Modeling Parameter Interactions in Ranking SVM

Yaogong Zhang[†§*] Jun Xu[‡] Yanyan Lan[‡] Jiafeng Guo[‡]
Maoqiang Xie[†§] Yalou Huang[†§] Xueqi Cheng[‡]
[†]College of Computer and Control Engineering ,Nankai University
[‡]CAS Key Lab of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences
[§]College of Software, Nankai University
ygzhang@mail.nankai.edu.cn, {junxu, lanyanyan, guojiafeng}@ict.ac.cn,
{xiemq, huangyl}@nankai.edu.cn, cxq@ict.ac.cn

ABSTRACT

Ranking SVM, which formalizes the problem of learning a ranking model as that of learning a binary SVM on preference pairs of documents, is a state-of-the-art ranking model in information retrieval. The dual form solution of Ranking SVM model can be written as a linear combination of the preference pairs, i.e., $\boldsymbol{w} = \sum_{(i,j)} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j)$, where α_{ij} denotes the Lagrange parameters associated with each pair (i, j). It is obvious that there exist significant interactions over the document pairs because two preference pairs could share a same document as their items. Thus it is natural to ask if there also exist interactions over the model parameters α_{ij}, which we may leverage to propose better ranking model. This paper aims to answer the question. Firstly, we found that there exists a low-rank structure over the Ranking SVM model parameters α_{ij}, which indicates that the interactions do exist. Then, based on the discovery, we made a modification on the original Ranking SVM model by explicitly applying a low-rank constraint to the parameters. Specifically, each parameter α_{ij} is decomposed as a product of two low-dimensional vectors, i.e., $\alpha_{ij} = \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle$, where vectors $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ correspond to document i and j, respectively. The learning process, thus, becomes to optimize the modified dual form objective function with respect to the low-dimensional vectors. Experimental results on three LETOR datasets show that our method, referred to as Factorized Ranking SVM, can outperform state-of-the-art baselines including the conventional Ranking SVM.

Categories and Subject Descriptors: H.3.3 [Information Systems Applications]: Information Search and Retrieval – Retrieval Models

Keywords: Parameter interaction; Ranking SVM

*The work was conducted when Yaogong Zhang was visiting CAS Key Lab of Network Data Science and Technology.

CIKM'15, October 19–23, 2015, Melbourne, Australia.
 ISBN 978-1-4503-3794-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2806416.2806595.

1. INTRODUCTION

Learning to rank has been widely used in information retrieval and recommender systems. Among the learning to rank models, Ranking SVM is a representative pairwise ranking model, evolving from the popular support vector machines (SVM) [1] for classification problems. In training, Ranking SVM first constructs the preference pairs of the documents based on their relevance labels (or click-through data [7]). Then, a binary SVM model is learned based on the preference pairs to capture the differences between documents with different relevance labels. In ranking, each document is assigned a relevance score based on the learned ranking model. Usually, the ranking model can be written in its dual form, the solution is a linear combination of the training preference pairs, i.e., $\boldsymbol{w} = \sum_{(i,j)} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j)$, where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the first and second document in the pair (i, j), and α_{ij} is the corresponding Lagrange multiplier.

It is obvious that the constructed preference pairs have significant interactions, since two preference pairs could share a same document as their items. In the dual form solution of Ranking SVM, each preference pair is associated with a Lagrange multiplier α. Therefore, it is natural to ask *whether there also exist interactions among these Lagrange multipliers.* If the answer is yes, how to utilize the interactions to improve Ranking SVM?

This paper tries to answer the above questions by analyzing the Lagrange multipliers of the trained Ranking SVM models. Specifically, we made an arrangement of the Lagrange multipliers and constructed a block diagonal matrix $\mathbf{A}$, where $A(i, j) = \alpha_{ij}$ if α_{ij} appears in the Ranking SVM model and zero otherwise. Then, we performed singular value decomposition (SVD) on each block of $\mathbf{A}$ and sorted the eigenvalues in descending order. We found that for all the queries, we just need 40% dimensions to capture 90% energy, but if we want to capture 100% energy, almost all the queries need at least 80% dimensions, it means there exists a low-rank structure in the matrix, which indicates strong interactions among the Lagrange multipliers.

Based on the discovery, we propose to improve the original Ranking SVM through explicitly modeling the parameter interactions in the training process. Specifically, we apply a low rank constraint over the Lagrange multipliers in the dual form solution of Ranking SVM. Each Lagrange multiplier α_{ij} in the dual form objective function is factorized as the dot product of two K-dimensional latent vectors, i.e., $\alpha_{ij} = \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle$, where $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ correspond to the first and second document in the preference pair, respectively. In this way,

the rank of the matrix $\mathbf{A}$ would not be larger than K. The learning of the ranking model, then, becomes optimizing the factorized dual form objective function with respect to the latent vectors. An effective algorithm based on gradient descent is proposed to conduct the optimization.

The proposed algorithm, referred to as Factorized Ranking SVM, offers several advantages: 1) The interactions among model parameters are explicitly captured when training the ranking model; 2) Compared with original Ranking SVM, the space complexity is dramatically reduced. Experimental results indicate that Factorized Ranking SVM can outperform several state-of-the-art baseline methods including Ranking SVM, on three LETOR benchmark datasets.

2. RANKING SVM

Ranking SVM formalizes the problem of learning a ranking model as the problem of learning a binary classification over preference document pairs. Suppose we are given a set of training label-query-document tuples $\{(y_i, q_i, \boldsymbol{x}_i)\}_{i=1}^{N}$, where N is the number of training tuples, $y_i \in \{r_1, \cdots, r_\ell\}$ is the relevance label for the i-th query-document pair, $q_i \in Q$ is the query, and $\boldsymbol{x}_i \in \mathcal{R}^n$ is the feature vector encoding the i-th query-document pair. There exists a total order between the relevance labels $r_\ell \succ r_{\ell-1}, \cdots, \succ r_1$, where '$\succ$' denotes a preference relationship. The set of preference pairs is defined as $P \equiv \{(i,j)|q_i = q_j, y_i \succ y_j\}$ and the linear Ranking SVM minimizes the loss function

$$\min_{\boldsymbol{w}\in\mathcal{R}^n} \frac{1}{2}||\boldsymbol{w}||^2 + C \sum_{(i,j)\in P} [1 - \langle \boldsymbol{w}, \boldsymbol{x}_i - \boldsymbol{x}_j\rangle]_+, \qquad (1)$$

where $C > 0$ is a parameter and $[x]_+ = \max(0, x)$. It is common to solve the dual problem of (1):

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \frac{1}{2}\boldsymbol{\alpha}^T M \boldsymbol{\alpha} - \boldsymbol{e}^T \boldsymbol{\alpha}, \\ &\text{s.t.} \qquad 0 \le \alpha_{ij} \le C, \forall (i,j) \in P \end{aligned} \qquad (2)$$

where $\boldsymbol{\alpha} \in \mathcal{R}^{|P|}$ is a vector of Lagrange multipliers indexed by pairs in P, $\boldsymbol{e} \in \mathcal{R}^{|P|}$ is the vector of ones, and $M_{(i,j),(u,v)} = \langle \boldsymbol{x}_i - \boldsymbol{x}_j, \boldsymbol{x}_u - \boldsymbol{x}_v\rangle, \forall (i,j),(u,v) \in P$ is a $|P| \times |P|$ matrix.

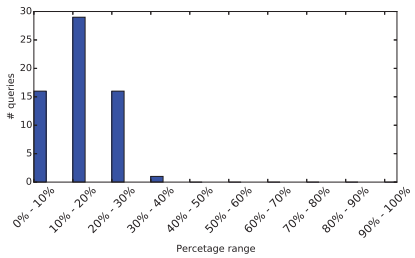
In ranking, the learned ranking model $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x}\rangle$ assigns a relevance score for any test document $\boldsymbol{x}$. From the prime-dual relationship, optimal $\boldsymbol{w}$ and $\boldsymbol{\alpha}$ satisfy

$$\boldsymbol{w} = \sum_{(i,j)\in P} \alpha_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j). \qquad (3)$$

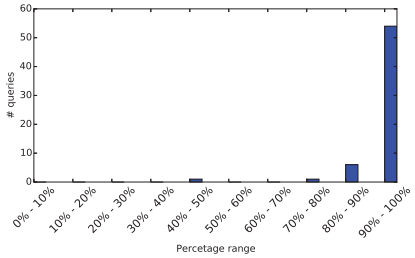
In machine learning, it is assumed that all of the training instances are independently identically distributed. In Ranking SVM, however, it is obvious that the preference pairs have significant interactions because two pairs (e.g.,(i,j) and (i,k))could share a document(e.g.,document i). Since the model parameters $\boldsymbol{\alpha}$ are associated with each preference pair, it is natural to ask whether these parameters also have interactions each other.

3. ANALYSIS OF PARAMETER INTERACTIONS

In this paper, we propose to analyze the interactions between model parameters based on singular value decomposition (SVD). Specifically, given a learned Ranking SVM model, we rearrange the model parameters $\boldsymbol{\alpha}$ as a matrix



(a) Capturing 90% energy



(b) Capturing 100% energy

Figure 1: Distribution of queries over percentage ranges when capturing 90% and 100% of the energy.

$A \in \mathcal{R}^{N\times N}$ and defined as $A(i,j) = \alpha_{ij}$ if $(i,j) \in P$ and 0 otherwise. Note that A is a block diagonal matrix, which consists of sub-matrices $A_i (i = 1, \cdots, |Q|)$, and each block A_i corresponds to a training query. This is because the preference pairs are constructed within the documents retrieved by one query.

We performed SVD on each submatrix A_i (corresponds to a query) and sorted the eigenvalues in descending order[1]. Figure 1(a) shows the statistics on the percentages of dimensions needed per query for capturing 90% of the energy. The statistics are based on a Ranking SVM model trained on one fold of OHSUMED dataset. The queries are grouped into different categories based on the ranges of the percentages. For example, all the queries in the category $0\% \sim 10\%$ should need $0\% \sim 10\%$ of the dimensions for capturing the 90% energy. We can see that all of the queries fall into the ranges less than 40%. Figure 1(b) shows the statistics for capturing 100% of the energy on the same data with the same Ranking SVM model,We can see that if we want to capture 100% of the energy, almost all of the queries fall in the ranges of $80\% \sim 90\%$ and $90\% \sim 100\%$, which indicates that A is a nearly full rank matrix.

We also conducted the analysis on the Ranking SVM models trained on the datasets of MQ2007 and MQ2008. Similar phenomenons were observed. The results indicate there exist significant interactions between the Ranking SVM model parameters. The interactions can be characterized with a low-rank structure.

4. FACTORIZED RANKING SVM

Based on the analysis above, we propose a new learning to rank model for capturing the parameter interactions. The model, referred to as Factorized Ranking SVM, is based on

[1]SVD can be performed on each submatrix independently because A is block diagonal.

original Ranking SVM and explicitly applies a low-rank constraint over the model parameters. Specifically, each parameter α_{ij} is assumed to be a product of two K-dimensional vectors $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$, i.e., $\alpha_{ij} = \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle$, where K is the size of latent dimension.

The ranking model in Equation (3), therefore, becomes

$$\boldsymbol{w} = \sum_{(i,j)\in P} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle (\boldsymbol{x}_i - \boldsymbol{x}_j). \tag{4}$$

Therefore, the loss function (1) can be written as

$$\begin{aligned} &\min_{\boldsymbol{v}_1,\cdots,\boldsymbol{v}_N} \frac{1}{2} \left\| \sum_{(i,j)\in P} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle (\boldsymbol{x}_i - \boldsymbol{x}_j) \right\|^2 + \\ &C \sum_{(k,l)\in P} \left[1 - \left\langle \sum_{(i,j)\in P} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle (\boldsymbol{x}_i - \boldsymbol{x}_j), \boldsymbol{x}_k - \boldsymbol{x}_l \right\rangle \right]_+ . \end{aligned} \tag{5}$$

Since the size of preference pairs is large, to conduct efficient optimization, we adopt a similar method used in [10, 12] in which the prime form of linear Ranking SVM is optimized. Specifically, It can be shown that the loss function in (5) can be written as

$$\begin{aligned} L(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N) = &\frac{1}{2} \|\boldsymbol{w}\|^2 + \\ &C \left(p_{\boldsymbol{w}} - \left\langle \boldsymbol{w}, \sum_{i=1}^{l} \left(l_i^+(\boldsymbol{w}) - l_i^-(\boldsymbol{w}) \right) \boldsymbol{x}_i \right\rangle \right), \end{aligned} \tag{6}$$

where $\boldsymbol{w}$ is defined in Equation (4), $p_{\boldsymbol{w}}, l_i^+(\boldsymbol{w})$, and $l_i^-(\boldsymbol{w})$ are defined as follows:

$$\begin{aligned} &SV(\boldsymbol{w}) \equiv \{(i,j) \in P, 1 - \boldsymbol{w}^T(\boldsymbol{x}_i - \boldsymbol{x}_j) > 0\}, p_{\boldsymbol{w}} = |SV(\boldsymbol{w})|, \\ &SV_i^+(\boldsymbol{w}) \equiv \{j | (j,i) \in SV(\boldsymbol{w})\}, l_i^+(\boldsymbol{w}) = |SV_i^+(\boldsymbol{w})|, \\ &SV_i^-(\boldsymbol{w}) \equiv \{j | (i,j) \in SV(\boldsymbol{w})\}, l_i^-(\boldsymbol{w}) = |SV_i^-(\boldsymbol{w})|. \end{aligned} \tag{7}$$

Please refer to [9] for the derivation details. The gradient of $L(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N)$ in Equation (6) , then, can be written as $\left(\frac{\partial L}{\partial \boldsymbol{v}_1}, \cdots, \frac{\partial L}{\partial \boldsymbol{v}_N} \right)$, where

$$\begin{aligned} \frac{\partial L}{\partial \boldsymbol{v}_k} = &\sum_{j:(k,j)\in P} \boldsymbol{v}_j \sum_{(i',j')\in P} \langle \boldsymbol{v}_{i'}, \boldsymbol{v}_{j'} \rangle \langle \boldsymbol{\phi}_{i'j'}, \boldsymbol{\phi}_{kj} \rangle \\ &+ \sum_{j:(j,k)\in P} \boldsymbol{v}_j \sum_{(i',j')\in P} \langle \boldsymbol{v}_{i'}, \boldsymbol{v}_{j'} \rangle \langle \boldsymbol{\phi}_{i'j'}, \boldsymbol{\phi}_{jk} \rangle \\ &- C \left(\sum_{j:(k,j)\in P)} \boldsymbol{v}_j \cdot \langle \boldsymbol{\phi}_{kj}, \boldsymbol{u} \rangle + \sum_{j:(j,k)\in P} \boldsymbol{v}_j \langle \boldsymbol{\phi}_{jk}, \boldsymbol{u} \rangle \right), \end{aligned} \tag{8}$$

for $k = 1, \cdots, N$. Here $\boldsymbol{u} = \sum_{i=1}^{N} \left(l_i^+(\boldsymbol{w}) - l_i^-(\boldsymbol{w}) \right) \boldsymbol{x}_i$ and $\boldsymbol{\phi}_{ij} = \boldsymbol{x}_i - \boldsymbol{x}_j$. Thus, the updating criteria for gradient descent is $\boldsymbol{v}_k^{(t)} \leftarrow \boldsymbol{v}_k^{(t-1)} - \eta \frac{\partial L}{\partial \boldsymbol{v}_k}$, where t is the iteration number and η is the learning rate. Algorithm 1 shows the pseudo code of the optimization algorithm.

The space complexity of Factorized Ranking SVM is $\mathcal{O}(N \times n + N \times K + N + \frac{N}{|Q|})$, where n is the number of features, $|Q|$ is the number of queries in training data, $N \times n$ is used for storing the original training data, $N \times K$ is used for storing $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N$, N is used for storing $l^+(\boldsymbol{w})$ and $l^-(\boldsymbol{w})$, and $\frac{N}{|Q|}$ is used for dynamically constructing the order-statistic tree [9]. Compared with the original Ranking SVM whose space complexity is $\mathcal{O}(N \times n + N^2)$ [9], Factorized Ranking SVM needs much less memory because $K \ll N$.

Algorithm 1 Factorized Ranking SVM

Input: training data $\{y_i, q_i, \boldsymbol{x}_i\}_{i=1}^N$, learning rate $\eta > 0$, number of hidden dimensions K, and parameter C
Output: model parameters $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N$

```
(v_1, ..., v_N) ← random values
repeat
    w = Σ_{(i,j)∈P} ⟨v_i, v_j⟩(x_i − x_j), where P = {(i,j)|q_i = q_j, y_i ≻ y_j}
    Calculate p_w, l_i^+(w), and l_i^−(w) {Equation (7)}
    for k = 1 to N do
        v_k ← v_k − η ∂L/∂v_k {Equation (8)}
    end for
until convergence
return v_1, ..., v_N
```

Table 1: Statistics on OHSUMED,MQ2007,and MQ2008.

Data Set	#labeled docs	#queries	#preference pairs
OHSUMED	16,140	106	582,588
MQ2007	42,158	1017	246,051
MQ2008	9,360	471	52,325

The time complexity of Factorized Ranking SVM is $\mathcal{O}(T \times (N^2 + N \times (\log K + \log N + n) + \frac{N^2}{|Q|} \times n))$, where T is the number of iterations. N^2 is for computing $\boldsymbol{w}$, $N \times (\log K + \log N + n)$ is for computing $p_w, l^+(\boldsymbol{w}), l^-(\boldsymbol{w}), \boldsymbol{u}$ with the trick of order-statistic tree (see [9] for details). $\frac{N^2}{|Q|} \times n$ is for calculating the gradient $\frac{\partial L}{\partial \boldsymbol{v}_k}$ for $k = 1, \cdots, N$.

5. EXPERIMENT

5.1 Experiment settings

We conducted experiments to test the performances of our approach using three LETOR benchmark datasets[14]: OHSUMED, Million Query track of TREC 2007 (MQ2007), and Million Query track of TREC 2008 (MQ2008). Each dataset consists of queries, corresponding retrieved documents, and human judged labels. The possible relevance labels are relevant, partially relevant, and not relevant. Statistics on the datasets are given in Table 1. The number of preference pairs for each dataset is also shown in the table.

Following the LETOR configuration, We conducted 5-fold cross-validation experiments on the three datasets. The results reported were the average over the five folds. In all of the experiments, LETOR standard features were used.

As for evaluation measures, MAP (mean average precision) and NDCG (normalized discounted cumulative gain)[5] at position of 1, 3, and 5 were used in our experiments. We compared the proposed Factorized Ranking SVM (denoted as "Fac-RSVM" in the experiments) with several state-of-the-art baseline methods, including the conventional Ranking SVM model (denoted as "RSVM"), and two representative learning to ranking models of RankNet [2] and ListNet [4]. As for Ranking SVM, we used the implementation released in [6][2] in all of the experiments. For RankNet and ListNet, we used the implementations in RankLib[3].

Factorized Ranking SVM has some parameters to tune. The learning rate parameter η, number of latent dimensions K, and parameter C were tuned based on the validation set

[2]http://svmlight.joachims.org
[3]http://pepple.cs.umass.edu/∼vdang/ranklib.html

Table 2: Ranking accuracies on dataset OHSUMED.

Method	MAP	NDCG@1	NDCG@3	NDCG@5
RSVM	0.4427	0.5289	0.4553	0.4392
RankNet	0.404	0.4007	0.3616	0.3388
ListNet	0.4443	0.5134	0.4664	0.4530
Fac-RSVM	**0.4463**	**0.5507**	**0.4798**	**0.4546**

Table 3: Ranking accuracies on dataset MQ2007.

Method	MAP	NDCG@1	NDCG@3	NDCG@5
RSVM	0.4442	0.3821	0.3796	0.3890
RankNet	0.4184	0.3527	0.3599	0.3660
ListNet	0.4466	**0.3897**	0.3897	**0.3956**
Fac-RSVM	**0.4483**	0.3824	**0.3900**	0.3940

during each experiment. For all of the baselines, we also tuned their parameters based on the validation set.

5.2 Experimental results

The experimental results on OHSUMED, MQ2007, and MQ2008 are reported in Table 2, Table 3, and Table 4, respectively. Boldface indicates the highest score among all runs. From the results, we can see that Fac-RSVM outperformed RSVM on all of the three datasets in terms of all of the evaluation measures (except on MQ2008 in terms of NDCG@1). The results indicate that Fac-RSVM effectively captures the parameter interactions and thus improved ranking performances. The results also showed that Fac-RSVM can outperform pair-wise and list-wise learning to rank methods of RankNet and ListNet (except for ListNet on MQ2007 in terms of NDCG@1 and NDCG@5), which further proved that Fac-RSVM is effective in relevance ranking.

In the experiments, Fac-RSVM underperformed RSVM in terms of NDCG@1 on MQ2008. MQ2008 has the most sparse labeled documents and preference pairs (see Table 1), which may make the parameter interactions weak. Note that the performances of Fac-RSVM degrade as the number of preference pairs gets smaller. We will analyze the phenomenon in our future work.

6. RELATED WORK

Learning to rank has become one of the most active research topics in IR [11]. State-of-the-art learning to rank models can be categorized into pointwise methods, pairwise methods, and listwise methods. Ranking SVM [7] is a representative pairwise learning to rank method. To make the algorithm more suitable to real-world IR applications, Joachims et al. [7] proposed to train Ranking SVM with users' click-through data from search engines. Cao et al. [3] adapted Ranking SVM to document retrieval by modifying the loss function so that the training can focus on the top ranked documents. Qiang et al. proposed Ranking FM to rank tweets in microblog retrieval [13]. In Ranking FM, the interactions between features are modeled with factorization machine [15]. In this paper, we also propose to improve Ranking SVM by explicitly capturing the parameter interactions in training.

Improving the scalability of Ranking SVM is another important research direction. Kuo et al. [8] proposed an efficient algorithm for optimizing large scale kernel ranking SVM. Lee and Lin [9] proposed to directly optimize the prime form of linear Ranking SVM and significantly improved the scalability. In this paper, we adopted the method proposed in [9] for conducting the optimization.

Table 4: Ranking accuracies on dataset MQ2008.

Method	MAP	NDCG@1	NDCG@3	NDCG@5
RSVM	0.4713	**0.3686**	0.4277	0.4730
RankNet	0.4522	0.3410	0.3991	0.4500
ListNet	0.4415	0.3244	0.3916	0.4396
Fac-RSVM	**0.4714**	0.3660	**0.4289**	**0.4731**

7. CONCLUSIONS

In this paper we investigated the parameter interactions in Ranking SVM. We empirically found that there exists a low-rank structure among the Lagrange multipliers of Ranking SVM model. Based on the discovery, we proposed a new ranking model in which the low-rank constraint is explicitly applied to the Lagrange multipliers, called Factorized Ranking SVM. In training, Factorized Ranking SVM decomposes each Lagrange multiplier as a dot product of two low-dimensional vectors. An efficient algorithm was developed to conduct the optimization. Advantages of factorized Ranking SVM include explicitly modeling parameter interactions in pairwise ranking and dramatically reduced space complexity of the ranking model. Experimental results based on three LETOR benchmark datasets show that Factorized Ranking SVM outperforms the state-of-the-art methods including Ranking SVM, RankNet, and ListNet.

Acknowledgements

The research work was funded by the 973 Program of China under Grants No. 2014CB340401, No. 2012CB316303, the 863 Program of China under Grants No. 2014AA015204, the National Natural Science Foundation of China under Grant No. 61203298, No. 61232010, No. 61425016, No. 61173064, No. 61472401, No. 61300166 and No. 61105049.

8. REFERENCES

[1] Bernhard E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learning Theory*, pages 144–152, 1992.

[2] Chris Burges, Tal Shaked, and et.al. Learning to rank using gradient descent. In *ICML*, pages 89–96, 2005.

[3] Yunbo Cao, Jun Xu, Tie yan Liu, Hang Li, Yalou Huang, and Hsiao wuen Hon. Adapting ranking SVM to document retrieval. In *SIGIR*, pages 186–193, 2006.

[4] Zhe Cao, Tao Qin, Tie yan Liu, Ming feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *ICML*, pages 129–136, 2007.

[5] Kalervo Järvelin and Jaana Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *SIGIR*, pages 41–48, 2000.

[6] Thorsten Joachims. Making large-scale SVM learning practical. In *Advances in Kernel Methods*. MIT Press, 1999.

[7] Thorsten Joachims. Optimizing search engines using clickthrough data. In *KDD*, pages 133–142, 2002.

[8] Tzu-Ming Kuo, Ching-Pei Lee, and Chih-Jen Lin. Large-scale kernel rankSVM. In *SDM*, 2014.

[9] Ching-Pei Lee and Chih-Jen Lin. Large-scale linear rankSVM. *Neural Computation*, 26(4):781–817, 2014.

[10] Kuan-Min Lin and Chih-Jen Lin. A study on reduced support vector machines. *IEEE Transactions on Neural Networks*, 14(6):1449–1559, 2003.

[11] Tie-Yan Liu. Learning to rank for information retrieval. *Springer*, 2011.

[12] O.Chapelle. Learning a support vector machine in the primal. In *Neural Computation*, pages 19(5):1155–1178, 2007.

[13] Runwei Qiang, Feng Liang, and Jianwu Yang. Exploiting Ranking Factorization Machines for Microblog Retrieval. In *CIKM*, pages 1783–1788, 2013.

[14] Tao Qin, Tie-Yan Liu, Jun Xu, and Hang Li. LETOR: A Benchmark Collection for Research on Learning to Rank for Information Retrieval. *Information Retrieval*, 2009.

[15] Steffen Rendle. Factorization machines with libFM. *ACM Trans. Intell. Syst. Technol.*, 3(3):57:1–57:22, May 2012.