Learning to Rank with Ties

Ke Zhou, Gui-Rong Xue
Dept. of Computer Science and Engineering
Shanghai Jiao-Tong University
No. 800, Dongchuan Road,
Shanghai, China 200240
{zhouke,grxue}@apex.sjtu.edu.cn

Hongyuan Zha
College of Computing
Georgia Institute of Technology
Atlanta, GA 30032
zha@cc.gatech.edu

Yong Yu
Dept. of Computer Science and Engineering
Shanghai Jiao-Tong University
No. 800, Dongchuan Road,
Shanghai, China 200240
yyu@apex.sjtu.edu.cn

ABSTRACT

Designing effective ranking functions is a core problem for information retrieval and Web search since the ranking functions directly impact the relevance of the search results. The problem has been the focus of much of the research at the intersection of Web search and machine learning, and learning ranking functions from preference data in particular has recently attracted much interest. The objective of this paper is to empirically examine several objective functions that can be used for learning ranking functions from preference data. Specifically, we investigate the roles of ties in the learning process. By ties, we mean preference judgments that two documents have equal degree of relevance with respect to a query. This type of data has largely been ignored or not properly modeled in the past. In this paper, we analyze the properties of ties and develop novel learning frameworks which combine ties and preference data using statistical paired comparison models to improve the performance of learned ranking functions. The resulting optimization problems explicitly incorporating ties and preference data are solved using gradient boosting methods. Experimental studies are conducted using three publicly available data sets which demonstrate the effectiveness of the proposed new methods.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Retrieval functions*; H.4.m [**Information Systems**]: Miscellaneous—*Machine learning*

General Terms

Algorithms, Experimentation, Theory

Keywords

Ranking function, machine learning, preference learning, paired comparison, ties, functional gradient descent, gradient boosting

SIGIR'08, July 20–24, 2008, Singapore.

1. INTRODUCTION

The ranking problem has been widely explored in many fields including information retrieval and web search, recommendation systems and sports such as chess playing. Ranking documents with respect to a given query is the most important problem for designing effective web search engines because the ranking functions directly influence the relevance of the search results. Several methodologies of designing ranking functions have been explored in the past by information retrieval researchers, including the vector space model [24] and language modeling based approaches [19,25]. Those methods have been proven to be effective in both experiments and real world applications. Due to the large number of heterogeneous features employed in current ranking functions, recent methods for document ranking tend to rely heavily on discriminative machine learning techniques: training sets are generated and ranking functions are constructed by fitting the training data. Some of these ranking algorithms, such as RankSVM [5, 16] and RankBoost [9], have shown encouraging improvements in performance.

One effective approach for learning ranking functions is based on learning from preference data and is called preference learning. In the case of learning to rank web documents, preference data are given in the form that one document is more relevant than another with respect to a given query. Such preference data can be generated from user clickthroughs, for example, and they also have the advantage of capturing user searching behaviors and preferences in a more timely manner [18]. Although several methods for learning ranking functions from preference data have been proposed [9, 16, 30], the existing methods have largely ignored one important type of preference relations, i.e., *ties*. A tie represents a relevance judgment indicating that two documents are of the same degree of relevance with respect to a query. In a sense, ties complement the preference data and they provide a different angle for learning a ranking function: preference data focus on the discriminative qualities of documents with different degrees of relevance, while ties provide information on the common characteristics of documents with the same degree of relevance.

In this paper, we consider the problem of incorporating ties in the context of preference learning. We propose a novel framework to explore the relevance judgments of ties for learning ranking functions. Both of the preference data

and the ties are naturally integrated in this framework by modeling the ties in a principled fashion. In particular, we explore statistical models for paired comparisons to derive the objective functions for our learning framework. Two approaches for paired comparisons are applied to model the ties and the preference data in this study, but other models can naturally be adapted to our framework as well. We then apply gradient descent in function spaces to optimize the proposed objective functions in order to learn the ranking functions [10, 30].

Our experimental studies on three publicly available data sets show that the performance of the learned ranking functions can be significantly improved by taking ties into account, so ties should be employed when they are available. More interestingly, we find that the performance improvement at the top-ranked results is more significant by including ties. This is because both the shared features of relevant documents and the discriminative qualities between relevant and irrelevant documents are captured by including ties. Thus incorporating ties are especially beneficial to web search engines since they tend to focus more on the top-ranked results. Another interesting observation is that ties are more effective when multiple levels of relevance judgements are available.

The rest of this paper is organized as follows: In Section 2, related studies of learning to rank are briefly surveyed. Then we analyze the properties of ties in Section 3. The loss functions of our framework are proposed based on the statistical models for paired comparisons discussed in Section 4 and 5. In Section 6, we discuss functional gradient descent methods for optimizing the proposed objective functions. In Section 7, we report and analyze the results of experimental studies on three publicly available data sets. Finally, we conclude this work and point out some directions for future research.

2. RELATED WORK

Many methods for designing ranking functions have been proposed in the past years. Vector space model [24] represents queries and documents as vectors of features, so the degree of relevance can be calculated by the similarity between feature vectors of the documents and those of the queries. The Okapi BM25 ranking function is proposed based on the vector space model and applied in many information retrieval systems [23]. Language modeling based methods consider the relevance of a document with respect to a query in a probabilistic framework and estimate the parameters in probability models that describe whether a document is relevant to a query [25].

In recent years, the ranking problem is frequently solved under the supervised machine learning framework [3, 7, 9, 16, 28–30]. This learning to rank approaches are capable of combining different kinds of features to train ranking functions. These approaches to learning ranking functions are usually based on the advanced techniques developed for other machine learning tasks. The problem of ranking is usually formulated as that of learning a ranking function from pair-wise preference data. The idea is to minimize the number of contradicting pairs in training data. For example, RankSVM [16] uses support vector machines to learn a ranking function from preference data. RankNet [3] applies neural network and gradient descent to obtain a ranking function. RankBoost [9] applies the idea of boosting to construct an efficient ranking function from a set of weak ranking functions. The studies reported in [30] proposed a framework called GBRank using gradient descent in function spaces [10, 11], which is able to deal with complicated features in the context of web search. Other methods for learning ranking functions are proposed in [4, 12]

Another approach to learning a ranking function address the problem of optimizing the loss function directly related to the performance measures of information retrieval, such as precision, mean average precision and Normalized Discount Cumulative Gain [6, 17, 28, 29]. The idea of these methods is to obtain a ranking function that is optimal with respect to some information retrieval performance measure. Since their objective functions are directly related to the performance measures, these methods show good performance in many practical situations. However, most of the performance measures are defined by absolute relevance judgements. As a result, these methods require the absolute relevant judgements, i.e., labeled data, in order to learn ranking functions. Considering the fact that collecting absolute relevant judgements is expensive and time-consuming, the application of these method can be limited.

3. EFFECTIVENESS OF TIES

Both ties and preference data arise in many scenarios of learning ranking functions for Web search engines. Preference data are usually extracted from two sources: user clickthroughs and absolute relevant judgments.

Clickthroughs. Commercial search engines can readily obtain large amounts of data recording the interactions between the users and the search results of search engines. Specifically, user clickthroughs can be used to analyze the preferences of users over the search results. Since this approach does not require manual labeling, it is the most frequently applied method to extract preference data. There are several studies on inferring preference data from clickthroughs [18]. Most of these methods make use of heuristic rules as well as the statistics of user behaviors, such as clickthrough rates.

Absolute relevant judgments. In absolute relevance judgments labels are assigned to each documents indicating explicitly the degree of the relevance of the document with respect to a query. For example, four relevance levels representing *prefect, excellent, good* and *bad* can be applied to each document. Whether a document is preferred to others can be directly inferred form the labels of absolute relevance judgments. Although those judgments are more expensive to obtain compared with those extracted from clickthroughs, they tend to contain less noise. Interestingly, the results of [30] show that it is more effective to transform the absolute relevant judgments into preference data before learning ranking functions.

An important observation is that ties can also be obtained from these two sources. For clickthroughs, heuristic rules can also be developed for extracting ties. For example, a tie can be generated if the clicks of two documents frequently occur together. In the case of absolute relevant judgements, a straightforward approach is to consider pairs of documents with the same grade of relevance as ties. Since ties can be

extracted through methods similar to the methods for preference data generation, training data for preference learning should be expanded to include ties. More importantly, we will show that incorporating ties leads to improvements in the performance of the learned ranking functions.

Existing preference learning methods only consider preference data and ignore the extra constraints that are provided by ties. To illustrate the effect of ties, we consider a toy problem shown in Figure 1. Suppose that we need to rank four documents d_1, d_2, d_3 and d_4 *in that order* with respect to some given query. For the sake of clarify, we consider the simple case of ranking those documents by their projection on a straight line. Assume we have the preference data: d_1 is preferred to d_3 and d_2 is preferred to d_4, but the preference relation between pairs (d_1, d_4) and (d_2, d_3) are not available. Such kind of situations usually occur when the preference data are extracted from user clickthroughs. The ranking function obtained may be represented by the thin line in the figure. We observe that the pair d_2 and d_3 are ranked incorrectly. However, if we include ties between pairs (d_1, d_2) or (d_3, d_4), the result of ranking is improved. The tie between (d_1, d_2) requires that d_1 and d_2 have the same relevant scores (similarly for d_3 and d_4). As a result, the ranking function becomes the thick line in the figure. In this case, ties provide additional descriptions of the user preference and the ranking function is refined by utilizing these information. From the above example, we can see that ties indeed provide new information about the ranking function and can enhance the performance of learned ranking functions.

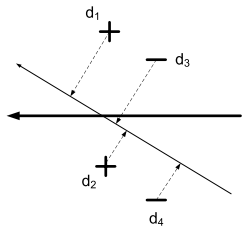

Figure 1: Illustration of ties

4. MODELS FOR PAIRED COMPARISONS WITH TIES

We now examine the question of how to make use of ties in learning ranking functions. We will propose a novel framework that integrates both ties and preferences data in a single objective function. To this end, let $\mathcal{D} = \{d_1, d_2, \ldots, d_c\}$ be the set of feature vectors of all query-document pairs in consideration.[1] We denote ties and preferences as follows: Given the feature vectors of query-document pairs x and y, let $x \succ y$ denote that x is preferred to y and $x = y$ denote that x and y is preferred equally.

The basic ingredient of our framework is the statistical models for paired comparisons and they provide a principled approach for modeling ties. Those models have been widely used in statistics, psychology and machine learning, two of the more well-known ones are Bradley-Terry model and Thurstone-Mosteller model [2,26]. Most of these models can be unified under the framework of general linear models as discussed below [8].

[1] A detailed discussion of the various types features extracted for query-document pairs can be found in [30].

4.1 General Linear Models

Given a nondecreasing function $F : \mathcal{R} \mapsto \mathcal{R}$ satisfying $F(+\infty) = 1$, $F(-\infty) = 0$ and $F(x) = 1 - F(-x)$, the *probability* that document x is preferred to document y is expressed as:

$$\mathrm{P}(x \succ y) = F(h(x) - h(y)) \tag{1}$$

where $h(x)$ and $h(y)$ are the scores of document x and y given by the ranking function h, x is ranked higher than y if $h(x) > h(y)$. The original general linear models do not have the ability to express the case that two items are equally preferred. However, it can be extended to incorporate ties as follows:

$$\mathrm{P}(x \succ y) = F(h(x) - h(y) - \epsilon) \tag{2}$$

$$\mathrm{P}(x = y) = F(h(x) - h(y) + \epsilon) - F(h(x) - h(y) - \epsilon) \tag{3}$$

The parameter ϵ is a threshold that controls the probabilities for ties. When the parameter $\epsilon = 0$, the new model is identical to the original general linear models. In this study, we apply two special cases of general linear models: the Bradley-Terry model and the Thurstone-Mosteller model.

4.2 Bradley-Terry Model

The Bradley-Terry model is widely used for paired comparisons, and its first use in learning ranking functions appears in [3]. Under this model, the probability of preference for a pair is given as follows:

$$\mathrm{P}(x \succ y) = \frac{e^{h(x)}}{e^{h(x)} + e^{h(y)}} \tag{4}$$

It can be observed that the Bradley-Terry model is a special case of the general linear model by letting $F(x) = \frac{1}{1+\exp(-x)}$. So the Bradley-Terry model can be extended to incorporate ties by substituting $\frac{1}{1+\exp(-x)}$ for $F(x)$ in Equation (2) and (3) [22]:

$$\mathrm{P}(x \succ y) = \frac{e^{h(x)}}{e^{h(x)} + \theta e^{h(y)}} \tag{5}$$

$$\mathrm{P}(x = y) = \frac{(\theta^2 - 1)e^{h(x)}e^{h(y)}}{(e^{h(x)} + \theta e^{h(y)})(e^{h(y)} + \theta e^{h(x)})} \tag{6}$$

Similar as in the general linear models, the parameter $\theta = e^{\epsilon}$ is a threshold that controls the probabilities of ties. When the parameter $\theta = 1$, the new model above is identical to the original Bradley-Terry model in Equation (4).

4.3 Thurstone-Mosteller Model

The Thurstone-Mosteller model is another probabilistic model for paired comparisons. Unlike the Bradley-Terry model, the Thurstone-Mosteller model is based on the Gaussian distribution. It can be obtained by letting $F(x)$ be the Gaussian cumulative distribution in the general linear models.

$$\mathrm{P}(x \succ y) = \Phi(h(x) - h(y) - \epsilon) \tag{7}$$

$$\mathrm{P}(x = y) = \Phi(h(x) - h(y) + \epsilon) - \Phi(h(x) - h(y) - \epsilon)$$

where $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-x}^{\infty} e^{-\frac{x^2}{2}} \mathrm{d}x$ is the Gaussian cumulative distribution function.

Unless otherwise stated, we use the terms Bradley-Terry model and Thurstone-Mosteller model to represent the corresponding model with ties in the rest of this paper.

5. LOSS FUNCTIONS

Assume that we have observed the preference data and ties $\{(x_i, y_i)\}_{i=1}^{N+M}$ where $x_i \succ y_i$ for all $i = 1 \ldots N$ and $x_i = y_i$ for all $i = N+1, \ldots, N+M$. Our goal is to learn a ranking function h from this set of data. Following the principle of empirical risk minimization [13], we need to measure how good a ranking function is with respect to the training data. The empirical risk is defined as follows:

$$\begin{aligned} \hat{\mathcal{R}}[h] &= \sum_{i=1}^{N+M} L(h, x_i, y_i) \\ &= \sum_{i=1}^{N} L(h, x_i \succ y_i) + \sum_{i=N+1}^{N+M} L(h, x_i = y_i) \quad (8) \end{aligned}$$

where $L(h, x_i \succ y_i)$ and $L(h, x_i = y_i)$ are the loss of the function h on the pairs $x_i \succ y_i$ and $x_i = y_i$, respectively. Based on the discussions in Section 4, we can use the negative log of the data likelihood as the loss:

$$\begin{aligned} L(h, x_i \succ y_i) &= -\log \mathrm{P}(x_i \succ y_i) \quad (9) \\ L(h, x_i = y_i) &= -\log \mathrm{P}(x_i = y_i) \quad (10) \end{aligned}$$

The probabilities $\mathrm{P}(x_i \succ y_i)$ and $\mathrm{P}(x_i = y_i)$ are determined by the paired comparison models in Section 4.

Bradley-Terry Model. The loss function for the Bradley-Terry model can be expressed as:

$$\begin{aligned} L^{\mathrm{BT}}(h, x \succ y) &= \log\left(1 + \theta e^{h(y)-h(x)}\right) \quad (11) \\ L^{\mathrm{BT}}(h, x = y) &= \log\left(1 + \theta e^{h(x)-h(y)}\right) \\ &+ \log\left(1 + \theta e^{h(y)-h(x)}\right) - \log(\theta^2 - 1) \end{aligned}$$

Thurstone-Mosteller Model. Analogously, the loss function for the Thurstone-Mosteller model is

$$\begin{aligned} L^{\mathrm{TM}}(h, x \succ y) &= -\log \Phi(h(x) - h(y) - \epsilon) \quad (12) \\ L^{\mathrm{TM}}(h, x = y) &= -\log(\Phi(h(x) - h(y) + \epsilon) \\ &\quad -\Phi(h(x) - h(y) - \epsilon)) \quad (13) \end{aligned}$$

Substituting L^{BT} and L^{TM} defined above into Equation (8), we obtain the loss functions we need to minimize in order to learn the ranking functions.

6. LEARNING BY FUNCTIONAL GRADIENT BOOSTING

In Section 5, we define the empirical risk function $\hat{\mathcal{R}}[h]$. The learning process is to obtain a function h^* from a function space $\mathcal{H}$ that minimizes the empirical risk $\hat{\mathcal{R}}[h]$. Formally,

$$h^* = \arg\min_{h \in \mathcal{H}} \hat{\mathcal{R}}[h] \quad (14)$$

In order to obtain h^*, we apply the gradient boosting algorithm [10,30,31]. The idea of gradient boosting is to approximate $h^*(x)$ by iteratively constructing a sequence of base learners. So we need to approximate $h^*(x)$ as:

$$h^*(x) = g_0(x) + \sum_{t=1}^{T} \beta_t g_t(x; \alpha_t) \quad (15)$$

$g_t(x; \alpha_t)(t = 0, \ldots, T)$ are called base learners and α_t represents the parameters of base learner $g_t(x)$, and $g_0(x)$ is the initial approximation of $h^\star(x)$. The main idea is to perform gradient descent in function spaces. At each iteration, a base learner g_t is chosen to be the gradient of $\hat{\mathcal{R}}[h]$ with respect to h at the current iterate. More precisely, the next iterate $h_t(x)$ is given by

$$h_t(x) = h_{t-1}(x) - \beta_t \nabla \hat{\mathcal{R}}[h_{t-1}(x)] \quad (16)$$

However, we can not compute the value of $-\nabla\hat{\mathcal{R}}[h_t(x)]$ at all x in general, but we can compute its values at a finite sample $\{x_i, y_i\}_{i=1}^{N+M}$,

$$g_{ti}^x = -\nabla \hat{\mathcal{R}}[h_t(x_i)],\ g_{ti}^y = -\nabla \hat{\mathcal{R}}[h_t(y_i)],\ i = 1, \ldots, N+M \quad (17)$$

Follow the techniques developed in [10, 30], we apply the base learner for constructing an approximation $g_t(x; \alpha_t)$ of $-\nabla\hat{\mathcal{R}}[h_t(x)]$ from the following finite training data,

$$(x_1, g_{t1}^x), \ldots, (x_{N+M}, g_{t,N+M}^x), (y_1, g_{t1}^y), \ldots, (y_{N+M}, g_{t,N+M}^y).$$

We summarize the above algorithm in Algorithm 1. In our experiments, the base learners are regression trees and we usually set the number of terminal nodes to a small number around 10. There are two other parameters need to determined. They are the number of the iterations T and the shrinkage factor β which are usually found by cross-validation.

Algorithm 1 Functional Gradient Boosting

Input: A finite sample of query document pairs $\{(x_i, y_i)\}_{i=1}^{N+M}$, the empirical risk functional $\hat{\mathcal{R}}[h(x)]$

Output: $h(x)$, which minimizes the empirical risk functional $\hat{\mathcal{R}}[h(x)]$

1: Initialize $h_0(x) = g_0(x)$
2: **for** t=1, 2, ... T **do**
3: Compute $g_{ti}^x = -\nabla\hat{\mathcal{R}}[h_t(x_i)]$, and $g_{ti}^y = -\nabla\hat{\mathcal{R}}[h_t(y_i)]$ for $i = 1, \ldots, N+M$
4: Fit a base learner $g_t(x; \alpha_t)$ based on training data
$$(x_1, g_{t1}^x), \ldots, (x_{N+M}, g_{t,N+M}^x),$$
and $(y_1, g_{t1}^y), \ldots, (y_{N+M}, g_{t,N+M}^y)$.
5: Update the approximation by $h_t(x) = h_{t-1}(x) + \beta g_t(x; \alpha_t)$, where β is the shrinkage factor.
6: **end for**
7: **return** $h_T(x)$

7. EXPERIMENTAL STUDIES

In this section, we apply the proposed algorithms to three publicly available real-world data sets and report the results of the experimental studies.

7.1 Data Description

We used the Letor[2] data collection [20]. This data collection contains three data sets: the OHSUMED data set, the

[2]http://research.microsoft.com/users/tyliu/LETOR/

TREC2003 data set and the TREC2004 data set which we now briefly describe.

OHSUMED. The OHSUMED data set is a subset of the MEDLINE database, which is popular in the information retrieval community. This data set contains 106 queries. The documents are manually labeled with absolute relevance judgements with respect to the queries. There are three levels of relevance judgments in the data set: *definitely relevant*, *possibly relevant* and *not relevant*. Each query-document pair is represented by a 25-dimensional feature vector that contains the most frequently used features in information retrieval, for example tfidf, BM25 score etc. The total number of query-document pairs is 16,140.

TREC2003. This data set is extracted from the topic distillation task of TREC2003[3]. The goal of the topic distillation task is to find good websites about the query topic. There are 50 queries in this data set. For each query, the human assessors decide whether a web page is an relevant result for the query, so two levels of relevance are used: *relevant* and *not relevant*. The documents in the TREC2003 data set are crawled from the `.gov` websites, so the features extracted by link analysis are also used to represent the query-document pair in addition to the content features used in the OHSUMED data set. The total number of features is 44 and total number of query-document pairs is 49,171.

TREC2004. This data set is extracted from the data set of the topic distillation task of TREC2004, so it is very similar to the TREC2003 data set. This data set contains 75 queries and 74,170 documents with 44 features.

All these data sets are labeled in the form of absolute judgements, so it is necessary to convert them to preference data. Given a query q, let x and y be the feature vectors for the query-document pairs (q, d_x) and (q, d_y), respectively. If d_x has higher grade than d_y, the preference $x \succ y$ is included, while if y has greater grade than x, the preference $y \succ x$ is included. Similarly, in the cases of ties, a tie $x = y$ is included if d_x and d_y have the same grade.

7.2 Evaluation Measures

In order to evaluate the performance of the proposed algorithms, three evaluation measures are applied: Precision, Mean average precision and Normalized Discount Cumulative Gain [1, 15]. All these evaluation measures are widely used for comparing information retrieval systems.

Precision. Given the binary relevance judgment, the precision of a ranked list is measured by the fraction of the retrieved documents that are relevant. In our experimental studies, precision at position n (P@n) is used to measure the quality of the top n results of the ranking list.

$$\text{P@n} = \frac{\text{No. of relevant docs in top } n \text{ results}}{n} \qquad (18)$$

Mean Average Precision. The average precision of a query is the average of the precision scores after each relevant document retrieved. Formally, average precision (AP) is calculated by the following equation.

$$\text{AP} = \frac{\sum_i \text{P@i} \times \text{rel}_i}{\text{No. of relevant documents}} \qquad (19)$$

where rel_i is the indicator function whether the i-th document of the ranking list is relevant to the query. Mean Average Precision (MAP) is obtained by the mean of the average precision over a set of queries. Compared with P@n measure, the MAP score is sensitive to the entire ranking list and contains the aspects of recall as well as precision.

Normalized Discount Cumulative Gain. P@n and MAP are defined based on binary judgements: relevant and irrelevant. In the case of multiple levels of judgements, a more sophisticated evaluation measure called Normalized Discount Cumulative Gain (NDCG) is used [15]. Unlike P@n and MAP, NDCG has the capability to deal with multiple levels of relevance. The NDCG value of a ranking list is calculated by the following equation:

$$\text{NDCG@n} = Z_n \sum_{i=1}^{n} (2^{r_i} - 1)/\log(i+1) \qquad (20)$$

where r_i is the grade assigned to the i-th document of the ranking list. In our experiments, r_i takes value of 0, 1 and 2 in OHSUMED data set for not, possibly and definitely relevant documents respectively. For data sets with binary judgments, such as TREC2003 and TREC2004 data set, r_i is set to 1 if the document is relevant and 0 otherwise. The constant Z_n is chosen so that the perfect ranking gives an NDCG value of 1.

7.3 Experimental Design and Results

We want to address the following questions:

1. Are ties effective for enhancing the performance of the ranking functions? How much improvement can we obtain by incorporating ties?
2. For what types of search problems will ties improve the performance of ranking significantly?
3. How does the performance of our algorithms compare with the existing methods. In this study, we consider two preference learning methods RankSVM, RankBoost and FRank [27] for comparisons. We also compare our methods to AdaRank [28], which learns a ranking function through directly optimizing the performance measures.
4. What is the convergence behaviors of our proposed algorithms? The ranking functions are obtained by gradient boosting method which is an iterative process, so we need to investigate the rate of convergence of our algorithms.

Each data set is partitioned on queries to perform 5 fold *cross-validation*. For each fold, there are three subsets of data: training, test and validation set. For each loss function described in Section 5, we tune the algorithms on the validation set to determine the parameters. All our experiments are performed on a server with four 3.2G CPUs and 2GB main memory. The gradient boosting algorithm takes about 1 hour for training both the Bradley-Terry model and the Thurstone-Mosteller model over the TREC2004 data set, which is the largest data set in our experiments. For AdaRank, we use the version for optimizing the mean average precision for comparison in this study. The results of RankSVM, RankBoost, AdaRank and FRank are reported in the Letor data set.

7.3.1 Experimental Results

For the first question, we apply both Bradley-Terry model and Thurstone-Mosteller model to the three data sets and

[3]http://trec.nist.gov/

Table 1: Performance comparisons over OHSUMED data set

	MAP	NDCG@1	NDCG@3	NDCG@5	P@1	P@3	P@5
BT	0.4537	0.5288	0.4911	0.4759	0.6650	0.6211	0.5806
BT-noties	0.4488	0.4753	0.4664	0.4623	0.6288	0.5911	0.5626
TM	0.4543	0.5269	0.4794	0.4714	0.6654	0.6146	0.6010
TM-noties	0.4476	0.5072	0.4764	0.4652	0.6252	0.6047	0.5803
RankSVM	0.4469	0.4952	0.4649	0.4579	0.6338	0.5924	0.5766
RankBoost	0.4403	0.4977	0.4726	0.4502	0.6048	0.5861	0.5446
AdaRank	0.4419	0.5420	0.4803	0.4554	0.6615	0.5831	0.5373
FRank	0.4463	0.5449	0.4995	0.4688	0.6710	0.6175	0.5597

Table 2: Performance comparisons over TREC2003 data set

	MAP	NDCG@1	NDCG@3	NDCG@5	P@1	P@3	P@5
BT	0.2601	0.4400	0.3704	0.3487	0.4400	0.3400	0.2720
BT-noties	0.2341	0.4200	0.3603	0.3456	0.4200	0.3333	0.2680
TM	0.2562	0.4600	0.3727	0.3583	0.4600	0.3600	0.2720
TM-noties	0.2258	0.4200	0.3512	0.3363	0.4200	0.3333	0.2640
RankSVM	0.2564	0.4200	0.3787	0.3473	0.4200	0.3400	0.2640
RankBoost	0.2125	0.2600	0.2704	0.2789	0.2600	0.2400	0.2200
AdaRank	0.1373	0.4200	0.2912	0.2424	0.4200	0.2667	0.1880
FRank	0.2451	0.4400	0.3690	0.3303	0.4400	0.3200	0.2320

Table 3: Performance comparisons over TREC2004 data set

	MAP	NDCG@1	NDCG@3	NDCG@5	P@1	P@3	P@5
BT	0.3852	0.4800	0.4550	0.4372	0.4800	0.4189	0.3230
BT-noties	0.3761	0.4600	0.4358	0.4294	0.4600	0.4108	0.3007
TM	0.3772	0.4800	0.4548	0.4363	0.4800	0.4112	0.3157
TM-noties	0.3674	0.4533	0.4456	0.4280	0.4533	0.4055	0.3128
RankSVM	0.3505	0.4400	0.4092	0.3935	0.4400	0.3511	0.2907
RankBoost	0.3835	0.4800	0.4640	0.4368	0.4800	0.4044	0.3227
AdaRank	0.3308	0.4133	0.4017	0.3932	0.4133	0.3422	0.2933
FRank	0.3809	0.4400	0.4479	0.4362	0.4400	0.3867	0.3227

compare the performance of the models with and without ties. For comparisons of the different models, we report the performance measured by both the precision and NDCG at position 1, 3 and 5 as well as the mean average precision (MAP). We average these performance measures over 5 folds for each data set. The results on OHSUMED, TREC2003 and TREC2004 data set are shown in Table 1, 2 and 3, respectively.

We can observe from the tables that the Bradley-Terry model and Thurstone-Mosteller model with the ties (labeled by BT and TM) outperform the corresponding models without ties (labeled by BT-noties and TM-noties). These tables show that the performance of ranking is improved (sometimes significantly) by learning ranking functions with ties. We conduct significant tests on the improvements of BT and TM over BT-noties and TM-noties. The results show that the improvements in terms of NDCG@5 is statistically significant (p-value < 0.05).

We report in Table 4 the performance improvements obtained by learning ranking functions with ties. The improvements over the OHSUMED data set are more significant than that over both TREC2003 and TREC2004 data sets in most cases. We think this is because ties are more effective when more relevant judgment levels are available while TREC2003 and TREC2004 data sets are labeled by only two relevant judgment levels. To further explore this point, we construct a new data set OHSUMED-2L by combining the definite relevant and possibly relevant judgements as a single level and perform experiments over this new data set. We report the improvements in performance in column five of Table 4. Compared with learning from three levels of judgements (column 2), the improvements is reduced by learning from two levels. This observation indicates that ties can enhance the ranking function more significantly when multiple relevant levels of judgements are applied. In the case of multiple relevant levels, it is usually difficult to assign documents to the right relevant levels with only preference data. Ties provide information about the distribution of each relevant levels. This information is able to constrain the learning process and thus enhance the ranking functions significantly.

Another observation from Table 4 is that the performance improvements on the top-ranked results, for example NDCG@1 and P@1, are more significant. The reason for the better performance on the top-ranked results could be that ties capture the common features of the relevant documents. To illustrate the effect of ties, we examined the top five results for Query 79 and Query 84 of OHSUMED data set in Table 5. The results are produced by the Bradley-Terry model with ties and without ties, respectively. We also list the document id and the ground-truth grade of each document. Taking Query 79 for example, it can be observed that the Bradley-Terry models with ties ranks the Document 45619 in the forth position, while the model without

Table 4: The performance improvements by including ties measured by NDCG of top ten positions

n	OHSUMED	TREC2003	TREC2004	OHSUMED-2L
1	11.26%	4.45%	4.35%	4.58%
2	6.71%	6.15%	4.48%	3.61%
3	5.30%	2.80%	4.40%	4.67%
4	5.58%	4.90%	4.27%	4.33%
5	2.94%	2.99%	1.82%	2.45%
6	5.77%	4.60%	4.53%	2.75%
7	4.94%	2.50%	3.60%	1.68%
8	1.57%	2.86%	0.98%	0.66%
9	1.94%	6.32%	2.61%	0.36%
10	2.62%	2.32%	1.87%	0.33%

Table 5: Top five results of example queries on OHSUMED data set

(a) Query 79

Rank	BT-noties		BT	
	Grade	Doc-Id	Grade	Doc-Id
1	2	65890	2	65890
2	2	263191	2	263191
3	2	257288	2	257288
4	2	101228	**2**	**45619**
5	**0**	**63458**	2	143269

(b) Query 84

Rank	BT-noties		BT	
	Grade	Doc-Id	Grade	Doc-Id
1	2	8653	2	145901
2	2	177517	2	177517
3	2	145901	2	8653
4	**0**	**101228**	1	205254
5	1	205254	**1**	**32155**

ties ranks it out of the top five. The reason is that the feature vector of Document 45619 can not distinguish this document from the irrelevant documents. Consequently, the model BT-noties does not return it in the top results. Thus, the performance of the model BT-noties will be reduced since it is actually labeled by definite relevant (represented by 2). On the other hand, since the common features of the relevant documents are explicitly captured by ties, the similarity between Document 45619 and other relevant documents is precisely addressed. As a result, Document 45619 is ranked at the forth position by the model Bradley-Terry model with ties. Similarly, the model with ties ranks Document 32155 in the fifth position for Query 84 as reported in Table 5(b).

It is interesting to investigate the effect of ties extracted from different levels of grades. To this end, we conduct experiments over the OHSUMED data set by extracting ties from *relevant*, *possibly relevant* and *irrelevant* documents respectively. We also extract ties from both the relevant and the possibly relevant documents. The performance measured by NDCG@5 is reported in Figure 2. It can be observed that when ties are extracted from the relevant documents or both the relevant and the possibly relevant documents, the performance of the resulting model slightly reduced. However, if the ties are extracted from the irrelevant documents, the decrease in performance is much more significant. One explanation is that the irrelevant documents are more diverse than the relevant document and thus ties become less effective in this case.

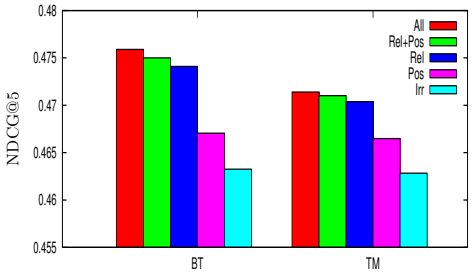

Figure 2: Performance vs. ties from different relevance levels

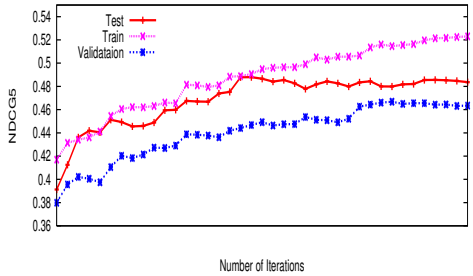

Figure 3: Learning curves of Bradley-Terry model

The performance of Bradley-Terry model and the Thurstone-Mosteller model are comparable in most cases. We also compare the performance of our algorithms with RankSVM, RankBoost, FRank and AdaRank as reported in Table 1, 2 and 3. The Bradley-Terry model and Thurstone-Mosteller model with ties outperform both RankSVM and RankBoost over the OHSUMED data set. On TREC2003 data set, our algorithms are comparable or slightly better than RankSVM, but RankBoost is much worse while on TREC2004 data set our algorithms are comparable or slightly worse than RankBoost, but RankSVM is worse. What causes the dramatic difference of RankSVM and RankBoost on TREC2003 and TREC2004 data set remains to be investigated. Again, we want to emphasize that the focus of the paper is on investigation of the role of ties in learning ranking functions not comprehensive comparisons of existing methods.

In order the study the convergence behaviors of our algorithms, we plot the NDCG@5 over the training, test and validation set with respect the iteration number of the gradient boosting algorithm in Figure 3. For each iteration, we average the NDCG@5 scores of all five folds of the OHSUMED data set. It can be observed from Figure 3 that performance on training set monotonically increases as the number of iterations grows, so the performance will gradually converge to a maximum point. Similar observations can be made also on the validation and test sets as well as over the TREC2003 and TREC2004 data sets.

8. CONCLUSION AND FUTURE WORK

In this paper, we demonstrated that incorporating can improve the performance of learned ranking functions based on

the experimental studies over three publicly available data sets. Furthermore, we find that the improvements over the top-ranked results are more significant since the common features of relevant documents are explicitly captured by ties. This promising property of ties makes them effective for web search engines.

Future directions of research include theoretical analysis of the effect of ties. We plan to explain the effectiveness of ties for the ranking problem in terms of the generalization bounds. Another direction of research is to develop new techniques for incorporate ties with other methods for learning to rank, such as RankSVM, RankBoost and RankNet. There several methods for online learning from preference data [14, 21], we hope to develop learning algorithms that both preference data and ties can be included to refine a ranking model incrementally. Also, since our framework is flexible to incorporate a large class of loss functions, it is interesting to compare and analysis other loss functions and determine what kind of characteristics the loss function should have for the ranking problems.

9. REFERENCES

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

[2] R. Bradley and M.Terry. The rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39, 1952.

[3] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine learning*, 2005.

[4] C. J. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In *Advances in Neural Information Processing Systems 19*. 2007.

[5] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking svm to document retrieval. In *Proceedings of the 29th ACM SIGIR*, 2006.

[6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, 2007.

[7] C. Cortes, M. Mohri, and A. Rastogi. Magnitude-preserving ranking algorithms. In *Proceedings of the 24th ICML*, 2007.

[8] H. A. David. *The Method of Paired Comparisons*. Oxford University Press, 2nd edition, 1988.

[9] Y. Freund, R. D. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In *Proceedings of the Fifteenth International Conference on Machine Learning*, 1998.

[10] J. Friedman. Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, 29, 2001.

[11] J. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38, 2002.

[12] G. Fung, R. Rosales, and B. Krishnapuram. Learning rankings via convex hull separation. In *Advances in Neural Information Processing Systems 18*. 2006.

[13] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[14] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, Cambridge, MA, 2000. MIT Press.

[15] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transaction of Information Systems*, 20(4), 2002.

[16] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of ACM SIGKDD*, 2002.

[17] T. Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd international conference on Machine learning*, 2005.

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

[19] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th ACM SIGIR*, 2001.

[20] T. Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of the Learning to Rank workshop in the 30th ACM SIGIR*, 2007.

[21] F. Radlinski and T. Joachims. Active exploration for learning rankings from clickthrough data. In *Proceedings of the 13th ACM SIGKDD*, 2007.

[22] P. Rao and L. Kupper. Ties in paired-comparison experiments: A generalization of the bradley-terry model,. *Journal of the American Statistical Association*, 62, March 1967.

[23] S. Robertson and D. A. Hull. The TREC-9 filtering track final report. *Proceedings of the 9th Text REtrieval Conference (TREC-9)*, 2001.

[24] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11), 1975.

[25] F. Song and W. B. Croft. A general language model for information retrieval. In *Proceedings of CIKM*, 1999.

[26] L. Thurstone. A law of comparative judgement. *Psychological Review*, 34, 1927.

[27] M.-F. Tsai, T.-Y. Liu, T. Qin, H.-H. Chen, and W.-Y. Ma. Frank: a ranking method with fidelity loss. In *Proceedings of ACM SIGIR*, pages 383–390, 2007.

[28] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th ACM SIGIR*, 2007.

[29] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of ACM SIGIR*, 2007.

[30] Z. Zheng, K. Chen, G. Sun, and H. Zha. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th ACM SIGIR conference*, 2007.

[31] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In *Advances in Neural Information Processing Systems*. 2007.