

# Learning to Rank with Supplementary Data

Wenkui Ding<sup>1</sup> \*, Tao Qin<sup>2</sup>, and Xu-Dong Zhang<sup>1</sup>

<sup>1</sup> Tsinghua University, Beijing, 100084, P.R. China  
dingwenkui@gmail.com, zhangxd@tsinghua.edu.cn

<sup>2</sup> Microsoft Research Asia, No.49, Zhichun Road, Haidian District,  
Beijing 100190, P.R.China  
taoqin@microsoft.com

**Abstract.** This paper is concerned with a new task of ranking, referred to as “supplementary data assisted ranking”, or “supplementary ranking” for short. Different from conventional ranking, in the new task, each query is associated with two sets of objects: the target objects that are to be ranked, and the supplementary objects whose orders are not of our interest. Existing learning to rank approaches (either supervised or semi-supervised) cannot well handle the new task, because they ignore the supplementary data in either training, test, or both. In this paper, we propose a general approach for the task, in which the ranking model consists of two parts. The first part is based on the matching between a target object and the query (which is similar to that in conventional approaches). The second part depends on the relationship between target objects and supplementary objects. The new ranking model is learned by minimizing a certain loss function on the training data. We call this approach “supplementary learning to rank”. As a showcase of the approach, we develop two Boosting-style algorithms. In these algorithms, we leverage the supplementary objects in the definition of weak rankers for the second part of the ranking model, and specify the relationship between target and supplementary objects as pairwise preference. Experimental results on both public and large-scale commercial datasets demonstrate the effectiveness of the proposed algorithms.

**Keywords:** supplementary ranking, learning to rank

## 1 Introduction

Ranking is the central problem for many applications in information retrieval (IR), such as document retrieval, web search, question answering, advertisement, and multimedia retrieval [6]. In the standard setting of ranking, a query is associated with a set of objects to be ranked, and a ranking model is employed to rank these objects in the descending order of their relevance to the query.

In this paper, we study a new task of ranking, called “supplementary data assisted ranking”, or “supplementary ranking” for short. Different from the standard setting of ranking, in the new task, two sets of objects are associated with

---

\* This work was done when the first author was visiting Microsoft Research Asia.

a given query. One set contains the target objects to be ranked, and the other set contains some supplementary objects related to the query, whose orders are however not of our interest. For ease of reference, we call these two sets the target set and the supplementary set respectively.

Many real ranking problems are in nature supplementary ranking problems. For example, in multi-lingual search [3], when a user issues an English query, a set of English documents containing the query terms compose the target set, while documents that are relevant to the query but written in other languages compose the supplementary set. In Web image search, the images compose the target set, while the web pages containing these images compose the supplementary set. As can be seen, in some applications, the target objects and the supplementary objects are of the same type (e.g., web pages), while in some other applications, they can be heterogeneous (e.g., images and web pages).

To solve the supplementary ranking task, one can directly apply existing methods developed for standard ranking. However, due to the differences between the two tasks, such a direct application may not be a good choice. For example, while supervised learning to rank techniques have been shown very effective for standard ranking, they completely ignore the supplementary data. Semi-supervised learning to rank methods leverage supplementary data in training, however, the supplementary data is still ignored in test. It is expected that if one can make good use of the supplementary data in both training and test, he/she can do a better job than existing approaches. This is exactly our proposal, and we refer to it as “supplementary learning to rank”.

In our proposed approach, the ranking score of a target object is determined by two sub ranking models. The first sub model is similar to the ranking model in previous work, which measures the matching between the target object and the query. The second sub model considers the relationship (e.g., similarity, preference, and parent-child relationship) between the target object and those supplementary objects. Then we learn the two sub models by minimizing a certain loss function (which evaluates whether the ranking of the target objects is correct) on the training data. In the test phase, the learned sub models are applied to rank the target objects associated with a new query, based on the information contained in both the target and supplementary sets. This new ranking mechanism has following advantages. First, the information contained in the supplementary set can be leveraged in a more comprehensive manner (i.e., in both training and test) than supervised and semi-supervised learning to rank. Second, from the machine learning point of view, the training and test processes become more consistent with each other, which ensures the learned model to generalize better than in semi-supervised learning to rank.

As a showcase of the proposed approach, we develop two Boosting-style supplementary learning to rank algorithms. These algorithms address two cases of supplementary ranking, one with homogeneous target and supplementary objects, and the other with heterogeneous objects. In both algorithms, we leverage the supplementary objects in the construction of weak rankers for the second sub ranking model, and specify the relationship between target and supplementary

objects as pairwise preference. In this way, the resultant algorithms regard the supplementary data as a reference, and ensure that a target object is ranked high if it is highly relevant to the query by itself, and it is more relevant than many supplementary objects. Experimental results on both public and large-scale commercial datasets show that the proposed algorithms (and thus the “supplementary learning to rank” approach) can make effective use of the supplementary data and outperform previous methods.

The rest of this paper is organized as follows. Section 2 introduces the new task of supplementary ranking and a general approach to tackle the task. In Section 3, we present two Boosting-style algorithms for supplementary ranking. Experimental results are reported in Section 4. Conclusions and future work are given in the last section.

## 2 Supplementary Ranking

### 2.1 Supplementary Data Assisted Ranking

In the standard setting of ranking, for each given query, the task is to rank a set of target objects according to their relevance to the query. While this is a general abstraction for ranking tasks, it ignores the fact that in many real applications, one can straightforwardly collect a set of supplementary objects related to the query, in addition to the target objects (for ease of reference, we call the sets containing the target objects and the supplementary objects the target set and the supplementary set respectively). The ranking of the objects in the supplementary set is usually not of our interest, however, these objects can be used to improve the ranking of the target objects.

Here we list several real applications in which supplementary objects naturally exist. Note that this is just an incomplete list, and one can find many other similar applications.

- *Multi-lingual Search*. In the scenario of multilingual search [9], when a user issues an English query, it is supposed to return a ranked list of English documents containing the query terms. Non-English documents that are also related to the query can be used to improve the ranking of the English documents, usually by means of machine translation and co-ranking. In this case, the English documents compose the target set, and the non-English documents compose the supplementary set. Gao et al. [3] have shown that the ranking accuracy of target English documents can be greatly boosted by considering supplementary non-English documents.
- *Document Re-ranking*. In the scenario of re-ranking [5], we are given a initial ranked list of  $n$  documents, which is produced by a light-weight ranker. Then a complex ranker is applied to re-rank the top  $k$  documents in the list. In such a case, the top  $k$  documents compose the target set, and the other  $n - k$  documents compose the supplementary set.
- *Web Image Retrieval*. Because of the semantic gap between textual queries and visual images, in commercial search engines, the web pages containing

images are usually used to assist the ranking of the images [11]. That is, the images compose the target set and the corresponding web pages compose the supplementary set. Note that in this case, the target set and the supplementary set actually contain heterogeneous objects (i.e., visual objects and textual objects), which is different from the sets in the above examples. Wang et al. [11] have shown that by leveraging the supplementary web pages, the ranking of the target images can be improved.

Given the wide availability of supplementary data as mentioned above, if we can well utilize them to assist the ranking of the target objects, we should be able to achieve better ranking performance in many applications. This is, however, not sufficiently studied in the literature of information retrieval. To emphasize its importance, we formally define the task of “supplementary data assisted ranking” (or “supplementary ranking” for short) in this paper. In the task, it is required that one leverages the supplementary objects to improve the ranking performance of the target objects.

## 2.2 Supplementary Learning to Rank

In this subsection, we discuss how to solve the problem of supplementary ranking.

One naïve choice is to directly apply existing methods developed for standard ranking, e.g., the learning to rank methods which have been widely used in the literature [6]. However, due to the differences between the two tasks, this might not be a good choice. For example, in most existing learning to rank methods (either supervised or semi-supervised), the ranking score of an object is only determined by the matching between the query and the object. If we use such ranking models, the supplementary data can at most be considered in the training process (e.g., to refine the loss functions as in semi-supervised learning to rank). In the test phase, however, no supplementary data can be leveraged. To tackle the problem, one needs to re-define the ranking model to explicitly incorporate the supplementary data. This is exactly our proposal.

**Table 1.** Notations

Symbol	Meaning
$q \in Q$	a query
$X^q = \{x_1^q, x_2^q, \dots\}$	feature vectors of the target objects
$Y^q = \{y_1^q, y_2^q, \dots\}$	relevance labels of the target objects
$Z^q = \{z_1^q, z_2^q, \dots\}$	feature vectors of the supplementary objects

For ease of description, we give some notations in Table 1. If without confusion, we will omit the superscript  $q$  in the following discussions. With these notations, we can describe our proposed new ranking model as follows.

$$f_Z(x) = f_0(x) + f_1(x; Z), \forall x \in X \quad (1)$$

where  $f_Z(x)$  indicates that the ranking score of an object depends not only on itself  $x$  but also on the supplementary set  $Z$ .

As can be seen, the proposed ranking model consists of two sub models. The first sub model  $f_0(x)$  is the same as the ranking model used in traditional learning to rank methods, which is determined by  $x$  itself. We refer to it as “individual sub model”. The second one  $f_1(x; Z)$  is, in contrast, determined by the relationship between  $x$  and the supplementary objects in  $Z$ . We refer to this sub model as “supplementary sub model”. Note that, in some applications like web image retrieval, the target objects and the supplementary objects may be of different types (e.g., one is visual image and the other is textual document). In such a case,  $x_i$  and  $z_i$  may locate in different feature spaces, and we need to carefully design the supplementary sub model to handle the situation (we will make more discussions on this in Section 3).

Suppose both sub models contain unknown parameters. Then one can use a training set to learn the parameters. We call the approach that automatically learns the parameters in the sub models “supplementary learning to rank”. In principle, any loss function can be used for this purpose, such as the hinge loss, the exponential loss, and the cross entropy loss, as long as it can measure whether the target objects in the training set are correctly ranked by the model.

After we learn the sub models  $f_0$  and  $f_1$ , we can apply them to rank the target objects associated with a new query. Note that although the ranking model considers the supplementary objects, we only need to compute the ranking scores for the target objects. The supplementary objects contribute to this process, but it is unnecessary to compute their ranking scores, since this is not of our interest.

As compared with supervised learning to rank, supplementary learning to rank can leverage more information contained in the supplementary data. Since the benefit of using supplementary data has been verified by many previous works [3, 11], supplementary learning to rank can be expected to achieve better ranking performance than supervised learning to rank. As compared with semi-supervised learning to rank, supplementary learning to rank makes the training and test processes more consistent with each other. According to statistical learning theory [10], the trained model can be expected to generalize better on the test set.

### 3 Boosting-based Algorithms

There are two steps to design a supplementary learning to rank algorithm: (1) defining the individual sub model and supplementary sub model, and (2) learning the parameters of the two sub models from the training data.

As mentioned in the previous section, any ranking loss can be used. In this section, we take the exponential loss as example and derive two Boosting-style algorithms for supplementary ranking. The first algorithm, RankBoost-Same, assumes that the target objects and the supplementary objects are of the same type. The second algorithm, RankBoost-Heter, can handle the case where the two sets contain heterogeneous objects.

### 3.1 RankBoost-Same

In this subsection, we consider a simple case, in which the target set and the supplementary set contain the same type of objects. Here we make discussions on the two sub models separately.

First, following the basic idea of Boosting, it is natural to define the individual sub model as a linear combination of a set of weak rankers. Assume there exist a set of weak rankers  $\mathcal{H} = \{h(x)\}$  and each weak ranker is a function of  $x$ . Then the individual sub model  $f_0(x)$  can be defined as follows,

$$f_0(x) = \sum_t \alpha_t h_t(x), \quad h_t(x) \in \mathcal{H}, \quad (2)$$

where  $\alpha_t$  is the combination weight of  $h_t(x)$ .

Second, we discuss the formulation of the supplementary sub model. We derive a new set of weak rankers from  $\mathcal{H}$  using supplementary objects. Specifically, for each weaker ranker  $h \in \mathcal{H}$ , we can define a new weak ranker as below,

$$h'(x; Z) = \frac{1}{|Z|} \sum_{z \in Z} I(h(x) - h(z)), \quad (3)$$

where  $|\cdot|$  is the number of elements in a set, and  $I(\cdot)$  is an indicator function.

From Eqn. (3), one can see that here we specify the relationship between target and supplementary objects as pairwise preference. That is, we use  $h'$  to count how many supplementary objects the target object  $x$  can beat in terms of the weak ranker  $h$ . The more supplementary objects a target object beats, the larger score it will get from  $h'$ . In this way, the supplementary objects actually serve as a reference, and a target object is ranked high if it is more relevant than many supplementary objects.

Note that all the  $h'$  functions compose a new set  $\mathcal{H}'$ . Again, following the idea of Boosting, we can define the supplementary sub model as the linear combination of the weak rankers in  $\mathcal{H}'$ ,

$$f_1(x; Z) = \sum_t \alpha_t h_t(x; Z), \quad h_t(x; Z) \in \mathcal{H}'. \quad (4)$$

After defining the set of the weak rankers, it is natural to use Boosting techniques to train the sub models. In the training process, we follow two common practices in Boosting-style algorithms: (i) we take each feature of a query-document pair as a weak ranker  $h$  in  $\mathcal{H}$ ; (ii) in order to select the best weak ranker, we go through all the weak rankers in the candidate set, compute their losses, and choose the one with the minimum loss. The detailed algorithm is shown in Table 2. We refer to the algorithm as RankBoost-Same.

### 3.2 RankBoost-Heter

In this sub section, we consider another case, in which the target objects and the supplementary objects are of different types. In other words,  $x_i$  and  $z_i$  have different feature representations. Again, we make discussions on the two sub models separately.

**Table 2.** RankBoost-Same algorithm

---



---

**Input:** A set of training queries  $\{(X^q, Y^q, Z^q)\}, q = 1, 2, \dots$ , and a set of weak rankers  $\mathcal{H}$

---

**Training :**

- 1: Generate the set  $\mathcal{H}'$  from  $\mathcal{H}$  by Eq. (3)
- 2: Construct the pool  $P$  of preference pairs for training: for each pair of query  $q$ , if  $y_i^q > y_j^q$ , add the pair  $(x_i^q, x_j^q)$  into  $P$
- 3: Initialize the weight of each pair in  $P$  as  $D_0(x_i^q, x_j^q) = \frac{1}{|P|}$
- 4: Initialize  $f_0 = 0, f_1 = 0$
- 5: **For**  $t = 1, \dots, T$
- 6: Find the best weak ranker  $h$  (together with its weight  $\alpha$ ) from  $\mathcal{H}$  using distribution  $D_t$
- 7: Find the best weak ranker  $h'$  (together with its weight  $\alpha'$ ) from  $\mathcal{H}'$  using distribution  $D_t$
- 8: Compare the loss of  $h$  and  $h'$ . If  $h$  corresponds to the smaller loss, set  $h_t = h, \alpha_t = \alpha$  and  $f_0 = f_0 + \alpha_t h_t$ ; else, set  $h_t = h', \alpha_t = \alpha'$  and  $f_1 = f_1 + \alpha_t h_t$
- 9: Update the weight  $D_{t+1}(x_i^q, x_j^q) = e^{\alpha_t(h_t(x_j^q) - h_t(x_i^q))} D_t(x_i^q, x_j^q)$ , and normalize it so that the sum of all the weights equals 1
- 10: **End For**

---

**Output:**  $f_Z(x) = f_0(x) + f_1(x; Z)$

---



---

First, for the individual sub model, the discussions can be very similar to those for RankBoost-Same. That is, we assume that there exists a set  $\mathcal{H} = \{h(x)\}$  of weak rankers based on the feature representation of the target objects, and define the individual sub model  $f_0$  as Eqn. (2).

Second, for the supplementary sub model, the situation is a little more complex, since the target objects and the supplementary objects do not share the same feature representation. As a result, weak rankers in  $\mathcal{H}$  cannot be applied to the supplementary objects. To tackle the problem, we assume that there is another set of weak rankers  $\mathcal{G} = \{g(z)\}$  defined on the supplementary objects. Then for each weaker ranker  $h \in \mathcal{H}$  and each weaker ranker  $g \in \mathcal{G}$ , we define a new weak ranker  $h' \in \mathcal{H}'$  as follows,

$$h'(x; Z) = \frac{1}{|Z|} \sum_{z \in Z} I(h(x) - g(z)). \quad (5)$$

The underlying assumption in the above definition is that although the features for  $x_i$  and  $z_i$  are heterogeneous, the outputs of the weak rankers based on these features can be comparable with each other since they all measure the relevance of an object to the query.

After defining the new weak rankers, the supplementary sub model can be defined as their linear combination in Eqn. (4).

Then we can use Boosting techniques to train the sub models. We call the corresponding algorithm RankBoost-Heter. The relationship between RankBoost-Heter and RankBoost-Same can be summarized as follows.

- RankBoost-Heter can also be used to deal with supplementary ranking with two homogeneous sets if setting  $\mathcal{G} = \mathcal{H}$ . However, the computational complexity of RankBoost-Heter is significantly higher than that of RankBoost-Same. Actually,  $\mathcal{H}'$  contains much more weak rankers than  $\mathcal{H}$  and  $\mathcal{G}$ . Specifically, we have  $|\mathcal{H}'| = |\mathcal{H}| \times |\mathcal{G}|$ . As a result, RankBoost-Heter is not a good choice for supplementary ranking with homogeneous sets, from the computational point of view.
- For the implementation, RankBoost-Same and RankBoost-Heter can be very similar. The only difference lies in the different ways of constructing the candidate set of new weak rankers. Therefore the algorithm flow of RankBoost-Heter is very similar to that of RankBoost-Same. We omit the details here.

## 4 Experimental Results

### 4.1 Settings

Two datasets were used in our experiments: one is from the LETOR benchmark collection [7, 8], and the other is from a commercial web search engine. We used the MQ2007-semi dataset in LETOR 4.0 in our experiments, because it contains both labeled and unlabeled data. There are about 1700 queries in this dataset. On average, each query is associated with about 40 labeled documents and about 1000 unlabeled documents. There are three levels of relevance labels. To evaluate the proposed algorithms in the real scenario of web search, we also used a dataset obtained from a commercial search engine. We refer to it as the ComSE dataset. There are 6,600 queries in the dataset. On average, each query is associated with about 20 labeled documents and more than 140 unlabeled documents. There are five levels of relevance labels.

The multi-fold cross validation strategy was used on both datasets. All the results reported in this section are the average results over multiple folds. We used NDCG [4] as the evaluation measure in our experiments, which is a widely-used IR measure for multi-level relevance judgments.

### 4.2 Supplementary Ranking on Homogeneous Data

Note that in both MQ2007-semi and ComSE, the labeled and unlabeled documents are of the same type, and represented in the same feature space. Therefore it is straightforward to use them to study the supplementary ranking with homogeneous data. In particular, we regard the labeled documents as the target objects and the unlabeled documents as the supplementary objects.

**Baselines** In addition to RankBoost-Same, we implemented three baselines.

- **RankBoost** RankBoost [2] is a supervised learning to rank algorithm that only uses labeled documents for training.
- **RankBoost-All** In this method, we treat unlabeled documents as irrelevant and train a model using RankBoost with all the documents.

- **RankBoost-Prop** This is the semi-supervised learning to rank method proposed in [1]. In the method, for each labeled training document, the same label is assigned to its nearest unlabeled document. When training with the data, the loss function is defined as the weighted sum of two parts: the loss function on the original labeled documents and the loss function on documents with pseudo labels.

For all these baselines and RankBoost-Same, we set the maximal number of selected weak rankers to 600. All the algorithms have converged within 600 iterations in our experiments. Compared with RankBoost, the computational complexity of RankBoost-Same is increased due to the usage of supplementary data. However, according to [2], the computational complexity of RankBoost-Same is only about twice that of RankBoost since the number of weak rankers is doubled. In our experiments RankBoost-Same took less than two seconds per iteration while RankBoost took one second.

**Results** The experimental results on the MQ2007-semi and ComSE dataset are listed in Table 3. Take the results on the MQ2007-semi for example, we have the following observations.

First, RankBoost-All does not perform well. Its performance is even worse than RankBoost. This indicates that improper use of the supplementary data (i.e., simply treating those supplementary objects as irrelevant) may hurt the ranking performance.

Second, RankBoost-Prop slightly outperforms RankBoost. This is also in accordance with the results reported in [1]. This result indicates that appropriately leveraging the supplementary data in the training process can lead to performance gain. However, the improvement of RankBoost-Prop over RankBoost is not statistically significant.

Third, RankBoost-Same outperforms all the baseline algorithms, including both supervised learning to rank methods and semi-supervised learning to rank algorithms. Furthermore, the improvement of RankBoost-Same over RankBoost is statistically significant in terms of NDCG@3, 5, and 10 (we use \* to indicate statistical significance in the table). This indicates that further considering the supplementary data in the test process can lead to more performance gain. This is in accordance with our discussions throughout the paper.

Similar conclusions can be drawn from the results on the ComSE dataset.

**Table 3.** Results of RankBoost-Same

NDCG	on MQ2007-semi				on ComSE			
	@1	@3	@5	@10	@1	@3	@5	@10
RankBoost	0.403	0.406	0.416	0.444	0.504	0.547	0.573	0.632
RankBoost-All	0.395	0.406	0.412	0.439	0.495	0.527	0.557	0.621
RankBoost-Prop	0.405	0.408	0.417	0.444	0.509	0.545	0.573	0.633
RankBoost-Same	0.414	0.419*	0.426*	0.454*	0.512	0.550*	0.577*	0.635*

### 4.3 Supplementary Ranking on Heterogeneous Data

Note that there is no publicly available heterogeneous dataset that can be used to test our proposed methods. In our experiments, we alternatively simulated such datasets based on MQ2007-semi and ComSE. Specifically, we randomly split the features in each dataset into two subsets  $A$  and  $B$ , each with half of the original features. We used the features in subset  $A$  as the representation of the target objects, and used the features in subset  $B$  as the representation of the supplementary objects. In this way, we can guarantee that the target objects and the supplementary objects locate in different feature spaces. For ease of reference, we call the new datasets generated in this way MQ2007-heter and ComSE-heter.

Note that not all the baselines used in Section 4.2 can still work on MQ2007-heter and ComSE-heter, mainly because they assume that the target objects and the supplementary objects share the same feature representation. For example, RankBoost-Prop needs to compute similarity between a target object and a supplementary object. When the target objects and the supplementary objects do not use the same feature representation, the similarity cannot be calculated. As a result, RankBoost becomes the only meaningful baseline on this task. Again, we set the maximal number of selected weak rankers to 600 for RankBoost and RankBoost-Heter.

Table 4 shows the results of RankBoost and RankBoost-Heter on the two heterogeneous datasets. From the table we can see that RankBoost-Heter performs much better than RankBoost. The improvements are statistically significant in terms of NDCG@1, NDCG@3 and NDCG@10 on the MQ2007-heter dataset, and in terms of all the four measures on the ComSE-heter dataset.

**Table 4.** Results of RankBoost-Heter

NDCG	on MQ2007-heter				on ComSE-heter			
	@1	@3	@5	@10	@1	@3	@5	@10
RankBoost	0.402	0.404	0.416	0.441	0.461	0.512	0.541	0.606
RankBoost-Heter	0.414*	0.413*	0.419	0.449*	0.469*	0.516*	0.546*	0.610*

To sum up, the experimental results show that our proposed two algorithms perform better than several supervised learning to rank and semi-supervised learning to rank methods. This verifies our claim that by using the supplementary data in both training and test, one can do a better job in supplementary ranking.

### 4.4 Discussions

In this section, we conduct some further study on our proposed algorithms, in order to understand how supplementary learning to rank leads to performance gains.

As can be seen from Section 3, the supplementary data is mainly used to derive new weak rankers. Therefore, we hypothesize that the good performance

of our proposed algorithms should come from these new weak rankers. That is, the new weak rankers  $h'$  are on average more effective than the original weak rankers  $h$ . To test this hypothesis, we have conducted the following analysis.

First, we looked at the models learned by the proposed algorithms. Note that there is a weight for each selected weak ranker in the models. The larger the weight is, the more important the corresponding weak ranker is. Here we take the MQ2007-semi and MQ2007-heter datasets for example. Given the models learned by RankBoost-Same and RankBoost-Heter from these datasets, we calculated the absolute sum of the weights (denoted as *Sum Of Weight* for ease of reference) for the original weak rankers and that for the new weak rankers, and list them in Table 5. From the table, we can see that for both RankBoost-Same and RankBoost-Heter, the new weak rankers earn much larger weights than the original weak rankers over all the five folds. This clearly shows that the new weak rankers play a major role in the learned models.

Second, we investigated the goodness of each individual weak ranker. Using the exponential loss of RankBoost, we can compute the ranking loss of the original rankers and new rankers as follows.

$$L(h) = \sum_q \sum_{x_i^q \succ x_j^q} e^{-(h(x_i^q) - h(x_j^q))}, \forall h \in \mathcal{H} \quad (6)$$

$$L(h') = \sum_q \sum_{x_i^q \succ x_j^q} e^{-(h'(x_i^q; Z^q) - h'(x_j^q; Z^q))}, \forall h' \in \mathcal{H}' \quad (7)$$

The smaller the loss is, the better the weak ranker is. For the MQ2007-semi dataset, we found that 12 of the top 20 best rankers belong to  $\mathcal{H}'$ , and the other 8 belong to  $\mathcal{H}$ . For the MQ2007-heter dataset, we found that 14 of the top 20 best rankers belong to  $\mathcal{H}'$  and the other 6 belong to  $\mathcal{H}$ . In other words,  $\mathcal{H}'$  contains more good rankers than  $\mathcal{H}$ . This verifies our hypothesis: on average, the supplementary data can help enhance the effectiveness of a weak ranker.

**Table 5.** Weights of the rankers in the models

		Fold1	Fold2	Fold3	Fold4	Fold5
RankBoost-Same on MQ2007-semi	Sum of Weight of $h$	2.470	2.595	2.604	2.522	2.646
	Sum of Weight of $h'$	3.631	3.398	3.516	3.693	3.657
RankBoost-Heter on MQ2007-heter	Sum of Weight of $h$	1.526	1.732	2.033	1.782	1.721
	Sum of Weight of $h'$	4.871	4.429	4.121	4.292	4.586

However, we would also like to point out that although on average the new weak rankers are more effective, the original weak rankers also play an important role in the models. They occupy a significant part in the top rankers, and their sum of weight is also not negligible. This indicates the necessity of using our proposed two sub models simultaneously.

## 5 Conclusions and Future Work

In this paper, we have proposed a new task of ranking, named supplementary data assisted ranking, which can cover many important applications. We have proposed a general approach to the task, named supplementary learning to rank, and developed two Boosting-style algorithms. Experimental results have shown that by use of supplementary learning to rank, significantly better ranking performance can be achieved, than using supervised and unsupervised learning to rank techniques.

For future work, we plan to investigate the following issues. (1) We will study other ways of using the supplementary data in the ranking model. (2) We will develop supplementary learning to rank algorithms based on support vector machines and neural networks. (3) We will apply the proposed approach to more real applications. (4) We will study the theoretical properties of supplementary learning to rank, e.g., its generalization ability and statistical consistency.

## Acknowledgments

We would like to thank Tie-Yan Liu and Hang Li for their valuable comments and suggestions on this work.

## References

1. Amini, M.R., Truong, T.V., Goutte, C.: A boosting algorithm for learning bipartite ranking functions with partially labeled data. In: SIGIR08, pp. 99–106. ACM (2008)
2. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. *JMLR*, vol. 4, pp. 933–969 (2003)
3. Gao, W., Blitzer, J., Zhou, M., Wong, K.F.: Exploiting bilingual information to improve web search. In: ACL09, pp. 1075–1083. Singapore (2009)
4. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. *ACM TOIS*, vol. 20(4), pp. 422–446 (2002)
5. Kurland, O., Lee, L.: PageRank without hyperlinks: Structural re-ranking using links induced by language models. In: SIGIR05, pp. 306–313. ACM (2005)
6. Liu, T.Y.: Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, vol. 3(3), pp. 225–331 (2009)
7. Liu, T.Y., Xu, J., Qin, T., Xiong, W., Li, H.: Letor: Benchmark dataset for research on learning to rank for information retrieval. In: Proceedings of SIGIR07 Workshop on Learning to Rank for Information Retrieval, pp. 3–10 (2007)
8. Qin, T., Liu, T.Y., Xu, J., Li, H.: LETOR: A benchmark collection for research on learning to rank for information retrieval. *IRJ*, pp. 346–374 (2010)
9. Savoy, J.: Comparative study of monolingual and multilingual search models for use with Asian languages. *ACM TALIP*, vol. 4(2), pp. 163–189 (2005)
10. Vapnik, V.N.: *Statistical Learning Theory*. Wiley, New York (1998)
11. Wang, X.J., Ma, W.Y., Xue, G.R., Li, X.: Multi-model similarity propagation and its application for web image retrieval. In: ACM Multimedia Conference04, pp. 944–951. ACM New York, NY, USA (2004)