

# OrdRank: Learning to Rank with Ordered Multiple Hyperplanes

Heli Sun<sup>1</sup>, Jianbin Huang<sup>2</sup>, Boqin Feng<sup>1</sup>, Tao Li<sup>3</sup>, Yingliang Zhao<sup>1</sup>, Jun Liu<sup>1</sup>

<sup>1</sup>Department of Computer Science & Technology  
Xi'an Jiaotong University  
Xi'an, China, 710049  
helisun.sunny@gmail.com  
{bqfeng, ylzhao, jliu}@mail.xjtu.edu.cn

<sup>2</sup>School of software  
Xidian University  
Xi'an, China, 710071  
jbhuang@xidian.edu.cn

<sup>3</sup>School of Computer Science  
Florida International University  
Miami, FL 33199, USA  
taoli@cs.fiu.edu

**Abstract**—Ranking is a central problem for information retrieval systems, because the performance of an information retrieval system is mainly evaluated by the effectiveness of its ranking results. Learning to rank has received much attention in recent years due to its importance in information retrieval. This paper focuses on learning to rank in document retrieval and presents a ranking model named OrdRank that ranks documents with ordered multiple hyperplanes. Comparison of OrdRank with other state-of-the-art ranking techniques is conducted and several evaluation criteria are employed to evaluate its performance. Experimental results on the OHSUMED dataset show that OrdRank outperforms other methods, both in terms of quality of ranking results and efficiency.

**Keywords**—learning to rank; order; multiple hyperplanes

## I. INTRODUCTION

Ranking is a central problem for information retrieval, because the performance of a search engine is mainly evaluated by the accuracy of its ranking results. Recently, machine learning techniques have been employed to construct ranking model automatically and is known as learning to rank. Learning to rank aims to learn a model which can be used to assign scores to objects and rank the objects on the basis of the scores.

Several methods for learning to rank have been developed and applied to information retrieval, including RSVM [1], RankBoost [2], RankNet [3] and some improved methods such as MHR [4], AdaRank [5] and ListNet [6]. RSVM and MHR are based on Support Vector Machine (SVM), RankBoost and AdaRank are based on boosting, RankNet and ListNet are based on neural nets. Recently, learning to rank functions has been a major issue in the machine learning community [7] and has produced many applications in information retrieval [8, 9, 10].

This paper focuses on learning to rank in document retrieval and presents an alternative approach to RSVM. We propose a new method OrdRank which performs the learning and ranking task based on the order relations between the ranks. It employs ordered multiple hyperplanes as the base decision functions and aggregates the rank lists of hyperplanes for final ranking. Experiments on OHSUMED dataset demonstrate the effectiveness of OrdRank.

The rest of this paper is organized as follows. Section 2 introduces learning to rank for information retrieval and the

RSVM method. Section 3 introduces our ranking model. Experimental results are reported in Section 4 and the conclusion and future work are given in the last section.

## II. ANALYSIS OF RSVM

Assuming that there exists an input space  $X \in R^n$ , where  $n$  denotes the number of features. There exists an output space of ranks represented by label set  $Y = \{r_1, r_2, \dots, r_m\}$ , where  $m$  denotes the number of ranks. Further, assume that there exists a total order between the ranks  $r_1 \succ r_2 \succ \dots \succ r_m$ , where  $\succ$  denotes a preference relationship. In training phase, a set of labeled instances  $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$  is given, where  $x_i \in X$  is the feature vector of instance  $i$ , and  $y_i \in Y$  is the rank label of instance  $i$ .

R. Herbrich et al. [4] proposed formalizing the above training problem as that of learning for classification on pairs of instances. Note that the relation  $x_i \succ x_j$  between instance pairs  $x_i$  and  $x_j$  can be expressed by a new instance  $x_i - x_j$ . If  $x_i$  is ranked ahead of  $x_j$ , we assign a label +1, otherwise -1 to the new instance. In this way, we produce a new training data set. Constructing the SVM model is equivalent to solving the following problem, where  $\|\omega\|^2$  denotes  $\ell_2$  norm measuring the margin of the hyperplane and  $\xi_{ij}$  denotes a slack variable.

$$\begin{aligned} \min_{\omega, \xi_{i,j}} \quad & \frac{1}{2} \|\omega\|^2 + C \sum \xi_{ij} \\ \text{s.t.} \quad & \langle \omega, x_i - x_j \rangle > 1 - \xi_{ij}, \forall x_i \succ x_j, \xi_{ij} \geq 0 \end{aligned} \quad (1)$$

To illustrate the limitation of RSVM, an example showing the distribution of instances for query 43 of OHSUMED dataset is presented in Fig. 1 Principle Component Analysis (PCA) is performed on the data of OHSUMED dataset and the first and second principle components are displayed in the figure. In Fig. 1, red circles denote “irrelevant” (R1) documents, blue diamonds denote “partially relevant” (R2) documents, and green squares denote “definitely relevant” (R3) documents. We can observe that

RSVM exploits a single hyperplane to rank all kinds of the documents and treats the instance pairs from all rank pairs equally. There are more instances in the ranks of R2 and R3, the ranking model of RSVM tends to be close to that of R2-R3, i.e., the hyperplane that separates R2 and R3. As a result, RSVM could not well separate the instances in the ranks of R1 and R2.

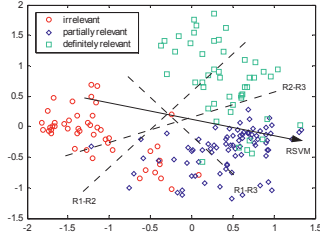


Figure 1. Distribution of instances in OHSUMED dataset (Query 43)

### III. ORDRANK

We propose a novel approach for learning to rank, referred to as OrdRank. It exploits ordered multiple hyperplanes as base decision functions and uses the vote strategy to aggregate the rankings of the base decision functions. OrdRank is a two-stage process. In the first stage it constructs hyperplanes based on the order relationships of the ranks as the base decision functions. The second stage of OrdRank is a decision process performed by the voting of the base decision functions.

#### A. Base decision function

Different from classification, the training instances of ranking are generally belong to certain rank, such as “definitely relevant”, “partially relevant” and “irrelevant”. There exists some order relations between the ranks. Using the order relations to build multiple hyperplanes, both the number of hyperplanes and the number of training instances will be greatly reduced. Firstly, we give the relation of ranks in Definition 1.

- **Definition 1.** Rank  $m$  is adjacent to rank  $n$  iff there is no rank between  $m$  and  $n$ . If rank  $m$  is adjacent to  $n$  and higher than  $n$ , the order relation between  $m$  and  $n$  is denoted as  $m \triangleright n$ .

We build a hyperplane for each rank pair  $m$  and  $n$  when  $m \triangleright n$ . If there are  $k$  ranks, then there will be only  $k-1$  base decision functions for the  $k-1$  rank pairs.

$$\begin{aligned} \min_{\omega_{m,n}, \xi_{i,j,k}} & \frac{1}{2} \|\omega_{m,n}\|^2 + C \sum_{i,j} \xi_{i,j,m,n} \\ \text{s.t.} & \langle \omega_{m,n}, x_i - x_j \rangle \geq 1 - \xi_{i,j,m,n}, \quad \xi_{i,j,m,n} \geq 0 \end{aligned} \quad (2)$$

where  $\omega_{m,n}$  denote that the parameter vector of base decision function for the rank pair  $m$  and  $n$ ,  $x_i$  indicates an instance from the rank  $m$  and  $x_j$  indicates an instance from the rank  $n$ . Fig. 2 shows an example of the OrdRank method, in a two-dimensional space described by the two principal coordinates.

Considering the order relations, OrdRank trains hyperplanes for every adjacent rank pairs: one is between “definitely relevant” and “partially relevant”, the other is between “partially relevant” and “irrelevant”.

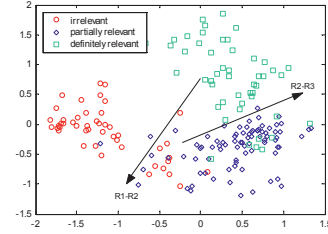


Figure 2. OrdRank for Query 43 of OHSUMED dataset

#### B. Voting

The rank label for any new instance is determined by a voting process. Let  $D$  denote the set of instances to be ranked and  $n$  the number of instances in  $D$ .  $s_q^i$  denotes the score of instance  $i$  given by decision function  $q$ . OrdRank regards the ranking of the base decision function as the vote for instance.

The vote  $v_{ij}^q$  of decision function  $q$  for instance  $i$  and  $j$  is:

$$v_{ij}^q = \begin{cases} 1 & \text{if } s_q^i > s_q^j; \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

Formally, the  $n \times n$  matrix  $V^q$  of decision function  $q$  is as the follows:

$$V^q = \begin{bmatrix} v_{11}^q & v_{12}^q & v_{13}^q & \cdots & v_{1n}^q \\ v_{21}^q & v_{22}^q & v_{23}^q & \cdots & v_{2n}^q \\ v_{31}^q & v_{32}^q & v_{33}^q & \cdots & v_{3n}^q \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ v_{n1}^q & v_{n2}^q & v_{n3}^q & \cdots & v_{nn}^q \end{bmatrix} \quad (4)$$

The sum of  $i$ -th row entries of  $V^q$  can be regarded as the vote of instance  $x_i$  given by decision function  $q$ . The vote  $v(x_i)$  for instance  $x_i$  is the average of  $v_{ij}^q$ . After the vote process, OrdRank sorts the instances according to the value of  $v(x_i)$ .

$$v_i^q = \sum_{j=1}^n v_{ij}^q \quad (5)$$

## IV. EXPERIMENTAL RESULTS

#### A. Evaluation criteria

In information retrieval, ranking results are usually evaluated in terms of performance measures such as Mean

Average Precision (MAP) [11], Normalized Discounted Cumulative Gain (NDCG) [12].

$$P@n = \frac{\text{number of relevant instances in top } n}{n} \quad (6)$$

$P@n$  measures accuracy of top  $n$  results for a query.

Given a query  $q_i$ , average precision  $AvgP_i$  is defined as

$$AvgP_i = \sum_{n=1}^N \frac{P@n \times pos(n)}{\text{number of relevant instances}} \quad (7)$$

where  $n$  is position,  $N$  is number of instances retrieved,  $pos(n)$  is a binary function indicating whether the instance at position  $n$  is positive. MAP is defined as the mean of average precisions over a set of queries.

NDCG is also a measure commonly used in IR, when there are more than two categories in relevance ranking. Given a query  $q_i$ , the NDCG score at position  $n$  in the ranking of documents is defined as

$$NDCG@n = Z_n \sum_{j=1}^n (2^{R(j)} - 1) / \log(1 + j) \quad (8)$$

where  $R(j)$  is the rating of the  $j$ -th document and  $Z_n$  is a normalization constant.  $Z_n$  is chosen to guarantee that a perfect ranking's NDCG score at position  $n$  is 1.

### B. Experiments

In the experiment, we use the OHSUMED collection [13] which was created for information retrieval research. The collection consists of 348,566 records and 106 queries. Each query has a number of associated documents. There are a total of 16,140 query-document pairs upon which relevance judgments are given by experts in advance. Each instance consists of a vector of features, determined by a query and a document. We adopted both 'low-level' and 'high-level' features used in document retrieval, which are summarized in Table 1. In order to conduct five-fold cross validation, we partitioned the OHSUMED collection into five parts. For each fold, we use three parts for training, one part for validation, and the remaining part for testing. All results reported in this section are obtained by averaging five trials.

In the experiments, we used LibSVM [14]. The experiments were conducted on a PC with 3.0 GHz CPU and 1G memory. Table 2 shows the  $P@n$  of RSVM, MHR and OrdRank respectively.

Fig. 3 and Fig. 4 show the performance comparison of RSVM, MHR and OrdRank on the OHSUMED dataset in terms of MAP and NDCG respectively. For MAP, OrdRank outperforms MHR with more than 1%, and outperforms RSVM with almost 6%. For most NDCG values, OrdRank outperforms MHR with more than 1% relative improvement.

For NDCG@4, the relative improvement is as large as 2%. These results show that the order relations can help improve the ranking performance.

TABLE I. LOW-LEVEL AND HIGH-LEVEL FEATURES FOR OHSUMED DATASET

Feature	Description
L1	$\sum_{q_i \in q \cap d} c(q_i, d)$
L2	$\sum_{q_i \in q \cap d} \log(c(q_i, d) + 1)$
L3	$\sum_{q_i \in q \cap d} \frac{c(q_i, d)}{ d }$
L4	$\sum_{q_i \in q \cap d} \log\left(\frac{c(q_i, d)}{ d } + 1\right)$
L5	$\sum_{q_i \in q \cap d} \log\left(\frac{ C }{df(q_i)}\right)$
L6	$\sum_{q_i \in q \cap d} c(q_i, C) \log\left(\frac{ C }{df(q_i)}\right)$
L7	$\sum_{q_i \in q \cap d} \log\left(\log\left(\frac{ C }{df(q_i)}\right)\right)$
L8	$\sum_{q_i \in q \cap d} \log\left(\frac{ C }{c(q_i, C)} + 1\right)$
L9	$\sum_{q_i \in q \cap d} \log\left(\frac{c(q_i, d)}{ d } \frac{ C }{c(q_i, C)} + 1\right)$
L10	$\sum_{q_i \in q \cap d} \log\left(\frac{c(q_i, d)}{ d } \log\left(\frac{ C }{df(q_i)}\right) + 1\right)$
H1	BM 25
H2	$\log(BM 25)$
H3	LMIR with DIR smoothing
H4	LMIR with JM smoothing
H5	LMIR with ABS smoothing

TABLE II. THE  $P@N$  VALUE OF RSVM, MHR AND ORDANK FOR OHSUMED DATASET

Algorithm	P@1	P@2	P@3	P@4
RSVM	0.61346	0.57245	0.53645	0.49834
MHR	0.64372	0.60043	0.57359	0.56017
OrdRank	0.65274	0.60462	0.56724	0.57673
Algorithm	P@5	P@6	P@7	P@8
RSVM	0.48745	0.46149	0.45806	0.44573
MHR	0.5387	0.53254	0.53003	0.5224
OrdRank	0.54341	0.54033	0.5321	0.52592

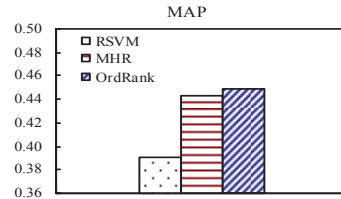


Figure 3. The MAP value of RSVM, MHR and OrdRank on the OHSUMED dataset

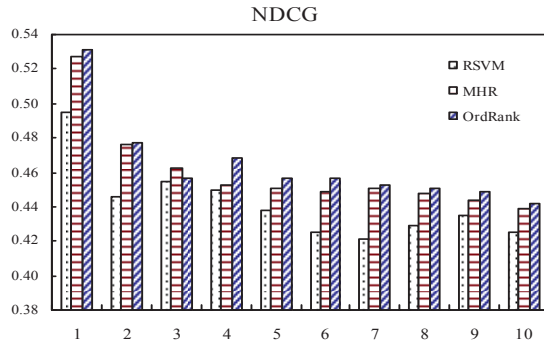


Figure 4. The NDCG value of RSVM, MHR and OrdRank on the OHSUMED dataset

TABLE III. TRAINING TIME OF THE RSVM, MHR AND ORDRANK

Minutes	RSVM	MHR			
		$\omega_{1,2}$	$\omega_{2,3}$	$\omega_{1,3}$	sum
Trial 1	1728.64	306.16	51.64	228.43	586.23
Trial 2	2013.57	456.41	38.72	283.57	778.7
Trial 3	1845.36	329.31	35.44	260.85	625.60
Trial 4	906.53	157.74	17.23	139.27	314.24
Trial 5	1113.72	202.01	14.63	163.51	380.15
Average	1521.56	290.33	31.53	215.13	536.98
Minutes	OrdRank			sum	
	Hyperplane 1	Hyperplane 2			
Trial 1	342.97	56.08		399.05	
Trial 2	494.22	41.28		535.5	
Trial 3	308.59	34.75		343.34	
Trial 4	158.14	17.39		175.53	
Trial 5	204.29	14.63		218.92	
Average	301.64	32.83		334.47	

As aforementioned, OrdRank uses the order relations between ranks to generate the base decision functions. In this way, less instance pairs are used and the computation time in training is greatly reduced. We also compared the training time between OrdRank, MHR and RSVM. Table 3 shows the results. In each trial, the total training time of OrdRank is only one-fifth of RSVM and two-third of MHR. In summary, the experimental results in this section demonstrate the effectiveness of OrdRank: it has good ranking performance with low training complexity.

## V. CONCLUSIONS

In this paper, we proposed a two-stage ranking method OrdRank for document retrieval. Experimental results show that OrdRank outperforms RSVM and MHR in ranking on the OHSUMED dataset and it is more effective and efficient.

There are several avenues for future research. One possible direction is the base decision function construction.

In this paper, we have only investigated one type of methods. A comprehensive investigation on base decision function construction is a natural next-step. Secondly, voting is another important component of OrdRank. In this paper, we have examined the effectiveness of using a simple voting method. There are many other methods proposed, such as Bayesian voting, weighted voting etc., and we will further study whether they can be incorporated in our OrdRank method. Finally, the optimization of the loss function also needs further investigations.

## ACKNOWLEDGMENT

This work was supported by the Nation 863 High Technology Program of China (2008AA01Z131) and Natural Science Basic Research Plan in Shaanxi Province of China (No. SJ08-ZT14).

## REFERENCES

- [1] Herbrich, R., Graepel, T. and Obermayer, K. "Large Margin Rank Boundaries for Ordinal Regression". Advances in Large Margin Classifiers, MIT Press, Cambridge, USA, 2000, pp.115-132.
- [2] Freund, Y., Iyer, R., Schapire, R., & Singer, Y., "An Efficient Boosting Algorithm for Combining Preferences". Journal of Machine Learning Research, 2003(4): pp.933-969.
- [3] Burges, C.J.C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N. and Hullender, G. "Learning to Rank using Gradient Descent". In Proceedings of ICML'05, 2005. pp.89-96.
- [4] T. Qin, Liu, T.-Y. Liu, W. Zhang, X.-D. Wang, D.-S. and H. Li. "Ranking with Multiple Hyperplanes". In Proceedings of SIGIR'07, 2007. pp.279-286.
- [5] J. Xu and H. Li. "Adarank: A Boosting Algorithm for Information Retrieval". In Proceedings of SIGIR'07, 2007. pp.391-398.
- [6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai and H. Li. "Learning to Rank: from Pairwise Approach to Listwise Approach". In Proceedings of ICML'07, 2007. pp.129-136.
- [7] F. Xia, T. Liu, J. Wang, W. Zhang, and H. Li. Listwise Approach to Learning to Rank: Theory and Algorithm. In Proceedings of ICML'08, 2008. pp. 1192-1199.
- [8] A. Veloso, H. M. de Almeida, M. A. Gonçalves, and W. M. Jr. "Learning to Rank at Query-Time Using Association Rules". In Proceedings of SIGIR'08, 2008. pp. 267-274.
- [9] X.-B Geng, T.-Y. Liu, T. Qin, A. Arnold, H. Li, and H.-Y. Shum. "Query Dependent Ranking Using K-Nearest Neighbor". In Proceedings of SIGIR'08, 2008. pp.115-122.
- [10] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, W.-Y. Xiong, and H. Li, "Learning to Rank Relational Objects and Its Application to Web Search" In Proceedings of WWW'08, 2008. pp.407-416.
- [11] Baeza-Yates, R., Ribeiro-Neto, B. "Modern Information Retrieval". Addison Wesley, 1999.
- [12] K. Jarvelin and J. Kekalainen. "IR Evaluation Methods for Retrieving Highly Relevant Documents". In Proceedings of SIGIR'00, 2000. pp.41-48.
- [13] Hersh, W. R., Buckley, C., Leone, T. J., and Hickam, D. H. "OHSUMED: An Interactive Retrieval Evaluation and New Large Test Collection for Research". In Proceedings of SIGIR'94, 1994. pp.192-201.
- [14] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM : A Library for Support Vector Machines", 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>