks to suit the specific a user. The process consists of the following steps:

First, the user’s query is submitted to an off-the-shelf search engine.

Then, the textual parts of the documents recommended by this search engine are downloaded.

Finally, the downloaded documents are ordered according to the knowledge induced from documents previously visited by the same user. The ranking criteria is based on Linear Vector Algorithm (LVA) discussed later in this chapter.

A recently recommended algorithm is known under the acronym VSA (Vector Space Algorithm). Unfortunately, this approach is relatively slow, and perhaps unnecessarily complicated for this particular application. We sought an improvement to this approach.

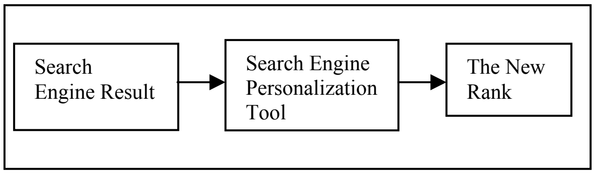


Figure (3-1): SEPT System, the Search Engine Personalization Tool SEPT reads the returned result from the search engine, compiles it and presents a new rank

3.1. System design

3.1.1. Search Engine Personalization Tool (SEPT)

Basically, a search engine personalization tool (SEPT) developed in the course of the work reported in this dissertation consists of a Graphical User Interface [GUI], a downloading algorithm, a ranking evaluation algorithm, and a training algorithm.

System goal and functions

As mentioned earlier, the goal of SEPT is to read the search engine's results, and to re-rank these documents in a descending order according to their "weights" (higher weights indicating higher relevance). Figure (3-1) helps clarify the essential functions of the SEPT system.

Let us first describe the Vector Space Algorithm (VSA, [63]) that is widely used in information retrieval ranking. Let us denote by *W(u,S)* the weight of term *u* in document *S*, and let us denote by *W(u,Q)* the weight of term *u* in query *Q*. The term document refers to the document exist in the database of a search engine. We want to find the document in the database which most similar the query. The term query refers to the document we have and want to find similar on in the database of the search engine. Then, the similarity between the document S and the query Q is calculated using the following formula:

$$\cos(S,Q) = \frac{\sum_{u \in S \cap Q} W(u,S).W(u,Q)}{\sqrt{\sum_{u \in S} (W(u,S))^2 . \sum_{u \in Q} (W(u,Q))^2}} \qquad 3\text{-}1;$$

Please refer to [63] for more details on VSA and equation 3.1.

To use this method we do the following:

1- Find all terms *u* in query Q.

2- Assign weight to each term *u* in query Q, weight = *W(u, Q)*, this weight is the frequency of the term u in the query Q according to its position. Terms in title and introduction have more weight than terms in the body of a document.

3- Find all terms *u* in document S which are similar to terms *u* in Q.

4- Assign weight to each term *u* in document S, weight = *W(u, S)*

5- Find the similarity sim(S, Q) = $\sum_{u \in S \cap Q} W(u,S).W(u,Q)$, all terms *u* should be found in S and Q.

6- Find the normalized similarity cos(S, Q) when we divide by the Euclidian distance between S and Q, ($\sqrt{\sum_{u \in S}(W(u,S))^2 . \sum_{u \in Q}(W(u,Q))^2}$). We normalize the similarity so that longer document does not take grater weight than shorter ones.

7- For the i^{th} document S_i, repeat step 4 to 6 to find the similarity vector cos(S_i, Q)

It is quite obvious that the similarity between S and Q is very complex as it uses the square root and the square function. As explained by Salton and other [63], the indexed term u in the document S might be weighted according to its importance with the weight W(u,S) and the indexed term u in the query Q is weighted according to its importance with the weight W(u,Q). The authors defined the cosine similarity presented in equation 3-1 as a method used in their vector space algorithm to measure the similarity between two vectors, a document S and a query Q.

Proposed solution

The input to SEPT is two-fold: (1) user-submitted "preferred document(s)" and (2) a set of URLs obtained in response to the user's query submitted to an off-the-shelf search engine. As indicated, the system reported here relies on a simple induction technique to "learn" how to assess the relevance of a document to the user's needs. The training phase uses positive and negative examples that have been labeled as such by the user that has scanned the output of the search engine. The positive ones are those that the user believes have a high degree of similarity to his or her query. The negative examples are those that the user deems irrelevant to the query.

Compare D and R in equation 3-2 to the terms W(u,S) and W(u,Q) in equation 3-1:

Lets assume that D is a frequency of a word in a document S. Lets assume that the word "Dinosaur" is repeated 13 times in the document S, then D = 13 for this particular term. However the same word "Dinasaur" in equation 3-1 is W(u,S), which is the weight awarded to the term u according to its significance in the document S but not a frequency of the term u as we used previously in equation 3-2. Let R be the frequency of a word in the reference document or the query.

In equation 3-2, the weight is found by the ratio $\frac{D}{R}$, as it reflects the similarity between the retrieved document and the reference document, because D and R represent the frequency of the same word in both documents, which means as long as the ratio is closer to one, the weight become higher.

We now define $P = \frac{D}{R}$

Weight = P * 100 if $P \leq 100$,

Weight = 100 – (P – 100) * 0.1 if $(100 < P \leq 1100)$,

Otherwise Weight = 0; (3-2) ;

Figure (3-2) illustrates the difference between the two approaches by showing the weight (“score”) they assign to a given term for different values of the $\frac{D}{R}$ -ratio. It is obvious why the new formula is called linear vector algorithm (LVA). Admittedly, the linearity is a just a simplification. This formula is much easier to calculate, and less prone to over-fit noisy training data.

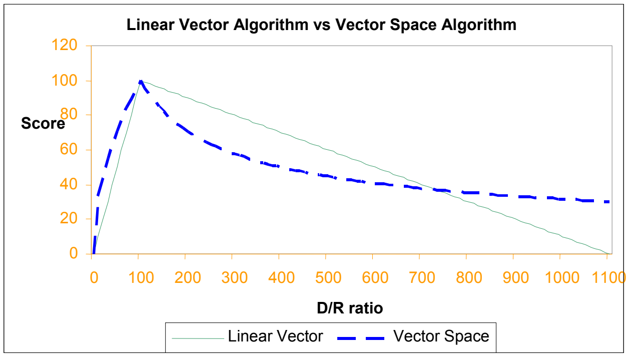


Figure (3-2): Calculating weight for each document using Linear Vector Algorithm vs. vector space.

Example 3-1: Calculating the weight for selected words

First, we need to clarify some definitions used in the calculation. Retrieved documents are those documents returned by the search engine, and reference documents are those documents submitted by the user as interesting documents. Let us show how our algorithm calculates the weights of the word “Audio”, please refer to (Table 3-1):

$$\mathrm{P} = \frac{\text{Frequency in the retrieved document (D)} * 100}{\text{Frequency in the reference document (R)}}$$

Table 3-1 tells us that the frequency of the word “Audio” in the reference document is 25 *words/document,* and in the retrieved document = 25 *words/document.* Therefore, the $\frac{D}{R}$ ratio is as follows:

$$\mathrm{P} = \frac{25 * 100}{25} = 100$$

According to the linear vector behavior's curve in Figure (3-2) and Formula (3-2), if P ≤ 100, then weight = P

Thus Audio (weight) = ratio = 100 *points.*

Similarly, the weight for the word "Food" is calculated as follows:

$$P = \frac{13*100}{14} = 92.8571$$

According to the linear vector behavior's curve in Figure (3-2) and Formula (3-2), if P ≤ 100, then weight = P

Thus Food (weight) = ratio = 92.8571 *points.*

Finally, the weight for the word "Connection" is calculated as follows:

$$P = \frac{14*100}{10} = 140$$

According to the linear vector behavior's curve in Figure (3-2) and Formula (3-2), if (100 < P < 1100), then

$$\text{weight} = 100 - [\frac{P - 100}{10}]$$

$$= 100 - [\frac{140 - 100}{10}]$$

$$= 100 - [\frac{40}{10}]$$

$$= 100 - 4 = 96 \text{ points.}$$

Table (3-1): Word frequency

Words	**Word frequency in the reference document**	**Word frequency in the retrieved document**
Audio	25	25
Food	14	13
Connection	10	14
Stability	0	7
Miami	12	0
Fund	23	5
Chair	14	14

The weight assigned to a retrieved document = $\sum_{i}^{n} weight(i)\,.\,C_i$,

where weight$_{(i)}$ is the weight for the word$_i$,

n is the number of distinct words in the document, [in this example n =7.]

ci is a factor used to panelize difference.

$C_i = P$ for $D < R$

$C_i = \frac{1}{P}$ for $R < D$

$C_i = 1$ for $D = R$

Document’s weight = Audio (weight). C_1 + Food (weight). C_2 + Connection (weight). C_3 + Stability (weight). C_4 + Miami (weight). C_5 + Fund (weight). C_6 + Chair (weight). C_7

$$= 100\ .\ 1 + 92.8571\ .\ \frac{13}{14} + 96\ .\ \frac{10}{14} + \ldots\ldots\ldots$$

$$= 100 + 86.22 + 68.571 + \ldots$$

The philosophy of this algorithm is to reward similarity, and penalize excessive repetition of similar words. Moreover, the algorithm also penalizes words that have lower retrieved document frequencies (D) than reference frequency (R). The algorithm computes the frequency of each word in the reference document and the retrieved document, and then assigns 100 points to each word if the frequencies of the word in both documents are identical where $\frac{D}{R}$ = 1. The word “Audio” in Table 3-1 is awarded 100 points regardless of frequency. As long as the frequencies in both documents are the same, the word has 100 points. In example 3-1, the algorithm assigned 100 points to the words “Audio,” and “Chair,” These two words have different frequencies; however, they have similar frequencies to their peers in the reference document. 100 points is the maximum number of points. In Table 3-1, the word “Food” has a frequency in the retrieved document = 13 *words/document*, which is less than the frequency in the reference document. In this case, the algorithm uses the first part of the graph where P * 100 is in the range [0, 100].

The algorithm operates in the [100, 1100] range if the frequency in the retrieved document (D) is greater than the frequency in the reference document (R). The word “Connection”, in Example 3-1, corresponds to this case. The frequency of the word

"Connection" in the retrieved document = 14 *words/document*, and its peer in the reference document = 10 *words/document*. The P * 100 ratio for the word "Connection" was calculated and found to be 140. This ratio is in the range [100, 1100]. According to Figure (3-2), the weight = 96 *points*. Here, we can see how the algorithm penalizes the excessive repetition of the word "Connection" in the retrieved document and assigns fewer points. This algorithm follows a linear function as illustrated in Figure (3-2), and applies a linear formula as presented in Formula (3-2).

On the other hand, the VSA awards similarity and penalizes excessive repetition in a non-linear behavior according to Formula (3-1) [63]. The vector space curve vs. linear vector curve is shown in Figure (3-2).

We assigned a slope equal to 1 where P € [0,100], therefore, the new weight equals P. However, if the frequency of the word in the retrieved document is greater than its peer in the reference document, the algorithm penalizes this amount linearly.

We designed the second slope to allow a maximum frequency of retrieved document D to be no more than 10R as discussed earlier in this chapter. Therefore, the slope was calculated as follows:

$$\frac{D}{R} * 100 = \frac{10R}{R} * 100 = 10 * 100 = 1000$$

This slope starts in the horizontal axis where $\frac{D}{R}$ = 100, thus it should intersect with the horizontal axis at $\frac{D}{R}$ * 100 + 100 = 1100, consequently the slope = -0.1.

3.1.2. Training algorithm

The SEPT was implemented with a “machine stability algorithm” to enhance its performance as a result of training. The training algorithm reads the “remove” files (*.rem) that contain Web pages with a high degree of dissimilarity to the user’s interests. It also reads the “add” files (*.add) that contain web pages with a high degree of similarity to the user’s interests.

Suppose the user is interested in animals and wildlife, and suppose that the user is searching for documents related to the keyword “Jaguar,” The training algorithm reads *.rem, and *.add files to train the system. All common words are added to a list called CommonWords list. This list contains irrelevant common words (e.g., and, the, they, here…) and common technical words (e.g., html, http, www ….). The system also adds irrelevant subjective words such as car, model, make, or vehicle to the IrrelevantWord list. In this example, web pages related to vehicles, the car industry, and any field of interest other than cars are stored in *.rem files. Only web pages related to animals are saved as *.add files. Therefore, the training algorithm trains the system to identify words related to wildlife and animals. These words are stored in the RelevantWords list. The system is trained to ignore anything other than this word list.

3.1.3. Downloading algorithm

The SEPT reads the search results, extracts URLs and downloads the text data found at each URL. Images and videos are ignored because they are not used in the weight calculations. Figure (3-3) illustrates how much downloading time can be thus saved.

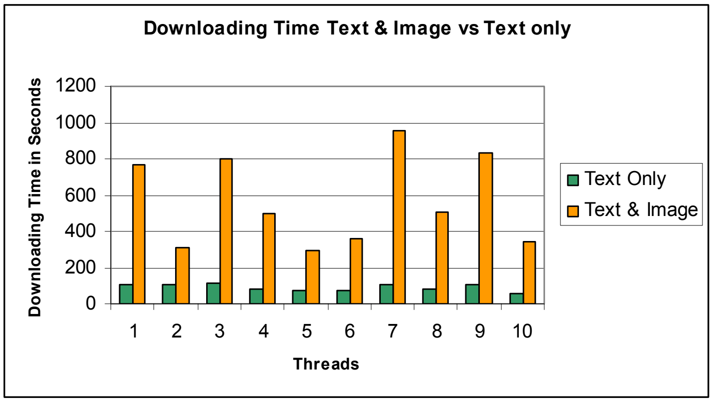


Figure (3-3): Using threads to downloading web pages. The figure compares between downloading web pages including images and web pages including text only.

3.1.4. Ranking efficiency algorithm

Let us now focus on how to evaluate the efficiency of the new ranking. The most widely used criteria are *precision* and *recall*. Unfortunately, these two criteria do not satisfy our requirements because they evaluates neither the rank of the retrieved documents nor their order. For instance, if we have five relevant documents in a retrieval system which retrieved ten items, the precision equals 50% no matter how the documents are ordered in the system. If the five items are in the top rank or in the middle or the bottom rank, the precision value does not change. However, recall of a retrieval system calculates the number of relevant documents retrieved from the total relevant documents in the database. As we are unable to find all the relevant documents with in the database of a search engine, computation of recall is infeasible. A full description of the tool used in this dissertation is found in chapter five.

3.2. Experimental results

To experimentally evaluate the system, a few test-beds were prepared, each defined by a different user query. For the queries, we chose keywords that could be expected to lead to multiple categories. For instance, by submitting the keyword “Jaguar”**,** a search engine is likely to return documents related to such topics as cats, cars, audios, etc. Likewise, the keyword “Research Fund” may return documents related to diverse scientific disciplines such as engineering or biology.

To start with, let us look at our system’s response to the keyword “Research Fund”. Suppose that the user is interested in “Cancer Research” and in “Medical Field” in general. Previously visited web pages containing these two words were labeled as positive examples. To these, we added some negative examples defined as documents that fail to contain at least one of these two terms. Having a sufficiently large pool of examples to choose from, we gave preference to those that displayed a high degree of relevance and other examples with high degree of irrelevance.

The goal of the first experiment is to establish LVA’s ability to predict the user’s perceived relevance of the returned documents and to find out how it depends on the number of training examples. The system has been run several times, each time on a training set of a different size. To begin with, we considered training sets that contained N positive examples and N negative examples (for N growing from 1 to 10).

3.1. Experiment # (3-1): Study of the LVA behavior in the "Research Funds" domain

This experiment shows how the LVA behavior responded to training.

Procedure:

1- We prepared the sample for the training; please refer to section (3.2.2).

2- We ran the training algorithm over different sets of examples (1 pair of add and remove files, 2 pairs, 3 pairs …. up to 10 pairs of files).

3- We assumed that the user submits the keyword “Research Funds” to the search engine.

4- The user is interested in the “Cancer Research” topic.

5- We carried out the training process as follows:

To train the system, we collected random positive and negative examples. We stored the positive examples in *.add files, and the negative examples in *.rem files. Each of them consists of a single Web page. The web pages with a high degree of similarity are stored in the *.add files. However, the web pages with a high degree of dissimilarity are stored in *.rem files. We have discarded all web pages, which have a low degree of similarity or a low degree of dissimilarity for experimental purposes only. We have created 10 *.add files and 10 *.rem files. These files have a tremendous amount of examples. Some files contain more than 7000 distinct terms. Once the samples were ready, we could start the training process. We launched the training agent to train the system. The training process takes about 10 to 30 seconds. However, the sample collection takes considerable effort and time. All the positive examples are regarding cancer research. However, all our negative examples are strictly related to non-medical interest. Once the system is fully trained, it behaves as if it is used by a cancer researcher for a while. Thus, the system responds effectively and rearranges the search’s results. The algorithm awards similar pages high weights, and penalizes dissimilar web pages.

6- We have trained the system and prepared it to assign weights according to VSA and LVA. In two distinct experiments, we collected the data illustrated in Figure (3-3), which was calculated according to the "ranking efficiency evaluation algorithm" discussed in chapter five.

Error rate

The percentage error rate is calculated as follows:

Percentage error rate = 100 - ranking efficiency evaluation

Figure (3-4) summarizes the results, quantifying the ranking performance by the error rate as obtained using the SEREET, an abbreviation of the term Search Engine Ranking Efficiency Evaluation Tool. The basic problem it addresses is the calculation of the efficiency of the rank such that the value it calculates changes with the change in the document's rank. For instance, if a system retrieves five relevant documents, all of which appear at the top of the system's ranking, the value will be much higher than the other system which has five relevant documents, all of which find themselves at the bottom of the list. When we exposed SPET to a learning algorithm and measured its performance using SEREET, we observed an improvement in the system's performance. The system behaved as indicated in Figure (3-4). The reader can see that, expectedly, the error rate tends to drop with the increasing size of the training set. When only one pair of training examples (one positive and one negative) was used, the accuracy of PT was unimpressive, indicating that a larger training set is needed. On the positive side, the number of examples needed for the system's performance to achieve reasonable levels does not appear to be as high as is typical for the field of machine stability.

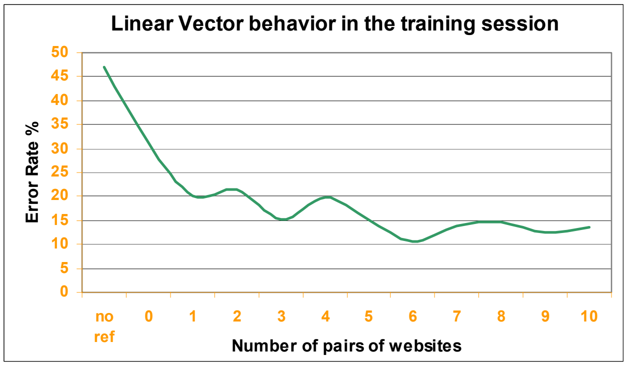


Figure (3-4): The response to the training algorithm using linear vector algorithm

3.2.2. Experiment # (3-2): Comparing LVA with VSA using "Research Funds" domain

In this experiment, we wanted to compare (in the same domain) LVA's performance with that of the more traditional (and more computationally expensive) VSA. Figure (3-5) summarizes the results obtained using the same training data as in the previous experiment. Again, the error rate was calculated using the SEREET mechanism described in chapter five. While both algorithms manifest the same error rate in the absence of reference pages, their performance improved with the growing size of the training set. Importantly, the LVA approach appears to be much more accurate in predicting the correct ranking than VSA, which has a comparably high error rate. The two curves tend to converge only for larger training sets.

Figure (3-5) clearly shows how the LVA outperforms the VSA in efficiency. The figure shows that both algorithms have the same error rate when there is no reference page because both algorithms are inactive. When we introduced a reference page, but no training was applied, the vector space and linear vector were found to have almost the same error rate of 31.36%. We considered the fact that the lack of sufficient examples in the reference page would cause this relatively high error rate. However, once the training algorithm is introduced to the system, we found that the linear vector method outperforms the VSA. The latter has a very high error rate. When we introduced more examples to the training algorithm, both methods started to have improvement in error rate, as is illustrated in Figure (3-5). The uneven behavior of both curves is expected in practical machine stability algorithms [64]. As more examples were introduced to the training algorithm, the curves converged further. We expect that the curve will keep improving the error rate if more examples are introduced.

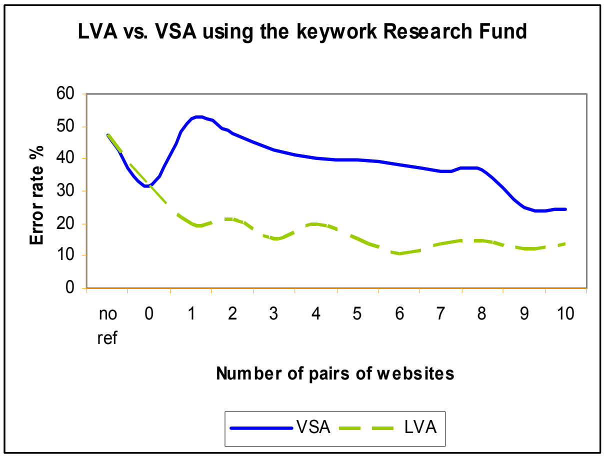


Figure (3-5): comparing the response to the training algorithm when it is used with vector space algorithm and linear vector algorithm for the keyword “research funds”

3.2.3. Experiment # (3-2): Comparing LVA with VSA using “Jaguar” domain

In this experiment, we used the keyword “Jaguar” and regarded as positive those documents that were related to an animal (rather than, say, a car). Figure (3-6) shows how an increased number of training examples leads to a higher ranking performance (along the SEREET criterion). Again, LVA tends to learn faster than the VSA.

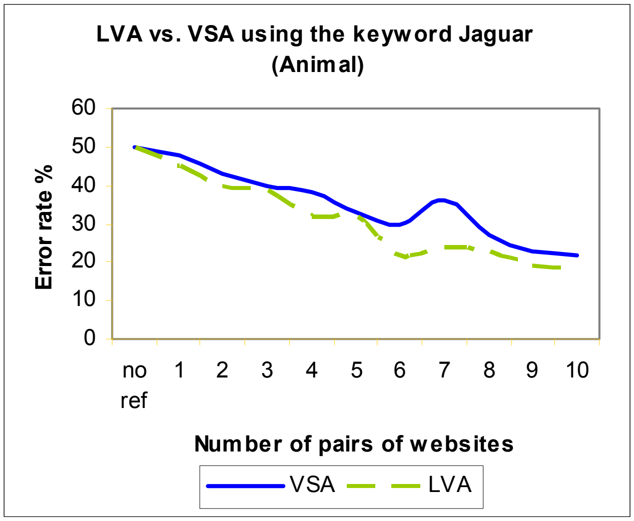


Figure (3-6): Comparing the response to the training algorithm when it is used with vector space algorithm and the linear vector algorithm for the keyword "jaguar"

3.3.4. Experiment #(3-3): Comparing LVA with VSA using "Beetle" domain

In this experiment, we used the keyword "Beetle". Figure (3-7) illustrates ranking performance. In all experiments, we found that the LVA can learn much faster than the VSA.

This experiment advocates our hypothesis that the LVA could substitute a widely used algorithm and improve the search's results for inexperienced users. However, knowledge and training are dominant factors in using the linear vector method.

3.2.5. Minimum amount of knowledge experiment

Downloading the contents of 100 web pages can take a considerable amount of time. However, we have introduced the concept of "minimum amount of knowledge," When a user submits a keyword, the search engine responds by providing relevant web-

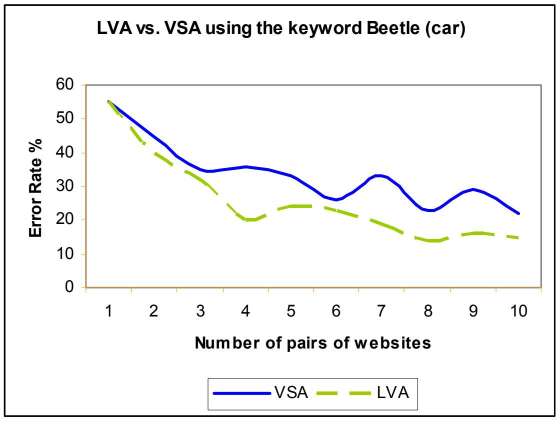


Figure (3-7): comparing the response to the training algorithm when it is used with vector space algorithm and the linear vector algorithm for the word "beetle"

page links, along with two to three lines of short description of each link. These lines are called "minimum amount of knowledge". Figure (3-8) is an example of the result.html file which contains the minimum amount of knowledge. If this amount is sufficient to construct a promising result, we can save a considerable amount of time and effort.

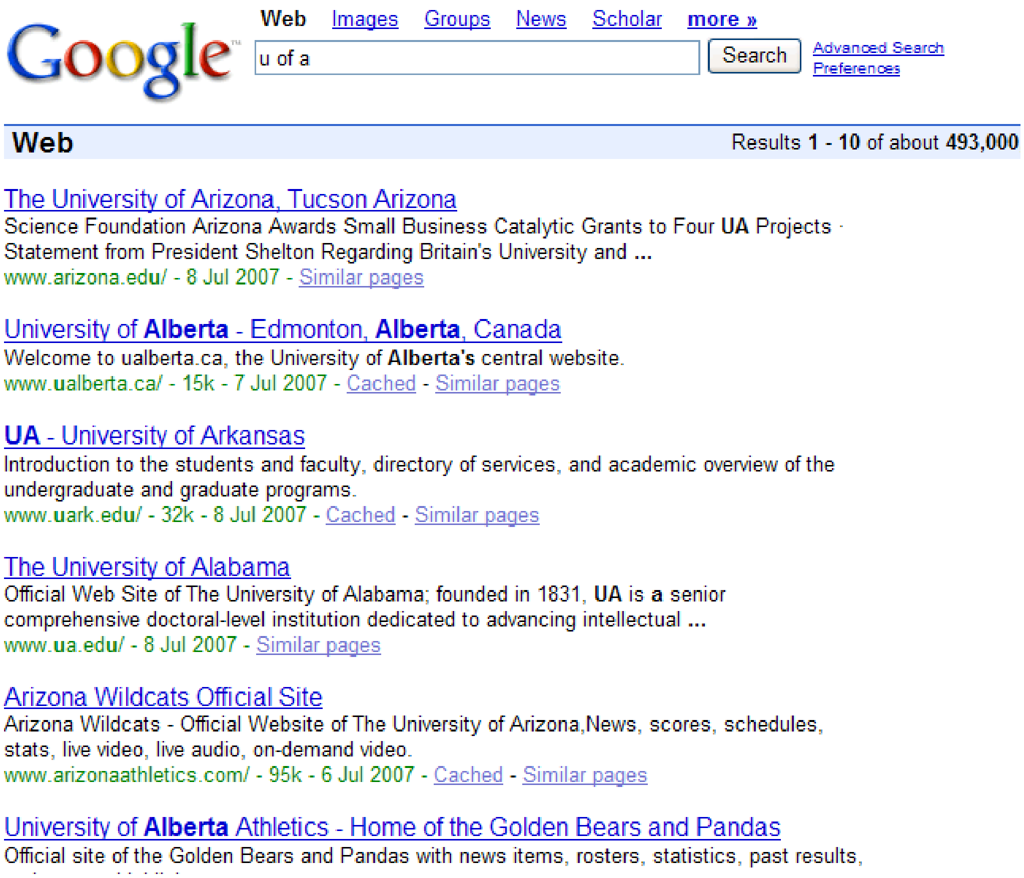
3.2.4. Experiment #(3-4): Linear vector with minimum amount of knowledge

In this experiment, SEPT reads only the result.html file, analyzes it and makes relevant conclusions.

Procedure:

1- We extracted the URLs from the result.html file.

2- We extracted the few lines provided by the search engine as a detail to each link.



Web Results **1** - **10** of about **493,000**

The University of Arizona, Tucson Arizona
Science Foundation Arizona Awards Small Business Catalytic Grants to Four **UA** Projects · Statement from President Shelton Regarding Britain's University and **...**
www.arizona.edu/ - 8 Jul 2007 - Similar pages

University of **Alberta** - Edmonton, **Alberta**, Canada
Welcome to ualberta.ca, the University of **Alberta's** central website.
www.**u**alberta.ca/ - 15k - 7 Jul 2007 - Cached - Similar pages

UA - University of Arkansas
Introduction to the students and faculty, directory of services, and academic overview of the undergraduate and graduate programs.
www.**ua**rk.edu/ - 32k - 8 Jul 2007 - Cached - Similar pages

The University of Alabama
Official Web Site of The University of Alabama; founded in 1831, **UA** is **a** senior comprehensive doctoral-level institution dedicated to advancing intellectual **...**
www.**ua**.edu/ - 8 Jul 2007 - Similar pages

Arizona Wildcats Official Site
Arizona Wildcats - Official Website of The University of Arizona,News, scores, schedules, stats, live video, live audio, on-demand video.
www.arizonaathletics.com/ - 95k - 6 Jul 2007 - Cached - Similar pages

University of **Alberta** Athletics - Home of the Golden Bears and Pandas
Official site of the Golden Bears and Pandas with news items, rosters, statistics, past results,

Figure (3-8): Example of the result.htm file which was used to extract web pages ranking data

3- We analyzed the words located in the web page's details, and extracted them and added them to the retrieved word list.

4- We ran the LVA and assigned a weight to each web page, then presented the new rank to the user.

5- We ran the training algorithm to train the system over different values of examples.

6- We ran the ranking efficiency evaluation algorithm to find out the ranking efficiency.

Figure (3-9) shows the behavior of the LVA when it is exposed to the minimum amount of knowledge experiment. The algorithm is inactive in the absence of a reference page; therefore, the error rate is relatively high. When a reference page was provided, in addition to the absence of training, the error rate dropped from 47.14% to 23.67%. Then, we introduced more examples to the system. This improves the error rate slightly. Thus, using the minimum knowledge method would slightly improve the ranking efficiency.

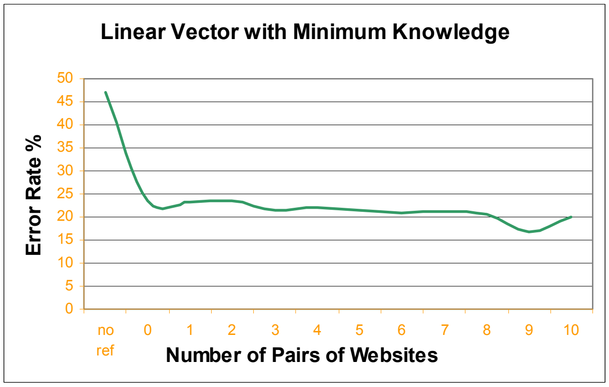


Figure (3-9): The response of the training algorithm for the linear vector space when it was used with the minimum amount of knowledge

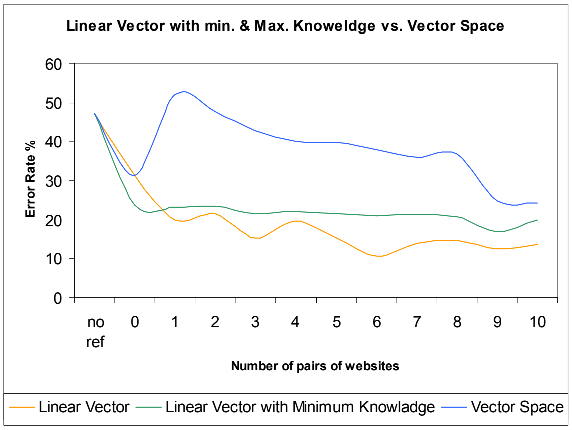


Figure (3-10): Comparing the response of the training algorithm for the LVA vs. VSA when it was used with the minimum and maximum amount of knowledge

Figure (3-10) shows the results of the previous experiments combined together. In this graph we can observe the efficiency of each algorithm. Thus, we can confidently say that the LVA provides greater results and a better error rate, as low as 10.64 %, when it is exposed to a suitable amount of knowledge. However, when the minimum amount of knowledge is provided to the algorithm, the error rate is somewhat higher than the latter method. Certainly, this is an expected result as the error rate would decrease with an increase in the amount of training examples.

The experiments in this chapter show how the LVA provides users with an improved result according to their interest. The study compares the results with an existing algorithm and presents the performance of both algorithms. In the training algorithm, the study concludes that the LVA's response to training is better than the VSA's. Another advantage added to the modified VSA over the existing one is found when we measure the minimum amount of knowledge. The modified VSA requires less data for the training whereas the VSA requires much more data to produce the same result.

Chapter 4

Time Spent on a Web Page is Sufficient to Infer a User's Interest

An important research problem deals with the question how to predict a frequent user's interest in certain types of websites. The motivation is to improve the ranking of the hyperlinks returned by a search engine. This chapter presents the experimental results that indicate that the most important criterion to gauge the user's interest is the time he or she has spent on a given web page.

Section 4.2 defines the problem which we have addressed in this chapter and the performance criteria. Section 4.3 discusses the experimental methodology by explaining how data have been collected. The results are then presented in section 4.4, which show the relation between the time spent on a web page and the user interest of that web page. The discussion and conclusion are the topic of section 4.5.

4.1. Problem and performance criteria

Many factors can be interpreted as an indication of a specific user's interest in a given web page: time spent on a web page, mouse movements, page scrolling, mouse click, etc. What we want to find out here is the importance of the amount of time a user spends viewing a web page. The answer to this question tells us how crucial – as compared to other criteria – it is to include this information in the user's profile. We have conducted three experiments using the inputs of dozens of volunteers experimenting with a transparent experimental set-up. When visiting a web page and spending time viewing its contents, the user had to submit his or her explicit rating for that particular web page. At the same time, the system *implicitly rated* the site based on the log file of the user's behavior.

To respect the user's privacy, the volunteers could switch off the monitoring system any time they wanted to work unsupervised (this might have modestly corrupted the reliability of the results). In order to obtain a sufficient amount of data, the measurements were taken for three months. More than one hundred users participated in the experiments.

4.2. Methodology of the study:

Let us begin by summarizing the hypotheses underlying this part of the research:

1- Time is a significant factor to infer a user's interest in a web page.

2- Factors other than time could be avoided without impairing the accuracy with which the user's interests are predicted.

The data have been collected by a transparent program that monitored the users' behavior while surfing the web. The users were able to turn on the monitoring system and search for a particular interest such as "computer networking", "operating systems" or "communication skills". The monitoring software then created for each user a specific log file that was then searched for such factors as the web page, the starting time and the finishing time. This information gives us the total time a user may spend on a particular web page. Then, the user is asked to rate the web page.

1. Evaluation based on the average time a user spent on a web page:

All in all, about one hundred user profiles have been created. The users rated the visited web pages by a grade from the interval [1,10] where 1 is interpreted as extremely interesting web page and 10 an absolutely uninteresting one. The monitoring software measured the time spent on each Web page, and then calculated the average time spent on each web page a given user has visited. (Note that the user may have visited a web page more than once.) Example 4.1 illustrates the process.

Example 4.1:

Table 4.1 gives an example of the output of the analyzer software. The table gives the average time a particular user spent on each web page. In this case we build Table 4-1 for user #1. WP1 is any web page which was rated 1 by user #1, WP2 is any web page which was rated 2 by user #1, and so on. Each column WPj presents a new web page rate starting from 1 to 10. The ratings for the web pages were submitted by user #1. Each row presents different web page in the same rate according to user #1. If we have two web pages with rate = 1, then we will have the first web page in row #1 and the second web page in row #2. To calculate the average time for the web page rated 1, we average the time that user #1 spent on the particular web page that was rated 1. WP1 for row #1 is not the same as WP1 for row #2. WP1 in row #1 could be www.xzy.com, and WP1 in row #2 could be www.abc.net. The value of 370 in WP1 in the first row is the average time user #1 spent on that particular web page (in seconds). If user #1 visited www.xyz.com five times, then the time spent on www.xyz.com is recorded each time and the average for the five visits is equal to 370 seconds. Then we calculate the average time for all web pages rated 1 as shown in Table 4.1.

Table (4-1): Average time spent in seconds on a web page for user #1

Visit #	WP1	WP2	WP3	…	WP10
1	370	180	180	…	18
2	989	587	98	…	9
3	878	98	78	…	9
…	…	…	…	…	…
10	896	567	89	…	23
Average	712	672	109	…	19

Average time spent on WP1 for user #1 $= \frac{370 + 989 + 878 + \ldots + 896}{10} = 712$ Seconds

Similarly the time spent on WP2 for user #1 $= \frac{180 + 587 + 98 + \ldots + 567}{10} = 672$ Seconds

and so on.

To build a user profile using average time, we built Table 4.1 for each user.

2. Evaluation based on the maximum time a user spent on a web page:

For each user we could now find the maximum time spent on each Web page. A user may visit a Web page more than once. In this case, we found the maximum time a user spent on a web page. In the following example we show how we calculate the maximum time spent on a web page.

Example 4.2:

Table 4.2 presents an example of the data that were found in the analyzer software. These data show that user #1 spent the maximum time in the visited web pages as follows:

WP1 = 476, 789, 767, …, 787 seconds

WP2 = 278, 70, 340, …, 650 seconds

and so on for each Web page.

WP1 in row 1 is not the same Web page as WP1 in row 2. WP1 in row 1 could be www.123.com, whereas WP1 in row 2 could be www.456.com.

Table (4-2): Maximum time spent on a web page for a particular user

Visit #	WP1	WP2	WP3	…	WP10
1	476	278	278	…	27
2	789	670	188	…	19
3	767	340	88	…	15
…	…	…	…	…	…
10	787	650	78	…	29
Average	821	678	127	…	45

To calculate the maximum time for the web page rated 1, we average the maximum time that user #1 spent on all Web pages rated 1. The value of 476 seconds in WP1 in the first row is the maximum time user #1 spent on this particular web page. However, the value of 789 seconds presents the maximum time the user spent on web page www.456.com. Then we calculate the average time for all web pages as shown in Table 4.2.

3. Evaluation based on the minimum time a user spent on a web page:

We found the minimum time a user spent on each web page because the user may visit a web page more than once. Table 4.3 shows an example of the data from the analyzer software. These data show that web pages rated 1 have the following minimum time: 90, 80, 70, …, 34 seconds. Web pages rated 2 have a minimum time spent of 80, 79, 76, …, 23 seconds, and so on for each web page.

Table (4-3): Minimum time spent on different web pages for a particular user

Visit #	WP1	WP2	WP3	…	WP10
1	90	80	79	…	7
2	80	79	68	…	5
3	70	76	23	…	5
…	…	…	…	…	…
	34	23	12	…	3
Average	67	57	34	…	9

We then calculated the average minimum time a user spent on each web page (67, 57, 34, …, 9 seconds).

4.3. Results:

Figure 4.1 shows the average time spent by all users on all web pages from the most interesting web page (WP1) to the least interesting web page (WP10). In this figure we can clearly see how the average time spent on the most interesting web pages is much greater than the average time spent on the least interesting web pages. The average time spent on web pages rated 1 is the longest time spent among all web pages. The figure depicts the relation between time and user-interest as a non-linear relation. Note that the average time spent reading a web page grows with its growing interestingness.

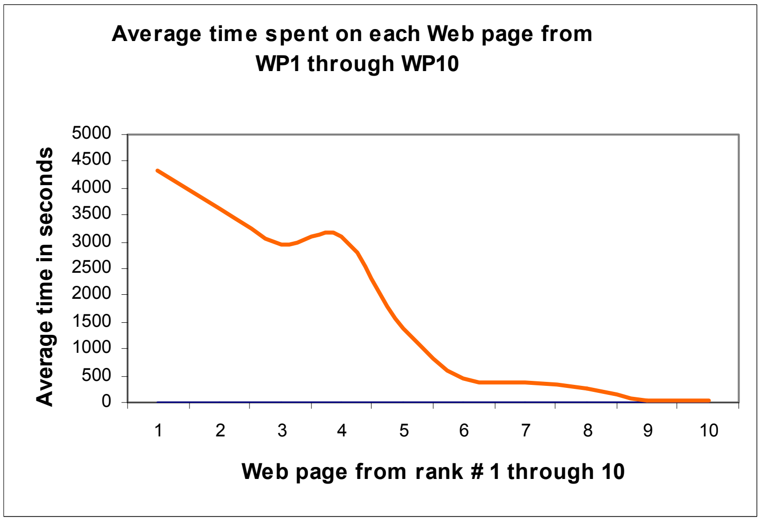


Figure (4-1): Average time a user spent on a Web page vs. the rate of each web page

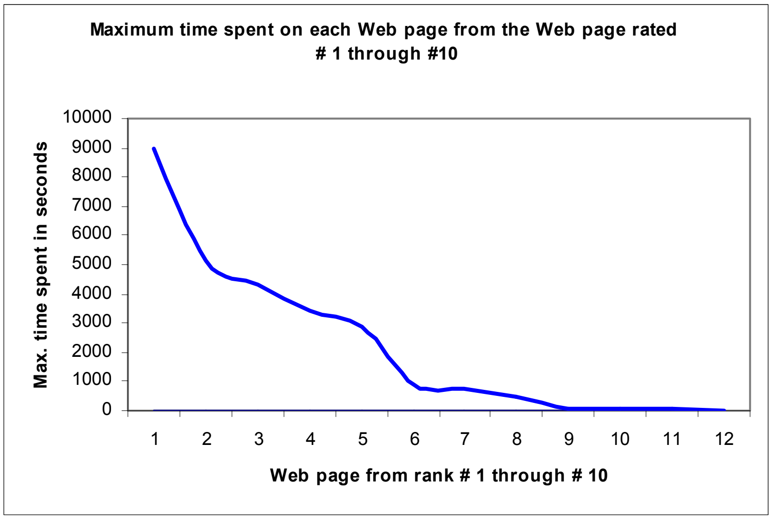


Figure (4-2): Maximum time a user spent on a Web page vs. the rate of each web page

The average maximum time spent by all users is plotted in figure 4.2 where we can clearly see that the maximum time spent on the most interesting web pages is much greater than the maximum time spent on the least interesting web pages. The maximum time spent on the web pages rated 1 is the longest time spent in the system. The figure describes the relation between time and interest as a proportional non-linear relation. As the web page becomes more interesting, the maximum time spent on a web page becomes greater, and as it becomes less interesting, the maximum time significantly reduces.

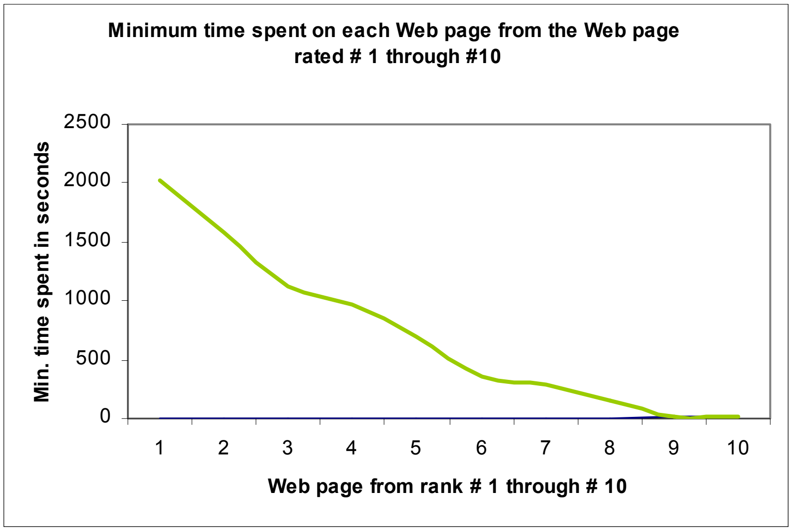


Figure (4-3): Minimum time a user spent on a Web page vs. the rate of each web page

The average minimum time spent by each user is plotted from the most interesting web page ranked 1 to the least interesting web page ranked 10, as shown in figure 4.3. In this figure we can clearly see that the minimum time spent on the interesting web pages is much greater than the minimum time spent on the least interesting web pages. The minimum time spent on the web page rated 1 is the longest time spent among all web pages. The figure describes the relation between time and interest as a proportional non-linear relation. As the web page becomes more interesting, the minimum time spent becomes greater, and as it becomes less interesting, the minimum time becomes tiny.

Figures 4.1, 4.2 and 4.3 indicate that time spent on a web page is a significant when measuring its interestingness. As such it can be used to infer a user's interest in the page. Three different attributes have been considered: the minimum time, the maximum time and the average time. We found that in all the three attributes, the most interesting web pages have the longest time, and the least interesting web pages have the shortest time. The relations are non-linearly proportional to the interest of a web page. Experiments suggest that we can ignore all factors other than time and rely on time only. This would reflect a sufficient result regarding user interest.

Chapter 5

An Algorithm to Evaluate the Efficiency of a Search Engine Ranking

The rapid developments in Internet search engines underline the need for reliable mechanisms for performance evaluation. So far, the vast majority of researchers have relied on the "precision" and "recall" measures known from information retrieval. Unfortunately, both of these measures are in our context somewhat inappropriate. This chapter discusses their shortcomings and proposes a new, better mechanism.

The principle of the tool proposed here is simple. Upon the entry of a user's query, the search engine checks its repository of stored web sites and returns a list of relevant hyperlinks ordered by their predicted relevance to the user's needs. Many mechanisms to assess this relevance have been used in the past, among them keyword frequency, page usage, link analysis, and various combinations of these three. Since each ranking algorithm leads to a different hyperlink ordering, we need to determine which of them is best. In the experiments reported here, a two-pronged strategy has been followed. The first step was to define an appropriate experimental procedure that submits to the search engine well-selected testing queries (to which the correct answers are known). Then, appropriate performance criterion was used to evaluate the quality of the search engine responses to the testing queries.

This chapter focuses on the latter aspect. As discussed earlier, previous research predominantly relied on precision and recall that are commonly used in the field of information retrieval. However, the utility of these metrics for search engine evaluation is limited: precision and recall establish whether the returned list contains the predominantly relevant links, and how many relevant links are missing. What they ignore is whether more relevant links find themselves higher up on the list.

5.1 Proposed mechanism for ranking

Precision and Recall are used to evaluate the efficiency of information retrieval systems. Precision is defined as the ratio of the relevant documents retrieved to the total number of retrieved documents. Recall is defined as the ratio of the relevant documents retrieved to the relevant documents in the database of the system. However, it is infeasible to find the number of relevant documents in the database of a search engine accurately. This makes it impossible to calculate the Recall value precisely. The problem with precision is presented in examples 5.1, 5.2, and 5.3.

$$precision = \frac{\#\text{of r.d.r}}{(\#\text{of r.d.r} + \#\text{of i.d.r})} * 100\% \qquad (5.1)$$

where r.d.r: relevant documents retrieved

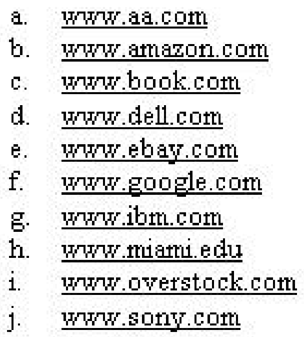
i.d.r: irrelevant documents retrieved

$$recall = \frac{\#\text{of r.d.r}}{\#\text{of r.d.db}} * 100\% \qquad (5.2)$$

where r.d.r: relevant documents retrieved

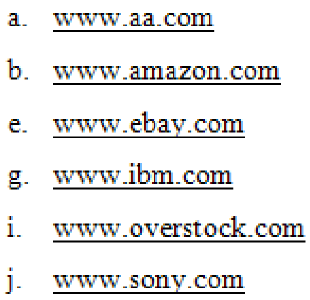
r.d.db: relevant documents in database

Example 5.1: Suppose the retrieved web pages have been ordered as shown in Figure 5.1, assuming that the web pages that are shown in Figure 5.2 are the only relevant documents to the query and the other documents (web pages) in Figure 5.1 are irrelevant.



a. www.aa.com
b. www.amazon.com
c. www.book.com
d. www.dell.com
e. www.ebay.com
f. www.google.com
g. www.ibm.com
h. www.miami.edu
i. www.overstock.com
j. www.sony.com

Figure (5-1): A list of retrieved web pages



a. www.aa.com
b. www.amazon.com
e. www.ebay.com
g. www.ibm.com
i. www.overstock.com
j. www.sony.com

Figure (5-2): A list of the retrieved web pages

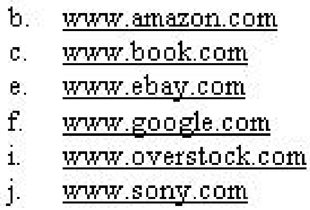
which are relevant to the query for example 5.1

Then

$$precision = \frac{\#\text{of r.d.r}}{(\#\text{of r.d.r} + \#\text{of i.d.r})} * 100\%$$

$$precision = \frac{6}{6+4} * 100\% \ = 60\%$$

Example 5.2: Suppose the documents in Figure 5.3 are the only relevant documents and the other documents that are shown in Figure 5.1 are irrelevant.



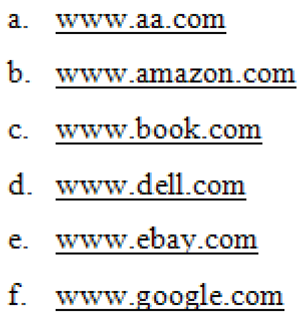
b. www.amazon.com
c. www.book.com
e. www.ebay.com
f. www.google.com
i. www.overstock.com
j. www.sony.com

Figure (5-3): A list of the retrieved web pages
which are relevant to the query for example 5.2

Assuming that the search engine ranks the web pages as presented Figure 5.1, then:

$$precision = \frac{6}{6+4} * 100\% \quad = 60\%$$

Example 5.3: Suppose the web pages that are shown in Figure 5.4 are the only relevant documents and the other documents that are presented in Figure 5.1 are irrelevant:



a. www.aa.com
b. www.amazon.com
c. www.book.com
d. www.dell.com
e. www.ebay.com
f. www.google.com

Figure (5-4): A list of the retrieved web pages
which are relevant to the query for example 5.3

Assuming that the search engine ranks the web pages as shown in Figure 5.1, then

$$precision = \frac{6}{6+4} * 100\% \ = 60\%.$$

With three different sequences, we have a fixed and unchanged precision value. Precision values in the three examples infer that the efficiencies of the three lists are equal which would conclude that the rankings of the three systems are identical. However, Examples 5.1, 5.2 and 5.3 show that we have three diverse results. Our hypothesis asserts that these systems have totally different ranks and they should have different ranking efficiencies.

Principles of operation of SEREET:

As an alternative approach that removes the aforementioned drawbacks, this section describes the *Search Engine Ranking Efficiency Evaluation Tool* (SEREET), which we propose to distinguish among any ranking systems. The purpose of this algorithm is to numerically evaluate the efficiency of a search engine rank.

Definition 5.1:

Let there be m hits and n misses.

Let i, , $1 \le i \le m + n$

represent the position of a website name on the search output list that is hit or missed, and let the position of the j_{th} hit, $1 \le j \le m$

be given by h_j Obviously 1<= h_j <= (m+n), for all j.

Let the W_i denote the weight of the i_{th}. website name, where $1 \le j \le m + n$

Define W_i as follows:

W_i = m + n + 1 - i, if the i_{th} name is a hit.

W_i = 0, if the i_{th} name is a miss.

Then the efficiency of ranking of search engine is given by the expression (5.3):

$$E = \sum_{i=1}^{i=m+n} Wi * \frac{2}{(m+n)*(m+n+1)} * 100\,\% \qquad (5.3)$$

Example 5-4:

Consider there are m=5 hits and n=4 misses in the order shown below:

1. h1
2. m1
3. m2
4. h2
5. h3
6. h4
7. h5
8. m3
9. m4

w_1= m+n+1-1= 5+4 =9

w_2=0

w_3=0

w_4= m+n+1-4 =6

w_5= m+n+1-5=5

w_6= m+n+1-6=4

w_7= m+n+1-7=3

w_8=0

w_9=0

Then

$$E = (\sum_{i=1}^{i=m} Wi) * \frac{2}{(m+n)*(m+n+1)} * 100\ \%$$

$$= (9+6+5+4+3)* \frac{2}{(5+4)*(5+4+1)} *100\%$$

$$= 27* \frac{2}{9*10} *100 = \frac{54}{90} *100\ \% = 60\ \%$$

Example 5-5:

Consider there are m=5 hits and n=4 misses in the order shown below:

1. h1
2. m1
3. m2
4. m3
5. m4
6. h2
7. h3
8. h4
9. h5

w_1= m+n+1-1= 5+4 =9

w_2=0

w_3=0

w_4= 0

w_5= 0

w_6= m+n+1-6=4

w_7= m+n+1-7=3

w_8= m+n+1-8=2

w_9= m+n+1-9=1

Then

$$E = \left(\sum_{i=1}^{i=m} Wi\right) * \frac{2}{(m+n)*(m+n+1)} * 100\ \%$$

$$= (9+4+3+2+1)* \frac{2}{(5+4)*(5+4+1)} *100\%$$

$$= 19* \frac{2}{9*10} *100 = \frac{38}{90} *100\ \% = 42\ \%$$

Example 5-6:

Consider there are m=5 hits and n=4 misses in the order shown below:

1. h1

2. h2

3. m1

3. m2

4. h3

5. h4

6. h5

7. m3

9. m4

w_1= m+n+1-1= 5+4 =9

w_2= = m+n+1-2= 5+4-1 =8

w_3=0

w_4=0

w_5= m+n+1-5=5

w_6= m+n+1-6=4

w_7= m+n+1-7=3

w_8=0

w_9=0

Then

$$E = \left(\sum_{i=1}^{i=m} Wi\right) * \frac{2}{(m+n)*(m+n+1)} * 100\ \%$$

$$= (9+8+5+4+3)* \frac{2}{(5+4)*(5+4+1)} *100\% = 29* \frac{2}{9*10} *100\ \%$$

$$= \frac{58}{90} *100\ \% = 64.44\ \%$$

Lemma 5-1: For integer m > 0,

$$1+2+\ldots+m = \frac{m*(m+1)}{2}$$

Proof:

Let S(m) = 1+2+ … + m (L1.1)

We can rewrite S(m) = m+ m-1+ … +1 (L1.2)

Adding both the sides of (L1.1) and (L1.2),

we get 2*S (m) = m * (m+1)

Hence $S(m) = \frac{m*(m+1)}{2}$

Hence we have proved the lemma.

Claim 5-1:

If there are no misses, the efficiency of the search engine is 100%.

Proof:

If there are no misses, n=0.

Hence Efficiency

$$E = (\sum_{i=1}^{i=m} Wi) * [\frac{2}{(m+n)*(m+n+1)}] * 100\ \%$$

Hence E= $$(\sum_{i=1}^{i=m} Wi) * [\frac{2}{(m)*(m+1)}] * 100\ \% \qquad (T1.1)$$

Also, $Wi = m+1-i$

Hence from (T1.1), Efficiency

$$E = (\sum_{i=1}^{i=m} Wi) * [\frac{2}{(m)*(m+1)}] * 100\ \% \qquad (T1.1)$$

$$= (m+1-1+m+1-2+...+m+1-m)*[\frac{2}{(m)*(m+1)}]*100\%$$

$$= (m+m-1+m-2+...+1)*[\frac{2}{m*(m+1)}]*100\%$$

$$= \frac{m*(m+1)}{2} * \frac{2}{(m)*(m+1)} * 100\%$$ from Lemma 5-1.

= 100%.

Claim 5-2:

Let there be m hits, m>0, and n misses, n>0 for two website name lists W_1 and W_2. W_1 is such that it has its hits only positions i , $1 \leq i \leq m$. W_2 is such that it has one miss in position M, $1\leq M \leq m$, all other hits in position i , $1 \leq i \leq m$ and one hit in position H, $m < H \leq (m + n)$. Then E1 >E2

where E_1 and E_2 are efficiencies for W_1 and W_2, respectively.

Proof:

We have

$$E_1 = \sum_{i=1}^{i=m} Wi * \frac{2*100}{(m+n)+(m+n+1)}$$

$$= K . \sum_{i=1}^{i=m} W_i + W_M$$

$$\text{where } K = \frac{2*100}{(m+n)+(m+n+1)} \qquad (5.2.1)$$

$$\therefore E_1 = K[\sum_{\substack{i=1\\ i\neq M}}^{i=m} Wi + (m+n+1-m)] \qquad (5.2.2)$$

$$\text{Now } E2 = K * \sum_{\substack{i=1\\ i\neq M}}^{i=m+n} Wi$$

$$= K[\sum_{\substack{i=1\\ i\neq H}}^{i=(m+n)} W_i + W_H]$$

$$= K[\sum_{i=1}^{i=m} Wi + W_H] = K\sum_{\substack{i=1\\ i\neq M}}^{i=m} Wi + W_M + W_H$$

$$= K[\sum_{\substack{i=1 \\ i \neq M}}^{i=m} Wi + 0 + (m + n + 1 - H)]$$

$$= K[\sum_{\substack{i=1 \\ i \neq M}}^{i=m} Wi + (m + n + 1 - H)] \qquad (5.2.3)$$

Observe that H > m > M in (5.2.3)by hypothesis. Hence comparing (5.2.1) and (5.2.2) in view of the fact (5.2.3), we have $E_1 > E_2$

Claim 5-3

Let there be m hits ,m>=0 and n misses, n>=0 for two website name lists W_1 and W_2. W_1 has all the hits in positions $1, \le i \le m,$

and all the misses in positions j, $m \le j \le (m + n)$

W_2 has p misses in positions k , $1 \le k \le m$ and $0 \le p \le m$

and (n-p) hits in positions l , $1 \le l \le m$, Then $E_1 > E_2$.

Proof

We will prove this claim by induction.

1. By Claim 2, the hypothesis is true for p=1.
2. Assume the hypothesis is true for p = e, e ≤ (m-1)

let E_e be the efficiency of this website name list denoted by W_e .

Then $E_1 > E_e$ (5.3.1)

let us exchange the positions of one hit at position s, s<m and one miss at t , t>m.

Thus now we have a new website name list W_{e+1} such that it has

e +1 misses in positions k , $1 \le k \le n$

Let E_{e+1} be the efficiency of W_{e+1}

Then

$$E_{e+1} \quad = \quad \sum_{i=1}^{m+n} Wi \; * \; \frac{2}{(m+n)+(m+n+1)} * 100$$

$$= \text{K} \; [\sum_{\substack{i=1, \\ i \neq s, \\ i \neq t}}^{i=m+n} Wi \; + Ws \; + Wt \;]$$

where $\text{K} = \; [\frac{2 * 100}{(m+n) * (m+n+1)}]$

Hence $E_{e+1} \quad = \text{K} \quad [\sum_{\substack{i=1, \\ i \neq s, \\ i \neq t}}^{i=m+n} Wi \; + Ws \; + Wt \;]$

$$= \text{K} \quad [\sum_{\substack{i=1, \\ i \neq s, \\ i \neq t}}^{i=m+n} Wi \; + 0 + (m + n + 1 - t)] \qquad (5.3.2)$$

Since s<n and t>m , we have s<t. (5.3.3)

Also , $E_e \; = \text{k} \quad [\sum_{\substack{i=1, \\ i \neq s, \\ i \neq t}}^{i=m+n} Wi \; + Ws \; + Wt \;]$

$$= \text{k} \; [\sum_{\substack{i=1, \\ i \neq s, \\ i \neq t}}^{i=m+n} Wi \; + Ws \; + Wt \;]$$

$$= \text{k} \; [\sum_{\substack{i=1, \\ i \neq s, \\ i \neq t}}^{i=m+n} Wi \; + 0 + (m + n + 1 - s)] \qquad (5.3.4)$$

Comparing (5.3.2) and (5.3.4) and using (5.3.3), we have $\text{E}_{\text{e}} > \text{E}_{\text{e+1.}}$ (5.3.4)

Hence the hypothesis is true for p=e+1.

5.2 The Average Normalized Modified Retrieval Rank (ANMRR):

ANMRR is an algorithm which can precisely evaluate efficiency of a retrieved list. ANMRR is being used by scientists in the information retrieval field. The limitation of the ANMRR algorithm is shown in this chapter. The ANMRR [65] algorithm can be used to evaluate a rank of retrieved documents according to equations (5.4) and (5.5), given below:

$$\text{ANMRR}= \frac{1}{Q * \sum \text{NMRR}(q)} \qquad (5.4)$$

$$\text{NMRR}(q)= \frac{\text{MRR}\ (q)}{(k + 0.5 - 0.5 * \text{NG}\ (q))} \qquad (5.5)$$

where NG(q) is the number of ground truth documents exist in the database of a search engine, Q is the number of queries, K is a value found by the following equation:

K=min(4NG(q), 2GTM) where GTM is max{NG(q)}

Because we do not have a precise number of relevant document in the database of a search engine NG(q), it is impossible to evaluate NMRR and consequently it is infeasible to evaluate ANMRR. This is a good reason to eliminate the ANMRR as an evaluation tool for search engine ranking.

Example 5.7:

If we have 20 retrieved images, 4 of which are relevant to the query and 16 are irrelevant, and if we know their ranks in the retrieved list, then we can evaluate the efficiency of the retrieval system using SEREET. However, with ANMRR we need to know how many ground truth images NG(q) are there in the database. Suppose NG(q) = 9. So we need this value to evaluate the system using ANMRR. In this chapter we introduced the SEREET tool and showed that SEREET is a totally different tool than the exiting algorithms such as precision and recall or ANMRR.

A close look at the evaluation tools previously used in search engine ranking reveals that they do not adequately address their needs. In particular, they only look at the presence of a document, but not at its ranking. By contrast, the Search Engine Ranking Efficiency Evaluation Tool (SEREET) introduced here provides a new way to precisely measure the ranking efficiency. This tool is very sensitive to changes in the order of documents as shown in Examples 5.4, 5.5 and 5.6. In these examples, we found that when the order of a document changes, the overall evaluation changes. This algorithm provides an accurate evaluation values. If few relevant documents are in the top of the rank and few are in the bottom of the rank, the algorithm averages their values and gives a precise weight to the rank. Moreover, if the relevant documents are clustered in the middle, it will have a value less than if they are in the top. If we have two different ranking algorithms with the same precision, we still can favor one over the other using SEREET. The design of the SEREET took into consideration space and time complexity. The worst case is executed in O(k) where k is the number of relevant documents retrieved. ANMRR needs more information infeasible to find in search engines. ANMRR is a powerful tool in other retrieval system efficiency evaluation applications, where such information can be found. However, with search engines, ANMRR is not suitable. For these reasons, all of the existing tools do not satisfy the requirements. Consequently, they are not qualified to evaluate the rank of a search engine. Search Engine Ranking

Efficiency Evaluation Tool (SEREET) provides a unique tool to precisely measure the ranking efficiency.

Chapter 6

A Fully Automated Recommender System

Using a Filtering Technique

The research reported in this dissertation focused on the development of (and experimentation with) a mechanism that combines traditional collaborative filtering approaches with explicit rating. This chapter describes the technique and illustrates its behavior by experiments with a test-bed created from real-world data collected in the course of this research. The results are most promising.

Let us start by a brief discussion of the field *collaborative filtering* that studies algorithms capable of inducing mechanisms to provide predictions of how a specific user might rate a given item offered by an e-commerce web site, be it a book, a movie or music. The prediction is based on the analysis of a database of previous ratings of the given item by customers of diverse preferences. For instance, a company that is going to release a new product (P) needs an estimate as to which customer segment would tend to buy P, so as to be able to design a marketing campaign focused on this particular segment. The situation is depicted in Figure 1. Suppose that the customers have been asked to rate the company's products by an integer from the interval [1, 5], where 5 is the highest grade. If many members of a given customer segment have rated the product (P) 5 out of 5, then other customers from this segment are also likely to have a positive opinion about P. Based on this assumption, the company will send the brochures primarily to this particular group.

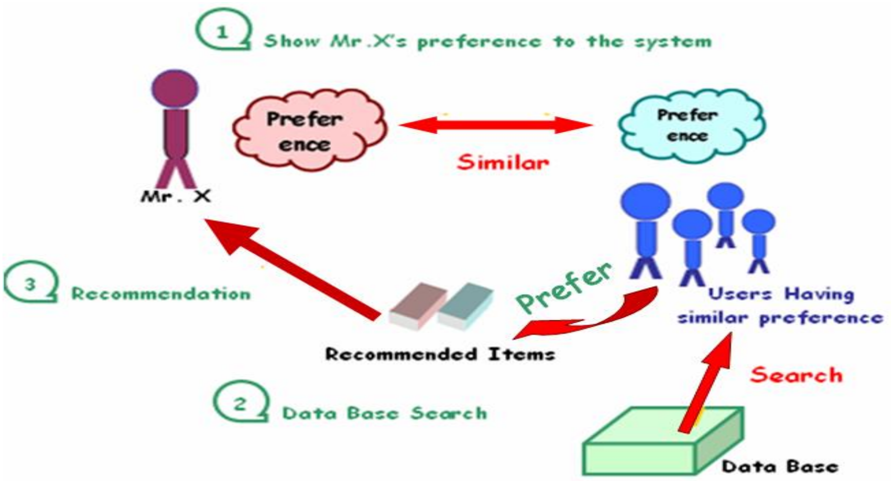


Figure (6-1): Collaborative filtering (adopted from www.kamishima.net)

The early stage of the work consisted in the collection of enormous amounts of data from more than one hundred volunteers – users of web services in ten distinct locations. These data were compiled using a filtering technique to find out whether the outputs of these techniques could be further improved. Section 6.2 introduces the problem and performance criteria. We pay special attention to studies of methods to exploit user profiles and of methods to induce a computer representation of this profile from historical data. Our methodology and experimental set-up are the topic of Section 6.3. The results of our experiments are summarized in Section 6.4, and the discussion is presented in Section 6.5. Our conclusions and some suggestions for future work are offered in Section 6.6. The historical background of collaborative filtering was presented in Chapter 2.

6.1 Problem and performance criteria

An important question is how to find a web page related to a topic that matches user-specific interests. We use the filtering technique which identifies a concrete web page that has been strongly recommended by users in the same user segment (also called "cluster"). The performance evaluation is based on a database of user reports that detail the users' satisfaction with the results. This information on satisfaction has been obtained by recording the ratings given by "test" users to the proposed web pages. The goal of the research is to implement a fully functional automated recommender system based on the a filtering technique and implicit rating method (where a system rates the web page based on the user's behavior).

6.2 Methodology

More than one hundred users voluntarily participated in the experiments that ran for about six months. The participants were divided into three groups, and each group was subdivided into two subgroups. The users were mutually unrelated. Some of them (but not all) worked in similar companies and they came from ten distinct geographical locations, but they all had backgrounds in computer science, computer engineering and information technology. They were provided with three topics from which each user chose one. Thus we had three groups, one for each topic. The topics were selected so as to make sure that the users did not have major experience in them: socket programming (computer networking), k-means classifier (machine learning) and helpdesk specialist (computer maintenance). Each group was asked to search the web for information necessary to make them understand their selected topic. Each subgroup consisted of 12-

20 participants and each participant then provided explicit ratings for the visited web pages. The ratings were collected and analyzed carefully by a computer program and are presented clearly in Section 6.4. Table 1 shows an example of data presented by a user.

Table (6-1): User/web page rate

Users	Web pages			
	Web page 1	Web page 2	…	Web page n
User 1	2	3	…	1
User 2	2	4	…	2
…	1	4	…	2
User n	1	3	…	2

Data collection:

In the data collection phase, we collected data from all users who belonged to a single subgroup into one table then repeated the process for all subgroups. We had six subgroups. For each subgroup, we took the average of all users to find the subgroup rating average. The output in most cases was a fraction: in this case, we rounded it to the nearest integer. For instance, we had the following ratings for a web page from 18 different users who belonged to a single subgroup: 2, 2, 1, 1, 1, 1, 2, 2, 2, 3, 2, 2, 1, 1, 2, 3, 3, 1.

The average rate $= \frac{32}{18} = 1.78$

Round (1.78) = 2

Therefore, the group rating for this web page is equal to 2.

We took this web page and recommended it to their peer group (the subgroup working on the same topic). Then we asked the new subgroup to rate the same web page explicitly. Table 2 shows the true data for clarification.

Table (6-2): Data fed to the filtering system

Subgroups	Web pages					
	W.P1	W.P2	W.P3	W.P4	W.P5	W.P6
Subgroup 1	1	1	2	5	3	2
Subgroup 2	2	1	1	4	4	2

6.3 Results

Figure 2 illustrates the results of the two subgroups by visualizing data found in a table similar to Table 2. The first subgroup visited ten different web pages and submitted their explicit ratings. Then we asked the second subgroup to visit the same web pages and submit their ratings explicitly. We noticed that the ratings for web pages 3, 4, 6, 7, 8 and 9 in both subgroups were identical. Little variation was found in web pages 1, 2, 5 and 10. From this graph we see that the two curves are matched at many points. However, we find few mismatched points in some web pages. Although there are some mismatches, they are minimal; the mismatches between the two curves were only one point away from each other.

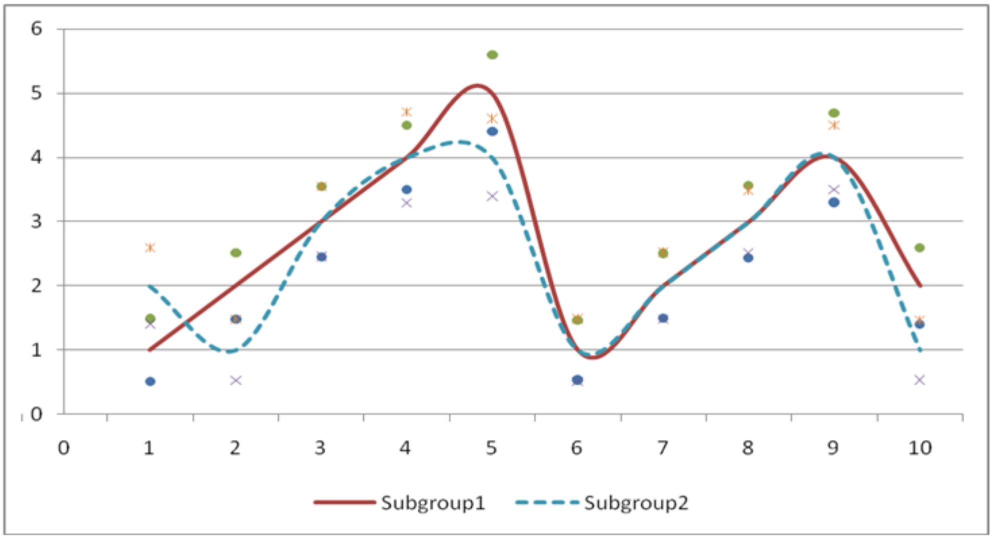


Figure (6-2): Experiment #1, the mean and standard deviation of the rating of each web page according to subgroup#1 and subgroup#2

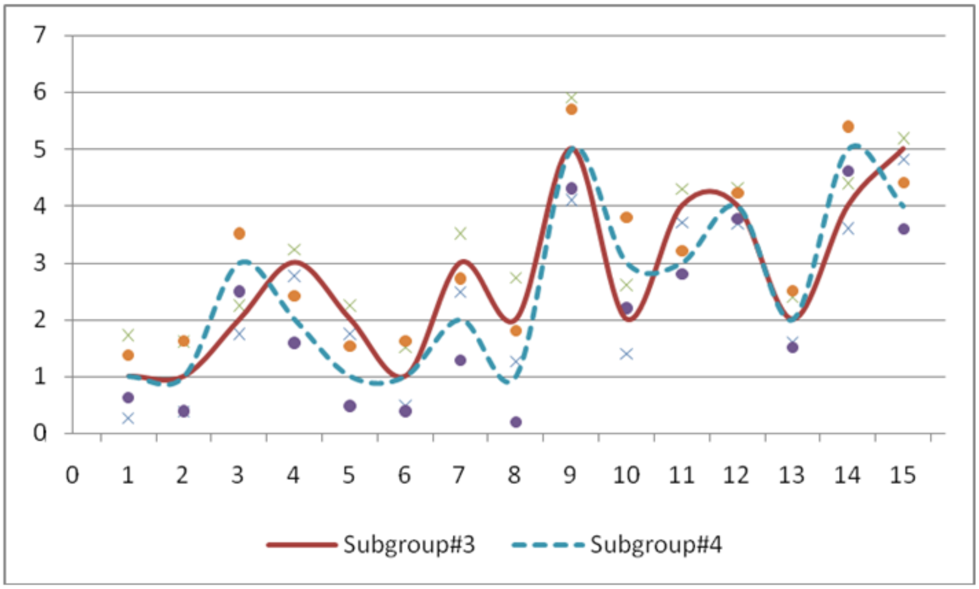


Figure (6-3): Experiment #2, the mean and standard deviation of the rating of each web page according to subgroup#3 and subgroup#4

In the experiment #2, we extended the experiment #1 with subgroups 3 and 4. We requested the ratings for 15 web pages. The plot of the results shows that there is no more than one point distance as a maximum mismatch. This is shown in Figure 3. We extended the experiment #2 with subgroups 5 and 6. Members of subgroup 5 were asked to rate 30 web pages, and those web pages were then presented to members of subgroup 6. Figure 4 shows that the ratings of both the subgroups were similar at many points. However the mismatches were again no more than one rating point.

In the experiment #4, we used an implicit rating where users were not required to submit any explicit rating manually. The system estimated the ratings automatically and was provided with a user inference engine (a plug-in software) which monitored the user's behavior according to our paper [43]. The software focused only on time spent on a web page. The system evaluated the ratings according to user interest (refer to [43] for more details). Implicit rating was used to build tables similar to Tables 1 and 2.

A fully automated filtering system was explored in the experiment #5. In this experiment the users of subgroup 1 visit the web pages regularly. The system now will rate the web pages visited by all users of subgroup 1. The system rates these web pages automatically with the implicit rating method that depends on the inference engine. Then, we proposed these web pages to the subgroup 2. We requested the subgroup 2 to rate these web pages explicitly. We found a larger number of mismatches, in this case. Moreover, the mismatch was quite large. We found a distance of 3 points which reflect a mismatch in the preferences. We refer this mismatch to the lack of efficiency of the inference engine. This means that if we have an efficient inference engine to precisely predict the user interest, the result would be improved substantially. We also found a proportional relation between the number of mismatches and the number of web pages in the recommender system.

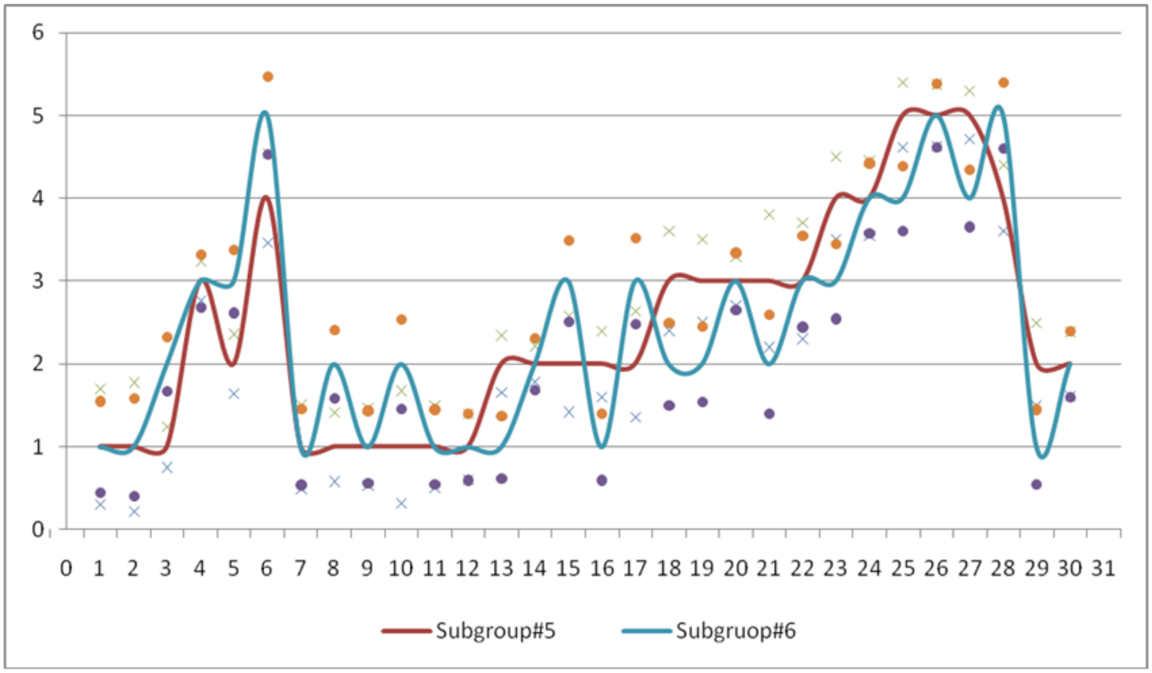


Figure (6-4): Experiment #3, the mean and standard deviation of the rating of each web page according to subgroup#5 and subgroup#6

6.4 Discussion:

One might expect a growth in the number of mismatches as the number of web pages grows. As we saw in Figure 5, a fully automated filtering system was able to infer specific user interest and to generate the corresponding rate implicitly based on the user profile. In this figure, we can clearly see a larger variation in the rating between the two curves. This variation could be due to the lack of efficiency in the implicit rating system. In this case, with a more accurate system, the two curves would merge and we would expect a maximum distance between the two to be no more than one rating point distance as in the explicit rating method in Figures 2, 3 and 4. However, we can still find distances of three points with the existing implicit rating system, but only at a few points. More points and experiments would definitely polish the result and construct a more vivid picture of the error rate in using the implicit rating recommender system.

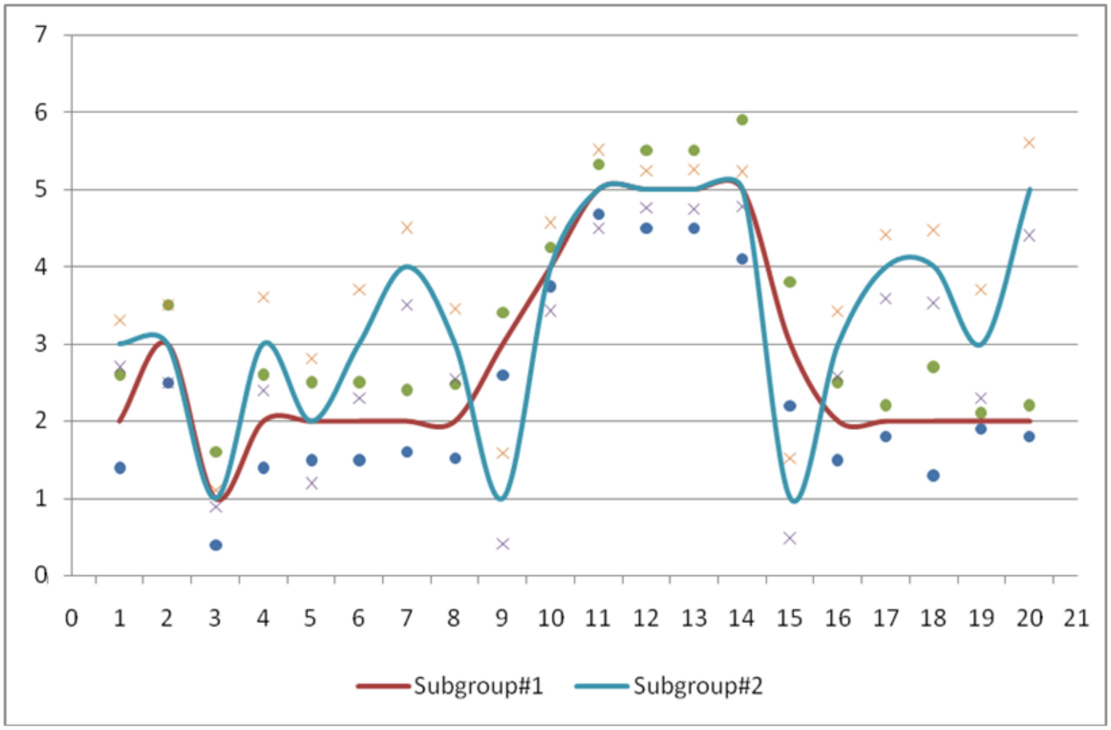


Figure (6-5): Experiment #4, the mean and standard deviation of the rating of each web page according to subgroup#1 and subgroup#2 using implicit rating

Based on these experiments, and on the results from similar studies published in existing literature, we conclude that the developed technique is indeed likely to benefit recommender systems on the web. What would remain to be done is the addition of an efficient inference engine and a user profile builder. The system described in the reported study inherits the advantages of the collaborative filtering recommender system such as non-textual web pages. With non-textual web pages, the filtering system would be superior to other recommender systems as it ignores text content.

Chapter 7

Conclusions and Suggestions for Future Works

This dissertation has contributed to the field of search engine personalization and its ranking algorithms. The contributions are as follows:

(1) Linear vector algorithm which was designed to assign weights to the retrieved documents according to their relevance to the submitted query. It is a new ranking algorithm which was designed as a modification to the widely used vector space algorithm. The goal in designing this algorithm is obtain a fast algorithm with acceptable performance.

(2) Evaluation of the time spent in a web page is an effective method to infer a user interest. The time spent on a web page could be used to infer user interest of a web page. A complete study was provided to address this aspect. From the experiments we found that the time spent on a web page is enough to infer user interest. Other factors could be eliminated without harming efficiency.

(3) We developed a new algorithm to numerically evaluate the ranking efficiency of a search engine. Precision and recall are commonly used in the field of information retrieval. However, the utility of these metrics for search engine evaluation is limited: precision and recall establish whether the returned list contains predominantly relevant links. Precision and recall ignore the position of the link in the list. This dissertation provides the tool to numerically evaluate a search engine ranking. With this tool we can compare the output of several queries or search engines.

(4) We presented a method for testing the filtering technique and evaluating its performance in recommending a web page to a user. Experiments with real data illustrate the behavior of the developed search engine personalization system and automatic recommender system. All of these observations suggest that we have proposed a unique solution to the problems associated with search engine personalization, search engine ranking, inference engine and web page recommender system. A filtering system is used in the recommender system to recommend an item to a user. The system was used in recommending items such as movies, books, and music. We have conducted several experiments in web pages recommendation. Results showed that filtering system is a powerful tool and can be implemented as a web pages recommender system.

The introduction to this dissertation listed seven research questions and five research goals. Of these, five questions have been answered here in terms of a novel solution.

7.1 Linear Vector Algorithm:

The field of internet search engines is a dynamic research area, full of challenging issues and questions that still wait for systematic investigation. The linear vector algorithm (LVA) is an efficient mechanism that can be used to rank the hyperlinks returned by a search engine. The basic principle of operation is quite simple. A user first selects several interesting web pages as his or her positive examples and other uninteresting web pages as negative examples. Based on these, the system builds a unique profile for each user. A search engine returns the retrieved documents and orders them according to their estimated relevance to the submitted query. Our personalization tool uses the user profile to rearrange the documents to better reflect the perceived user’s interests and preferences.

The experiments presented in Chapter 3 show how the LVA provides users with an improved document ranking. The study compares the results with an older algorithm, VSA, and compares them with those achieve by LVA. Experiments indicate that the latter responses to the training much faster, and offers better ranking than the former.

7.2 Time Spent on a Web page is Sufficient to Infer a User's Interest

An important question is how to predict a frequent user's interest in certain types of web site. This part of the research reported here has presented the results of experiments that indicate that the most important criterion here is the time the user has spent on a given web page.

The user uses the Internet regularly. The system starts to build a user profile. A brief overview of previous work indicates that several mechanisms for building this profile by regular observations of user's behavior have already been suggested. The research reported here focused on the time spent by a user on a web. Our experiments show that this time can be used to infer a user's interest in a web page The experiments examined three different time attributes and concluded that time can be easily recruited to infer a user interest in a web page. The experiments presented a non-linear relation between time and user interest of a web page.

7.3 An Algorithm to Numerically Evaluate the Efficiency of a Search Engine Rank:

As for the mechanisms for performance evaluation, the vast majority of scientists have relied on the "precision" and "recall" measures known from the field of Information Retrieval. Unfortunately, both of these measures fail to measure how successful the ranking of the returned documents is according to their relevance to the search query. Precision evaluates the number of relevant documents retrieved from the database to number of irrelevant documents retrieved from the database. This assessment does not pay any attention to the location in which the retrieved documents are sorted. For instance, if the relevant documents were in the top of the list or the bottom of the list, the precision value does not change. In this case, we cannot differentiate between two lists if the first list has some relevant documents sorted in the top of the rank and the other list has the same number of relevant documents, but they were sorted in the bottom of the list. Recall evaluates the number of relevant documents retrieved to the number of relevant documents in the database. However, it is almost impossible to evaluate the number of relevant documents in the database, a circumstance that almost eliminates recall as an assessment tool.

7.4 A Fully Automated Recommender System Using a Filtering Technique

Recommender systems have enjoyed significant attention from the community that seeks to develop algorithms to optimize the performance of recommender systems based on historical data. This part of the dissertation focused on a mechanism that combines traditional filtering approaches with explicit rating. The task of the filtering technique is used to provide predictions as to how a specific user might rate a given item offered by an e-commerce web site, be it a book, movie, or music. The prediction is based on the analysis of a database with previous rating by users.

The research started by collecting massive amount of data from more than a hundred users that used web services in ten distinct locations. These data were compiled using a filtering system. Experimental results with the developed recommender system strongly indicate the utility of using a filtering system to this end. Filtering can facilitate relatively accurate. Figure 5 presented satisfactory results of a fully automated filtering system. In a system to measure the user interest in a web page, we would expect a maximum distance between the two curves to be no more than one rating point distance as it was in the explicit rating method in Figure 2, Figure 3 and Figure 4. However, we still can find distances of 3 points with the existing implicit rating system, but it is only in few points.

Based on this model, we recommend a similar fully automated filtering system to be implemented as a recommender system for web page recommendation and therefore for search engines. Adding inference engine and user profile builder would fully automate the system. This system inherits the advantages of collaborative filtering recommender system such as non-textual web pages. With non-textual web pages, collaborative filtering would be superior according to other recommender systems as it ignores text content.

We implemented the LVA algorithm (ch3) on text retrieval, we would recommend implementing this LVA algorithm on non-textual information. We would also recommend farther experiment on inference engine ch4 to improve the inference quality. Such improvement would result in unique search engine. We would also suggest building our own database rather than implementing the system on an on-the-shelf search engine.

Bibliography

[1] W. Fan, M.D. Gordon, P. Pathak. "Personalization of search engine services for effective retrieval and knowledge management," *In Proceedings of the 2000 International Conference on Information Systems (ICIS), 2000, Brisbane, Australia.*

[2] Boyan, J. A., D. Freitag and T. Joachims, "A Machine learning architecture for optimizing web search engines," *In Proceedings of the AAAI workshop on Internet-Based Information Systems, AAAI Technical Report WS-96-06, 1996*

[3] Eugene Agichtein, Eric Brill and Susan Dumais, "Improving Web Search Ranking b Incorporating User Behavior Information," *29th Annual., International., ACM SIGIR, 2006.*

[4] Philippe Poinçot, Soizick Lesteven, and Fionn Murtagh, "Comparison of two document similarity search engines," *Library and Information Services in Astronomy III, ASP Conference Series, Vol. 153, 1998*

[5] S. Chakrabarti, B.Dom, D. Gilbson, J. Kleinberg, S. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "Mining the link structure of the world wide web," *IEEE Computer, 32(8): 60-67, 1999.*

[6] D. Maltz and K. Ehrlich, "Pointing the way: Active collaborative filtering," *Proceedings of the Conference on Human Factors in Computing Systems CHI'95, New York, 1995. ACM.*

[7] H. Ahonen, O. Heinonen, M. Klemettinen, and A. Verkamo, "Finding co-occurring text phrases by combining sequence and frequent set discovery," *In R. Feldman, editor, Proceedings of 16th International Joint Conference on Artificial Intelligence IJCAI-99 Workshop on Text Mining: Fiundations, Techniques and Applications, page 1-9, 1999.*

[8] Hypung R. Kim and Philip K. Chan, "Learning Implicit User's interest Hierarchy for Context in Personalization," *In Proceedings of the 8th international., conference on Intelligent user interfaces, Miami, Florida, USA, 2003, pp. 101 - 108*

[9] Bracha Shapira, Meirav Taieb-Maimon and Anny Moskowitz, "Study of Usefulness of Known and New Implicit Indicators and Their Optimal Combination for Accurate Inference of Users Interest," *In Proceedings of the 2006 ACM symposium on Applied computingPages: 1118 – 1119, Year of Publication: 2006*

[10] Clement Yu, King-lup Liu, Zonghuan Wu, and Naphtali Rishe, "A Methodology to retrieve text documents from multiple databases," *IEEE Transactions on knowledge and data engineering, Vol 14, No. 6, page 1347-1361, Nov/Dec 2002.*

[11] Martin E. Muller, “An Intelligent multi-agent architecture for information retrieval from the Internet,” *Technicalreport,UniversityofOsnabruk,1999.*

[12] S. K. Madria, S. S. Bhowmick, W. K. Ng, and E.-P. Lim, “Research issues in web data mining,” *In Proceedings of Data Warehousing and Knowledge Discovery, First International Conference, DaWaK ’99, Pages 303-312, 1999.*

[13] Oard, D., & Kim, J, “Implicit feedback for recommender systems,” *In Proceedings of the AAAI workshop on recommender systems, Madison, WI, USA , 1998, pp. 80–82.*

[14] Nichols, D, “Implicit Rating and Filtering,” *In Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering, Hungary, 1997, 31-36.*

[15] Shian-Hua Lin; Meng Chang Chen; Jan-Ming Ho; Yueh-Ming Huang, “ACIRD: intelligent Internet document organization and retrieval,” *IEEE Transactions on Knowledge and Data Engineering, Volume: 14, Issue: 3, Page(s): 599-614, May/Jun 2002.*

[16] Dion H. Goh and Rebecca P. Ang, “Relevancy Rankings – Pay for Performance Search Engines in the Hot Seat,” *Online Information Review, 27(3), 87-93.*

[17] Taher H. Haveliwala, “Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search*,” IEEE Transactions on Knowledge and Data Engineering, 15(4):784-796. IEEE, August 2003.*

[18] Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J.L., Gordon, L.R. and Riedl, J, “Applying Collaborative Filtering to Usenet News,” *Communications of the ACM, 40(3), 77-87.*

[19] Hill, W.C. and Terveen, L, “Using frequency-of-mention in public conversations for social filtering,” *In Proceedings of the ACM Conference on Computer Supported Cooperative Work(CSCW'96), Cambridge, MA, ACM PRess, 106-12.*

[20] W. C. Hill, J. D. Hollan, D. Wroblewski and T. McCndless, “Edit Wear and Read Wear*,” In Proceeding of ACM CHI Conference on Human Factors in Computing Systems, pages 3-9, 1992.*

[21] Stevens, C, “Knowledge-based assistance for accessing large, poorly structured information spaces,” *Ph.D. thesis, University of Colorado, Department of Computer Science, Boulder, (1993)*

[22] Kim, H. and Chan, P. K, “Implicit indicator for interesting web pages,” *International Conference on Web Information Systems and Technologies, 2005, 270-277.*

[23] Jung K, “Modeling Web User's interest with Implicit Indicators,” *Master Thesis, Florida Institute of Technology.*

[24] Claypool, M., Le, P., Wased, M., and Brown, D, “Implicit Interest Indicator,” *In Proc. 6th international conference on Intellegent User Interfaces, 33-40, 2001*

[25] Goecks, J. and Shavlik, J, “Learning users’ interests by unobtrusively observing their normal behavior,” *In Proceedings of the 5th international conference on Intelligent user interfaces, 129-132, 2000.*

[26] Middleton, S. E., Shadbolt, N. R. and De Roure, D. C, “Capturing Interest through Inference and Visualization: Ontological., User Profiling in Recommender Systems,” *In Proceedings of the Second Annual., Conference on Knowledge Capture 62-69, 2003.*

[27] White, R.W., Jose, J.M. and Ruthven, I, “The use of implicit evidence for relevance feedback in Web retrieval,” *In Proceedings of 24th ECIR Conference, 93-109. 2002.*

[28] Lieberman, H, “Letizia: An Agent That Assists Web Browsing,” *In Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI ‘95) Montreal, Canada, 20-25. August (1995)*

[29] Rajiv Badi, Soonil Bae, J. Michael Moore, Konstantinos Meintanis, Anna Zacchi, Haowei Hsieh, Frank Shipman and Catherine C. Marshall, “Recognizing User's interest and Document Value form Reading and Organizing Activities in Document Triage,” *International Conference on Intelligent User Interfaces archive, In Proceedings of the 11th international conference on Intelligent user interfaces. Sydney, Australia Pages: 218 – 225, 2006*

[30] Grudin, J, “Groupware and Social., Dynamics: Eight Challenges for Developers,” *Communications of the ACM, 35 : 92–105, 1994.*

[31] Chan, P. K, “A non-invasive learning approach to building web user profiles,” *In B. Masand & M. Spiliopoulou, editors, Proceedings of the Workshop on Web Usage Analysis and User Profiling (WEBKDD’99).*

[32] Sarwar, B., Konstan, J., Borchers, A., Herlocker, J., Miller, B., Reidl, J, “Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System,” *In Proceedings of ACM Conference on Computer Supported Collaborative Work (CSCW), 1998, pp. 345-354.*

[33] Alhalabi W., Kubat M, Tapia M, “A Search Engine Personalization Tool,” Maser Thesis, The University if Miami, June 2004.

[34] Morita, M., & Shinoda, Y, “Information filtering based on user behavior analysis and best match text retrieval,” *In Proceedings of the 17 th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 272-281, 1996.*

[35] D. W. Oard and J. Kim, “Modeling information content using observable behavior,” *In Proceedings of the 64th Annual Conference of the American Society for Information Science and Technology, pages 481–488, Washington, 2001.*

[36] Pazzani, M. A “Framework for Collaborative, Content-Based and Demographic Filtering,” *Artificial Intelligence Review. 13(5-6) 393-408, 1999*

[37] Miha Gr è ar, “User Profiling: Collaborative Filtering,” *SIKDD 2004 at multi conference IS 2004, 12-15 Oct 2004, Ljubljana, Slovenia.*

[38] Alhalabi W., Kubat M, Tapia M, “Search Engine Personalization Tool Using Linear Vector Algorithm,” *In Proceedings of the Fourth Saudi Technical Conference, Vol. 2, Page: 334-344 (December 2-6, 2006)*

[39] Toshio Oka Hiroyuki Morikawa Tomonori Aoyama, “Vineyard: A Collaborative Filtering Service Platform in Distributed Environment,” *Applications and the Internet Workshops, 2004. SAINT 2004 Workshops. 2004 International Symposium Page(s): 575 – 581, 26-30 Jan. 2004*

[40] Prem Melville, Raymond J. Mooney and Ramadass Nagarajan, “Content-Boosted Collaborative Filtering for Improved Recommendations,” *In Proceedings of the Eighteenth National Conference on Artificial Intelligence(AAAI-2002), pp. 187-192, Edmonton, Canada, July 2002*

[41] Resnick, P., Varian, H. R, “Recommender Systems,” *Communications of the ACM 40, 56-58,1997.*

[42] Atsuyoshi Nakamura and Naoki Abe, “Collaborative Filtering using Weighted Majority Prediction Algorithms,” *In Proceedings of the Fifteenth International Conference on Machine Learning, Pages: 395 - 403 Year of Publication: 1998*

[43] S. Lawrence and C. L. Giles, “Inquirus, the NECI meta search en-gine,” *in Seventh International World Wide Web Conference, (Bris-bane, Australia), pp. 95–105, 1998.*

[44] Dell Zhang and Yisheng Dong, “An efficient algorithm to rank Web resources,” *In Proceedings of the 9th international World Wide Web conference on Computer networks: the international journal of computer and telecommunications networking, Amsterdam, The Netherlands, Pages: 449 – 455, 2000*

[45] S. M. Shafi and Rafiq A. Rather, “Precision and Recall of Five Search Engines for Retrieval of Scholarly Information in the Field of Biotechnology,” *Webology , Volume 2, Number 2, August, 2005*

[46] Heting Chu and Marilyn Rosenthal, “Search Engines for the World Wide Web: A Comparative Study and Evaluation Methodology,” *ASIS 1996 Annual Conference Proceedings, Octobr 1996.*

[47] Shih-Hao Li and Peter B. Danzig, “Precision and Recall of Ranking Information Filtering Systems,” *Journal of Intelligent Information Systems 1-22, Kluwer Academic Publishers, Boston.*

[48] Charles L. A. Clarke and Gordon V. Cormack, “Shortest-Substring Retrieval and Ranking,” *ACM Transactions on Information Systems, Vol 18, No. 1, January 2000, Pages 44-78.*

[49] Caroline M. Eastman and Bernard J. Jansen, “Coverage, Relevance and Ranking: The Impact of Query Operators on Web Search Engine Results,” *ACM Transaction on Information Systems, Vol. 21, No. 4, October 2003, Page 383-411.*

[50] Pedro Goncalves, Jacques Robin, Thiago Santos, Oscar Miranda and Silvio Meira, “Measuring the Effect of Centroid Size on Web Search Precision and Recall,” *In Proceedings 8th Annual Conference of the Internet Society (INET’98). Geneva, Switzerland, July, 1998.*

[51] Budi Yuwono, Savio L. Y. Lam, Jerry H. Ying and Dik L. Lee, “A World Wide Web Resource Discovery System,” *In Proceedings of the Fourth International World Wide Web Conference, Boston, MA., Dec. 1995.*

[52] David Hawking, Nick Craswell, Paul Thistlewaite and Donna Harman, “Results and Challenges in Web Search Evaluation,” *Computer Networks, 31(11-16): 1321-1330, May 1999. Also in Proceedings of the 8th International World Wide Web Conference.*

[53] Sergey Brin and Lawrence Page, “The anatomy of a large-scale hypertextual Web search engine,” Computer Networks and ISDN Systems archive, Volume 30 , Issue 1-7 Pages: 107 – 117, 1998

[54] B. Yuwono and D.L. Lee, “Search and ranking algorithms for locating resources on the World Wide Web,” *12th International Conference on Data Engineering (ICDE'96) p. 164*

[55] Ricardo Baeza-Yates and Emilio Davis, “Web Page Ranking using Link Attributes,” *In Alt. track papers & posters, WWW Conf., pp. 328-329, New York, NY, USA, 2004.*

[56] Andrew Trotman and Richard A. O'Keefe. Identifying and Ranking Relevant Document Elements. In INEX '03, 2003.

[57] J. Kleinberg, “Authoritative sources in a hyperlinked environment,” *Journal of ACM (JASM), 46, 1999.*

[58] Mildrid Ljosland, “Evaluation of Web Search Engine and the Search for Better Ranking Algorithms,” *SIGIR99 Workshop on Evaluation of Web Retrieval (1999)*

[59] Ryen W. White, Ian Ruthven and Joemon M. Jose, “Finding Relevant Documents using Top Ranking Sentences: An Evaluation of Two Alternative Schemes,” *In Proceedings of the 25th Annual International ACM SIGIR Conference (SIGIR 2002). Tampere. Pages 57-64. 2002.*

[60] Albert Bifet and Carlos Catillo, “An Analysis of Factors Used in Search Engine Ranking,” *In Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb).*

[61] Robert M. Losee and Lee Anne H. Paris, “Measuring Search Engine Quality and Query Difficulty: Ranking with Target and Freestyle,’ *Journal of the American Society for Information Science archive Volume 50 , Issue 10 (1999), Pages: 882 – 889, 1999*

[62] Sepandar Kamvar, Taher Haveliwala and Gene Golub, “Adaptive Methods for the Computation of PageRank,” *Numerical Solution of Markov Chains, pages 31-44.*

[63] G. Salton, A. Wong and C. S. Yang, “A Vector Space Model for Automated Indexing,” *Communication of the ACM, Vol. 18, 11, November 1975, pages 613- 620.*

[64] Ryszad S. Michalski, Ivan Bratko , Miroslav Kubat, “Machine Learning and Data Mining: Methods and Applications,” *John Wiley & Sons; (April 9, 1998).*

[65] Abdel-Mottaleb, M. Krishnamachari, S., “Multimedia descriptions based on MPEG-7: extraction and applications,” *IEEE Transactions on Multimedia, Vol. 6, No. 3, June 2004, page(s): 459- 468*