Semi-Supervised Document Retrieval

Ming Li [a], Hang Li [b], Zhi-Hua Zhou [a,*]

[a] *National Key Laboratory for Novel Software Technology*
Nanjing University, Nanjing 210093, China

[b] *Microsoft Research Asia, 49 Zhichun Road, Beijing 100080, China*

[Submitted: August 3, 2007; Revised: October 25, 2008; Accepted: November 9, 2008]

Abstract

This paper proposes a new machine learning method for constructing ranking models in document retrieval. The method, which is referred to as SSRank, aims to use the advantages of both the traditional Information Retrieval (IR) methods and the supervised learning methods for IR proposed recently. The advantages include the use of limited amount of labeled data and rich model representation. To do so, the method adopts a semi-supervised learning framework in ranking model construction. Specifically, given a small number of labeled documents with respect to some queries, the method effectively labels the unlabeled documents for the queries. It then uses all the labeled data to train a machine learning model (in our case, Neural Network). In the data labeling, the method also makes use of a traditional IR model (in our case, BM25). A stopping criterion based on machine learning theory is given for the data-labeling process. Experimental results on three benchmark data sets and one web search data set indicate that SSRank consistently and almost always significantly outperforms the baseline methods (unsupervised and supervised learning methods), given the same amount of labeled data. This is because SSRank can effectively leverage the use of unlabeled data in learning.

Key words:
Information Retrieval, Machine Learning, Data Mining, Learning to Rank,
Semi-Supervised Learning,

[*] Corresponding author. Tel: +86-25-8368-6268; Fax: +86-25-8368-6268; Email: zhouzh@nju.edu.cn

1 Introduction

Recently, supervised machine learning methods have been applied to ranking function construction in document retrieval (Joachims, 2002; Burges et al., 2005; Gao et al., 2005; Cao et al., 2006; de Almeida et al., 2007; Cao et al., 2007; Xu and Li, 2007; Yue et al., 2007). This approach offers many advantages, because it employs a rich model for document ranking. For instance, it is easy to add new 'features' into the ranking model. In fact, recent investigations have demonstrated that supervised learning approach works better than the conventional IR methods for relevance ranking. On the other hand, the machine learning approach also suffers from a drawback that traditional IR approaches such as BM25 and Language Modeling do not. That is, it needs a large amount of labeled data for training and usually the labeling of data is expensive. (In that sense, the traditional IR methods are 'unsupervised learning methods').

One question arises here: can we leverage the merits of the two approaches and develop a method that combines the uses of the two? This is exactly the issue we address in this paper. Specifically, we propose a method on the basis of semi-supervised learning. To the best of our knowledge, there has been no previous work focusing on this problem.

A ranking function based on unsupervised learning can always be created without data labeling. Such a function can work reasonably well. Thus, the problem in this paper can be recast as that of how to enhance the ranking accuracy of a traditional IR model by using a supervised learning method and a small amount of labeled data. On the other hand, supervised learning for ranking usually requires the use of a large amount of labeled data to accurately train a model, which is very expensive. The addressed problem can also be viewed as that of how to train a supervised learning model for ranking by using a small amount of labeled data and by leveraging a traditional IR model.

The key issue for our current research, therefore, is to design a method that can effectively use a small number of labeled data and a large number of unlabeled data, and can effectively combine supervised learning (e.g., RankNet) and unsupervised learning (e.g., BM25) methods for ranking model construction.

Our method, referred to as SSRank (Semi-Supervised Rank), naturally utilizes the machinery of semi-supervised learning to achieve our goal. In training, given a certain number of queries and the associated labeled documents, SSRank ranks all the documents for the queries using a supervised learning model trained with the labeled data, as well as using an unsupervised learning model. As a result, for each query, two ranking results of the documents with respect to the query are obtained. SSRank then calculates the relevance score of each *unlabeled* document for each query, specifically, the probability of being relevant or being in a high rank of relevance. It labels the unlabeled documents if their relevance scores are high enough. With the labeled data, a new supervised learning model can be constructed. SSRank repeats the process, until a stopping criterion is met. In this paper, we propose a stopping criterion on the basis of machine learning theory.

Experimental results on three benchmark data sets and one web search data set show that the proposed method can significantly outperform baseline methods (either a supervised method using the same amount of labeled data or an unsupervised method).

The setting of SSRank is somewhat similar to that of relevance feedback (or pseudo relevance feedback). There are also some clear differences between SSRank and relevance feedback (or pseudo relevance feedback), however, as will be explained in Section 2.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 explains the semi-supervised learning method: SSRank. Section 4 gives the experimental results. Section 5 provides our conclusion and discusses future work.

2 Related Work

2.1 Learning for Document Retrieval

In Information Retrieval, traditionally ranking models are constructed in an unsupervised fashion, for example, BM25 (Robertson and Hull, 2000) and Language Model (e.g., (Lafferty and Zhai, 2001)) are functions based on degree of matching between query and document. There is no need of data labeling, which is no doubt an advantage. Many experimental results show that these models are very effective and they represent state-of-the-art methods for document retrieval.

In Machine Learning, the problem of 'learning to rank' became a popular research topic recently and many methods have been proposed. The ranking problem is defined as that of assigning scores to instances and sorting the instances by using the scores. A typical setting in learning to rank is that instances labeled with a number of ordered categories or 'ranks' are given and a ranking model is created using the labeled data. For example, Herbrich et al. (2000) proposed transforming the problem of learning to rank into a problem of classifying instance pairs and learning the classification model by means of support vector machines. The method is referred to as Ranking SVM. Freund et al. (2003) proposed a similar approach to learning to rank, but using the framework of boosting.

Learning to rank (supervised learning) can also be applied to document retrieval, as document retrieval is in nature a ranking problem. Recently, there have been many investigations in the IR community along this direction. For example, Joachims (2002) trained Ranking SVM for document retrieval using click-through data. Gao et al. (2005) trained a linear discriminant model with features generated by a language model, and made use of the model in document retrieval. Burges et al. (2005) utilized cross entropy as the loss function in learning and employed Neural Network as the ranking model. Their method, called RankNet, was applied to general web search. Cao et al. (2006) adapted Ranking SVM to document retrieval by modifying the loss function such that the model is trained with more considerations on higher ranks and queries with fewer relevant documents. Besides, genetic programming has been applied to ranking function construction for document retrieval (Fan et al., 2004; Trotman, 2005; Cummins and O'Riordan, 2006; de Almeida et al., 2007). Recently, learning to rank has been extended from pairwise training approach to listwise training approach, and successfully applied to document retrieval problem (Cao et al., 2007; Xu and Li, 2007; Yue et al., 2007). Since it is easy to add new features into the rank model, the supervised learning approach enjoys higher accuracy and better adaptability. The previous work shows that this is exactly the case and a ranking method based on supervised learning usually performs better than an unsupervised traditional IR method.

2.2 Semi-Supervised Learning

Semi-supervised learning (Chapelle et al., 2006; Zhu, 2005) is a machine learning paradigm in which the model is constructed with a small number of labeled instances and a large number of unlabeled instances. One key idea in semi-supervised learning is to label unlabeled data using certain techniques and thus increase the amount of labeled training data.

Many semi-supervised learning methods have been proposed. Typical methods include those using the EM algorithm (Dempster et al., 1977) to estimate the parameters of a generative model and the labels of unlabeled data (Shahshahani and Landgrebe, 1994; Miller and Uyar, 1997; Nigam et al., 2000), those defining a graph over the data instances on the basis of certain similarity metric and determining the labels of unlabeled data (Blum and Chawla, 2001; Zhou et al., 2003; Zhu et al., 2003; Belkin and Niyogi, 2004), and those applying 'co-training' (Blum and Mitchell, 1998) to construct multiple learners to label unlabeled data (Blum and Mitchell, 1998; Goldman and Zhou, 2000;

Zhou and Li, 2005b; Li and Zhou, 2007; Yu et al., 2007; Zhou et al., 2007). Previous work on semi-supervised learning mainly focused on classification (e.g., (Blum and Mitchell, 1998; Nigam et al., 2000)) and regression (e.g., (Zhou and Li, 2005a; Brefeld et al., 2006; Zhou and Li, 2007)).

There are only a few studies on semi-supervised learning for ranking. Usunier et al. (2005) presented a theoretical work, which extended the generalization bound of semi-supervised learning to ranking and theoretically demonstrated that unlabeled data is helpful for ranking. Chu and Ghahramani (2005) extended Gaussian Process for preference learning to a semi-supervised setting by incorporating the graph Laplacian which is constructed using all the training examples and the pairwise relationship among them. Note that in document retrieval, ranking function models the ordering of retrieved document *within* each query rather than across queries. In other words, documents retrieved according to different queries are not directly comparable. So, the general method proposed by (Chu and Ghahramani, 2005) is not suitable for document retrieval tasks.

Semi-supervised learning has also been applied to applications, such as text classification (e.g., (Nigam et al., 2000; Li and Liu, 2003; Liu et al., 2003)), image retrieval (e.g., (Zhou et al., 2004, 2006)) and computer-aided diagnosis (Li and Zhou, 2007). Recently, it has been used to classify relevant documents for pseudo-relevance feedback (Huang et al., 2006).

Note that existing semi-supervised learning methods may not be directly applicable to learning of ranking functions in document retrieval. The reason is that ranking is an issue different from conventional learning problems such as classification and regression. In learning for ranking, one needs to learn a model that can map instances to *ordered categories*. Possibly the first semi-supervised learning method that can be applied to "learning to rank" in retrieval task is (Zhou et al., 2004, 2006) which was designed for image retrieval. In that method, features are extracted from either the query image or the retrieved image separately, while in learning to rank for document retrieval methods (e.g., (Joachims, 2002), (Cao et al., 2006) and (Xu and Li, 2007)), features are extracted based on query-document pairs. Recently, while the current paper is being reviewed, two methods that exploit unlabeled data are proposed and can be adapted to document retrieval task. Amini et al. (2008) labeled the nearest unlabeled instance of each labeled instance with the same label of this labeled instance, and then adapted RankBoost (Freund et al., 2003) to learn ranking function based on the both the originally and newly labeled training set. Duh and Kirchhoff (2008) exploited unlabeled data in a *transductinve* settings, where KPCA was repeatedly applied to the unlabeled instances of each query. Then, all labeled instances and the unlabeled instances of this query were projected into this new space, and a ranking function was learned using the projected labeled instances to rank all the unlabeled instances. To the best of our knowledge, our current paper is the first work that leverages the learning to rank machinery and conventional document retrieval model to address the semi-supervised document retrieval problem.

2.3 Relevance Feedback

Relevance feedback (Rocchio, 1971; Salton and Buckley, 1990; Harman, 1992; Shen and Zhai, 2005) and pseudo relevance feedback (Attar and Fraenkel, 1977; Xu and Croft, 1996; Sakai et al., 2005; Tao and Zhai, 2006) are known to be effective methods for improving the performances of document retrieval. In relevance feedback, given a query by the user, the retrieval system returns a number of documents and asks the user to make judgments on the relevance of the documents with respect to the query. Then, the system uses the judged documents to modify the query using techniques such as query expansion and query term reweighting, and re-retrieves documents with the modified query. In the re-retrieval process, Rochio's algorithm (Rocchio, 1971) etc are employed. Instead of asking explicit feedbacks from the user, pseudo relevance feedback takes the top k retrieved documents as "relevant documents".

There are similarities between the settings of relevance feedback (or pseudo relevance feedback) and that of SSRank in this paper. Specifically, both approaches attempt to leverage a certain number of relevance judgements to improve the performance of document retrieval. However, there are also clear differences between conventional relevance feedback (or pseudo relevance feedback) and SSRank. Firstly, relevance feedback usually makes use of the labeled documents to *reform the query*, while SSRank makes use of the labeled documents to *refine* the ranking model. Secondly, relevance feedback (or pseudo relevance feedback) is usually an *online* process, which is conducted for each individual query . In contrast, learning of SSRank is an *offline* process, which is conducted for all the queries with partial relevance judgments (some documents are labeled, but the remaining are not). Thirdly, while relevance feedback aims to improve the retrieval results for the current query, while SSRank is targeted at improvements on the relevance of new queries. Fourthly, the co-training style algorithm in SSRank largely differs from Rocchio's algorithm etc, used in relevance feedback (or pseudo relevance feedback).

3 The proposed method: SSRANK

3.1 General Framework

Suppose that there is a document collection. In retrieval, the documents retrieved with a given query are sorted using a ranking model such that the documents relevant to the query are on the top, while the ranking model is created using machine learning. In learning, a number of queries are given, and for each query a number of documents are retrieved and the corresponding labels are attached. The labels associated with the documents for a query represent the relevance degrees of the documents with respect to the query. For each query and document pair, we construct a feature vector. TF-IDF score, for example, can be a feature. We construct the ranking model using all the feature vectors and their corresponding labels. For simplicity, we also refer a feature vector as a document (associated with a certain query).

Let x denote an instance (feature vector), $x \in \mathcal{X} : \mathcal{X} \subseteq \mathbb{R}^d$ and let y denote a label representing a relevance degree, or a rank, $y \in \mathcal{Y} : \mathcal{Y} = \{r_1, r_2, ..., r_M\}$. There exists a total order between the labels in $\mathcal{Y}$: $r_M \succ r_{M-1} \succ ... \succ r_1$, where $r_i \succ r_j$ implies that r_i has higher relevance than r_j. Let $f : \mathcal{X} \mapsto \mathbb{R}$ be a ranking function. In ranking, instances (corresponding to documents) with respect to a query are sorted according to f such that $x_i \succ x_j$ if $f(x_i) > f(x_j)$. The learning of the ranking function can be performed by employing supervised learning methods such as Ranking SVM and RankNet.

In this paper we consider the case in which for each query in the training data only a small number of documents (instances) associated with it are labeled and the remaining documents (instances) are unlabeled. Note that this is commonly true in IR. Let $X = \{x_1, x_2, ..., x_N\}$ be the set of training instances from all the training queries. Some instances in X have been manually labeled. Let $L = \{(x_l, y_l)\}_{l=1}^{|L|}$ and $U = \{x_u\}_{u=|L|+1}^{N}$ respectively denote the sets of labeled instances and unlabeled instances.

We propose a semi-supervised learning method to accomplish the learning task. For any unlabeled instance x_u we calculate the scores for all the possible labels, and then choose the most likely label for it. With the labeled data set augmented with these newly labeled instances, we train a more accurate ranking model.

We consider using multiple *base* ranking functions representing multiple ‘views’ and then combining the uses of them for labeling the unlabeled data, following the idea of co-training. Specifically, there are V base ranking functions $f_1 : \mathcal{X} \mapsto \mathbb{R}$, ..., $f_V : \mathcal{X} \mapsto \mathbb{R}$. Each base ranking function can assign scores to the instances with respect to a query. Suppose that for each view

v, x_u is assigned a score representing the likelihood of its being in rank r_m: $S_v(y_u = r_m|x_u), r_m \in \mathcal{Y}$ with the base ranking function f_v. We can then calculate the final score of x_u's being in rank r_m: $S(y_u = r_m|x_u)$, from the scores of all the views, and choose the rank that has the highest score as the rank of x_u (Ranks are randomly picked up when there is a tie). Several strategies for the combination can be considered. First, we can employ linear combination

$$S\left(y_u = r_m|x_u\right) = \sum_{v=1}^{V} w(v) S_v(y_u = r_m|x_u) \tag{1}$$

where $w(v)$ is weight of view v and $\sum_v w(v) = 1$. Here, we can define $w(v)$ as the confidence of judgment by f_v. Alternatively, we can employ majority voting

$$S\left(y_u = r_m|x_u\right) = \frac{1}{V}\sum_{v=1}^{V} \delta\left(r_m = \arg\max_{r_i \in \mathcal{Y}} S_v(y_u = r_i|x_u)\right) \tag{2}$$

where $\delta(B)$ takes 1 as value if B is true and 0 otherwise.

Note that there is a total order relationship existing in $\mathcal{Y}$, and thus the two strategies are not the same as those in learning for multi-class classification.

3.2 Score Calculation

We propose a way of calculating scores of unlabeled data *for each view* in the above semi-supervised learning method.

Using one of the base ranking functions f_v, we can rank the instances (corresponding to documents) associated with a query. Note that some of the instances are labeled while the others are unlabeled. If $f_v(x_i)$ is larger than $f_v(x_j)$, then it is likely x_i has a higher rank than x_j, i.e., $y_i \succ y_j$.

We assign a probability vector to each instance (either labeled or unlabeled) using the scores of all the labeled instances given by the base ranking function.

First, we define the probability of x_i being ranked no lower than x_j by f_v (i.e., $y_i \succeq y_j$) with respect to query q as

$$P_v\left(y_i \succeq y_j|x_i, x_j, q\right) = \frac{e^{f_v(x_i)-f_v(x_j)}}{1 + e^{f_v(x_i)-f_v(x_j)}} \tag{3}$$

following the proposal in (Burges et al., 2005). We next define the probability of instance x_i having a rank no lower than r_m as

$$P_v\left(y_i \succeq r_m | x_i, q\right) = \frac{1}{l_m} \sum_{\substack{x_j \in \sigma q \\ y_j = rm}} P_v\left(y_i \succeq y_j | x_i, x_j, q\right) \tag{4}$$

where σ_q denotes the *labeled* instances with respect to q and l_m denotes the number of instances in σ_q labeled as r_m.

Since there are M ranks, we calculate M such probabilities. Each instance, both labeled and unlabeled, then is assigned an M-dimensional probability vector, calculated according to Eq. 4. All the probability vectors from *all the queries* are collected together in the new probability space. In the new space, we then employ the k-Nearest Neighbor method (Mitchell, 1997) to calculate $S_v(y_u = r_m|x_u)$, the score of possible rank r_m of instance x_u, from the ranks of its k nearest labeled instances, where Euclidean distance is used as the metric.

It is noteworthy that mapping instances from the feature space into the probability space is essential for our score calculation method. Specifically, the mapping makes instances from different queries comparable. This is because in the probability space the probability vectors represent the likelihood values of instances in different ranks, which do not depend on queries. Moreover, the probability vectors contain the ordering information in the ranking lists. Consequently, we can employ a method like kNN to make predictions on the ranks of unlabeled instances from all the labeled instances .

We note that alternative ways for labeling unlabeled data may exist. For instance, one can make use of $P(r_{k+1} \succ y_i \succeq r_m|x_i, q)$ in the score calculation. It seems, however, that it is hard to accurately estimate the probability, according to our experiment.

3.3 Theoretical Analysis

In a semi-supervised learning method, unlabeled instances can be incorrectly labeled and noise can be introduced. It is important, therefore, to clarify the condition under which data labeling can be continued, in order to enhance the accuracy of the learning. The following proposition provides such a condition.

Proposition 1 *Let m_0 denote the number of labeled instance pairs in the training data and m_1 denote the number of labeled instance pairs in the first iteration of semi-supervised learning. Let e_1 denote the error rate in the newly labeled instance pairs in the first iteration. If the following inequality holds*

$$e_1 < \frac{(a+1) - \sqrt{a+1}}{2a} \tag{5}$$

where $a = m_1/m_0$, then the accuracies of ranking functions can be improved in terms of the lower bound of average precision in the first iteration of the semi-supervised learning.

Let m_t and m_{t-1} respectively denote the number of labeled instance pairs in the t-th iteration and the $(t-1)$-th iteration of semi-supervised learning. Let e_t and e_{t-1} respectively denote the error rate in the newly labeled instance pairs in the t-th iteration and the $(t-1)$-th iteration. If the following inequalities hold

$$0 < \frac{e_t}{e_{t-1}} < \frac{m_{t-1}}{m_t} < 1 \tag{6}$$

where $e_{t-1} < 0.5$ and $e_t < 0.5$, then the accuracies of ranking functions can be improved in terms of the lower bound of average precision in the t-th iteration $(t > 1)$ of the semi-supervised learning.

It is not difficult to verify that the proposition holds.

PROOF.

We use two theoretical results obtained in previous work.

First, let us consider using a learning to rank method, for example RankNet (Burges et al., 2005) and Ranking SVM (Joachims, 2002) to create the ranking model . Such a method transforms the ranking problem into that of classifying instance pairs. The learning process, thus, is equivalent to constructing a classifier $h : \mathcal{X} \times \mathcal{X} \mapsto \{+1, -1\}$, where $+1$ and -1 stand for 'ordering the first instance before the second instance' and 'ordering the first instance after the second instance', respectively. Errors made by h imply pair inversions in a ranking. According to Joachims (2002), the performance of a ranking function in terms of average precision in the setting is approximately bounded from below by the inverse of the number of instance pair inversions (errors).

Next, let us analyze the error rate introduced in data labeling, following a similar analysis in (Goldman and Zhou, 2000) and (Zhou and Li, 2005b). We actually utilize the theoretical results on learning from noisy data proposed by Angluin and Laird (1988). Let m and $\eta(< 0.5)$ denote the size of training set and the noise rate in the training set. Let h denote a learned hypothesis that minimizes the disagreement on a sequence of noisy training instances and ϵ denote the worst-case error rate of h. If m, η and ϵ satisfy the following condition

$$m = \frac{c}{\epsilon^2(1-2\eta)^2} \tag{7}$$

where c is a constant, the difference between h and the true hypothesis h^* will be small with very high probability. Letting $u = c/\epsilon^2$, the equation can be re-formalized as the following utility function.

$$u = \frac{c}{\epsilon^2} = m(1 - 2\eta)^2 \tag{8}$$

Let h_0 denoted the hypothesis learned from the labeled instance pairs and h_1 denote the hypothesis learned in the first iteration . To make h_1 have smaller classification error rate than h_0, the utility of h_1 should be larger than that of h_0, i.e.

$$m_0(1 - 2\eta_0)^2 < (m_0 + m_1)(1 - 2\eta_1)^2 \tag{9}$$

where

$$\eta_1 = \frac{\eta_0 m_0 + e_1 m_1}{m_0 + m_1} \tag{10}$$

Assume that there exists no noise in the original training set, and thus $\eta_0 = 0$. Solving the inequity in Eq. 9 yields Eq. 5. It follows that when Eq.5 is satisfied, h_1 makes fewer pair inversions than h_0 and hence the corresponding ranking function f_1 has higher average precision lower bound than f_0.

It is also easy to verify that Eq. 6 holds for the $(t-1)$-th and t-th iterations in a similar way. □

3.4 Algorithm

Now we can build the semi-supervised learning algorithm SSRank on the basis of the discussions above. Fig. 1 shows the pseudo code of the algorithm. We can see that significant differences exist between SSRank and relevance feedback (or pseudo relevance feedback).

In this paper we only consider the uses of two views (i.e., $V = 2$). One ranking function is based on machine learning namely RankNet and the other is based on IR namely BM25. The two views are denoted as *Learning View* and *IR View* respectively. Note that in SSRank only the base ranking function in Learning View is iteratively updated, while the base ranking function in IR View does not change, because the latter is an unsupervised function. We use the theoretical result in Section 3.3 to derive the stopping criterion. The algorithm iterates until the stopping criterion is met.

```
Algorithm: SSRank
    Input: labeled instance set $L$, unlabeled instance set $U$, combining strategy $\mathcal{C}$
           conventional document retrieval method: IR, (e.g. BM25),
           machine learning method for ranking: ML, (e.g. RankNet)
  Process:
           Construct $f^{(i)}$ using IR
           Calculate the scores of the instances w.r.t each query $q$ using $f^{(i)}$
           Calculate the probabilities of all the ranks in $\mathcal{Y}$ for each instance
           Assign scores to the unlabeled instances in $U$ using the method in Section 3.2
           $t \leftarrow 1$
           Learn a ranking function $f^{(l)}$ from $L$: $f^{(l)} \leftarrow \mathrm{ML}(L)$
           Repeat Until $f^{(l)}$ does not change
               Calculate the scores of the instances w.r.t each query $q$ using $f^{(l)}$
               Calculate the probabilities of all the ranks in $\mathcal{Y}$ for each instance
               Assign scores to the unlabeled instances in $U$ using the method in Section 3.2
               Combine the scores from $f^{(l)}$ and $f^{(i)}$ using $\mathcal{C}$ (Section 3.1) to label unlabeled instances
               Construct $L'$ using newly labeled instances
               Calculate the number of newly labeled pairs $m_t$ and estimate the error rate $\hat{e}_t$
               if $t = 1$                                                     % the first iteration
                   if $\hat{e}_t < \frac{1}{2m_t/m_0}\left((m_t/m_0+1) - \sqrt{(m_t/m_0+1)}\right)$    % refer to Eq. 5
                       Learn a ranking function from $L \cup L'$: $f^{(l)} \leftarrow \mathrm{ML}(L \cup L')$
               else                                                           % the other iterations
                   if $m_{t-1} < m_t$ and $\hat{e}_t m_t < \hat{e}_{t-1} m_{t-1}$                 % refer to Eq. 6
                       Learn a ranking function from $L \cup L'$: $f^{(l)} \leftarrow \mathrm{ML}(L \cup L')$
               $t \leftarrow t + 1$
   Output: the learned ranking function $f^{(l)}$
```

Fig. 1. The SSRank algorithm

In each iterations, the main computational cost is on the refinement of the ranking functions. Suppose that the cost of training one ranking function is $O(v)$, where v is a variable indicating the order of the computational cost of the ranking function learning method. For example, for RankNet, $v = cWN^2$ where N is the total number of training examples, c is the number of epochs in training, and W is the total number of weights in the neural network. Since the rank computation and ranking model evaluation in both views (e.g., RankNet and BM25) are extremely fast, the cost is dominated by the refinement of the ranking function generated by the machine learning method, and hence is roughly $O(v)$. Assume the algorithm stops after t iterations, the total cost will be $O(tv)$. Usually t is a small integer (e.g. in most cases t is less than 3). So, the cost of SSRank is just slightly expensive than running a pure supervised algorithm on the labeled data, but the reward is a significant improvement of the performance.

4 Experiments

4.1 Benchmark Data Sets

We used three benchmark data sets on document retrieval in our experiments.

The first two data sets are from the TREC ad-hoc retrieval track. The document collections are from *The Wall Street Journal* (*WSJ*) and *Associated Press* (*AP*), which can be found in TREC Data Disk 2 and 3. *WSJ* contains 74,521 articles from 1990 to 1992, and *AP* contains 158,241 articles from 1988 and 1990. The queries are from the *description fields* of 200 TREC topics (No.101 $\sim$ No.300). Each query has a number of documents associated and they are labeled as 'Relevant' or 'Irrelevant' (to the query). Following a similar practice in (Trotman, 2005), the queries that have less than 10 relevant documents were discarded.

The third data set is the OHSUMED collection (Hersh et al., 1994) from the TREC filtering track. The data set contains 348,566 documents and 106 queries; in total 16,140 documents have been judged as 'Definitely Relevant', 'Partially Relevant', or 'Irrelevant' (to the queries).

Table 1
Statistics of data sets

Data Set	# Queries	# Docs	# Docs Per Query
AP	116	24727	213.16
WSJ	126	40230	319.29
OHSUMED	106	16140	152.26

Table 1 gives the statistics of the data sets. For all the three data sets, stop words were removed and terms were stemmed with Potter Stemmer (Baeza-Yates and Ribeiro-Neto, 1999). Table 2 gives the details of the features used, where $tf(t,d)$ and $idf(t,C)$ respectively denote term frequency of term t in document d and inverse document frequency of t in document collection C, respectively. The features, defined based on query-document pairs, are those widely used in learning methods for IR (e.g., (Nallapati, 2004) and (Cao et al., 2006)).

Table 2
Features defined based on query-document pairs

ID	Feature Value	ID	Feature Value
1	$\sum\limits_{t \in q \cap d} \log\left(tf(t,d)+1\right)$	2	$\sum\limits_{t \in q \cap d} \log\left(\frac{\lvert C \rvert}{tf(t,d)}+1\right)$
3	$\sum\limits_{t \in q \cap d} \log\left(idf(t,C)\right)$	4	$\sum\limits_{t \in q \cap d} \log\left(\frac{tf(t,d)}{\lvert d \rvert}+1\right)$
5	$\sum\limits_{t \in q \cap d} \log\left(\frac{tf(t,d)}{\lvert d \rvert} \cdot idf(t,C)+1\right)$	6	$\sum\limits_{t \in q \cap d} \log\left(\frac{tf(t,d)}{\lvert d \rvert} \cdot \frac{\lvert C \rvert}{tf(t,C)}+1\right)$
7	$\log\left(BM25(q,d)\right)$		

4.2 Evaluation Measures

In the experiments, Normalized Discounted Cumulative Gain (NDCG) (Jarvelin and Kekalainen, 2000) was used to evaluate the performance of the ranking methods. Given a query q_i, the NDCG score at position p in a ranking list ordered by a ranking function is defined as

$$N_i = n_i \sum_{j=1}^{p} \frac{2^{r_j} - 1}{\log(1+j)} \tag{11}$$

where r_j is the rank of the j-th document, and the normalization constant n_i is chosen such that the NDCG score of the ideal ordering becomes 1. The final NDCG score is averaged over the scores of all the queries. In this paper, the NDCG scores at positions of 1, 3, 5 and 10 are reported.

Mean Average Precision (MAP) was also used. MAP stands for the mean of Average Precisions over all the queries. Given a query q_i, Average Precision is defined as

$$AvgPre_i = \sum_{j=1}^{m_i} \frac{I(j)(R_j/j)}{R} \tag{12}$$

where R and R_j denote the number of relevant documents and the number of documents before the position $(j+1)$ respectively, m_i is the number of retrieved documents, and $I(j)$ is an indicator which takes value 1 if the document at position j is relevant and value 0 otherwise. Note that, unlike NDCG, MAP can only handle the cases, in which there are two relevance ranks, i.e. relevant and irrelevant. When there are more than two ranks of relevance, e.g., OHSUMED, the highest rank is treated as relevance and the others irrelevant in calculation of MAP.

4.3 Experiment 1: Comparison with Baselines

We conducted four-fold cross validation on all the data sets in all the experiments. In each fold, for each query in the training set, the documents were randomly split into two groups according to a ratio. In one group the labels on relevance of the documents were used, and in the other group the labels were *withheld* and the documents were viewed as unlabeled. The ratio is referred to as *labeling rate* (μ). For instance, if there are 100 documents and the labeling rate is 10%, then 10 documents are used as labeled data, and 90 documents are used as unlabeled data. In our experiments, we used four different labeling rates: 10%, 20%, 30% and 40%. Methods were evaluated and compared under different labeling rate for each data set. As a result, there are 12 different groups of results (i.e. 3 data sets × 4 labeling rates). To ensure that for most queries relevant instances were selected into the labeled data set, for each query, documents with BM25 scores lower than 0.01 were discarded and were not used in the experiment. The number of relevant instances was roughly one tenth in the experiments.

We then applied SSRank to all the data sets. In our experiments, for Learning View of SSRank we employed RankNet (Burges et al., 2005) and for IR View we employed BM25 (Robertson and Hull, 2000). For the score calculation with k-Nearest Neighbor, k was fixed at 10 (cf., Section 3). For combination of the two views RankNet and BM25 at SSRank, we tried both strategies: linear combination (c.f., Eq. 1) and agreement (a special case of Eq. 2 when $V = 2$), denoted as SSRank-Lin and SSRank-Agr, respectively. For comparison, we also tested SSRank with only one view. The one using RankNet is referred to as SSRank-RN, and the other one is referred to as SSRank-BM.

Table 3
Methods compared in the experiments

Type	Name	Information
Semi-supervised	SSRank-lin	SSRank using linear combination of the two views (c.f. Eq. 1)
	SSRank-Agr	SSRank using agreement combination of the two views (c.f. Eq. 2)
	SSRank-RN	SSRank using only one view where RankNet is used
	SSRank-BM	SSRank using only one view where BM25 is used
Supervised	RankNet-L	RankNet trained only on labeled data
	RankNet-LU	RankNet trained on labeled and unlabeled data with the true labels
Unsupervised	BM25	BM25 is a traditional document retrieval method

Here, we only experiment with two implementations of RankNet as baseline methods. The first one, RankNet-L, uses only the labeled data to train the model. This is what we can obtained with RankNet in real-world tasks. The second one, RankNet-LU, uses both the labeled data and unlabeled data to train the model. Note that this is a “cheating” method, which assumes that it could know the ground-truth labels of all the unlabeled examples. Thus, it is evident that such a method is *infeasible* in real-world tasks. However, it is good to include it in the comparison since it might be the upper performance of SSRank. Besides, BM25 is used as another baseline method. We use the labeled data as the validation set and tune the parameters of BM25 for the best performance. The detailed information of the compared algorithms is tabulated in Table 3.

For each data set and each labeled rate, the proposed methods and baseline

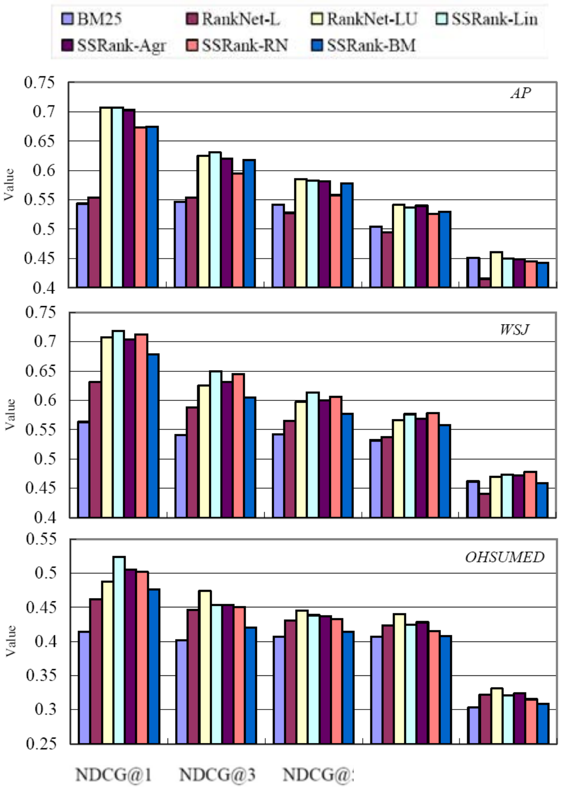

Fig. 2. Performances of methods on three data sets averaged over four labeling rates

methods were evaluated in terms of NDCG and MAP . Figure 2 and Fig. 3 show the results. Due to space limitation, the results of the three data sets are combined together by data sets and by labeling rates.

Fig. 2 shows the average performances of the methods on different data sets. We can see from the figure that SSRank-Lin and SSRank-Agr outperform RankNet-L and BM25. Significant improvements can be observed on *AP* and *WSJ*, while improvement on *OHSUMED* is small. We can also see that SSRank-Lin and SSRank-Agr are superior to SSRank-RN and SSRank-BM.

Fig. 3 shows the average performances of the methods in labeling rates. We can see that SSRank can significantly performs better than the baseline methods under all the four labeling rates. We can also see that SSRank with two views works better than SSRank with only one view.

Statistical significance testing (t-test) at significant level 0.05 shows that SSRank-Lin and SSRank-Agr significantly outperform the baselines in more than half of the twelve settings (three data sets by four labeling rates) in terms of NDCG and MAP. For example, for NDCG@10, SSRank-Lin and SSRank-Agr significantly outperform RankNet-L in 7 and 8 settings, respectively, and they outperform BM25 in 9 and 7 settings, respectively. SSRank-RN is signifi-

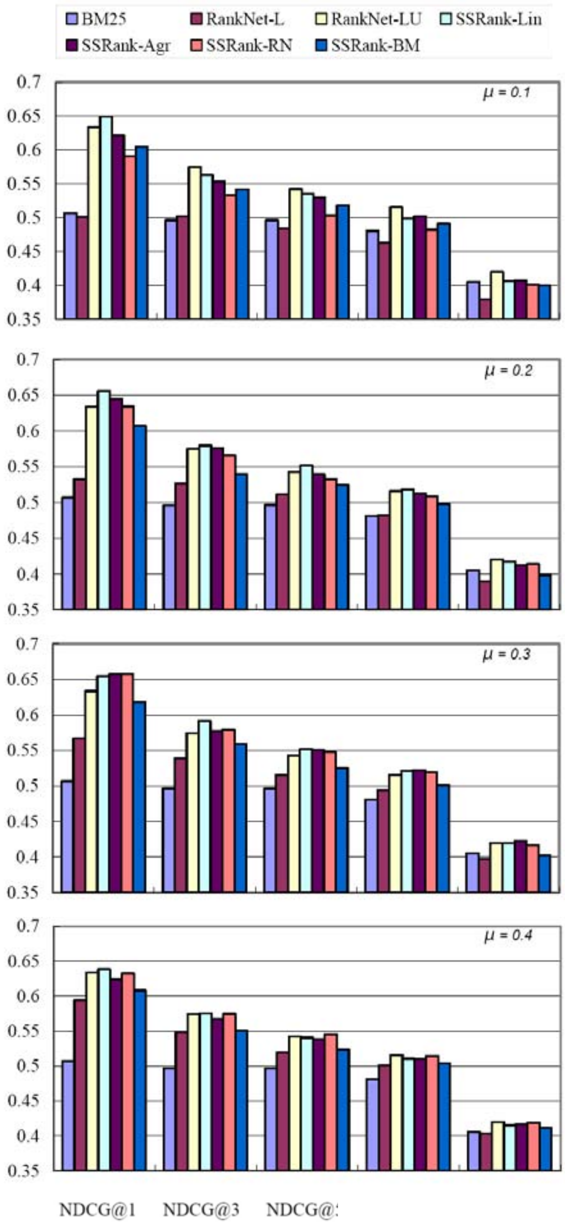

Fig. 3. Performances of methods on four labeling rates averaged over three data sets

cantly better than the two baselines in 7 and 5 settings, respectively, and SSRank-BM is significantly better than the two baselines in 6 and 4 settings, respectively.

The relative improvements of SSRank over RankNet-L and BM25 on the 12 settings are further summarized in Table 4 and Table 5, respectively. The highest numbers are highlighted in boldface. We can see that SSRank-Lin and SSRank-Agr outperform the baseline methods of RankNet-L and BM25 *consistently*. Furthermore, SSRank-Lin and SSRank-Agr work better than SSRank-RN and SSRank-BM. Additionally, SSRank-Lin performs slightly better than SSRank-Agr.

We can conclude, therefore, that SSRank can perform better than the baseline methods, and SSRank with two views can perform better than SSRank with one view.

Table 4
Improvements of SSRank over RankNet-L

Measures	SSRank-Lin	SSRank-Agr	SSRank-RN	SSRank-BM
NDCG@1	**18.9%**	16.4%	14.6%	11.3%
NDCG@3	**8.8%**	7.2%	6.1%	3.1%
NDCG@5	**7.1%**	6.2%	4.6%	2.8%
NDCG@10	**5.4%**	**5.4%**	4.0%	2.5%
MAP	5.2%	**5.3%**	4.6%	2.2%

Table 5
Improvement of SSRank over BM25

Measures	SSRank-Lin	SSRank-Agr	SSRank-RN	SSRank-BM
NDCG@1	**28.1%**	25.4%	23.8%	19.9%
NDCG@3	**16.1%**	14.3%	13.4%	9.8%
NDCG@5	**9.6%**	8.6%	7.2%	5.0%
NDCG@10	**6.4%**	**6.4%**	5.0%	3.4%
MAP	2.7%	**2.8%**	2.1%	-0.3%

4.4 Experiment 2: Learning Curve

To investigate how the performance of SSRank improves as the labeling rate increases (10%, 20%, 30% and 40%), we conducted an additional experiment. Fig. 4 to Fig. 6 give the learning curves of SSRank methods and RankNet-L and RankNet-LU, in terms of NDCG@5 and MAP for the experimental data sets. It can be observed from the figures that as the amount of labeled data increases, the performances of all the SSRank methods approach to RankNet-LU. Note that the performance of the methods, either semi-supervised methods or the pure supervised method such as RankNet-L, fluctuate slightly as the amount of labeled data increase. This might due to the fact that the experimental data are real-world data which contains much noise. Anyway, in general, SSRank-Lin and SSRank-Agr perform better than SSRank-RN and SSRank-BM, particularly when the labeling rate is low.

4.5 Experiment 3: Stopping Criterion

We also investigated the effectiveness of the proposed stopping criterion (Proposition 1). Specifically we tested the cases in which we had a fixed number of iterations T in data labeling, $T = 10$. (Recall that in SSRank data labeling is performed until the stopping criterion is satisfied). We refer to the corresponding methods as SSRank^T-Lin and SSRank^T-Agr, respectively.

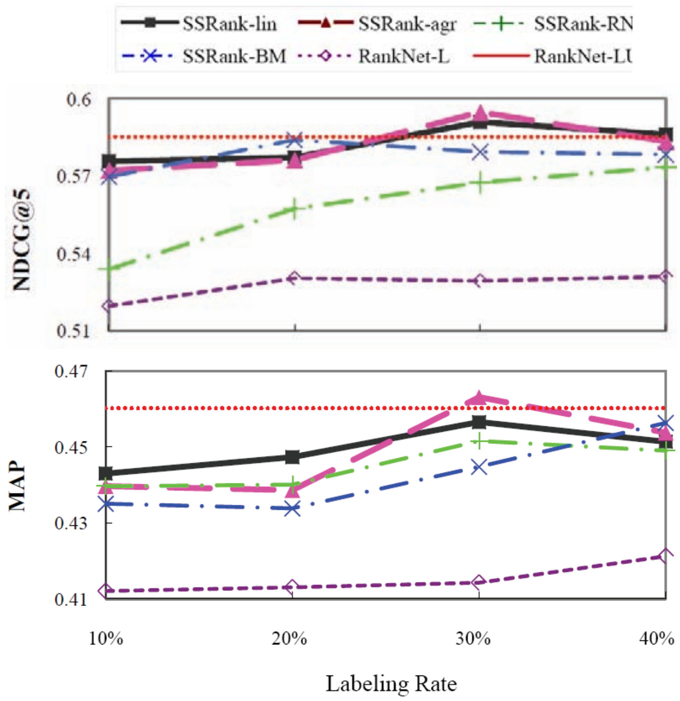

Fig. 4. Learning curves of semi-supervised learning methods on *AP*

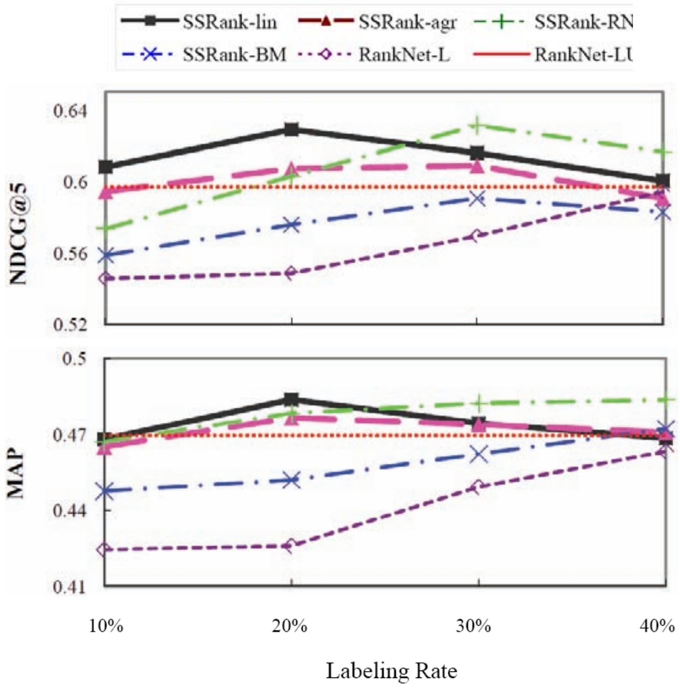

Fig. 5. Learning curves of semi-supervised learning methods on *WSJ*

Experimental results show that SSRank-Lin and SSRank-Agr usually perform better than SSRank^T-Lin and SSRank^T-Agr on all the 12 settings (3 data sets by 4 labeling rates). For example, in terms of MAP, SSRank-Lin outperforms SSRank^T-Lin on 12 settings and SSRank-Agr outperforms

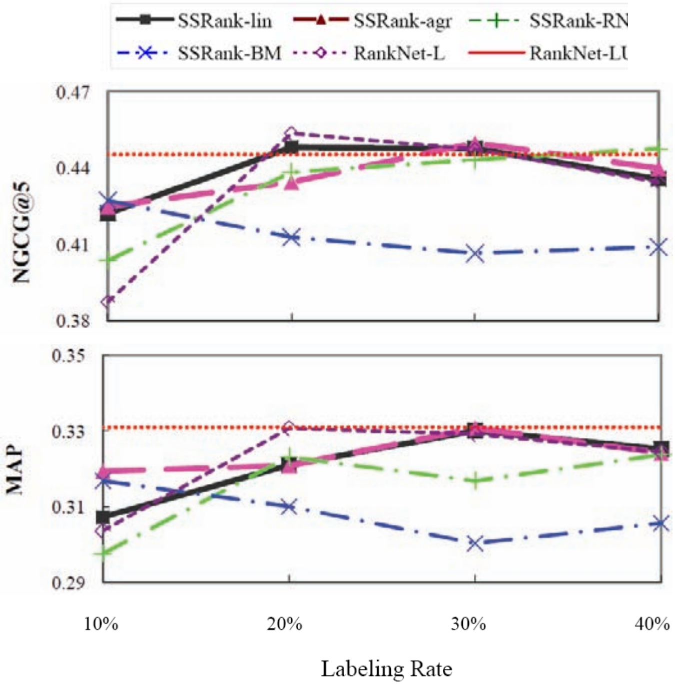

Fig. 6. Learning curves of semi-supervised learning methods on *OHSUMED*

SSRANK^T-Agr on 11 settings. Fig. 7 plots the MAP ratios averaged across different labeling rates of SSRANK-Lin and SSRANK-Agr, respectively, on each experimental data set. Such a ratio is computed by the MAP of the method stopped after a fixed number of iterations over the corresponding method using the stopping criterion proposed in Section 3.3. Thus, a ratio less than 1 means that the method stopping after a fixed number of iterations has lower MAP value that that using our proposed criterion. It is obvious from the figure that both SSRANK^T-Lin and SSRANK^T-Agr perform worse than SSRANK-Lin and SSRANK-Agr, respectively, which suggests the proposed stopping criterion is effective.

Furthermore, since the unlabeled data actually had labels (they were only withheld in the experiments), accuracies on ranking instance pairs as the training iterates may give some insight of the two different stopping criterion employed by SSRANK. For example, Fig. 8 shows the accuracies of SSRANK-Lin and SSRANK^T-Lin during the iterations of data-labeling on *WSJ* under labeling rate of 10% (i.e., starting from 10% of data labeled). We can see from the figure that SSRANK-Lin stops after two iterations when it should and the accuracy keeps on increasing in the training process. In contrast, the accuracy of SSRANK^T-Lin fluctuates. It seems hard to find an optimal point to stop for the fixed number approach. The same tendencies are observed in the other settings. Note that the accuracies of the two methods differ slightly at the second iteration. The reason is that RankNet randomly selects the initial values for training, and thus there is no guarantee that the same model will be obtained in two different trials. Note that Fig. 8 also reveals that the semi-supervised

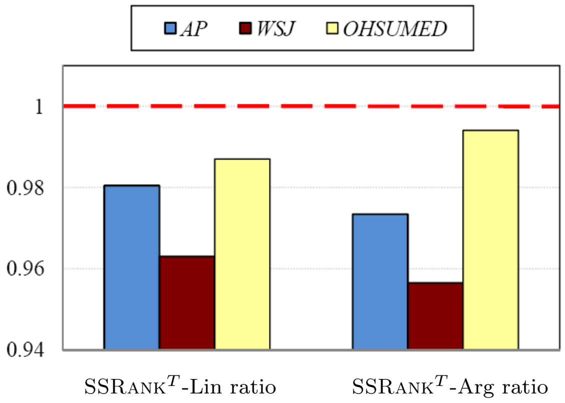

Fig. 7. MAP ratios of two SSRANK methods using different stopping criteria on experimental data sets

process could not perfectly label the unlabeled examples. Thus, using only the semi-supervised process could hardly reach the maximum performance that could be reached by RankNet-LU when all the examples are labeled. This is easy to understand since when all the data are labeled, semi-supervised learning is not needed.

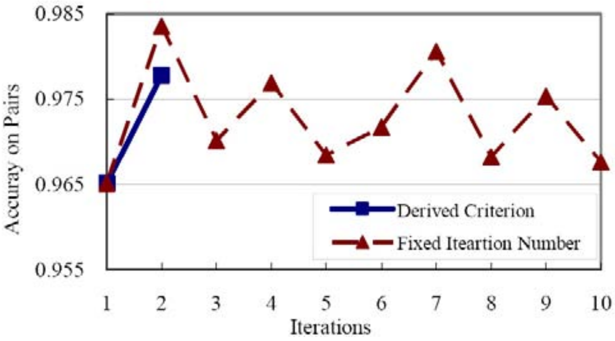

Fig. 8. Accuracies of SSRANK-Lin using different stopping criteria on instance pairs

We note that SSRANK based on a fixed iterations might still work here, but the use of the stopping criterion appears to be better. The superiority of the use of the criterion seems to be more evident on MAP than on NDCG, because the criterion is derived from number of inverse instance pairs and is more closely related to MAP (Joachims, 2002).

4.6 Discussions

The experimental results show that SSRANK outperforms RankNet-L (using the same amount of labeled data). It indicates that SSRANK can indeed effectively leverage the use of unlabeled data to enhance the ranking performance of the supervised learning method. This is because the only extra information used by SSRank is the unlabeled data set, when compared with RankNet-L.

In addition, the performance of BM25 can be improved by using SSRank and a small amount of labeled data. This finding will be valuable for IR, because it points out a new approach to improving the performance of the conventional IR model.

The experimental results also show that SSRank performs better than SSRank-BM and SSRank-RN. It suggests that the uses of two views are better than the uses of one view. For most of time, SSRank outperforms both single view methods, suggesting that SSRank does not simply 'average' the performances of the two views. This finding is in accordance with the theory on semi-supervised learning. That is, if the learner in each view can make predictions with high accuracy, and the two views are not highly correlated, then co-training can work well (Balcan et al., 2005).

For combining strategies in SSRank. Linear combination performs slightly better than agreement (i.e. the special case of majority voting when $V = 2$). One possible explanation is that the weights used in linear combination can provide more information.

The stopping criterion of SSRank, which is derived on the basis of machine learning theory, appears to be effective. Since in semi-supervised learning noise will be inevitably involved, the use of the stopping criterion seems to be better.

4.7 Experiment 4: Application to Web Search

We also applied the proposed method SSRank to a real search system, in which the amount of labeled data was exactly small. The training data was created from 150 real user queries. The instances for each query were constructed. In total, there were 646 features generated for each query document pair. For each query only a small number of instances were manually labeled to represent the degree of relevance, while others were left unlabeled.

SSRank-Lin and SSRank-Agr, as well as SSRank-RN and SSRank-BM, were used to learn ranking functions, and then the models were evaluated with a hold-out test set. The test set consisted of instances generated from 50 queries, and with all the instances being manually labeled. BM25 and RankNet-L were also used as the baselines. Note that RankNet-LU was not created, because not all the training data were actually labeled as in the other experiments.

Fig. 9 shows the results in terms of NDCG and MAP. It can be seen from

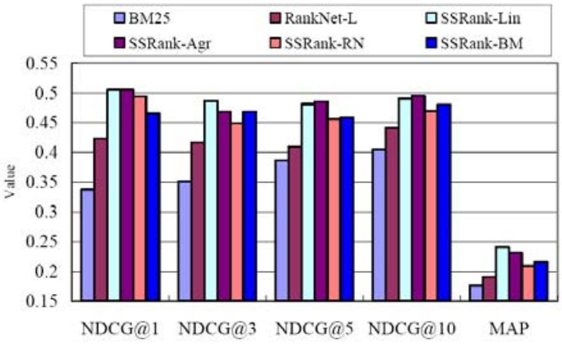

Fig. 9. Performances of methods on Web search

the figure that the four semi-supervised learning methods outperform the two baseline methods. Furthermore, SSRank-Lin and SSRank-Agr performs better than SSRank-BM and SSRank-RN. For example, with the use of SSRank-Lin NDCG@1, NDCG@3, NDCG@5, NDCG@10 and MAP are improved by 19.6%, 16.9%, 17.5%, 11.1% and 26.0%, respectively, when compared with RankNet-L.

5 Conclusion and Future Work

This paper addresses the issue of ranking model construction in document retrieval, particularly when there are only a small amount of labeled data available. The paper proposes a semi-supervised learning method SSRank for performing the task. It leverages the uses of both labeled data and unlabeled data, utilizes views from both conventional IR and supervised learning to conduct data labeling, and relies on a criterion to control the process of data labeling. Several conclusions can be drawn from the experimental results. First, SSRank can work better than the baseline methods of using BM25 or using a supervised learning model with only labeled data. It demonstrates that SSRank can effectively leverage the use of unlabeled data. Second, among the variants of SSRank, the methods of using two views are always better than those using one single view. This agrees with the findings in semi-supervised learning studies. Third, the stopping criterion used in SSRank is indeed effective to control the quality of data labeling.

In this paper, a stopping criterion for the semi-supervised learning method has been proposed on the basis of the theoretical results in (Angluin and Laird, 1988). We must note that the bounds used to derive the stopping criterion are still not tight enough, although the criterion seems to work well empirically. Further studies on the issue may be needed. In the paper, we have addressed the cases in which all the training queries have some documents labeled, but did not consider the cases in which some training queries have labeled documents while the others do not. How to extend our method to the cases will also be an interesting research topic. How much initially labeled data is needed in order to get the bootstrapping process roll out is another question which we have not addressed in this paper. This will also be a research topic in the future.

6 Acknowledgement

We want to thank the anonymous reviewers for their helpful comments and suggestions. Part of the research was supported by the National Science Foundation of China (60505013, 60635030, 60721002), Jiangsu Science Foundation (BK2008018), the Foundation for the Author of National Excellent Doctoral Dissertation of China (200343), the Microsoft Professorship Award and the Microsoft Research Asia Internet Services Program.

References

Amini, M.-R., Truong, T.-V., Goutte, C., 2008. A boosting algorithm for learning bipartite ranking functions with partially labeled data. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 99–106.

Angluin, D., Laird, P., 1988. Learning from noisy examples. Machine Learning 2 (4), 343–370.

Attar, R., Fraenkel, A. S., 1977. Local feedback in full-text retrieval systems. Journal of the ACM 24 (3), 397–417.

Baeza-Yates, R., Ribeiro-Neto, B., 1999. Modern Information Retrieval. ACM Press.

Balcan, M.-F., Blum, A., Yang, K., 2005. Co-training and expansion: Towards bridging theory and practice. In: NIPS 17. pp. 89–96.

Belkin, M., Niyogi, P., 2004. Semi-supervised learning on riemannian manifolds. Machine Learning 56 (1-3), 209–239.

Blum, A., Chawla, S., 2001. Learning from labeled and unlabeled data using graph mincuts. In: Proceedings of the 18th International Conference on Machine Learning. pp. 19–26.

Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with co-training. In: Proceedings of the 11th Annual Conference on Computational Learning Theory. pp. 92–100.

Brefeld, U., Gartner, T., Scheffer, T., Wrobel, S., 2006. Efficient co-regularised least squares regression. In: Proceedings of the 23rd International Conference on Machine Learning. pp. 137–144.

Burges, C. J. C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G. N., 2005. Learning to rank using gradient descent. In:

Proceedings of the 22nd International Conference on Machine Learning. pp. 89–96.

Cao, Y., Xu, J., Li, H., Huang, Y., Hon, H.-W., 2006. Adapting ranking SVM to document retrieval. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 186–193.

Cao, Z., Qin, T., Liu, T.-Y., Tsai, M., Li, H., 2007. Learning to rank: From pairwise approach to listwise approach. In: Proceedings of the 24th International Conference on Machine Learning. pp. 129–136.

Chapelle, O., Schölkopf, B., Zien, A. (Eds.), 2006. Semi-Supervised Learning. MIT Press, Cambridge, MA.

Chu, W., Ghahramani, Z., 2005. Extension of gaussion process for ranking: Semi-supervised and active learning. In: Proceedings of the NIPS 2005 Workshop on Learning to Rank. pp. 29–34.

Cummins, R., O'Riordan, C., 2006. Term-weighting in information retrieval using genetic programming: A three stage process. In: Proceedings of 17th European Conference on Artificial Intelligence. pp. 793–794.

de Almeida, H. M., Gonçalves, M. A., Cristo, M., Calado, P., 2007. A combined component approach for finding collection-adapted ranking functions based on genetic programming. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 399–406.

Dempster, A. P., Laird, N. M., Rubin, D. B., 1977. Miximum likelihood from incomplete data via the em algorithm. Journal of Royal Statistical Society 39 (1), 1–38.

Duh, K., Kirchhoff, K., 2008. Learning to rank with partially-labeled data. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 251–258.

Fan, W., Gordon, M. D., Pathak, P., 2004. A generic ranking function discovery framework by genetic programming for information retrieval. Information Processing and Management 40 (4), 587–602.

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

Gao, J., Qi, H., Xia, X., Nie, J.-Y., 2005. Discriminant model for information retrieval. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 290–297.

Goldman, S., Zhou, Y., 2000. Enhancing supervised learning with unlabeled data. In: Proceedings of the 17th International Conference on Machine Learning. pp. 327–334.

Harman, D., 1992. Relevance feedback revised. In: Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1–10.

Herbrich, R., Graepel, T., Obermayer, K., 2000. Large margin rank boundaries for ordinal regression. In: Smola, A. J., Bartlett, P., Schölkopf, B., Schuurmans, D. (Eds.), Advances in Large Margin Classifiers. pp. 115–132.

Hersh, W. R., Buckley, C., Leone, T., Hickam, D. H., 1994. OHSUMED: An interative retrieval evaluation and new large test collection for research. In: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 192–201.

Huang, X., Huang, Y. R., Wen, M., An, A., Liu, Y., Poon, J., 2006. Applying data mining to pseudo-relevance feedback for high performance text retrieval. In: Proceedings of the 6the IEEE International Conference on Data Mining. pp. 295–306.

Jarvelin, K., Kekalainen, J., 2000. IR evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 41–48.

Joachims, T., 2002. Optimizing search engines using clickthrough data. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 133–142.

Lafferty, J., Zhai, C., 2001. Document language models, query models, and risk minimization for information retrieval. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 111–119.

Li, M., Zhou, Z.-H., 2007. Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples. IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans 37, 1088 – 1098.

Li, X., Liu, B., 2003. Learning to classify text using positive and unlabeled data. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence. pp. 587–594.

Liu, B., Dai, Y., Li, X., Lee, W. S., Yu, P. S., 2003. Building text classifiers using positive and unlabeled examples. In: Proceedings of the 3th IEEE International Conference on Data Mining. pp. 179–188.

Miller, D. J., Uyar, H. S., 1997. A mixture of experts classifier with learning based on both labeled and unlabeled data. In: Advances in Neural Information Processing Systems 9. pp. 571–577.

Mitchell, T. (Ed.), 1997. Machine Learning. McGraw-Hill.

Nallapati, R., 2004. Discriminative models for information retrieval. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 64–71.

Nigam, K., McCallum, A. K., Thrun, S., Mitchell, T., 2000. Text classification from labeled and unlabeled documents using em. Machine Learning 39 (2-3), 103–134.

Robertson, S., Hull, D. A., 2000. The TREC-9 filtering track final report. In: Proceedings of the 9th Text Retrieval Conferece. pp. 25–40.

Rocchio, J. J., 1971. Relevance feedback in information retreival. In: The SMART Retrieval System. pp. 313–323.

Sakai, T., Manabe, T., Koyama, M., 2005. Flexible pseudo-relevance feedback via selective sampling. ACM Transactions on Asian Language Information Processing 4 (2), 111–135.

Salton, G., Buckley, C., 1990. Improving retrieval performance by retrieval feedback. Journal of the American society for Information Science 41 (4), 288–297.

Shahshahani, B., Landgrebe, D., 1994. The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon. IEEE Transactions on Geoscience and remote sensing 32 (5), 1087–1095.

Shen, X., Zhai, C., 2005. Active feedback in ad hoc information retrieval. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 59–66.

Tao, T., Zhai, C., 2006. Regularized estiamtion of mixture models for robust pseduo-relevance feedback. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 162–169.

Trotman, A., 2005. Learning to rank. Information Retrieval 8 (3), 359–381.

Usunier, N., Truong, V., Amini, M. R., Gallinari, P., 2005. Ranking with unlabeled data: A first study. In: Proceedings of the NIPS 2005 Workshop on Learning to Rank. pp. 24–28.

Xu, J., Croft, W. B., 1996. Query expansion using local and global document analysis. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 4–11.

Xu, J., Li, H., 2007. AdaRank: A boosting algorithm for information retrieval,. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 391–398.

Yu, S., Krishnapuram, B., Rosales, R., Steck, H., Rao, R. B., 2007. Bayesian co-training. In: Advances in Neural Information Processing Systems 20. Cambridge, MA: MIT Press.

Yue, Y., Finley, T., Radlinski, F., Joachims, T., 2007. A support vector method for optimizing average precision. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 271–278.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., Schöolkopf, B., 2003. Learning with local and global consistency. In: Advances in Neural Information Processing Systems 17. Cambridge, MA: MIT Press, pp. 1633–1640.

Zhou, Z.-H., Chen, K.-J., Dai, H.-B., 2006. Enhancing relevance feedback in image retrieval using unlabeled data. ACM Trans. Information Systems 24 (2), 219–244.

Zhou, Z.-H., Chen, K.-J., Jiang, Y., 2004. Exploiting unlabeled data in content-based image retrieval. In: Proceedings of the 15th European Conference on Machine Learning. pp. 525–536.

Zhou, Z.-H., Li, M., 2005a. Semi-supervised regression with co-training. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence. pp. 908–913.

Zhou, Z.-H., Li, M., 2005b. Tri-training: Exploiting unlabeled data using three classifier. IEEE Transactions on Knowledge and Data Engineering 17 (11), 1529–1541.

Zhou, Z.-H., Li, M., 2007. Semi-supervised regression with co-training style algorithms. IEEE Transactions on Knowledge and Data Engineering 19, 1479–1493.

Zhou, Z.-H., Zhan, D.-C., Yang, Q., 2007. Semi-supervised learning with very few labeled training examples. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence. pp. 675–680.

Zhu, X., 2005. Semi-supervised learning literature survey. Tech. Rep. 1530, Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI.

Zhu, X., Ghahramani, Z., Lafferty, J., 2003. Semi-supervised learning using gaussian fields and harmonic functions. In: Proceedings of the 20th International Conference on Machine Learning. pp. 912–919.