Personalized Ranking for Digital Libraries Based on Log Analysis

Yang Sun[1], Huajing Li[2], Isaac G. Councill[1], Jian Huang[1],
Wang-Chien Lee[2] and C. Lee Giles[1,2]
[1]College of Information Sciences and Technology
[2]Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA, USA
{ysun, icouncill, jhuang, giles}@ist.psu.edu, {huali, wlee}@cse.psu.edu

ABSTRACT

Given the exponential increase of indexable context on the Web, ranking is an increasingly difficult problem in information retrieval systems. Recent research shows that implicit feedback regarding user preferences can be extracted from web access logs in order to increase ranking performance. We analyze the implicit user feedback from access logs in the CiteSeer academic search engine and show how site structure can better inform the analysis of clickthrough feedback providing accurate personalized ranking services tailored to individual information retrieval systems. Experiment and analysis shows that our proposed method is more accurate on predicting user preferences than any non-personalized ranking methods when user preferences are stable over time. We compare our method with several non-personalized ranking methods including ranking SVM^{light} as well as several ranking functions specific to the academic document domain. The results show that our ranking algorithm can reach 63.59% accuracy in comparison to 50.02% for ranking SVM^{light} and below 43% for all other single feature ranking methods. We also show how the derived personalized ranking vectors can be employed for other ranking-related purposes such as recommendation systems.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering, Retrieval models*; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Measurement, Experimentation, Performance

Keywords

personalized ranking, web usage mining

1. INTRODUCTION

With the exponential increase of searchable information on the Web, ranking has become an essential component of information retrieval systems. The quality of ranking order determines which search tool is preferred. The task of ranking is to order the retrieval results according to a given criteria providing documents that are most relevant to user needs.

Traditionally, document text and metadata are the most prevalent information cues on which users make relevance judgments. As stated in *Information Foraging Theory*, "people must assess the relevance or utility of information based on available cues, such as bibliographic citations, abstracts, keywords, titles, etc." [28]. Ranking can be considered as a prediction of how users will judge the relevance of a set of documents to a given query. From this perspective, ranking can be viewed as a subjective service which is tightly related with specific users. For example, in academic paper digital libraries, user U_a may want to find the most classical paper in a specific domain to understand the source and fundamental description of a problem. On the other hand, another user U_b may be eager to know the most recent publications in the domain in order to determine current research trends and discover new findings. Although the query issued by U_a and U_b may be the same, each user has different preferences (U_a prefers the *number of citations* feature while U_b prefers *date* feature). Thereafter, to better serve users with different interests, the ranking mechanism needs to be customizable based on user preferences.

Personalized services have been delivered by many web systems for customized appearances and search results. User preferences are manually customized by users or automatically mined in order to create custom user interfaces and search filters. As for ranking, an effective information system needs to automatically mine access logs of user behavior since user preferences are often difficult to express explicitly. Observing this, it is of great potential benefit to explore recent user browsing history to improve ranking facilities and discover information regarding the temporal trends of user preferences.

Web server logs are important resources for mining user preferences, which makes log mining an important technology to discover user preferences. Log mining is widely applied in a broad range of research topics including Web personalization, recommender systems, and Web site design and evaluation [23, 31, 32]. A typical Web usage mining process includes three phases which are (1) data preprocessing, (2) pattern discovery, and (3) pattern analysis [32]. From Web logs, information such as IP addresses, time stamps, and requested pages can be extracted, which can then be used by the Web application to infer implicit feedbacks of users such as motivations, goals and preferences. Much research has been done to apply data mining and machine learning technologies to study implicit user feedback to improve ranking [15, 30, 35]. However, such work has not modeled the preferences for each individual user to provide personalized ranking.

In this paper, we introduce a user preference model and propose a personalized ranking method using web log mining results of a large-scale academic digital library. Our solution is based on the observation that a user's ranking preference is actually a *multiple criteria decision problem*, which can then be decomposed into a set of unit preferences, each of which is placed on specific data features. From click-through data harvested from Web logs, we periodically compute and update user preference vectors in a given feature space so that the ranker can always be used to describe a user's needs as exhibited in recent logs. In summary, we make the following contributions to the personalized ranking research:

- We introduce a user preference model and propose a personalized ranking method to improve the accuracy in predicting user preferences. We present a ranker training algorithm that uses the summarized partial user preferences to improve ranking quality.

- We identify the patterns of user click-through history that indicate users' subjective judgments of document relevance and propose a method of efficiently extracting and summarizing user preferences from large volumes of Web logs.

- The results of extensive experiments on a real-world document digital library are presented to show the applicability and quality of our methods.

The rest of this paper is organized as follows. In section 2 we briefly review current approaches to ranking and personalization. Section 3 describes our method to model user preference and extract implicit feedback from access logs. The details of experiments, findings and analysis are described in section 4. Section 5 discusses our future work and applications. Section 6 concludes.

2. RELATED WORK

In this section, we briefly review personalization techniques and their applications in information systems with emphasis on search engine systems.

2.1 Ranking

Effectively ranking documents for large scale information retrieval systems continues to be a challenging task. In Web search, PageRank [26] and Hypertext Induced Topic Selection (HITS) [18] algorithms were created to measure the importance or authority of individual documents as a key ranking factor, showing great success in combining these metrics with lexical similarity among queries and documents. More recently, machine learning technology has been applied to rank documents in search engines [15, 30, 35]. Clickthrough data extracted from search engine query logs are used to train ranking models and globally optimize the search results[15]. Machine learning methods are also used for static ranking and show an increase in relevance compared to PageRank [30], where a standard neural network back-propagation algorithm RankNet is trained on the features of web pages. Boosting is also used in ranking documents for information retrieval systems [35] where explicit user feedback is used as the training data to improve the ranking performance.

Much personalized ranking research has been focused on making use of the *"personalization vector"* of the PageRank algorithm [26]. A set of representative topics is used as the personalization vector to bias the PageRank computation and obtain a topic-sensitive ranking [12]. User profiles based on URL features are used to obtain a topical and geographically biased PageRank [2]. From the topology of the Web, the Hilltop algorithm provides topic-driven ranking based on"expert" documents which are found through link analysis [4]. There exists work focusing on the scalability and performance of personalized PageRank algorithms [14, 9] using graph-mining methods to generate personalized views of document importance. These algorithms present a categorical perspective on personalization. It is difficult to expand the idea to a large number of categories since these algorithms require a pre-computed set of ranking scores for each category, which is given a priori and is difficult to adjust dynamically. In addition, PageRank-based algorithms only incorporate a structural aspect of the World Wide Web. Recently, research work suggests that PageRank may not perform better than other simple measures on ranking webpages in general purpose search engines [3, 30].

Another approach to personalized ranking is to use full-text preferences interactively[20]. A top-ranked list of documents is provided to users based on a set of pre-defined preferences. According to users' relevance feedback on these documents, preferences can be adjusted to fit each individual's information need. Partial orders, which are defined as a binary relation $\preceq$ over a limited set P, are studied to understand a user's goal so that when this user continues to expand the search results, the user's preferences can be considered.

2.2 Personalization Techniques

Personalized systems mainly take advantage of personal information for the following services:

- Recommender systems [6, 11, 17]: The system studies correlations among between users and products to proactively suggest selection candidates.

- Searching [34, 7]: Personal information is supplied to the search engine to reorganize the query to reduce the ambiguity of search terms.

- Filtering [6]: Unlike searching in the general case, personalized filtering services employ user profiles to narrow the search scope.

Generally speaking, a personalized service requires knowledge of users, which can be expressed as *user models.* Typically, a user model is expressed as a set of preferences and access patterns. These user-associated metadata are obtained in many ways, including relevance feedback [34, 5] given or implied by user behavior, machine learning techniques, and web mining. Machine learning techniques are widely applied in personalized services for their ease of maintenance and their capability to capture recenct user preferences. For example, as of recommender systems, often *content based* and *collaborative filtering* methods are used [1]. Content based analysis uses the similarity in object contents to generate recommendation lists, whereas collaborative filtering takes another perspective by studying ratings given by users to find user groups with similar interests for the purpose of recommendation. Web mining [24, 19, 8, 23] explores logged user access history to generalize user activity patterns and characteristics, which can then be used for personalization purposes.

3. USER PREFERENCE MODEL

Our goal is to predict the relevance of documents for users which is different from typical goal in user modeling research. Much research has been focused on describing and predicting user actions by mining the web usage data. Association rule extraction, collaborative filtering, and sequential pattern analysis are the most common and noticeable methods related to user preference models. Association rule extraction has been used to identify sets of items that are accessed together [21]. Collaborative filtering algorithms [27, 29] have been used to identify similar users based on the overlap between their requests, and recommend the given user webpages accessed by similar users. Sequential pattern analysis algorithms are trying to discover patterns such that the history of actions is evidence to future actions based on time ordered sessions. Probabilistic user behavior models are also proposed to describe and predict user actions [22]. User actions are described by the conditional probability of user performs an action given the user's action history and the clustering probability of the user.

All these models obtain significant achievements on predicting user actions. However, it is not feasible to apply these models in the context of ranking because these models typically do not consider document features. A user model in ranking problem should incorporate document features so that the model can be used to predict the ordering for any documents in the IR system.

In our research, the user preference is modeled as a vector in the document feature space where features include all the metadata presented to users in a system. One assumption in this feature space vector model is that users can only rely on these features (document metadata) to determine the relevance of documents. The notions of the user preference model are introduced as the following:

- A *feature* is the basic unit of discrete data. Each feature is defined to be an item from a complete set of features F that are presented to users to determine relevance.
- For a given subset of N features, a user is defined as a N-vector $U = \{u_1, u_2, ..., u_N\}$ such that $u_i > 0$ if the user prefers higher score of the feature i and $u_i < 0$ otherwise. $u_i = 0$ is also allowed to represent no preference on feature i. u_i is the weight that user U evaluate feature i of documents.
- A document D is defined as a N-vector $D = \{d_1, d_2, ..., d_N\}$ in the feature space where d_i is the weight of feature i for the document.
- The *level of relevance* between a document D and a user U is defined as the *inner product* of the document vector D and the user vector U.

According to the vector model, the inner product of a document vector D and a user vector U represents the level of relevance of the document D to the user U and therefore can be used to rank the documents for user U (see Eq1).

$$Score(D, U) = < D \cdot U > = \sum_{i=1}^{N} d_i \times u_i. \qquad (1)$$

3.1 Implicit feedback

In the user preference model, user preference vectors are obtained by training on user feedback extracted from web access history. Thus, correctly interpreting user feedback from web access logs is a key to the effectiveness of the model. The web access logs typically record all requests of users. Clearly, users do not request documents randomly. However, simply comparing the clicked documents with non-clicked documents is biased by current ranking of the documents because users tend to request the top ranked documents they see. Click-through data considering the logic of a website can be used to eliminate the bias. The web logic embedded click-through only compares two documents presented to or requested by users with equal opportunity. It has been reported that click-through data representing partial user feedback in search engine systems can be extracted to reflect the users' judgment on document rankings [15, 16].

More generally, any two documents D_1, D_2 in a website requested by or presented to a user U for the same purpose form a document pair p. A user prefers one document over the other if he/she requests additional information (including purchase in e-commerce websites) of one document but not the other. Such pair of documents represents an instance of implicit relevance feedback from the user. The document pairs can be denoted as $p = \{D_1 > D_2\}$ where document D_1 is preferred over document D_2 by user U. In the vector model, the preference vector of the user U should rank D_1 higher than D_2 (i.e. $Score(D_1, U) > Score(D_2, U)$). Thus, the document pairs representing user implicit feedback can be used as training samples to train the user preference vectors.

3.2 User Preference Vector

For a given set of n document pairs $P = \{p_1, p_2, ..., p_n\}$ representing the implicit feedback of a user, an optimal user preference vector U^* should correctly rank all pairs of documents. That is, for any pair of documents $\forall p_i = \{D_{i_1} > D_{i_2}\} \in P$, $Score(D_{i_1}, U^*) > Score(D_{i_2}, U^*)$. Let $Score(p, U) = Score(D_{i_1}, U) - Score(D_{i_2}, U)$. $Score(p, U) > 0$ represents a document pair p is correctly ranked by a user preference vector U. The actual preference vectors typically do not achieve the optimal results. In practice, we use the preference vector that maximizes the correctly ranked document pairs of a user as the user's preference vector (see Eq2).

$$U = \arg\max \sum_{i=1}^{n} |Score(p_i, U) > 0| \qquad (2)$$

The training of user preference vectors can be described as finding a preference vector that maximizes the number of correctly ranked document pairs. We use a power iteration approach [25] to estimate user preference vectors. Let $U = \{u_1, u_2, ...u_N\}$ be the user preference vector where u_i is the weight for the ith feature out of total N features. U^j denotes the preference vector in jth iteration. Each element weight u_i in each iteration j is computed by maximizing the correctly ranked document pairs assuming the rest of the weights $u_k|_{k \in N, k \neq i}$ are known from previous iteration $j - 1$. Thus, each calculation is reduced to a one-dimension maximization problem. The algorithm is described in Algorithm 1.

Algorithm 1 Maximization(P)

```
1: $P = p_1, p_2, ..., p_n$
2: Initialize $U^0$: $u_1 = u_2 = ... = u_m = \frac{1}{k}$, $j = 0$
3: repeat
4:     $j \leftarrow j + 1$
5:     for k=1, ...,m do
6:         Find $u_k$ that maximizes correctly ranked document pairs
7:         $u_k = \arg\max \sum_{i=1}^{n} |Score(p_i, U^{j-1}) > 0|$
8:         Update $U^j$ with $u_k$
9:     end for
10: until Converge
11: RETURN $U^i$
```

3.3 Personalized Ranking

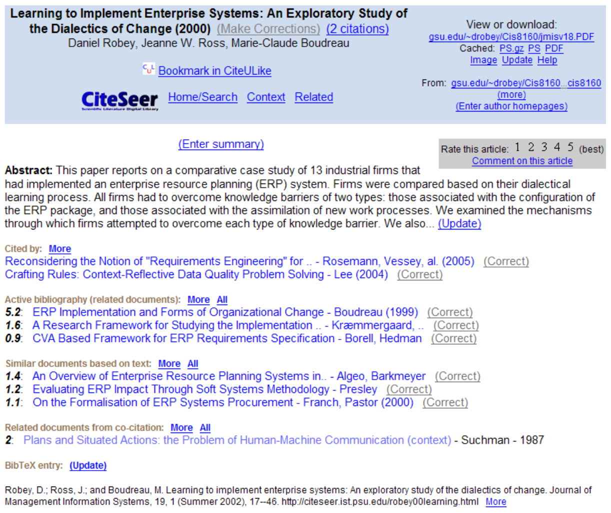
Based on the user preference model, personalized ranking can be described as reordering documents by the similarity score between documents and user preference vectors. When a user submits a query to the system, the system retrieves all document based on lexical similarity first, and then reranks the documents based on the preference vector of the user. If the preference vector is not successfully obtained for a specific user, the default preference vector defined by all users will be employed for this user.

4. EXPERIMENTS

4.1 Extracting Implicit Feedback

We use CiteSeer data demonstrating our personalized ranking method. There is a link on each document summary page linking to the PDF version of the document in CiteSeer (see Figure 1). User clicks on these links can be used to represent further evidence of relevance judgments since users are interested enough in the document to download the source content. For a given query, if a user downloads document D_1 but does not download document D_2 after reviewing both documents' summary pages, it is reasonable to assume that that this user judges document D_1 to be more relevant than document D_2. Thus, D_1 and D_2 form a document pair $p = \{D_1 > D_2\}$ representing the user preference.

The sequence of user actions can be extracted from web access logs. A web access log record includes the client IP, request time, request file (link on the website), reference link, user-agent string, and session id. Such data is sufficient to identify each user session and the click sequence of that session. In CiteSeer, the download link and document detail page link are presented in the log records as shown in Table 1.



Learning to Implement Enterprise Systems: An Exploratory Study of the Dialectics of Change (2000) (Make Corrections) (2 citations)
Daniel Robey, Jeanne W. Ross, Marie-Claude Boudreau

Bookmark in CiteULike

CiteSeer Home/Search Context Related

View or download:
gsu.edu/~drobey/Cis8160/jmisv18.PDF
Cached: PS.gz PS PDF
Image Update Help

From: gsu.edu/~drobey/Cis8160...cis8160
(more)
(Enter author homepages)

(Enter summary)

Rate this article: 1 2 3 4 5 (best)
Comment on this article

Abstract: This paper reports on a comparative case study of 13 industrial firms that had implemented an enterprise resource planning (ERP) system. Firms were compared based on their dialectical learning process. All firms had to overcome knowledge barriers of two types: those associated with the configuration of the ERP package, and those associated with the assimilation of new work processes. We examined the mechanisms through which firms attempted to overcome each type of knowledge barrier. We also... (Update)

Cited by: More
Reconsidering the Notion of "Requirements Engineering" for .. - Rosemann, Vessey, al. (2005) (Correct)
Crafting Rules: Context-Reflective Data Quality Problem Solving - Lee (2004) (Correct)

Active bibliography (related documents): More All
5.2: ERP Implementation and Forms of Organizational Change - Boudreau (1999) (Correct)
1.6: A Research Framework for Studying the Implementation .. - Kræmmergaard, .. (Correct)
0.9: CVA Based Framework for ERP Requirements Specification - Borell, Hedman (Correct)

Similar documents based on text: More All
1.4: An Overview of Enterprise Resource Planning Systems in.. - Algeo, Barkmeyer (Correct)
1.2: Evaluating ERP Impact Through Soft Systems Methodology - Presley (Correct)
1.1: On the Formalisation of ERP Systems Procurement - Franch, Pastor (2000) (Correct)

Related documents from co-citation: More All
2: Plans and Situated Actions: the Problem of Human-Machine Communication (context) - Suchman - 1987

BibTeX entry: (Update)

Robey, D.; Ross, J.; and Boudreau, M. Learning to implement enterprise systems: An exploratory study of the dialectics of change. Journal of Management Information Systems, 19, 1 (Summer 2002), 17--46. http://citeseer.ist.psu.edu/robey00learning.html More

Figure 1: The document summary page in CiteSeer.

Request of document detail page	
130.*.*.* - - [23/Sep/2007:04:22:01 -0400] "GET /419972.html HTTP /1.1" 200 4096 "http://www.google.com /search?hl =en&rlz=1B3GGGL_enUS 241US241&q=Enterprise+Systems" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"	
Request: 419972.html	Query: Enterprise Systems
Request of document detail page	
130.*.*.* - - [23/Sep/2007:04:23:21 -0400] "GET /robey00learning.html HTTP /1.1" 200 4096 "http:// www.google.com/search?hl=en&rlz=1B3GGGL_enUS 241US241 &q=learning+Enterprise+Systems" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"	
Request: robey00learning.html	Query: Enterprise Systems
Request of download paper	
130.*.*.* - - [23/Sep/2007:04:23:55 -0400] "GET /cache/papers/cs/16140/ http:zSzzSzwww.cis.gsu. eduzSzdrobeyzSzCis8160zSzjmisv18.PDF/robey00learning.pdf HTTP/1.0" 200 115257 "http://citeseer .ist.psu.edu/robey00learning.html" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"	
Request: robey00learning.pdf	Reference :robey00learning.html

Table 1: Examples of CiteSeer access log records.

Through log analysis, we can identify the complete user session of a user behavior sequence. In this example the user reviewed the document "robey00learning.html" and "419972.html" for the query "Enterprise System" and then downloaded the PDF version of the later document. Thus, we conclude from this sequence of user behavior that this user thought document "robey00learning.html" is more relevant than document "419972.html" for the query "Enterprise System". We assume that user decisions are based solely on the information provided on the document summary pages.

4.2 Features of Documents

The features of documents presented to users in CiteSeer include citation statistics, publication venues, author information, publication dates, similarity to individual queries, etc. In practice, features are converted to real numbers in order to fit into our preference vector model. In our experiment, we choose eight features presented to users in the document summary page (see Table 2).

We use a cosine similarity score to measure the similarity between the query and the title and abstract. The score

Feature	Definition
title similarity	The lexical similarity between the query and title of document
abstract similarity	The similarity between the query and abstract of document
citation count	The number of times the document has been cited[10]
venue weighted ranking	Ranking documents based on the venue weighted citation analysis[33].
coauthor count	Ranking score defined as the average number of coauthors for each author in a document
average h-index	the average h-index[13] for each author in a document
author citation ranking	average on the citation received for each author
recency	How recent a paper is published.

Table 2: Features of documents in academic paper search engine.

ranges from 0 to 1 where 0 represents that none of the terms in the query appears in the title/abstract and 1 represents that the query is exactly the same as the title/abstract. *Citation count* is the number of times a document is cited by other documents. We also implement a *venue-weighted PageRank* based on the citation graph to represent the impact of venue. In addition, three author-related features are implemented for each document: the *coauthor count*, average *h-index* and *author citation score*. The coauthor count averages the number of coauthors for each author of a document. The score is to reflect the level of collaboration of authors. The average *h-index* is a score averaging the *h-indices* of all the authors of a document. *h-index* is defined as follows: "A scientist has index h if h of his N_p papers have at least h citations each, and the other (N_p - h) papers have at most h citations each.[13]" The *h-index* is typically used as an academic performance evaluator in certain institutions and countries. The author citation score is generated by counting the citations an author received and averaging among all the authors of a document. The recency score is calculated by normalizing the difference between publication date and the current date, reflecting the freshness of a paper. The features we used in the experiments are based on heuristics and availability. There is much room for future improvement of the result by tuning feature selection and weights as well as implementing new features. We also test the effectiveness of current features and show how these features predict users decisions to download documents.

4.3 Data Preprocessing

It is important to separate robot-generated logs from user generated logs. However, with the development and evolution of Web technologies, the access patterns of robot traffic change dramatically and are thus difficult to capture. It has been observed that the requests initialized by robots contribute a considerable portion of the network traffic. We use a two-step log filter in order to identify each user session. The first step checks the User-Agent string field recorded in the logs to identify obvious robots. For unethical robots that do not declare themselves as robots when accessing CiteSeer services, or mimic the access patterns of real users, we identify them by a set of features including total visits per day, the temporal distribution of visits and the distribution of session lengths.

With the filtered log records, the IP address is used as the identity of users. We regard every IP address as a single user, for which requests from a same IP address are taken as from the same user. Although this is not guaranteed to be accurate, this is the best available approach. The log entries that are generated from a single IP address are grouped and recorded sequentially by their time stamps. If we find the successive visit interval by a user exceeds a time threshold, the latter request is taken as the start of a new session.

4.4 Experiment Setting

Our data set consists of CiteSeer log files covering a period of approximately one and a half months which contain 11,856 unique users (IPs) with 3,051,670 click-download document pairs. To test our personalized ranking algorithm, the data set is divided into two parts based on the timestamp. The first part including one month of data is used as training samples and the second part including half month of data is used in testing. Since not all the IPs in the testing data have corresponding records in the training data, our test is conducted on a subset of the complete testing data containing 852 IPs with 20,964 document pairs in which 57.7% of the IPs have prior records in the training data. Since ranking is inteded to predict the users' preferences, it is reasonable to evaluate the ranking algorithms using later generated logs by users in the same system because the testing can be viewed as the accuracy in predicting future user behavior. The accuracy of the preference vectors can be defined as the percentage of the correctly ranked document pairs (see Eq3).

$$accuracy(U) = \frac{||\{p_j|Score(p_j,U) > 0\}||}{||\{p_j \in P\}||} \tag{3}$$

4.5 Results

The complexity of our algorithm is determined by testing the accuracy of current weight vectors for the data sets in each iteration. The average time complexity of our algorithm is $O(n * \lg n)$ based on quick sort (see Figure 2). It shows that our algorithm is highly scalable to large data sets. Our algorithm is able to train the linear combination ranking function using a weight vector on 2.9 million data points in less than 2 hours on a Xeon 3.0GHz PC with 1GB RAM.

According to the experimental setting, the accuracy is defined as the percentage of correctly ranked document pairs over the total document pairs that were extracted. This is used as a measure of how well the ranking predicts the user preference. We compare our personalized ranking algorithm with non-personalized user preference vector and SVM^{light} software[15]. The ranking SVM^{light} software could only handle a small portion of the training data in a reasonable time period (3000 training samples for 4 hours). All the ranking functions are tested on the same testing data and the accuracy of predicting user preferences are compared in Figure 3. The result shows that personalized ranking method performs better than ranking SVM^{light} and other non-personalized ranking methods.

Individual features are used to rank documents in many digital libraries and search engines. The ranking accuracy using each individual feature score is examined and com-

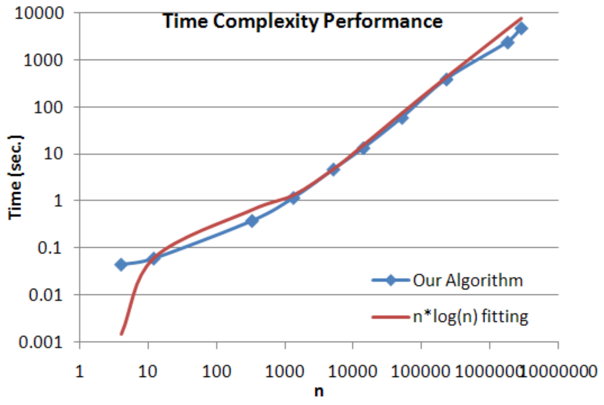


Figure 2: Time complexity performance of personalized ranking weight training algorithm.

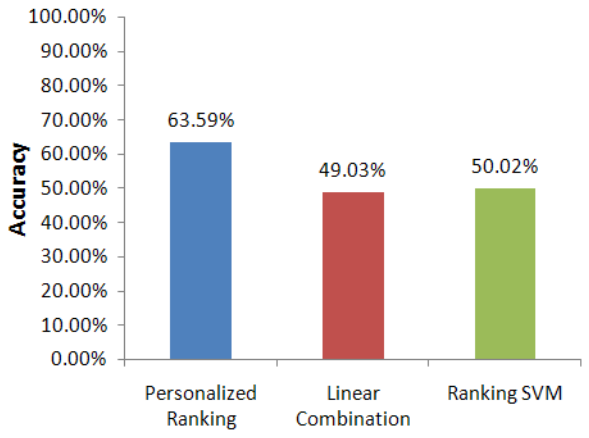


Figure 3: The comparison of the accuracy of our personalized ranking, ranking SVM^{light} and a linear weight vector trained on the sample data on testing data.

pared (see Table 3). Notice that all the single feature ranking accuracies are less than 43%. The result is due to the fact that many documents have the same value for a feature, rendering the feature useless for discriminating between certain documents. Thus, the baseline ranking accuracy can be far less than 50% for these features. The results show that none of the single feature rankings have a comparable performance to the personalized ranking algorithm in terms of predicting user actions.

4.6 Result Analysis

42.3% IPs in our selected testing data set do not have prior behavior records. The non-personalized ranking is used in such cases, which may result in lower ranking accuracy. We also experiment on a subset of the testing data including only the IPs that have prior records and obtain an accuracy result of 67.74%. The result implies that the personalized ranking method will be more accurate with more history data of users.

In the experiment, we also filter out the training samples that are less than 30 document pairs. Our assumption is that more document pairs in the training data improve the accuracy and stability of the algorithm. To test this assumption, we set the threshold to 10, 15, 20 and 30 to compare the total accuracy of the ranking method under each condition. Figure 4 shows a clear trend that more document pairs in the training sample significantly improve the overall accuracy.

Ranking function	Accuracy
title similarity	16.91%
abstract similarity	25.77%
citation count	21.67%
popularity ranking	32.25%
co-author ranking	42.35%
average h-index	41.68%
author citation ranking	42.72%
recency	38.76%

Table 3: The ranking accuracy results using each single feature as the document ranking function.

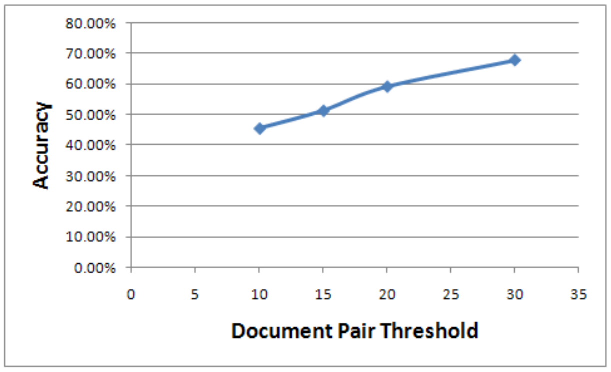


Figure 4: Comparison of different training sample threshold.

The features we implement in the experiment also affect the results significantly. Less informative features may impose overfitting problem. We run a test of different combination of features on the data subset that all the IPs have prior records. Table 4 shows that the performance of some feature combinations have better accuracy results than using all the features.

Feature Combination	Accuracy
[0, 1, 2, 5, 7]	**73.55%**
[0, 1, 2, 3, 4]	72.74%
[0, 1, 3, 4, 5]	71.29%
[0, 3, 4, 6]	68.39%
[0, 1, 2, 3, 4, 5, 6, 7]	67.74%

0:title similarity, 1:abstract similarity, 2:citation count, 3:venue weighted ranking, 4:coauthor count, 5:average h-index , 6:author citation ranking, 7:recency.

Table 4: The feature combinations that produce a better accuracy result on predicting user preferneces.

The features can be roughly divided into four groups. *Titlesimilarity* and *Abstractsimilarity* describe the lexical relevance features of a document. (The text-related features are query dependant.) Citation count and venue weighted ranking describe the citation analysis result of a document. Coauthor count, average h-index, and author citation ranking describe the quality of authors in a document. The *recency* feature describes the temporal property of a document. The feature combination tuning results suggests that at least three features, each from a different feature group, should be considered and the text similarity features are always necessary in improving the accuracy. The click-download document pairs do not show a significant difference in text similarity features because the click of documents already implied the text similarity in the click-download actions. However, if a document is clicked with a low text similarity to the query for other reasons, users will tend to prefer the documents with higher text similarity. For the same reason, citation-based features are also underestimated because the original ranking algorithm is based on citation features in CiteSeer. Since not all click-download document pairs can be rooted to the search function in CiteSeer (there are a significant number of click-download pairs generated by the reference from other search engines), it is hard to estimate the impact of the original ranking algorithms since they are unknown.

There are a few drawbacks in the data set used in the experiment. Other factors may affect the user preferences in that specific time period of the logs used in the research such as conference submission deadlines and temporal trends in topic popularity. User preferences may change significantly after that period of time. There are also many documents that have only partial features due to missing data in the original document collection. These defects in the training data and testing data will be the focus of future investigations. The above analysis shows that an appropriate selection of data and features promises much room for improving the personalized ranking method.

4.7 Advantages of User Preference Vectors

Since the personalized ranking method is based on the preference vector model in feature space, there are benefits for practical systems. In practice, a large system can return millions of documents in response to a general query. Any complex model of ranking (e.g. HITS, SVM) yields a significantly slower performance on ranking results for large data sets. With a linear vector of features as a ranking function, it is possible to provide fast ranking based on the optimization of indexed feature scores.

The personalized ranking method also has the advantage in preventing ranking spam. Since the user preference vector are trained on user access history data, it can be spammed only if the user browsing history is contaminated which is very difficult due to current IP technologies. Even if one preference vecotr is spammed, the rest of the vectors remain untouched for individual users.

5. APPLICATIONS AND FUTURE WORK

Our personalized ranking model is also useful in other application areas. The preference vectors generated from the experiment represent the users in the feature space. Thus, it is possible to define distance or similarity between two users. The similarity between two users can be defined as the inner product or cosine similarity of the preference vectors of the two users. Therefore documents can be suggested to similar users with similar preference in recommendation systems. A preference network can be constructed based on the similarity measure of user preference. As we discussed in the previous section, the user preferences may change over time. the user preference vector can also help in a temporal study of the preference trend of users.

In our user preference model, we assume each user is represented by only one preference vector. However, probabilistic models of the preference vectors can be used to improve the model. User preference can be modeled as a probability distribution of multiple preference vectors. Future work will extend our personalized ranking method by considering the probabilistic model of user preference.

As we analyzed above, the accuracy is significantly affected by previous records of users. The training data will be able to cover most of the records for the "returned" users by spanning the log mining to a longer period of time. Longer period training data can also capture the variations in preference, hence increase the accuracy of ranking.

The user preference model is only tested on one particular website. The generalizability of the model is not tested. Our future work will also test the user preference model for other types of websites that could fit in the model.

6. CONCLUSION

Ranking remains a difficult problem in information retrieval systems. It is difficult for a single ranking function to meet the variety of individual users' information needs. Thus, a personalized ranking method is needed. We propose a personalized ranking method based on user preference models, representing users by sparse vectors in a feature space that accounts for regularities in specific site designs. The user preference vectors are obtained by training on user implicit feedback extracted from web access logs. Within the context of a popular scholarly paper search engine, we model user preferences based on an analysis of specific features of the search tool, showing significant improvements in rank predictions.

We formalize the ranking problem based on the user preference model and show that the personalized ranking method is more accurate on predicting user actions when user preferences are stable over time. To evaluate our personalized ranking method, we test out ranking method on a data set extracted from later access logs and compare it with other non-personalized ranking methods. The experiments show that our personalized ranking method significantly improves the ranking accuracy on predicting user actions. The accuracy result for personalized ranking is 63.59% compared to 50.02% with ranking SVM^{light} and below 43% for single feature ranking functions.

7. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[2] M. S. Aktas, M. A. Nacar, and F. Menczer. Personalizing pagerank based on domain profiles. In *Proc. of WebKDD*, 2004.

[3] B. Amento, L. Terveen, and W. Hill. Does "authority" mean quality? predicting expert quality ratings of web docu-ments. In *Proceedings of the 23rd ACM SIGIR conference*, pages 296 – 303, 2000.

[4] K. Bharat and G. A. Mihaila. Hilltop: A search engine based on expert documents. In *Proc. of the 9th International WWW Conference (Poster)*, 2000.

[5] W. B. Croft, S. Cronen-Townsend, and V. Larvrenko. Relevance feedback and personalization: A language modeling perspective. In *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001.

[6] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international WWW conference*, pages 271–280, New York, NY, USA, 2007. ACM Press.

[7] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *WWW*, pages 581–590, 2007.

[8] M. Eirinaki and M. Vazirgiannis. Web mining for web personalization. *ACM Trans. Inter. Tech.*, 3(1):1–27, February 2003.

[9] D. Fogaras and B. Racz. Towards scaling fully personalized pagerank. In *Proc. of Third Workshop on Algorithms and Models for the Web*, 2004.

[10] E. Garfield. Is citation analysis a legitimate evaluation tool? *Scientometrics*, 1:359–375, 1979.

[11] M. Gker and C. Thompson. Personalized conversational case-based recommendation, 2000.

[12] T. H. Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. In *Proc. of the 11th International WWW Conference*, 2002.

[13] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences*, 102:16569–16572, 2005.

[14] G. Jeh and J. Widom. Scaling personalized web search. In *Proc. of the 12th International WWW Conference*, 2003.

[15] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. of the 8th ACM SIGKDD*, pages 133 – 142, 2002.

[16] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proc. of the 28th ACM SIGIR conference*, pages 154 – 161, 2005.

[17] S. Jung, K. Harris, J. Webster, and J. L. Herlocker. Serf: Integrating human recommendations with search. In *Proc. of 13th CIKM*, 2004.

[18] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of ACM*, 48:604–632, 1999.

[19] R. Krishnapuram, A. Joshi, O. Nasraoui, and L. Yi. Low-complexity fuzzy relational clustering algorithms for web mining. *IEEE-FS*, 9:595–607, Aug. 2001.

[20] A. Leubner and W. Kiebling. Personalized nonlinear ranking using full-text preferences. In *Proc. SIGIR Workshop on Customised Information Delivery*, 1999.

[21] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[22] E. Manavoglu, D. Pavlov, and C. L. Giles. Probabilistic user behavior models. *icdm*, 00:203, 2003.

[23] B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization based on web usage mining. *Commun. ACM*, 43(8):142–151, 2000.

[24] M. D. Mulvenna, S. S. Anand, and A. G. Büchner. Personalization on the net using web mining: introduction. *Commun. ACM*, 43(8):122–125, 2000.

[25] A. Packard, M. K. H. Fan, and J. Doyle. A power method for the structured singular value. In *Proceedings of the 27th IEEE Conference on Decision and Control*, pages 2132–2137, 1988.

[26] L. Page and S. Brin. The pagerank citation ranking: bring-ing order to the web. In *Tech. report SIDL-WP-1999-0120, Stanford University*, 1999.

[27] D. Pavlov, E. Manavoglu, D. M. Pennock, and C. L. Giles. Collaborative filtering with maximum entropy. *IEEE Intelligent Systems*, 19(6):40–48, 2004.

[28] P. Pirolli and S. Card. Information foraging in information access environments. In *Proc. of the SIGCHI conference*, pages 51 – 58, 1995.

[29] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994. ACM.

[30] M. Richardson, A. Prakash, and E. Brill. Beyond pagerank: machine learning for static ranking. In *Proc. of the 15th WWW Conference*, pages 707 – 715, 2006.

[31] M. Spiliopoulou. Web usage mining for web site evaluation. *Commun. ACM*, 43(8):127–134, 2000.

[32] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: discovery and applications of usage patterns from web data. *SIGKDD Explor. Newsl.*, 1(2):12–23, 2000.

[33] Y. Sun and C. L. Giles. Popularity weighted ranking for academic digital libraries. In *29th ECIR*, pages 605–612. Springer-Verlag, 2007.

[34] J. Teevan, S. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities, 2005.

[35] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th ACM SIGIR*, pages 391–398. ACM Press, 2007.