Semi-Supervised Ensemble Ranking

Steven C.H. Hoi
School of Computer Engineering
Nanyang Technological University
Singapore 639798
chhoi@ntu.edu.sg

Rong Jin
Dept. of Computer Sci. & Eng.
Michigan State University
East Lansing, MI 48824, U.S.A.
rongjin@cse.msu.edu

Abstract

Ranking plays a central role in many Web search and information retrieval applications. Ensemble ranking, sometimes called meta-search, aims to improve the retrieval performance by combining the outputs from multiple ranking algorithms. Many ensemble ranking approaches employ supervised learning techniques to learn appropriate weights for combining multiple rankers. The main shortcoming with these approaches is that the learned weights for ranking algorithms are query independent. This is suboptimal since a ranking algorithm could perform well for certain queries but poorly for others. In this paper, we propose a novel semi-supervised ensemble ranking (SSER) algorithm that learns query-dependent weights when combining multiple rankers in document retrieval. The proposed SSER algorithm is formulated as an SVM-like quadratic program (QP), and therefore can be solved efficiently by taking advantage of optimization techniques that were widely used in existing SVM solvers. We evaluated the proposed technique on a standard document retrieval testbed and observed encouraging results by comparing to a number of state-of-the-art techniques.

Introduction

Ranking is crucial to many real-world applications, especially in the fields of Web search and information retrieval. One way to address the ranking problem is ensemble ranking, sometimes called meta-search. It improves retrieval performance by combining the outputs from multiple ranking algorithms. The ensemble ranking techniques can be beneficial for many applications. For example, it can be used to improve Web search by combining the ranking lists from several Web search engines.

One simple approach for ensemble ranking is to first normalize the output ranking scores from multiple ranking algorithms, and then combine the normalized ranking scores by weights that are usually tuned empirically. Despite its simplicity, this heuristic approach often fails to deliver the optimal performance because of the hand tuned weights.

In recent studies, ensemble ranking is usually formalized as a machine learning problem, in which various supervised learning techniques have been applied to learn optimal weights for combining multiple ranking algorithms. Unlike typical supervised learning where each training example consists of an input pattern $\mathbf{x}$ and a class label y, in learning to rank, training data usually consist of a number of queries, and each query is associated with a list of objects (e.g., documents) whose relevances have been manually judged by human subjects[1]. The goal of the ensemble ranking is to learn optimal weights from the training data to combine the ranking results output from multiple ranking algorithms.

Many studies on ensemble ranking cast it into a supervised learning problem. For example, a well-known ranking technique is Ranking Support Vector Machine (SVM) (Herbrich, Graepel, & Obermayer 2000; Joachims 2002). To decide an appropriate order for a set of objects, it considers solving the ordering problem between any two objects $\mathbf{x}_a$ and $\mathbf{x}_b$, which can be further converted into a binary classification problem, i.e., whether or not $\mathbf{x}_a$ be ranked before $\mathbf{x}_b$. We refer to these approaches as "*supervised ensemble ranking*", or **SER** for short.

The main drawback of the SER approaches is that the weight learned for each ranking algorithm is query independent. This is only suboptimal since a ranking algorithm could perform well for certain queries but poorly for others. This motivates us to develop a query-dependent solution for ensemble ranking, in which the weights assigned to different ranking algorithms are affected by the characteristics of queries. To this end, we propose a novel algorithm for *Semi-Supervised Ensemble Ranking* (**SSER**), which learns query-dependent combination weights for ensemble ranking.

The major contributions in this paper include: (1) we present a new formulation for supervised ensemble ranking, which significantly improves the learning efficiency compared to the existing approaches such as Ranking SVM; (2) we propose a novel algorithm for semi-supervised ensemble ranking, which employs graph-based regularization to exploit the geometric relationship of the top retrieved objects for a given query; (3) we evaluate the proposed algorithm on a benchmark dataset by comparing it to several state-of-the-

[1]In certain applications (e.g., document retrieval), the relevance judgments can be inferred indirectly from user interaction data (e.g., click-through data) collected by Web search engines (Joachims 2002)

art ranking algorithms.

The rest of this paper is organized as follows: first, we briefly review the related work on the ensemble ranking and supervised learning methods for ranking; second, we give a formal formulation of the ensemble ranking problem and present the proposed ensemble ranking solution; third, we present an empirical study on a benchmark testbed for document retrieval; finally, we set out our conclusion.

Related Work

Document retrieval has been extensively studied in the information retrieval and Web community (Baeza-Yates & Ribeiro-Neto 1999). Traditional document retrieval methods often focus on the extraction of effective features (Salton 1971). The features can be either structural features, such as term frequency and inversed document frequency, or query-independent features, such as PageRank for Web document retrieval. The traditional document retrieval methods usually formulate some ranking functions on the extracted features, and then empirically tune the parameters of the ranking functions as well as the combination weights for aggregating the ranking scores output by different ranking functions. Some well-known approaches include the Okapi BM25 model (Robertson, Walker, & Beaulieu 1998) and languages models in information retrieval (LMIR) (Ponte & Croft 1998). We refer to these approaches as "unsupervised ensemble ranking" methods.

Recently, the ranking problem of document retrieval has been formalized as a supervised machine learning task, in which a variety of machine learning techniques have been actively investigated for solving the learning tasks. Regular approaches usually adapt typical supervised machine learning techniques to solve the learning task. For example, Herbrich et al. (Herbrich, Graepel, & Obermayer 2000) propose to solve the ranking issue by applying SVM to solve document retrieval tasks. We refer to their approach as the Ranking SVM method. Joachims also applies the similar method to learn an ensemble ranking solution for combining the results from multiple search engines (Joachims 2002), which even achieved better retrieval quality than Google after proper training. Other variants of SVM related approaches for ranking can also be found in (Cao *et al.* 2006; Nallapati 2004).

In addition to SVMs, there are also other machine learning techniques studied for ranking. For example, Freund et al. proposed to solve the ranking problem by Boosting techniques (Freund *et al.* 2003). Burges et al. proposed a neural network approach to model the ranking function, which is called RankNet (Burges *et al.* 2005). More other related work on learning to rank topics can be found in (Usunier *et al.* 2005; Li & Lin 2007; Agarwal 2006; Cao *et al.* 2007; Agarwal & Chakrabarti 2007). In addition to general ranking topics, there are some related work focusing on the ranking aggregation problem, such as Markov Chain based rank aggregation (Dwork *et al.* 2001) and supervised rank aggregation (Liu *et al.* 2007b), etc. Most of these work are often based on either unsupervised or supervised learning approaches, which are different from our solution.

Semi-supervised Ensemble Ranking

In this section, we first introduce the ensemble ranking problem in the context of information retrieval. We will then formulate ensemble ranking as a convex optimization problem and propose a novel semi-supervised algorithm for ensemble ranking. It is important to note that we focus on score-based ensemble ranking approaches, which are different from the order-based ensemble ranking approaches.

Problem Statement

We consider a general ensemble ranking problem that aggregates results from multiple ranking algorithms for information retrieval tasks. Let $\mathcal{Q} = \{q_i, i = 1, \ldots, N_q\}$ denote a collection of N_q queries in the training data, and each query q_i is associated with a list of N_i objects $\mathcal{D}^i = \{d_1^i, \ldots, d_{N_i}^i\}$, in which each object $d_j^i$is manually judged with relevance $y_j^i \in \{-1, 0, +1\}$ where $+1$ stands for highly relevant, 0 for possibly relevant, and -1 for irrelevant. The relevance judgement y_j^i could be a degree of relevance that is either provided directly by human subjects or inferred indirectly from user computer interaction data, such as the click-through data collected by search engines.

Let $\mathcal{G} = \{g_1(\cdot), \ldots, g_m(\cdot)\}$ denote the ensemble of m ranking functions where each function $g_i(d) : \mathcal{X} \rightarrow \mathbb{R}$. The goal of ensemble ranking is to combine the ranking functions in $\mathcal{G}$ to produce ranking lists that are better than any individual ranking functions. In the simple form, the combined ranking function, denoted by $f_{\mathbf{w}}(\cdot)$, is expressed as

$$f_{\mathbf{w}}(d) = \sum_{k=1}^{m} w_k g_k(d) \tag{1}$$

where w_k is the weight assigned to ranking function $g_k(\cdot)$. Hence, for most ensemble ranking methods, the key is to learn the combination weights $w_k, k = 1, \ldots, m$.

To simplify the representation, we introduce an m-dimensional feature vector $\mathbf{x}_j^i \in \mathbb{R}^m$ to represent the ranking scores output by m rankers for each object d_j^i, i.e.,

$$\mathbf{x}_j^i = (\mathbf{x}_{j,1}^i, \ldots, \mathbf{x}_{j,m}^i) = (g_1(d_j^i), \ldots, g_m(d_j^i)) \tag{2}$$

Then, the ranking score for d_j^i by the combined ranking function is calculated as $f_{\mathbf{w}}(d_j^i) = \mathbf{w}^\top \mathbf{x}_j^i$.

Supervised Ensemble Ranking

One way to solve the ensemble ranking problem is to adopt the Ranking SVM method. The basic idea is to decompose a ranking list into a number of ordered example pairs. Given a query q_i and two objects (d_j^i) and (d_k^i), one can create a training example $(\mathbf{x}_j^i - \mathbf{x}_k^i, z_{jk}^i)$, where the training label z_{jk}^i is defined as:

$$z_{jk}^i = \begin{cases} +1 & \text{if } d_j^i \succ d_k^i; \\ -1 & \text{if } d_k^i \succ d_j^i. \end{cases} \tag{3}$$

where $\succ$ stands for the prefer operator. Using the training examples defined above, one builds an SVM model by solving the following optimization:

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N_q}\sum_{j=1}^{N_i}\sum_{k=j+1}^{N_i}\xi_{jk}^i \tag{4}$$
$$s.t. \quad z_{jk}^i(\mathbf{w}\cdot(\mathbf{x}_j^i - \mathbf{x}_k^i)) \geq 1 - \xi_{jk}^i, i = 1\ldots N_q,$$
$$\xi_{jk}^i \geq 0, j < k \text{ and } j,k \in [1,\ldots,N_i].$$

The above ranking SVM approach has a major drawback in training efficiency because the number of example pairs is quadratic in the number of objects. Thus, it is inefficient when the number of objects is large.

In this paper, we propose a new approach for supervised ensemble ranking that can improve the training efficiency of Ranking SVM significantly. First, for each query q_i, we construct a *relevance matrix* A^i to represent the relevance judgement, in which each element $A_{j,k}^i$ is defined as follows:

$$A_{j,k}^i = \begin{cases} +1 & \text{if } y_j^i = 1 \text{ and } y_k^i = -1; \\ -1 & \text{if } y_j^i = -1 \text{ and } y_k^i = 1; \\ \theta & \text{if } y_j^i = 1 \text{ and } y_k^i = 0; \\ -\theta & \text{if } y_j^i = 0 \text{ and } y_k^i = 1; \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

where $\theta \in (0,1)$ is a constant that represents the chance for a highly relevant object to be ranked before a possibly relevant object. Furthermore, for each training query q_i, we summarize the ranking results output by a ranking function $g_j(\cdot)$ by a matrix $\widetilde{R}^{i,j}, j = 1,\ldots,m$, where $\widetilde{R}_{k,l}^{i,j} = 1$ if d_k^i is ranked before d_l^i by $g_j(\cdot)$ and 0 otherwise. We smooth the ranking matrix by $\widetilde{R}^{i,j} + \delta I$ where δ is a small constant for smoothing, and normalize the smoothed ranking matrix to be a transition matrix $R^{i,j}$ by a column-based normalization, i.e., $\sum_{k=1}^{N_i} R_{k,l}^{i,j} = 1$. Similarly, for a testing query q and a collection of objects to be ranked, we obtain the ranking scores generated by ranking function $g_i(\cdot)$, and summarize the ranking results into a transition matrix B^i, which is subjected to the same summarization procedure used for constructing $R^{i,j}$.

We formulate ensemble ranking as a problem of learning optimal weights to minimize the difference between the relevance matrix and the combined transition matrix. We cast this idea into the following optimization problem:

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N_q}\xi_i \tag{6}$$
$$s.t. \quad tr\left([A^i]^\top\left[\sum_{j=1}^{m} w_j R^{i,j}\right]\right) \geq 1 - \xi_i,$$
$$\xi_i \geq 0, w_j \geq 0, i = 1\ldots N_q, j = 1\ldots m.$$

where $tr(\cdot)$ is a trace function that measures the similarity between the relevance matrix A^i and the combined transition matrix $\sum_{j=1}^{m} w_j R^{i,j}$. In the above optimization, we formulate the problem by adopting the regularization framework used in SVM. It is important to note that the supervised ensemble ranking formulation in (6) differs from the ranking SVM approach in (4) in that our formulation only engages $\mathcal{O}(N_q)$ constraints, which is significantly smaller than the number of constraints in (4).

We can further generalize the formulation in (6) by introducing a function $s(A,B)$ to measure the similarity between two matrices A and B. By replacing $tr(A,B)$ with $s(A,B)$, we have (6) generalized as

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N_q}\xi_i \tag{7}$$
$$s.t. \quad \sum_{j=1}^{m} w_j s(A^i, R^{i,j}) \geq 1 - \xi_i,$$
$$\xi_i \geq 0, w_j \geq 0, i = 1\ldots N_q, j = 1\ldots m.$$

One choice for $s(A,B)$ is to generalize the RBF function for matrices, i.e., $s(A,B) = \exp(-d(A,B)/\lambda)$ where λ is the scaling constant and $d(A,B)$ is distance between matrices A and B. For instance, we can measure $d(A,B)$ by the induced norm of matrix $A-B$; we can also measure $d(A,B)$ by the Bregman matrix divergence.

Semi-Supervised Ensemble Ranking

As pointed out in the introduction section, one major drawback of the supervised ensemble ranking approaches is that they learn query-independent weights for different ranking algorithms. We propose a Semi-Supervised Ensemble Ranking (SSER) method that learns query-dependent weights by exploiting the correlation among the objects retrieved for a given query. The intuitive idea behind our method is that if the contents of two retrieved objects are similar, they should be assigned with similar ranking scores. Taking this factor into consideration, we can extend (6) into a semi-supervised ensemble ranking method:

$$\min_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N_q}\xi_i \tag{8}$$
$$+\frac{\gamma}{2}\sum_{k=1}^{N_u}\sum_{l=1}^{N_u} s_{kl}\Big(f_{\mathbf{w}}(d_k^q) - f_{\mathbf{w}}(d_l^q)\Big)^2$$
$$s.t. \quad tr\left([A^i]^\top\left[\sum_{j=1}^{m} w_j R^{i,j}\right]\right) \geq 1 - \xi_i,$$
$$\xi_i \geq 0, w_j \geq 0, i = 1\ldots N_q, j = 1\ldots m.$$

where $\mathcal{D}^q = (d_1^q,\ldots,d_{N_u}^q)$ is the set of objects to be ranked for query q; s_{kl} is the similarity between d_k^q and d_l^q. Note that the term $\sum_{k,l=1}^{N_u} s_{kl}\Big(f_{\mathbf{w}}(d_k^q) - f_{\mathbf{w}}(d_l^q)\Big)^2$ is introduced to capture the idea that similar objects should be assigned with similar ranking scores. It is this term that makes the weights to be query dependent. $\gamma \geq 0$ is a regularization parameter introduced for balancing the query-dependent regularization term. We can simplify the query-dependent regularization term by using graph Laplacian (Chung 1997), which has been successfully used in many semi-supervised learning applications (Chapelle, Schölkopf, & Zien 2006). Specifically, we put all the similarities $s_{k,l}$ into a matrix $S = [s_{k,l}]_{N_u\times N_u}$. We construct the normalized graph Laplacian matrix L for S as follows:

$$L = I - D^{-1/2}SD^{-1/2}, \tag{9}$$

where $D = diag(d_1, d_2, \ldots, d_{N_u})$ is the degree matrix with the diagonal elements defined as $d_i = \sum_{j=1}^{N_u} s_{ij}$. By adopting the normalized graph Laplacian, we can rewrite (8) as:

$$\min_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^\top (I + \gamma G^\top L G)\mathbf{w} + C\sum_{i=1}^{N_q} \xi_i \tag{10}$$

$$s.t. \quad \mathbf{w}^\top \mathbf{b}^i \geq 1 - \xi_i,$$
$$\xi_i \geq 0, w_j \geq 0, i = 1 \ldots N_q, j = 1 \ldots m.$$

where G is an $N_u \times m$ matrix with element $G_{ij} = g_i(d_j^q)$, $1 \leq i \leq m$, and $1 \leq j \leq N_u$; $\mathbf{b}^i = (b_1^i, \ldots, b_m^i)$ with element $b_j^i = tr([A^i]^\top R^{i,j})$.

Using the Lagrange multipliers method, we can derive the dual problem of (10) as follows:

$$\max_{\alpha} \quad \sum_{j=1}^{N_q} \alpha_j - \frac{1}{2}\Big(\sum_{i=1}^{N_q} \alpha_i b^i + \eta\Big)^\top H \Big(\sum_{i=1}^{N_q} \alpha_i b^i + \eta\Big)$$

$$s.t. \quad 0 \leq \alpha_i \leq C, i = 1, \ldots, N_q \tag{11}$$

$$\eta_j \geq 0, j = 1, \ldots, m \tag{12}$$

where $H = \big(I + \gamma G^\top L G\big)^{-1}$. This is a standard quadratic program (QP), similar to the dual optimization problem of SVMs. We can solve the above optimization efficiently by using either the standard QP solvers or the QP solvers used for solving SVMs. Once the dual optimization is solved, we can find $\mathbf{w}$ by applying the KKT condition, i.e.,

$$\mathbf{w} = \big(I + \gamma G^\top L G\big)^{-1} \sum_{i=1}^{N_q} \alpha_i \mathbf{b}^i \tag{13}$$

Experimental Results

Experimental Dataset

In our experiments, we adopt a standard experimental testbed, i.e., the OHSUMED collection (Hersh *et al.* 1994), which is a benchmark dataset widely used for document information retrieval (Liu *et al.* 2007a; Cao *et al.* 2006; Xu & Li 2007; Cao *et al.* 2007). This dataset is a collection of documents and queries on medicines, which contains 348,566 references and 106 queries. Relevance judgements were made on 16,140 query-document pairs, in which three relevance degrees are assigned: *definitely relevant*, *possibly relevant*, and *not relevant*.

In our experiments, each instance is represented by an m-dimensional vector, in which each dimension represents the ranking score produced by a ranker for a given query-document pair. In particular, 25 different ranking models are used in the experiments, including 10 kinds of models on low-level features from the fields of title and abstract respectively, and 5 kinds of models on high-level features from the combination of title and abstract. Table 1 and Table 2 show a list of ranking models with low-level and high-level features respectively.

Comparison Methods

To evaluate the performance of the proposed technique, we engage several state-of-the-art ranking techniques recently proposed in the information retrieval field. In particular, we compare our solution with 5 state-of-the-art ranking techniques as follows:

(1) **RankBoost**: the well-known ranking method that employs a typical boosting approach by combining multiple weak rankers to get a strong ranker (Freund *et al.* 2003);

(2) **Ranking SVM**: the well-known ranking SVM method, i.e., formulating the ranking task as a binary classification problem on example pairs, and then to solve the problem using SVMs (Herbrich, Graepel, & Obermayer 2000);

(3) **ListNet**: a ranking method that considers a list of objects as "instance" in the learning task (Cao *et al.* 2007);

(4) **AdaRank**: a ranking method using AdaBoosting (Xu & Li 2007), i.e., repeatedly constructing 'weak rankers' on the basis of re-weighted training queries and finally linearly combining the weak rankers for making ranking predictions.

(5) **MHR-BC**: a recently proposed ranking method using multiple hyperplanes, i.e., by learning multiple hyperplanes from training data, and then aggregating the ranking results of these hyperplanes to get the final ranked result (Qin *et al.* 2007).

(6) **SER**: the proposed supervised ensemble ranking method, which is equivalent to the SSER when setting the regularization γ to 0;

(7) **SSER**: the proposed semi-supervised ensemble ranking method.

In our experiments, we implemented and evaluated the proposed methods in MATLAB on a Windows PC with Duro-Core 3.4GHz CPU and 3GB RAM. Specifically, for implementing the SER and SSER algorithms, we employ a

Table 1: A list of rank models based on low-level features

IDs	Rank Models	Descriptions
L1	$\sum_{q_i \in q \cap d} c(q_i, d)$	Term Frequency (TF)
L2	$\sum_{q_i \in q \cap d} \log(c(q_i, d) + 1)$	Described in (Cao *et al.* 2006)
L3	$\sum_{q_i \in q \cap d} \frac{c(q_i, d)}{\lvert d \rvert}$	Normalized TF
L4	$\sum_{q_i \in q \cap d} \log\big(\frac{c(q_i, d)}{\lvert d \rvert}\big) + 1$	Described in (Cao *et al.* 2006)
L5	$\sum_{q_i \in q \cap d} \log\big(\frac{\lvert C \rvert}{df(q_i)}\big)$	Inv. Doc. Frequency (IDF)
L6	$\sum_{q_i \in q \cap d} \log\Big(\log(\frac{\lvert C \rvert}{df(q_i)})\Big)$	Described in (Cao *et al.* 2006)
L7	$\sum_{q_i \in q \cap d} \log\big(\frac{C}{c(q_i, C)} + 1\big)$	Described in (Cao *et al.* 2006)
L8	$\sum_{q_i \in q \cap d} \log\big(\frac{c(q_i, d)}{\lvert d \rvert}\log(\frac{\lvert c \rvert}{df(q_i)}) + 1\big)$	Described in (Cao *et al.* 2006)
L9	$\sum_{q_i \in q \cap d} c(q_i, d)\log(\frac{\lvert C \rvert}{df(q_i)})$	TF*IDF
L10	$\sum_{q_i \in q \cap d} \log\big(\frac{c(q_i, d)}{\lvert d \rvert}\frac{\lvert C \rvert}{\lvert c(q_i, C) \rvert} + 1\big)$	Described in (Cao *et al.* 2006)

Table 2: A list of rank models based on high-level features

IDs	Rank Models	Descriptions
H1	BM25 score	Okapi BM25 Model (Robertson et al. 1998)
H2	log(BM25 score)	Modified Okapi BM25 (Robertson et al. 1998)
H3	LMIR-DIR	Language model by DIR smoothing (Zhai et al. 2001)
H4	LMIR-JM	Language model by JM smoothing (Zhai et al. 2001)
H5	LMIR-ABS	Language model by ABS smoothing (Zhai et al. 2001)

standard QP solver to solve the optimization problems. The normalized graph Laplacian matrix L is constructed by a standard graph construction method, i.e., first building k-nearest neighbor graph (k=5 in our experiments) with Euclidean distance to obtain a sparse weight matrix S, and then getting L by normalizing the weight matrix S with Eq. (9).

For the experimental setup, we conducted a set of 5 experiments with 5-fold cross validation, in which 3-fold data are used for training, 1-fold data are used for validation to determine parameters, and the rest 1-fold data are used for testing. For the SSER algorithm, there are several parameters, including γ, C, θ and δ. In our experimental approaches, C, δ and θ are simply set to 1, 1 and 0.5 respectively without tuning. We only determined the best parameter γ from the validation set. In most cases, we found that setting $\gamma = 0.5$ tends to work fairly comparable to the best performance.

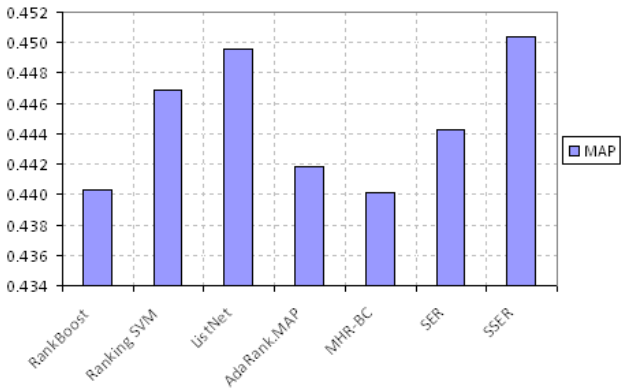

Figure 1: Comparison of MAP performance for the proposed method with respect to other approaches.

Performance Evaluation

To evaluate the performance of the ensemble ranking algorithms, we adopt the standard performance metric, average precision (AP) and Mean average precision (MAP), which are widely used in the information retrieval field (Baeza-Yates & Ribeiro-Neto 1999). For a given query, the average precision is defined as:

$$AP = \frac{1}{rel} \sum_{k=1} Prec@k \tag{14}$$

where rel is the number of relevant documents, and $Prec@k$ is the percentage of relevant documents in the top k ranked documents. MAP is the mean of the average precision scores over a set of queries.

Figure 1 shows the MAP performance of the compared methods obtained by averaging the results from 5 experiments. From the results, we can see that the proposed supervised ensemble ranking (SER) method outperforms the RankBoost, AdaRank, and MHR-BC methods, but is slightly worse than the ranking SVM approach. Among all compared methods, the proposed semi-supervised ensemble ranking (SSER) method achieved the best MAP performance, outperforming all of the state-of-the-art approaches in our comparisons. This shows that the proposed scheme is effective for improving the retrieval performance.

Table 3: Evaluation on training time efficiency (seconds).

Algorithm	Trial1	Trial2	Trial3	Trial4	Trial5	Avg.
Rank.SVM*	43,035.1	50,681.3	31,280.6	30623.3	29761.9	37076.4
Rank.SVM†	1,327.4	1,295.2	1,422.1	1132.0	965.6	1228.5
SER	5.7	7.1	7.2	5.8	5.5	6.3
SSER	7.1	8.3	8.3	7.0	6.6	7.5

*Rank.SVM is based on $\text{SVM}^{\text{light}}$ (Joachims 1999), which is implemented in C.
†Rank.SVM is based on $\text{SVM}^{\text{struct}}$ (Yue *et al.* 2007), which is implemented in C.
SER and SSER are implemented using the default quadprog solver in MATLAB.

To further examine the performance in detail, we also show the evaluation results of average precision on top retrieval documents in Figure 2. From the results, we found that the proposed SSER method outperforms all of the compared methods on rank-1 and rank-2 results. This is important for an information retrieval task since users often click only on top several ranked results. For example, some recent survey on analyzing Web search engine click rates [2] showed that the percentage of clicks on rank-1 and rank-2 results accounts for more than 50% of all clicks.

Further, we are aware that the regularization parameter γ is critical to the performance of the SSER algorithm. Figure 3 illustrates the influence of different values of γ on the retrieval performance in one experiment. From this result, we can see that the optimal γ seems close to 0.5. Currently we can only determine the best parameter by the validation set. Advanced techniques may be studied in future work.

Finally, we evaluate the time efficiency of the proposed algorithms. Table 3 shows the experimental results of time performance by comparing our algorithms with two variants of ranking SVMs that use $\text{SVM}^{\text{light}}$ and $\text{SVM}^{\text{struct}}$ respectively. Clearly, the proposed algorithms are significantly more efficient than ranking SVMs. Specifically, both ranking SVMs took more than 10 hours and 20 minutes respectively for each training on average, while our algorithms only need several seconds. As we explained the reason before, the ranking SVM in each trial (63 queries in the training set) has to engage about $350,000$ rank constraints, but our approach only requires about 126 constraints on average.

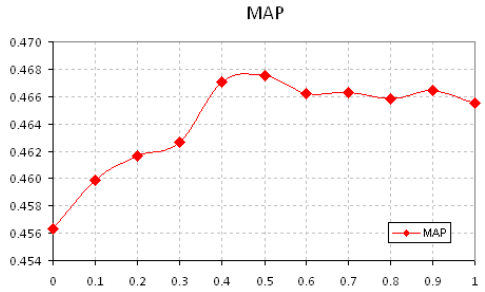

Figure 3: Performance evaluation of the SSER method with different values of the regularization parameter (γ).

Conclusion

This paper proposed a novel semi-supervised ensemble ranking technique for document retrieval. We first formu-

[2] http://www.jimboykin.com/click-rate-for-top-10-search-results/

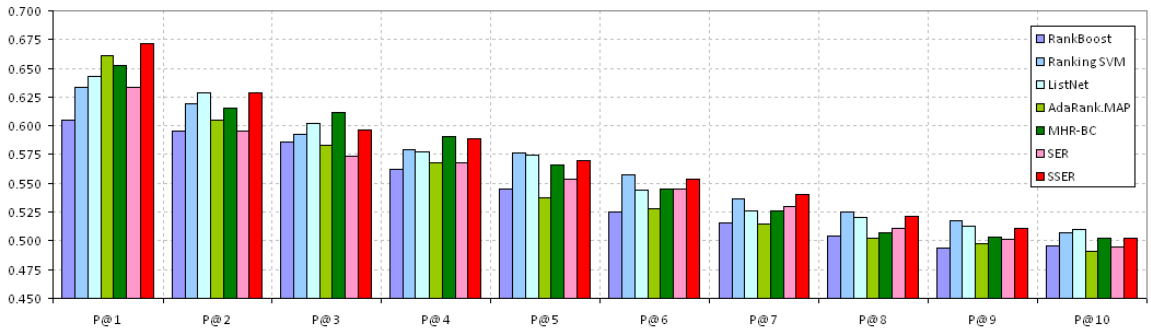

Figure 2: Evaluation of average precision performance on rank-1 to rank-10 retrieval documents.

lated a new supervised ensemble ranking method similar to the ranking SVM approach but is significantly more efficient. Based on the proposed scheme, we then presented the novel semi-supervised ensemble ranking technique, which is formulated as a quadratic program that can be solved efficiently. Encouraging experimental results validated the effectiveness of the proposed method.

One main drawback with the proposed framework is that it has to learn a different set of weights *online* for each query as opposed to supervised ranking approaches that learn combination weights offline. Clearly, this drawback will make the proposed ranking framework infeasible for very large-scale applications. In the future, we plan to examine more efficient approaches to alleviate the high computational cost for online learning. In addition, the regularization parameter used in the proposed framework is decided empirically. In the future, we plan to investigate more systematical approaches for finding the optimal regularization parameter.

Acknowledgments

We would like to thank Dr. Liu at Microsoft Research Asia for sharing the LETOR testbed for our experiments. The work was supported in part by the National Science Foundation (IIS-0643494), National Institute of Health (1R01-GM079688-01), and Singapore NTU Academic Research Grant (RG67/07). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF and NIH.

References

Agarwal, A., and Chakrabarti, S. 2007. Learning random walks to rank nodes in graphs. In *Proc. ICML2007*.

Agarwal, S. 2006. Ranking on graph data. In *Proc. ICML2006*.

Baeza-Yates, R. A., and Ribeiro-Neto, B. A. 1999. *Modern Information Retrieval*. ACM Press / Addison-Wesley.

Burges, C.; Shaked, T.; Renshaw, E.; Lazier, A.; Deeds, M.; Hamilton, N.; and Hullender, G. 2005. Learning to rank using gradient descent. In *Proc. ICML'05*, 89–96.

Cao, Y.; Xu, J.; Liu, T.-Y.; Li, H.; Huang, Y.; and Hon, H.-W. 2006. Adapting ranking svm to document retrieval. In *Proc. SIGIR'06*, 186–193.

Cao, Z.; Qin, T.; Liu, T.-Y.; Tsai, M.-F.; and Li, H. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proc. ICML'07*, 129–136.

Chapelle, O.; Schölkopf, B.; and Zien, A. 2006. *Semi-Supervised Learning*. Cambridge, MA: MIT Press.

Chung, F. 1997. *Spectral Graph Theory*. American Mathematical Society.

Dwork, C.; Kumar, R.; Naor, M.; and Sivakumar, D. 2001. Rank aggregation methods for the web. In *Proc. WWW'01*, 613–622.

Freund, Y.; Iyer, R.; Schapire, R. E.; and Singer, Y. 2003. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research* 4:933–969.

Herbrich, R.; Graepel, T.; and Obermayer, K. 2000. Large margin rank boundaries for ordinal regression. In Smola; Bartlett; Schoelkopf; and Schuurmans., eds., *Advances in Large Margin Classifiers*. MIT Press, Cambridge, MA.

Hersh, W.; Buckley, C.; Leone, T.; and Hickam, D. 1994. Ohsumed: an interactive retrieval evaluation and new large test collection for research. *Proc. SIGIR 1994*.

Joachims, T. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*. Cambridge, MA: MIT Press. chapter 11, 169–184.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *Proceeding of ACM SIGKDD Conference*, 133–142.

Li, L., and Lin, H.-T. 2007. Ordinal regression by extended binary classification. In *NIPS 19*, 865–872.

Liu, T.-Y.; Qin, T.; Xu, J.; Xiong, W.; and Li, H. 2007a. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *SIGIR'07 Workshop: LR4IR 2007*.

Liu, Y.-T.; Liu, T.-Y.; Qin, T.; Ma, Z.-M.; and Li, H. 2007b. Supervised rank aggregation. In *Proc. WWW'07*, 481–490.

Nallapati, R. 2004. Discriminative models for information retrieval. In *Proc. SIGIR'04*, 64–71.

Ponte, J. M., and Croft, W. B. 1998. A language modeling approach to information retrieval. In *Proc. SIGIR'98*, 275–281.

Qin, T.; Zhang, X.-D.; Wang, D.-S.; Liu, T.-Y.; Lai, W.; and Li, H. 2007. Ranking with multiple hyperplanes. In *SIGIR'07*, 279–286.

Robertson, S.; Walker, S.; and Beaulieu, M. 1998. Okapi at trec-7: Automatic ad hoc, filtering, vlc and interactive track. In *Proc. the Seventh Text REtrieval Conference*.

Salton, G. 1971. *The SMART retrieval system; experiments in automatic document processing*. Prentice-Hall, NJ.

Usunier, N.; Truong, V.; Amini, M. R.; and Gallinari, P. 2005. Ranking with unlabeled data: A first study. In *NIPS 2005 workshop: Learning to Rank*.

Xu, J., and Li, H. 2007. AdaRank: a boosting algorithm for information retrieval. In *SIGIR'07*, 391–398.

Yue, Y.; Finley, T.; Radlinski, F.; and Joachims, T. 2007. A support vector method for optimizing averageprecision. In *SIGIR'07*.