Learning to Rank Relational Objects and Its Application to Web Search

Tao Qin[*]
Tsinghua University
Beijing, P.R.China
tsintao@gmail.com

Tie-Yan Liu
Microsoft Research Asia
Beijing, P.R.China
tyliu@microsoft.com

Xu-Dong Zhang
Tsinghua University
Beijing, P.R.China
zhangxd@mail.thu.edu.cn

De-Sheng Wang
Tsinghua University
Beijing, P.R.China
wangdsh_ee@mail.thu.edu.cn

Wen-Ying Xiong*
Peking University
Beijing, P.R.China
flykitej@gmail.com

Hang Li
Microsoft Research Asia
Beijing, P.R.China
hangli@microsoft.com

ABSTRACT

Learning to rank is a new statistical learning technology on creating a ranking model for sorting objects. The technology has been successfully applied to web search, and is becoming one of the key machineries for building search engines. Existing approaches to learning to rank, however, did not consider the cases in which there exists relationship between the objects to be ranked, despite of the fact that such situations are very common in practice. For example, in web search, given a query certain relationships usually exist among the the retrieved documents, e.g., URL hierarchy, similarity, etc., and sometimes it is necessary to utilize the information in ranking of the documents. This paper addresses the issue and formulates it as a novel learning problem, referred to as, 'learning to rank relational objects'. In the new learning task, the ranking model is defined as a function of not only the contents (features) of objects but also the relations between objects. The paper further focuses on one setting of the learning problem in which the way of using relation information is predetermined. It formalizes the learning task as an optimization problem in the setting. The paper then proposes a new method to perform the optimization task, particularly an implementation based on SVM. Experimental results show that the proposed method outperforms the baseline methods for two ranking tasks (Pseudo Relevance Feedback and Topic Distillation) in web search, indicating that the proposed method can indeed make effective use of relation information and content information in ranking.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - Retrieval models; H.3.4 [**Information Systems Applications**]: Systems and Software - performance evaluation

General Terms

Algorithms, Experimentation, Theory

---

*This work was conducted at Microsoft Research Asia.

Keywords

Learning to Rank Relational Objects, Relational Ranking SVM, Pseudo Relevance Feedback, Topic Distillation

1. INTRODUCTION

Ranking is a central problem for web search, because the goodness of a search system is mainly evaluated by the accuracy of its ranking results. Traditionally, ranking model is constructed by tuning a few parameters with a small amount of labeled data. It is only recently machine learning technologies called 'learning to rank' have been intensively applied to the task. In the approach, a large number of features and a large amount of training data are used to create the ranking function and various optimization techniques (loss functions and algorithms) are employed to train the ranking function. Previous work shows that learning to rank has certain advantages when compared with the traditional approaches. Many methods have been proposed including Ranking SVM [15, 18], RankBoost [12], RankNet [4], ListNet [6], AdaRank [34], MHR [23], and FRank [31].

Existing technologies on learning to rank are limited to one setting of ranking, namely ranking based on content information. Specifically, in web search given a query and a number of retrieved documents containing the query, a ranking function is defined as a function of query and document.

There are other search applications in which relation information between documents can be or must be exploited. The relation information can be represented in a graph, or more generally, a matrix. For example, web pages from the same site form a sitemap hierarchy. If both a page and its parent page are about the topic of the query, then it would be better to rank higher the parent page for this query. This is a problem referred to as Topic Distillation at TREC [33]. As another example, similarities between documents are available, and we can leverage the information to enhance relevance ranking. This is a problem close to Pseudo Relevance Feedback [29] in IR. Other problems like Subtopic Retrieval [36] also need utilize relation information. Existing learning to rank methods, however, cannot handle the problems.

In this paper, we propose a new learning framework referred to as learning to rank relational objects (e.g., rank relational documents in web search). In the ranking task, we make use of both content information and relation information. In conventional learning to rank, (when applied to

web search), the ranking function is defined as that of query and document pair. In contrast, in the new learning task, the ranking function is defined as that of not only query and document pair, but also relations between documents. The framework is very general, and in this paper we address two specific tasks: Pseudo Relevance Feedback and Topic Distillation as examples.

We focus on one setting of learning to rank relational objects, in which the way of using the relationships between objects is predefined, while the parameters of the ranking function need to be trained by using labeled training data. We formalize the learning task as an optimization problem. The optimization problem appears to be a challenging issue, because of the nested structure of the ranking function. We know of no existing method that can be directly applied. We then propose a new method to solve the problem, and specifically an implementation method based on SVM. Experimental results show that the proposed method outperforms the baseline methods for Pseudo Relevance Feedback and Topic Distillation.

The remaining part of the paper is organized as follows. In Section 2, we introduce related work. In Section 3, we give a definition on the problem 'learning to rank relational objects' and in Section 4 we propose one solution to the problem. An SVM-based algorithm is then presented in Section 5, followed by experiments in Section 6. Finally, conclusions are given in the last section.

2. RELATED WORK

2.1 Learning to rank

Learning to rank is a new area in statistical learning, in parallel with learning for classification, regression, etc. The area is attracting broad interests recently, in part because there are many application issues which can be formalized as ranking, for example, web search, and in part because it is a novel learning task and there are many unknown issues which need to be addressed.

Previously, researchers have tried to transform the problem of ranking into that of classification and apply existing classification techniques to perform the task. For example, as classification techniques one can employ SVM, Boosting, and Neural Network, and this leads to the methods of Ranking SVM [15], RankBoost [12], and RankNet [4], previously proposed in the literature. Methods for employing the learning methods in web search have been proposed (e.g., [18]). Furthermore, methods for making the transformation more appropriate for search have also been studied (e.g., [5, 23, 31]). See also [30, 8, 11, 13] for other work.

More recently, a number of authors have proposed directly defining a loss function on list of objects and directly optimizing the loss function in learning [6, 34, 35, 26]. This approach formalizes the ranking problem in a more straightforward way and thus tends to be more effective.

All the learning to rank methods, however, are based on the assumption that there is no relation information between the objects that should be used in ranking. This is not the case in practice, for example, in search, as we pointed out. As a result, existing methods cannot be directly applied to such kind of setting. Simply extending existing methods to the setting would not work well, as will be seen in the experimental results section.

2.2 Using Relation Information in Search

Relation information between documents plays an important role in many web search tasks. For example, ranking web pages on the basis of importance, improving relevance ranking by using similarity information, diversifying search results.

Relation information has been used for importance ranking [22, 19]. The basic idea is that if a page has many inlinks, then the page is likely to be important, and the importance score can be propagated through the link graph. PageRank [22] and HITS [19] are well known algorithms for computing importance of web pages. They rank web pages based on the Markov chain model and authority-hub model respectively; both leverage the use of hyperlink (relation) information on the web.

Similarity between documents is useful information for search ranking as well. Subtopic Retrieval is an example [36]. In the task, given a query, the returned documents should cover as many subtopics as possible. If there are documents about the same subtopic, then only one document should be selected and ranked high. (See also [7].) In other search tasks, similarity between documents is used to boost relevance, for example, in Pseudo Relevance Feedback [29]. The technique first conducts a round of relevance ranking, assumes that the top ranked documents are relevant, and extracts new terms from the relevant documents. It then formulates a new query and conducts a second round of ranking. With two rounds of ranking, some relevant documents dropped in the first round can be ranked higher.

Topic Distillation is another example of using relation information in web search. Here, Topic Distillation refers to the search task in which we select a few pages that can best represent a topic by exploiting structure (relation) information on the web. It is found that propagating the relevance of a document to its neighborhood on the hyperlink graph can improve performance of Topic Distillation [27]. Furthermore, propagating the relevance of a web page to its parent pages can also boost the accuracy of the task [24].

Although relation information between documents has been used in search. So far there has been no previous work on learning to rank which leverages relation information in ranking. In this paper, we propose a new learning to rank method which utilizes both content information and relation information. We take Pseudo Relevance Feedback and Topic Distillation as examples.

2.3 Learning Using Relation Information

There are a number of learning problems in which relation information is used.

Semi-supervised learning on graph data is an example, which is a problem as follows. Given a graph (either directed or undirected), some of the nodes are labeled, we are to use the graph, i.e., relation information, to label the remaining nodes [38, 37, 3]. Some existing methods perform the task by only using relation information [38, 37], while the others try to use both content and relation information [3].

Cluster on graph data is another example of using relation information in learning [9, 28]. Some existing methods make use of similarity relation [28], while the others make use of co-occurrence relation [9].

The major difference between our work and these work is that ours is based on supervised learning, while they are based on semi-supervised learning or unsupervised learning.

There are also methods proposed for learning to rank network entities, such as papers in citation graph [1] and pages in Web graph [32]. In the methods, only relation information is utilized.

3. GENERAL FRAMEWORK

3.1 Motivation

There are many application problems in which one wants to learn a model to rank objects by using both content information and relation information. Let us take web search as example.

The central problem in web search is ranking. Given a query, we retrieve a number of web pages which contain the query from the index, rank the web pages based on a number of factors such as the relevance of the pages to the query, importance of the pages, diversities of the pages, and then present the top ranked pages (for example, 1000) to the users. The ranking function (model) is usually created and tuned in advance and off-line, using certain amount of labeled data (e.g., relevant, irrelevant).

Relevance mainly depends on the contents of the query and the web pages. If the query matches well against the content of a web page, for example, the query words occur many times in the page, then the relevance of the pages should be high. The content information can be and is widely used in search ranking.

Sometimes relation information between the web pages is also available, and we want to make effective use of it as well. For instance, similarities between web pages can be calculated, and if two pages are similar with each other and one of the pages is relevant to the query, then it is very likely that the other page is also relevant. If for some reason the former page is ranked in top position, while the latter is not, then we can use the similarity information to further boost the latter. This is close to the technique Pseudo Relevance Feedback in IR. In this paper, for simplicity we also refer to it as Pseudo Relevance Feedback. As another example, if two pages form a parent-child relation at a website, even both of them are relevant to the query (it is very likely that such phenomenon occurs), we may only want to rank high the parent page. The parent-child relationship between web pages can be found from the URL hierarchy of a website [25]. This is a task referred to as Topic Distillation at TREC.

The question then becomes how to combine both the content information and relation information in learning and ranking.

3.2 Formulation

We consider a new problem of learning to rank, referred to as learning to rank relational objects. In the task, we consider the use of both contents of objects and relations between objects.

Let X be an $n \times d$ matrix representing d dimensional feature vectors of n objects; each row corresponds to one object and each column corresponds to one feature. Let R denote an $n \times n$ matrix representing relationships between the n objects; each row and each column respectively correspond to one object. Let y be a vector representing ranking scores of the n objects.

In learning, we are given N training samples: (X_1, R_1, y_1), (X_2, R_2, y_2), ..., (X_N, R_N, y_N), drawn from an unknown joint probability distribution $P(X, R, y)$ where X, R, and y denote feature vectors of objects, relations between objects, and ranking scores of objects respectively. Here, each sample contains n objects to be ordered[1]. Ranking only happens within a sample which consists of a 'group of objects'. It does not make sense to make comparison between objects across samples.

Suppose that f is a function, such that

$$y = f(X, R). \tag{1}$$

The goal of learning is to select the best function $\widehat{f}$ from a function space $\mathscr{F}$ using the training data.

In prediction, given features X_t and relations R_t for n objects drawn from the same joint distribution, we output y_t using the learned function, i.e., $y_t = \widehat{f}(X_t, R_t)$. y_t actually represents the ranking scores of the given n objects having features X_t and relation R_t. Obviously we make use of both feature (content) information and relation information in the ranking function.

We define a loss function and formulate the learning problem as that of minimizing the total loss with respect to the given training data.

$$\widehat{f} = \arg \min_{f \in \mathscr{F}} \sum_{k=1}^{N} L(f(X_k, R_k), y_k)$$

where L denotes loss function.

We refer to the above learning problem as 'learning to rank relational objects'.

It is easy to see that the conventional 'learning to rank' problem is a special case of the current problem. Specifically, if we ignore the relations R, then we get the same function as that in conventional learning to rank,

$$f(X, R) = f(X).$$

3.3 Application to Web Search

Many problems in web search can be formalized as learning to rank relational objects. In this paper, we only consider two examples: Pseudo Relevance Feedback and Topic Distillation.

Given a query, there are n retrieved documents. X is a matrix representing the feature vectors derived from the query and the documents. R is a matrix representing the relations between the documents. y is a vector representing the ranking scores of the documents.

In training, the labeled data about N queries is given. Each consists of the feature vectors X, the relations R, and the 'true' ranking scores y.[2] We learn a ranking function f using the training data.

In ranking, given n documents of a new query, we use the trained ranking function to assign scores to the documents and sort them.

In the Pseudo Relevance Feedback defined in this paper, R is simply a matrix representing the similarities between the documents. In Topic Distillation, R is a matrix representing the parent-child relationship between web pages in a web site.

---

[1]For simplicity, we assume that all the samples contain the same number of objects. It is easy to generalize to the case in which different samples have different objects.

[2]Again, for simplicity, we assume that all the queries have the same number of retrieved documents n.

4. OUR METHOD

We propose a novel method for learning to rank relational objects.

4.1 Setting

First, we specify the ranking function (1) in the following way

$$y = f(h(X), R) \tag{2}$$

where X and R denote features and relations respectively, h is a function of X. That is to say, ranking based on relations is defined in the outer function f, while ranking based on contents is defined in the inner function h.

There are three settings for learning the nested ranking function.

Setting 1 : inner function h is predefined, but outer function f is to be learned

Setting 2 : outer function f is predefined, but inner function h is to be learned

Setting 3 : both inner function h and outer function f are to be learned

In this paper, we address setting 2. We leave settings 1 and 3 to future work since they are somewhat complex. Specifically, we define the ranking function as

$$y = f(h(X; \omega), R).$$

where ω is an unknown parameter. Then, learning becomes the following optimization problem:

$$\min_{\omega} \sum_{k=1}^{N} L(f(h(X_k; \omega), R_k), y_k)$$

where L denotes loss function, $f(h(X; \omega), R)$ denotes ranking function, and y denotes ground truth. Again in $f(h(X; \omega), R)$, the outer function of f is predefined, but the inner function of h with parameters ω is not. Now the key question is how to define the function f to integrate relation information.

4.2 Ranking Function

We propose defining f as a solution of minimizing the linear combination of two objectives.

$$f(h(X; \omega), R) = \arg\min_{z}\{l_1(h(X; \omega), z) + \beta l_2(R, z)\} \tag{3}$$

where z denotes any possible ranking scores, the first objective $l_1(h(X; \omega), z)$ measures the difference between h and z, and the second objective $l_2(R, z)$ measures the inconsistency between elements in z under R. Furthermore, β is a non-negative coefficient, representing the trade-off between the two objectives.

The first objective $l_1(h(X; \omega), z)$ can simply be

$$l_1(h(X; \omega), z) = \|h(X; \omega) - z\|^2 \tag{4}$$

where $\|.\|$ denotes L_2 norm.

When $\beta = 0$ and $h(X; \omega)$ is a linear model, the ranking model Eq.(3) degenerates to the model:

$$f(X, R; \omega) = h(X; \omega) = X\omega, \tag{5}$$

and the learning problem in Eq.(3) becomes that of learning of a linear ranking model such as linear Ranking SVM.

The second objective $l_2(R, z)$ may have different forms, depending on the types of the relationships between objects.

An intuitive explanation of function f in Eq.(3) is as follows. There are n objects. Each object receives a score from the function h. The objects then propagate the scores with each other on the basis of the relationships given in R (for example, R can be a undirected graph). The propagation must reach a stable state and the n objects obtain their final scores z. The propagation is defined as minimization of a total objective function, or an energy function. The total objective function represents a trade-off between maintaining local consistence with the output from h and maintaining global consistence with the restraints from R.

Similar techniques have been employed in graph based semi-supervised learning and unsupervised learning. To the best of our knowledge, this is the first time, the technique is used in a supervised learning setting. This is also the key idea of our method for learning to rank relational objects.

In the following subsections, we give explicit forms of the function f in different applications in which different relationships are defined and thus different objective functions $l_2(R, z)$ are utilized.

4.3 Pseudo Relevance Feedback

We consider a variant of Pseudo Relevance Feedback, in which we make use of both the relevance of documents to the query and the similarities between documents in ranking. We represent the similarity relationship between documents in an undirected graph, in which the nodes represent objects, and the weights on the edges represent the similarities between objects. Such kind of graph is widely used in clustering and semi-supervised learning. It is easy to see that we can use a matrix notation to represent the graph.

The second objective function $l_2(R, z)$ can then be defined as

$$l_2(R, z) = 1/2 \sum_{i} \sum_{j} R_{i,j}(z_i - z_j)^2 \tag{6}$$

where $R_{i,j}$ is the i-th row j-th column element of matrix R, and represents the similarity between documents i and j. z_i is the i-th element of vector z. The rationale behind is that by minimizing this objective we can guarantee that if two documents are similar, then their final ranking scores should also be similar. Specifically, the larger the value of $R_{i,j}$ is, the more similar the objects i, j are, and thus the closer the ranking scores z_i and z_j are.

With Eq.(4) and Eq.(6), the ranking model in Eq.(3) can be re-written as

$$f(X, R; \omega) = \arg\min_{z}\{\|h(X; \omega) - z\|^2 + \beta/2 \sum_{i} \sum_{j} R_{i,j}(z_i - z_j)^2\} \tag{7}$$

Let us denote the total objective in Eq.(7) as

$$l(z) = \|h(X; \omega) - z\|^2 + \frac{\beta}{2} \sum_{i} \sum_{j} R_{i,j}(z_i - z_j)^2$$

Let D be a diagonal matrix with $D_{i,i} = \sum_{j} R_{i,j}$. Then we can write the total objective as

$$l(z) = \|h(X; \omega) - z\|^2 + \beta z^T(D - R)z.$$

Note that $D - R$ is in fact the Laplacian matrix of the relationship graph. Setting the derivative of $l(z)$ with respect

to z to 0, we obtain

$$\frac{\partial l(z)}{\partial z} = 2(z - h(X;\omega)) + 2\beta(D - R)z = 0.$$

This yields

$$(I + \beta(D - R))z = h(X;\omega) \qquad (8)$$

where I denotes an $n \times n$ identity matrix. Since β is non-negative, thus matrix $I + \beta(D - R)$ is diagonally dominant, and so $I + \beta(D - R)$ is invertible, according to the Levy-Desplanques theorem. In this way, we obtain the explicit form of the ranking model for Pseudo Relevance Feedback as follows:

$$f(X, G;\omega) = (I + \beta(D - R))^{-1}h(X;\omega) \qquad (9)$$

For n objects, the time complexity of straightforwardly computing the ranking model is of order $O(n^3)$ and thus is expensive. The main cost of the computation comes from matrix inversion.

We employ the following technique to quickly carry out the computation. First, we note that solving the system of linear equation (8) is enough to obtain the ranking model. Let $A = I + \beta(D - R)$. If A is a banded matrix with bandwidth $k \ll n$, then the score z in Eq.(8) can be solved with time complexity $O(n)$ [14]. If R is a banded matrix, then A is also a banded matrix, and the bandwidth of A is the same as that of R. In order to make R a banded matrix, we should not compute the similarity between all document pairs, and in fact this is not necessary in practice. Instead, for each document we only consider the k nearest neighborhoods of it. As a result, R becomes a sparse matrix, which has at most k non-zero values in each row and each column. By Gibbs-Poole-Stockmeyer algorithm [20], we can convert a sparse matrix to a banded matrix with linear time. In this way, the time complexity of creating a relational ranking model becomes of linear order of n and thus is comparable with that of the existing learning to rank methods.

Note that the 'Pseudo Relevance Feedback (PRF)' in this paper slightly differs from the conventional Pseudo Relevance Feedback. Conventional PRF only considers the similarity between the top ranked documents and the other documents in the second round ranking. By contrast, our PRF in principle takes into consideration the similarity between all document pairs.

4.4 Topic Distillation

We consider Topic Distillation, in which we make use of both the relevance of web pages to the query and the parent-child relation between the web pages in a web site. We represent the parent-child relationship between web pages in a matrix R,

$$R_{i,j} = \begin{cases} 1 & \text{if object } i \text{ is the parent of } j, \\ 0 & \text{other.} \end{cases} \qquad (10)$$

Note that the matrix is asymmetric in the sense if i is the parent of j then the converse is not true.

The second objective function $l_2(R, z)$ can then be defined as

$$l_2(R, z) = \sum_i \sum_j R_{i,j} \exp(z_j - z_i) \qquad (11)$$

Note that each exponential function $\exp(z_j - z_i)$ only contributes when its corresponding $R_{i,j}$ equals one, which means

i is the parent of j. In that case, if i has a larger score than j, then the objective will be low; in contrast, if i has a smaller score than j, then the objective will be high. The relationship is represented as an exponential function. This is similar to the use of exponential objective function in supervised learning such as Boosting.

With Eq.(4) and Eq.(11), the ranking model in (3) can be written as

$$f(X, R;\omega) = \arg\min_z \{\|h(X;\omega) - z\|^2 + \beta \sum_i \sum_j R_{i,j}\exp(z_j - z_i)\} \qquad (12)$$

Let us denote the total objective in Eq.(12) as

$$l(z) = \|h(X;\omega) - z\|^2 + \beta \sum_i \sum_j R_{i,j}\exp(z_j - z_i)$$

It appears difficult to find an analytic solution of minimization of the total objective function. Here, we choose to make an approximation of the function.

First, we approximate $\exp(z_j - z_i)$ using the Taylor expansion[3]:

$$\exp(z_j - z_i) \approx 1 + (z_j - z_i) + \frac{1}{2}(z_j - z_i)^2.$$

Then, we approximate $l(z)$ as

$$l(z) \approx \|h(X;\omega) - z\|^2 + \beta \sum_i \sum_j R_{i,j}\left\{1 + (z_j - z_i) + \frac{1}{2}(z_j - z_i)^2\right\}.$$

We have

$$l(z) = \|h(X;\omega) - z\|^2 + \beta(g_0 + g_1^T z + z^T(D - R)z),$$

where $g_0 = \sum_i \sum_j R_{i,j}$, $g_{1k} = \sum_i R_{i,k} - \sum_j R_{k,j}, k = 1, 2, ..., n$, D is a diagonal matrix with $D_{k,k} = \frac{1}{2}(\sum_i R_{i,k} + \sum_j R_{k,j}), k = 1, 2, ..., n$. Setting the derivative of $l(z)$ with respect to z to 0, we obtain

$$\frac{\partial l(z)}{\partial z} = 2(z - h(X;\omega)) + \beta g_1 + \beta(2D - R - R^T)z = 0.$$

This yields

$$(2I + \beta(2D - R - R^T))z = 2h(X;\omega) - \beta g_1. \qquad (13)$$

Similarly, matrix $2I + \beta(2D - R - R^T)$ is invertible, according to the Levy-Desplanques theorem. In this way, we obtain the explicit form of the ranking model for Topic Distillation:

$$f(X, R;\omega) = (2I + \beta(2D - R - R^T))^{-1}(2h(X;\omega) - \beta g_1) \quad (14)$$

The time complexity of computing the ranking model is of order $O(n^3)$ for a general matrix R. In fact, in sitemap hierarchy, a webpage has at most one parent page, and so matrix R has at most n non-zero elements. That is, R is naturally very sparse. Similarly to Pseudo Relevance Feedback, for a sparse matrix R we can compute the ranking model in linear time.

5. OPTIMIZATION BASED ON SVM

The learning (optimization) task in Eq.(3) is specified, when the ranking function is defined, for example, as in Eq.(9) and Eq.(14). One can use different techniques to implement the optimization problem, such as SVM, Boosting and Neural Network. In this section we consider using

---

[3]In our experiments, we found that $|z_j - z_i|$ is very small, and so this approximation is reasonable.

the SVM technique. We refer to the method as 'Relational Ranking SVM'. For simplicity, we consider the linear function $h(X; \omega) = X\omega$.

5.1 Ranking SVM

We first make a review of Ranking SVM. Ranking SVM is a state of the art method for learning to rank, and its learning task is defined as the following quadratic programming problem:

$$\min_{\omega, \xi_{q,i,j}} \frac{1}{2} \|\omega\|^2 + c \sum_{q,i,j} \xi_{q,i,j}$$
$$\text{s.t.} \quad \omega^T x_{q,i} \geq \omega^T x_{q,j} + 1 - \xi_{q,i,j}, \qquad (15)$$
$$\forall x_{q,i} \succ x_{q,j}, \xi_{q,i,j} \geq 0$$

where $x_{q,i} \succ x_{q,j}$ implies that object i is ranked ahead of object j for query q in the training data, $\xi_{q,i,j}$ denotes slack variable, and $\|\omega\|^2$ denotes structural loss. For ease of explanation, we re-write the above optimization in a matrix form:

$$\min_{\omega, \xi_q} \frac{1}{2} \omega^T \omega + c \sum_q \mathbf{1_q^T} \xi_q$$
$$\text{s.t.} \quad \forall q \in Q, C_q f(X_q; \omega) \geq \mathbf{1}_q - \xi_q, \qquad (16)$$
$$f(X_q; \omega) = X_q \omega, \xi_q \geq 0$$

where Q is the set of training queries, C_q denotes a constraint matrix for query q, $\mathbf{1_q}$ denotes a vector with all the elements being 1, and its dimension is the same as that of ξ_q. Each row of C_q represents a pairwise constraint: one element is 1, one element is -1, and the other elements are all 0. For example, for query q, if document 1 is ranked ahead of document 3, and document 2 ahead of document 4, then we have

$$C_q = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}$$

Note that Eq.(16) can be written in a similar form as Eq.(3):

$$\min_{\omega, \xi_q} \frac{1}{2} \omega^T \omega + c \sum_q \mathbf{1}_q^T [\mathbf{1}_q - C_q f(X_q; \omega)]_+$$
$$\text{s.t.} \quad \forall q \in Q, f(X_q; \omega) = \arg\min_z \|X_q \omega - z\|^2 \quad (17)$$

where $[x]_+$ indicates the positive part of x. This implies that our formulation of the learning problem in Eq.(3) is quite general.

5.2 Relational Ranking SVM

We next describe Relational Ranking SVM. Combining Eq.(9) with Eq.(16), we can get the optimization problem of Relational Ranking SVM for Pseudo Relevance Feedback,

$$\min_{\omega, \xi_q} \frac{1}{2} \omega^T \omega + c \sum_q \mathbf{1}_q^T \xi_q$$
$$\text{s.t.} \quad \forall q \in Q, C_q f(X_q, R_q; \omega) \geq \mathbf{1}_q - \xi_q, \xi_q \geq 0 \quad (18)$$
$$f(X_q, R_q; \omega) = (I + \beta(D_q - R_q))^{-1} X_q \omega$$

This is a Linear Constrained Quadratic Programming problem, and can be solved by employing existing optimization techniques.

Similarly, for Relational Ranking SVM in Topic Distillation, we have an optimization problem as follows:

$$\min_{\omega, \xi_q} \frac{1}{2} \omega^T \omega + c \sum_q \mathbf{1}_q^T \xi_q$$
$$\text{s.t.} \quad \forall q \in Q, C_q f(X_q, R_q; \omega) \geq \mathbf{1}_q - \xi_q, \xi_q \geq 0 \quad (19)$$
$$f(X_q, R_q; \omega) = (2I + \beta(2D_q - R_q - R_q^T))^{-1}(2X_q \omega - \beta g_{q,1})$$

where $g_{q,1}$ is the vector g_1 in Eq.(13) for query q. This problem can also be solved by employing existing optimization techniques.

6. EXPERIMENTS

We applied Relational Ranking SVM to the tasks of Pseudo Relevance Feedback and Topic Distillation. We used LETOR [21] in our experiments, which is a dataset created for learning to rank research[4]. We used OHSUMED in LETOR for Pseudo Relevance Feedback and TREC in LETOR for Topic Distillation. As evaluation measure, we utilized NDCG@n (Normalized Discounted Cumulative Gain) [17].

6.1 OHSUMED: Pseudo Relevance Feedback

We compared the performances of Relational Ranking SVM and several baseline methods in Pseudo Relevance Feedback using the OHSUMED data set in LETOR.

6.1.1 Data Set

The OHSUMED data set in LETOR has been derived from the OHSUMED benchmark data [16] for information retrieval research. The document collection is a subset of MEDLINE, a database on medical publications. The collection consists of 348,566 records (out of over 7 million) from 270 medical journals during the period of 1987-1991. The fields of a record include title, abstract, MeSH indexing terms, author, source, and publication type.

There are 106 queries in OHSUMED data set, each with a number of associated documents. The relevance degrees of documents with respect to the queries are judged by humans, on three levels: *definitely relevant*, *partially relevant*, or *not relevant*. There are in total 16,140 query-document pairs with relevance judgments. Each query-document pair is represented by a 25 dimension feature vector. For the details of the features, please refer to [21].

Similarity between documents is provided as relation information. The similarity between two documents is calculated in the following way. First stop words are removed from the documents. Each document is represented by a term vector in the vector space model [2]. The similarity $R_{i,j}$ between two documents i and j is defined as cosine between the term vectors of the two documents. Note that a term vector differs from a feature vector in learning; the former is a function of document, while the latter a function of query document pair.

6.1.2 Experiment Procedure

As baseline, we adopted Ranking SVM and Ranking SVM plus relation [10]. For reference purposes, we also tested BM25 and Pseudo Relevance Feedback based on BM25. For BM25 and Pseudo Relevance Feedback, we used the tools provided in Lemur toolkit[5].

In Ranking SVM, we only use content information. In comparison with this baseline, we can see whether Relational Ranking SVM can effectively leverage relation information to perform better ranking. Actually, the results of Ranking SVM are already provided in LETOR.

In Ranking SVM plus relation, we make use of both content information and relation information. Following the proposal in [10], we conduct regularization on the scores output by Ranking SVM, using similarities between documents . In comparison with this baseline, we can verify whether it is better to integrate relation information into learning process, as in Relational Ranking SVM.

---

[4]The data set can be downloaded from http://research.microsoft.com/users/LETOR/.
[5]http://www.lemurproject.org/

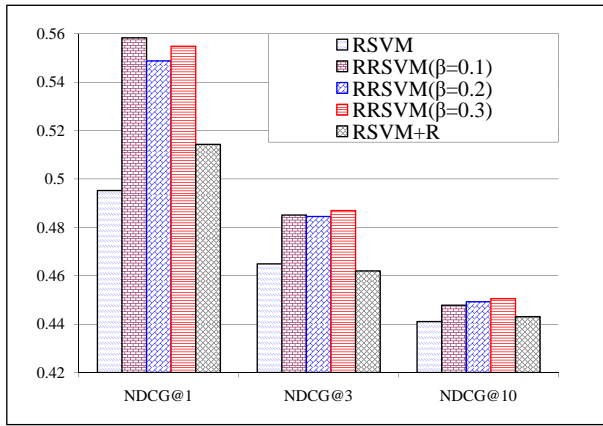**Figure 1: Comparison with Learning Methods**

Figure 2: Comparison with Non-Learning Methods

We conducted 5 fold cross validation experiments, using the partitions provided in LETOR.

For all the SVM models in the experiments, we employed Linear SVM, because the result of Ranking SVM in the LETOR data set is based on Linear Ranking SVM.

There is a parameter β in the ranking model Eq.(3). In the experiments, we heuristically set β as $0.1, 0.2, 0.3$, and conducted experiments with the values for Relational Ranking SVM and Ranking SVM plus relation.

6.1.3   Experimental Results

Figure 1 show the average performances of Ranking SVM, Ranking SVM plus relation, and Relational Ranking SVM over 5 folds. Here 'RSVM' stands for Ranking SVM, 'RSVM+R' stands for Ranking SVM plus relation and 'RRSVM' stands for Relational Ranking SVM.

For all the β values, RRSVM performs much better than RSVM and RSVM+R in terms of NDCG at all positions. Particularly for NDCG@1, RRSVM works significantly better than RSVM, with more than 10% relative improvement. Furthermore, RRSVM also works better than RSVM+R.

Figure 2 show the results of BM25 and Pseudo Relevance Feedback (PRF) based on BM25. We can see that PRF is better than BM25. RRSVM significantly outperforms BM25 and PRF, which verifies the correctness of the claim that learning methods using relation works better than non-learning methods using relation.

In summary, RRSVM can really outperform the baseline methods.

6.1.4   Discussion

Table 1 and 2 show the top 10 results of RSVM and RRSVM for query "FIBROMYALGIA/FIBROSITIS, DIAGNOSIS AND TREATMENT" separately. The documents in red are 'definitely relevant', documents in blue are 'partially relevant', and documents in black are 'not relevant'.

It is easy to find that RRSVM is better than RSVM for this query. Document 114913 is a 'definitely relevant' document. RSVM is only able to rank it to position 5. This document is in fact very similar to document 142171 which is ranked at position 1. Using the similarity information between them, RRSVM can effectively boost it to top 3. This example shows why we can improve the performance of relevance by using similarity information.
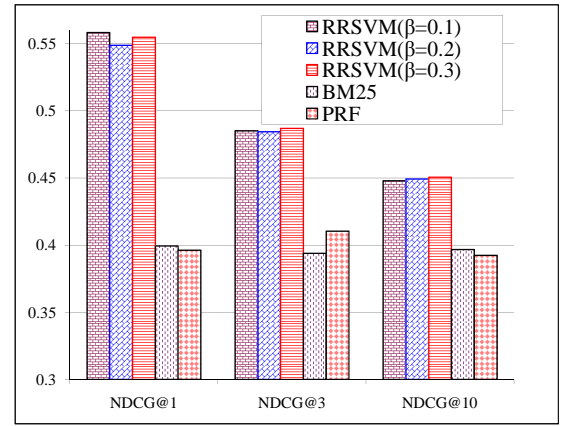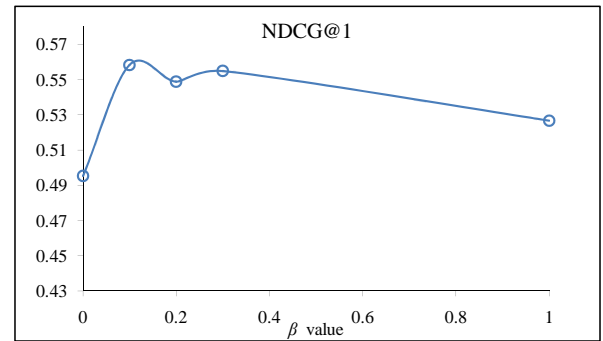

Figure 3: RRSVM with Different β Values

The coefficient β represents a trade-off between the uses of content information and relation information. Figure 3 shows the performance curve of RRSVM with different β values. Note that when $\beta = 0$ RRSVM degenerates to RSVM. RRSVM achieves the highest NDCG@1 value when $\beta = 0.1$. When β gets larger, RRSVM's performance begins to deteriorate. That is, we need to select relatively small β for this task.

6.2   TREC: Topic Distillation

We compared the performances of Relational Ranking SVM and several baseline methods in Topic Distillation using the TREC2004 data set in LETOR.

6.2.1   Data Set

In TREC 2004, there was a special track for web search. The goal of this track was to investigate the behaviors of search methods when the document collection is from the web. The track used the .gov data as document collection, which is from a crawl of the .gov domain on January, 2002. There are in total 1,053,110 HTML documents, and 11,164,829 hyperlinks. The track also provided queries and relevance judgments on documents.

The TREC dataset in LETOR has been derived from the .gov collection. There are 75 queries, each associated with about 1,000 documents. The relevance degrees of documents with respect to the queries are offered by TREC, on two levels: *relevant* or *not relevant*. There are 44 features defined as functions over a query-document pair [21].

Table 1: Top 10 Results of RSVM

Doc ID	Title
142171	Current pharmacologic therapy of arthritis.
264930	Incidence of transient nephrotic syndrome during pregnancy in diabetic women with and without pre-existing microalbuminuria
180409	Molecular relationships between the class II HLA antigens and susceptibility to rheumatoid arthritis
103801	Impaired carbohydrate metabolism of polymorphonuclear leukocytes in glycogen storage disease Ib.
114913	Current pharmacologic management of narcolepsy.
188977	Reiter's syndrome precipitated by a typhoid vaccination
197502	Neuropeptides in synovium of patients with rheumatoid arthritis and osteoarthritis.
114422	Electromyographic abnormalities in neurologic injury associated with pelvic fracture: case reports and literature review.
288849	Synthesis and release of phospholipase A2 by unstimulated human articular chondrocytes.
266203	Role of the general practitioner in managing patients with myocardial infarction

Table 2: Top 10 Results of RRSVM

Doc ID	Title
142171	Current pharmacologic therapy of arthritis.
264930	Incidence of transient nephrotic syndrome during pregnancy in diabetic women with and without pre-existing microalbuminuria
114913	Current pharmacologic management of narcolepsy.
114422	Electromyographic abnormalities in neurologic injury associated with pelvic fracture: case reports and literature review.
103801	Impaired carbohydrate metabolism of polymorphonuclear leukocytes in glycogen storage disease Ib.
288849	Synthesis and release of phospholipase A2 by unstimulated human articular chondrocytes.
180409	Molecular relationships between the class II HLA antigens and susceptibility to rheumatoid arthritis
264301	Cardiopulmonary consequences of obstructive sleep apnea.
197502	Neuropeptides in synovium of patients with rheumatoid arthritis and osteoarthritis.
315397	Defective CD2 pathway T cell activation in systemic lupus erythematosus

Relation between web pages in a web site is given as a matrix R. The element $R_{i,j}$ equals 1 if page i is parent of page j, and equals 0 for other cases.

6.2.2 Experiment Procedure

As baseline methods, we used Ranking SVM (RSVM).

We also tested existing non-learning method of sitemap based relevance propagation [24]. The basic idea of sitemap based relevance propagation is to use the relevance of a child page to enhance the relevance of its parent page. This method makes use of the parent-child relationship. There are two variants of the method: sitemap based term propagation ('ST' for short) and sitemap based score propagation ('SS' for short).

Considering the fact that Ranking SVM does not use relation information, we tested another method, Ranking SVM plus relation (RSVM+R). In this method, we use the ranking score of a child page output by RSVM to enhance the ranking score of its parent also output by RSVM. The idea is similar to that of sitemap based relevance propagation [24].

We conduced 5-fold cross validation experiments, using the partitions in LETOR. For Ranking SVM, we can make use of the results of it provided in LETOR. For all the SVM models in the experiment, we employed Linear SVM. This is because the LETOR data set offers results of Linear Ranking SVM.

In the experiments, we heuristically set β as $0.1, 0.2, 0.3$, and applied them to Relational Ranking SVM.

6.2.3 Experimental Results

Figure 4 show the average performances of RRSVM, RSVM, and RSVM+R in 5-fold cross validation.

RRSVM outperforms RSVM and RSVM+R, in all settings of β, especially when $\beta = 0.1$. Although RSVM+R boosts NDCG@1 score higher than RSVM, its NDCG@3

and NDCG@10 scores are not as good as those of RSVM. The results indicate that learning relation information can indeed boost the performance of Topic Distillation.

Moveover, Figure 5 show the average performances of RRSVM, ST and SS in 5-fold cross validation.

We can see that RRSVM work much better than ST and SS in terms of NDCG. For example, The NDCG values of RRSVM are 10 points higher than the relevance propagation methods, which represents 30% relative improvements. The results show that the learning based method can achieve better accuracy than non-learning methods.
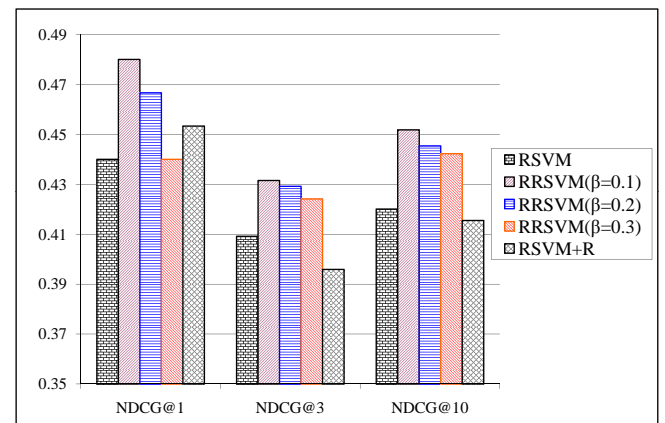

Figure 4: Comparison with Learning Methods

6.2.4 Discussions

We investigated the reason that RRSVM can achieve better results than RSVM and concluded that it is because RRSVM can successfully leverage the relation information.
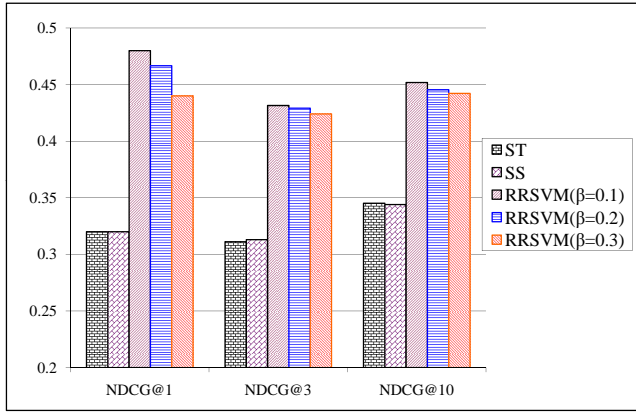
Figure 5: Comparison with Non-Learning Methods

Table 3: Top 10 Results of RSVM

Doc ID	URL
4074	http://www.nasa.gov/
880	http://liftoff.msfc.nasa.gov/
22352	http://www.hq.nasa.gov/osf/heds/
20325	http://www.hq.nasa.gov/osf/
409215	http://spacelink.nasa.gov/nasa.../.index.html
53139	http://core.nasa.gov/cu.html
89693	http://www.hq.nasa.gov/osf/heds/hedsplan.html
2443	http://sse.jpl.nasa.gov/
183570	http://www.lanl.gov/csse/
375621	http://www.pnl.gov/microcats/apps/space/

Table 4: Top 10 Results of RRSVM

Doc ID	URL
4074	http://www.nasa.gov/
880	http://liftoff.msfc.nasa.gov/
20325	http://www.hq.nasa.gov/osf/
22352	http://www.hq.nasa.gov/osf/heds/
409215	http://spacelink.nasa.gov/nasa.../.index.html
183570	http://www.lanl.gov/csse/
2443	http://sse.jpl.nasa.gov/
899308	http://technologyplan.nasa.gov/default.cfm?id=8.0
375621	http://www.pnl.gov/microcats/apps/space/
692871	http://www.jpl.nasa.gov/forum/

Figure 6: RRSVM with Different β Values

Table 3 and 4 show the case of query "space exploration". There are ten results returned by each of the methods RRSVM and RSVM. The answer pages for this query are red colored. In Table 3, the answer page (id 20325) is ranked below its child page (id 22352) by RSVM, because its content feature is not so strong as that of its child page. In Table 4, the answer page (id 20325) is ranked higher than its child page (id 22352), because RRSVM can effectively use the parent-child relation information to boost the position of it.
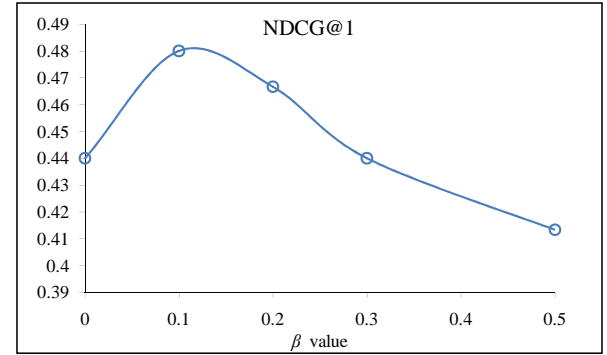
The coefficient β represents a trade-off between the uses of content information and relation information. Figure 6 shows the performance curve of RRSVM with different β values. Note that with $\beta = 0$ RRSVM degenerates to RSVM. Again, RRSVM achieves the highest NDCG@1 score at $\beta = 0.1$. When β gets larger, RRSVM' performance begins to deteriorate. That is, we need to select relatively small β for this task.

7. CONCLUSIONS

In this paper, we have proposed a new and general problem referred to as learning to rank relational objects, in which the ranking model is defined as a function of both content information and relation information of objects. Learning to rank relational objects can be applied to a large variety of problems in web search, such as Pseudo Relevance Feedback and Topic Distillation. We have focused on one setting of the learning task and proposed a new learning method for it on the basis of optimization. We have applied the method to the above two web search tasks by defining suitable ranking models. Furthermore, we have developed SVM techniques for solving the optimization problem. Experimental results on two public data sets show that the proposed method performs significantly better than the baseline methods.

There are still many issues which need further investigations. (1) We have assumed that all the training queries and documents are labeled. It is interesting to study how to perform learning to rank relational objects when some of the queries and documents are unlabeled. (2) We have looked at one setting of the problem (setting 2 in Section 4.1), it is interesting to know how to address the learning problems in the other settings (setting 1 and 3). (3) We have formalized the learning task as an optimization problem with a nested ranking function. It is worth to see whether there are any other approaches. (4) We have proposed one learning method to solve the optimization issue. The generalization ability of the method is not known. (5) We have applied the SVM techniques to the problem. It is also interesting to see how to apply other techniques such as Boosting and Neural Network here. (6) We have studied two applications: Pseudo Relevance Feedback and Topic Distillation. We also need to look at other web search tasks.

8. ACKNOWLEDGMENTS

9.   REFERENCES

[1] A. Agarwal, S. Chakrabarti, and S. Aggarwal. Learning to rank networked entities. In *KDD '06*, pages 14–23, New York, NY, USA, 2006. ACM Press.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, May 1999.

[3] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7:2399–2434, 2006.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 89–96, New York, NY, USA, 2005. ACM Press.

[5] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking svm to document retrieval. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 186–193, New York, NY, USA, 2006. ACM Press.

[6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 129–136, New York, NY, USA, 2007. ACM Press.

[7] W. Dai and R. Srihari. Minimal document set retrieval. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 752–759, New York, NY, USA, 2005. ACM Press.

[8] H. M. de Almeida, M. A. Gonçalves, M. Cristo, and P. Calado. A combined component approach for finding collection-adapted ranking functions based on genetic programming. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 399–406, New York, NY, USA, 2007. ACM.

[9] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD '01*, pages 269–274, New York, NY, USA, 2001. ACM Press.

[10] F. Diaz. Regularizing query-based retrieval scores. *Information Retrieval*, 2007.

[11] W. Fan, M. D. Gordon, and P. Pathak. A generic ranking function discovery framework by genetic programming for information retrieval. *Inf. Process. Manage.*, 40(4):587–602, 2004.

[12] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969, 2003.

[13] X. Geng, T.-Y. Liu, T. Qin, and H. Li. Feature selection for ranking. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 407–414, New York, NY, USA, 2007. ACM.

[14] G. H. Golub and C. F. V. Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[15] R. Herbrich, T. Graepel, and K. Obermayer. Support vector learning for ordinal regression. In *ICANN1999*, pages 97–102, 1999.

[16] W. Hersh, C. Buckley, T. J. Leone, and D. Hickam. Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *SIGIR '94*, pages 192–201, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[17] K. Jarvelin and J. Kekanainen. IR evaluation methods for retrieving highly relevant documents. In *SIGIR2000*, pages 41–48, 2000.

[18] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, New York, NY, USA, 2002. ACM Press.

[19] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[20] J. G. Lewis. Algorithm 582: The gibbs-poole-stockmeyer and gibbs-king algorithms for reordering sparse matrices. *ACM Trans. Math. Softw.*, 8(2):190–194, 1982.

[21] T.-Y. Liu, J. Xu, T. Qin, W.-Y. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, 2007.

[22] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[23] T. Qin, T.-Y. Liu, W. Lai, X.-D. Zhang, D.-S. Wang, and H. Li. Ranking with multiple hyperplanes. In *SIGIR '07*, pages 279–286, New York, NY, USA, 2007. ACM Press.

[24] T. Qin, T.-Y. Liu, X.-D. Zhang, Z. Chen, and W.-Y. Ma. A study of relevance propagation for web search. In *SIGIR '05*, pages 408–415, New York, NY, USA, 2005. ACM Press.

[25] T. Qin, T.-Y. Liu, X.-D. Zhang, G. Feng, D.-S. Wang, and W.-Y. Ma. Topic distillation via sub-site retrieval. *Information Processing & Management*, 43(2):445–460, 2007.

[26] T. Qin, X.-D. Zhang, M.-F. Tsai, D.-S. Wang, T.-Y. Liu, and H. Li. Query-level loss functions for information retrieval. *Information Processing & Management*, 2007.

[27] A. Shakery and C. Zhai. A probabilistic relevance propagation model for hypertext retrieval. In *CIKM2006*, pages 550–558, New York, NY, USA, 2006. ACM Press.

[28] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[29] T. Tao and C. Zhai. Regularized estimation of mixture models for robust pseudo-relevance feedback. In *SIGIR '06*, pages 162–169, New York, NY, USA, 2006. ACM.

[30] A. Trotman. Learning to rank. *Inf. Retr.*, 8(3):359–381, 2005.

[31] M.-F. Tsai, T.-Y. Liu, T. Qin, H.-H. Chen, and W.-Y. Ma. Frank: a ranking method with fidelity loss. In *SIGIR '07*, pages 383–390, New York, NY, USA, 2007. ACM Press.

[32] A. C. Tsoi, G. Morini, F. Scarselli, M. Hagenbuchner, and M. Maggini. Adaptive ranking of web pages. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 356–365, New York, NY, USA, 2003. ACM.

[33] E. Voorhees and D. Harman. *TREC: Experiment and Evaluation in Information Retrieval*. MIT Press, 2005.

[34] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *SIGIR '07*, pages 391–398, New York, NY, USA, 2007. ACM Press.

[35] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *SIGIR '07*, pages 271–278, New York, NY, USA, 2007. ACM Press.

[36] C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 10–17, New York, NY, USA, 2003. ACM Press.

[37] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency, 2003. In 18th Annual Conf. on Neural Information Processing Systems.

[38] D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 1036–1043, New York, NY, USA, 2005. ACM.