SOLAR: Scalable Online Learning Algorithms for Ranking

Jialei Wang[1], **Ji Wan**[2,3], **Yongdong Zhang**[2], **Steven C. H. Hoi**[3*]
[1] Department of Computer Science, The University of Chicago, USA
[2] Key Laboratory of Intelligent Information Processing, ICT, CAS, China
[3] School of Information Systems, Singapore Management University, Singapore
jialei@cs.uchicago.edu, {wanji,zhyd}@ict.ac.cn, chhoi@smu.edu.sg

Abstract

Traditional learning to rank methods learn ranking models from training data in a *batch* and *offline* learning mode, which suffers from some critical limitations, e.g., poor scalability as the model has to be re-trained from scratch whenever new training data arrives. This is clearly non-scalable for many real applications in practice where training data often arrives sequentially and frequently. To overcome the limitations, this paper presents SOLAR — a new framework of Scalable Online Learning Algorithms for Ranking, to tackle the challenge of scalable learning to rank. Specifically, we propose two novel SOLAR algorithms and analyze their IR measure bounds theoretically. We conduct extensive empirical studies by comparing our SOLAR algorithms with conventional learning to rank algorithms on benchmark testbeds, in which promising results validate the efficacy and scalability of the proposed novel SOLAR algorithms.

1 Introduction

Learning to rank [27, 8, 29, 31, 7] aims to learn some ranking model from training data using machine learning methods, which has been actively studied in information retrieval (IR). Specifically, consider a document retrieval task, given a query, a ranking model assigns a relevance score to each document in a collection of documents, and then ranks the documents in decreasing order of relevance scores. The goal of learning to rank is to build a ranking model from training data of a set of queries by optimizing some IR performance measures using machine learning techniques. In literature, various learning to rank techniques have been proposed, ranging from early pointwise approaches [15, 30, 28], to popular pairwise [26, 18, 3], and recent listwise approaches [5, 38]. Learning to rank has many applications, including document retrieval, collaborative filtering, online ad, answer ranking for online QA in NLP [33], etc.

Most existing learning to rank techniques follow batch and offline machine learning methodology, which typically assumes all training data are available prior to the learning task and the ranking model is trained by applying some batch learning method, e.g., neural networks [3] or SVM [4]. Despite being studied extensively, the batch learning to rank methodology has some critical limitations. One of serious limitations perhaps is its poor scalability for real-world web applications, where the ranking model has to be re-trained from scratch whenever new training data arrives. This is apparently inefficient and non-scalable since training data often arrives sequentially and frequently in many real applications [33, 7]. Besides, batch learning to rank methodology also suffers from slow adaption to fast-changing environment of web applications due to the static ranking models pre-trained from historical batch training data.

To overcome the above limitations, this paper investigates SOLAR — a new framework of Scalable Online Learning Algorithms for Ranking, which aims to learn a ranking model from a sequence of training data in an online learning fashion. Specifically, by following the pairwise learning to rank framework, we formally formulate the learning problem, and then present two different SOLAR algorithms to solve the challenging task together with the analysis of their theoretical properties. We conduct an extensive set of experiments by evaluating the performance of the proposed algorithms under different settings by comparing them with both online and batch algorithms on benchmark testbeds in literature.

As a summary, the key contributions of this pa-

[*] The corresponding author. This work was done when the first two authors visited Dr Hoi's group.

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, pages 1692–1701, Beijing, China, July 26-31, 2015. ©2015 Association for Computational Linguistics

per include: (i) we present a new framework of Scalable Online Learning Algorithms for Ranking, which tackles the pairwise learning to ranking problem via a scalable online learning approach; (ii) we present two SOLAR algorithms: a first-order learning algorithm (SOLAR-I) and a second-order learning algorithm (SOLAR-II); (iii) we analyze the theoretical bounds of the proposed algorithms in terms of standard IR performance measures; and (iv) finally we examine the efficacy of the proposed algorithms by an extensive set of empirical studies on benchmark datasets.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 gives problem formulations of the proposed framework and presents our algorithms, followed by theoretical analysis in Section 4. Section 5 presents our experimental results, and Section 6 concludes this work and indicates future directions.

2 Related Work

In general, our work is related to two topics in information retrieval and machine learning: *learning to rank* and *online learning*. Both of them have been extensively studied in literature. Below we briefly review important related work in each area.

2.1 Learning to Rank

Most of the existing approaches to learning to rank can be generally grouped into three major categories: (i) pointwise approaches, (ii) pairwise approaches, and (iii) listwise approaches.

The pointwise approaches treat ranking as a classification or regression problem for predicting the ranking of individual objects. For example, [12, 19] formulated ranking as a regression problem in diverse forms. [30] formulated ranking a binary classification of relevance on document objects, and solved it by discriminative models (e.g., SVM). In [15], Perceptron [32] ranking (known as "Prank") [15] formulated it as online binary classification. [28] cast ranking as multiple classification or multiple ordinal classification tasks.

The pairwise approaches treat the document pairs as training instances and formulate ranking as a classification or regression problem from a collection of pairwise document instances. Example of pairwise learning to rank algorithms include: neural network approaches such as RankNet [3] and LambdaRank [2], SVM approaches such as RankSVM [26], boosting approaches such as RankBoost [18], regression algorithms such as GBRank [43], and probabilistic ranking algorithms such as FRank [35]. The pairwise group is among one of widely and successfully applied approaches. Our work generally belongs to this group.

The listwise approaches treat a list of documents for a query as a training instance and attempt to learn a ranking model by optimizing some loss defined on the predicted list and the ground-truth list. In general, there are two types of approaches. The first is to directly optimize some IR metrics, such as Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG) [25]. Examples include AdaRank by boosting [39], SVM-MAP by optimizing MAP [42], PermuRank [40], and SoftRank [34] based on a smoothed approximation to NDCG, and NDCG-Boost by optimizing NDCG [37], etc. The other is to indirectly optimize the IR metrics by defining some listwise loss function, such as ListNet [5] and ListMLE [38].

Despite being studied actively, most existing works generally belong to batch learning methods, except a few online learning studies. For example, Prank [15] is probably the first online pointwise learning to ranking algorithm. Unlike Prank, our work focuses online pairwise learning to rank technique, which significantly outperforms Prank as observed in our empirical studies. Besides, our work is also related to another existing work in [10], but differs considerably in several aspects: (i) they assume the similarity function is defined in a bi-linear form which is inappropriate for document retrieval applications; (ii) their training data is given in the form of triplet-image instance (p_1, p_2, p_3), while our training data is given in a pairwise query-document instance (q_t, d_t^1, d_t^2); (iii) they only apply first order online learning algorithms, while we explore both first-order and second-order online algorithms. Finally, we note that our work differs from another series of online learning to rank studies [21, 22, 23, 36, 41] which attempt to explore reinforcement learning or multi-arm bandit techniques for learning to rank from implicit/partial feedback, whose formulation and settings are very different.

2.2 Online Learning

Our work is closely related to studies of online learning [24], representing a family of efficient

and scalable machine learning algorithms. In literature, a variety of online algorithms have been proposed, mainly in two major categories: first-order algorithms and second-order algorithms. The notable examples of first-order online learning methods include classical Perceptron [32], and Passive-Aggressive (PA) learning algorithms [13]. Unlike first-order algorithms, second-order online learning [6], e.g., Confidence-Weighted (CW) learning [16], usually assumes the weight vector follows a Gaussian distribution and attempts to update the mean and covariance for each received instance. In addition, Adaptive Regularization of Weights Learning (AROW) [14] was proposed to improve robustness of CW. More other online learning methods can be found in [24]. In this work, we apply both first-order and second-order online learning methods for online learning to rank.

3 SOLAR — Online Learning to Rank

We now present SOLAR — a framework of Scalable Online Learning Algorithms for Ranking, which applies online learning to build ranking models from sequential training instances.

3.1 Problem Formulation

Without loss of generality, consider an online learning to rank problem for document retrieval, where training data instances arrive sequentially. Let us denote by $\mathcal{Q}$ a query space and denote by $\mathcal{D}$ a document space. Each instance received at time step t is represented by a triplet $(q_t^{(i)}, d_t^{(1)}, d_t^{(2)})$, where $q_t^{(i)} \in \mathcal{Q}$ denotes the i-th query in the entire collection of queries $\mathcal{Q}$, $d_t^{(1)} \in \mathcal{D}$ and $d_t^{(2)} \in \mathcal{D}$ denote a pair of documents for prediction of ranking w.r.t. the query $q_t^{(i)}$. Without loss of clarity, for the rest of this paper, we simplify the notation $q_t^{(i)}, d_t^{(1)}, d_t^{(2)}$ as q_t^i, d_t^1, d_t^2, respectively.

We also denote by $y_t \in \{+1, -1\}$ the true ranking order of the pairwise instances at step t such that if $y_t = +1$, document d_t^1 is ranked before d_t^2; otherwise d_t^1 is ranked after d_t^2. We introduce a mapping function $\phi : \mathcal{Q} \times \mathcal{D} \rightarrow \mathbb{R}^n$ that creates a n-dimensional feature vector from a query-document pair. For example, consider $\phi(q, d) \in \mathbb{R}^n$, one way to extract one of the n features is based on term frequency, which counts the number of times the query term of q occurs in document d. We also introduce $\mathbf{w}_t \in \mathbb{R}^n$ as the ranking model to be learned at step t, which is used to form the target ranking function below:

$$f(q_t^i, d_t^1, d_t^2) = \mathbf{w}_t^\top \phi(q_t^i, d_t^1, d_t^2) = \mathbf{w}_t{}^\top (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))$$

Assume that we have a total of Q queries $\{q^{(i)}\}_{i=1}^Q$, each of which is associated with a total of D_i documents and a total of T_i training triplet instances. In a practical document retrieval task, the online learning to rank framework operates in the following procedure:

(i) Given a query q_1, an initial model $\mathbf{w}_1$ is first applied to rank the set of documents for the query, which are then returned to users;
(ii) We then collect user's feedback (e.g., click-through data) as the ground truth labels for the ranking orders of a collection of T_1 triplet training instances;
(iii) We then apply an online learning algorithm to update the ranking model from the sequence of T_1 triplet training instances;
(iv) We repeat the above by applying the updated ranking model to process the next query.

For a sequence of T triplet training instances, the goal of online learning to rank is to optimize the sequence of ranking models $\mathbf{w}_1, \ldots, \mathbf{w}_T$ during the entire online learning process. In general, the proposed online learning to rank scheme is evaluated by measuring the online cumulative MAP [1] or online cumulative NDCG [25]. Let us denote by NDCG_i and MAP_i the NDCG and MAP values for query q_i, respectively, which are defined as follows:

$$\text{NDCG}_i = \frac{1}{N_n} \sum_{r=1}^{D_i} G(l(\pi_f(r)))D(r) \quad (1)$$

$$\text{MAP}_i = \frac{1}{m} \sum_{s:l(\pi_f(s))=1} \frac{\sum_{j \leq s} \mathbb{I}_{\{l(\pi_f(j))=1\}}}{s} \quad (2)$$

where $\mathbb{I}_{\{\cdot\}}$ is an indicator function that outputs 1 when the statement is true and 0 otherwise; $G(K) = 2^K - 1$,$D(K) = \frac{1}{\log_2(1+K)}$, $N_n = \max_\pi \sum_{r=1}^m G(l(\pi(r)))D(r)$, $l(r)$ is the corresponding labels as K-level ratings, π_f denote a rank list produced by ranking function f, m is the number of relevant documents. The online cumulative IR measure is defined as the average of the measure over a sequence of Q queries:

$$\text{NDCG} = \frac{1}{Q} \sum_{i=1}^{Q} \text{NDCG}_i \qquad \text{MAP} = \frac{1}{Q} \sum_{i=1}^{Q} \text{MAP}_i \quad (3)$$

3.2 First-order SOLAR Algorithm

The key challenge of online learning to rank is how to optimize the ranking model $\mathbf{w}_t$ when receiving a training instance (q_t^i, d_t^1, d_t^2) and its true label y_t at each time step t. In the following, we apply the passive-aggressive online learning technique [13] to solve this challenge. First of all, we formulate the problem as an optimization:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|^2 + C\ell(\mathbf{w}; (q_t^i, d_t^1, d_t^2), y_t)^2 \quad (4)$$

where $\ell(\mathbf{w}_t)$ is a hinge loss defined as $\ell(\mathbf{w}_t) = \max(0, 1 - y_t \mathbf{w}_t{}^\top(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)))$, and C is a penalty cost parameter.

The above optimization formulation aims to achieve a trade-off between two concerns: (i) the updated ranking model should not be deviated too much from the previous ranking model $\mathbf{w}_t$; and (ii) the updated ranking model should suffer a small loss on the triplet instance (q_t^i, d_t^1, d_t^2). Their trade-off is essentially controlled by the penalty cost parameter C. Finally, we can derive the following proposition for the solution to the above.

Proposition 1. *This optimization in (4) has the following closed-form solution:*

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \lambda_t y_t(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)) \quad (5)$$

where λ_t is computed as follows:

$$\lambda_t = \frac{\max(0, 1 - \mathbf{w}_t{}^\top y_t(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)))}{\|\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))\|^2 + \frac{1}{2C}} \quad (6)$$

It is not difficult to derive the result in the above proposition by following the similar idea of passive aggressive online learning [13]. We omit the detailed proof here. We can see that if $\mathbf{w}_t{}^\top y_t(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)) \geq 1$, then the model remains unchanged, which means that if the current ranking model can correctly rank the order of d_t^1 and d_t^2 w.r.t. query q_t^i at a large margin, we can keep our model unchanged at this round; otherwise, we will update the current ranking model by the above proposition. Figure 1 gives the framework of the proposed online learning to rank algorithms. We denote by the first-order learning to rank algorithm as "SOLAR-I" for short.

3.3 Second-order SOLAR Algorithm

The previous algorithm only exploits first-order information of the ranking model $\mathbf{w}_t$. Inspired by recent studies in second-order online learning [6, 16, 14], we explore second-order algorithms for online learning to rank.

Algorithm 1: SOLAR — Scalable Online Learning to Rank

```
Initialize w_1 = 0, t = 1
for i = 1, 2, ..., Q do
    receive a query q_i and documents for ranking
    rank the documents by current model w_t
    acquire user's feedback in triplet instances
    for j = 1, ..., T_i do
        update w_{t+1} with (q_t^i, d_t^1, d_t^2) and y_t by
        Eqn. (5) (SOLAR-I) or by Eqn.(8) (SOLAR-II)
        t = t + 1
    end for
end for
```

Figure 1: SOLAR: scalable online learning to rank

Specifically, we cast the online learning to ranking problem into a probabilistic framework, in which we model feature confidence for a linear ranking model $\mathbf{w}$ with a Gaussian distribution with mean $\mathbf{w} \in \mathbb{R}^d$ and covariance $\Sigma \in \mathbb{R}^{d\times d}$. The mean vector $\mathbf{w}$ is used as the model of the ranking function, and the covariance matrix Σ represents our confidence on the model: the smaller the value of $\Sigma_{p,p}$, the more confident the learner has over the p-th feature w_p of the ranking model $\mathbf{w}$.

Following the similar intuition of the above section, we want to optimize our ranking model $\mathcal{N}(\mathbf{w}, \Sigma)$ by achieving the following trade-off: (i) to avoid being deviated too much from the previous model $\mathcal{N}(w_t, \Sigma_t)$; (ii) to ensure that it suffers a small loss on current triplet instance; and (iii) to attain a large confidence on the current instance. Similar to [16], we employ the Kullback-Leibler divergence to measure the distance between the current model $\mathbf{w}$ to be optimized and the previous model $\mathbf{w}_t$, and the regularization terms include both the loss suffered at current triplet instance and the confidence on current triplet instance.

Specifically, we formulate the optimization of second-order online learning to rank as:

$$\begin{aligned}\{\mathbf{w}_{t+1}, \Sigma_{t+1}\} &= \arg\min_{\mathbf{w},\Sigma} D_{KL}(\mathcal{N}(\mathbf{w}, \Sigma)||\mathcal{N}(\mathbf{w}_t, \Sigma_t)) \\ &\quad + \frac{\ell(\mathbf{w})^2 + \Omega(\Sigma)}{2\gamma} \quad (7)\end{aligned}$$

$$\Omega(\Sigma) = (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))^\top \Sigma(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))$$

where γ is the trade-off parameter. The following proposition gives the closed-form solution.

Proposition 2. *This optimization problem in (7) has the following closed-form solution:*

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha_t \Sigma_t y_t(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)) \quad (8)$$

$$\Sigma_{t+1} = \Sigma_t - (1/\beta_t)\Sigma_t A \Sigma_t \quad (9)$$

where A, β_t, and α_t are computed as follows:

$$\begin{aligned} A &= (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))^\top \\ \beta_t &= (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))^\top \Sigma_t(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)) + \gamma \\ \alpha_t &= \max(0, 1 - y_t \mathbf{w}_t{}^\top(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)))/\beta_t \end{aligned}$$

The above can be proved by following [14]. We omit the details. We denote the above algorithm as "SOLAR-II" for short.

4 Theoretical Analysis

In this section, we theoretically analyze the two proposed algorithms by proving some online cumulative IR measure bounds for both of them.

In order to prove the IR measure bounds for the proposed algorithms, we first need to draw the relationships between the cumulative IR measures and the sum of pairwise squared hinge losses. To this purpose, we introduce the following Lemma.

Lemma 4.1. *For one query q_i and its related documents, the NDCG and MAP is lower bounded by its sum of pairwise squared hinge loss suffered by rank model w.*

$$NDCG_i \geq 1 - \gamma_{NDCG} \sum_t \ell^2(w, (q_t^i, d_t^1, d_t^2))$$

$$MAP_i \geq 1 - \gamma_{MAP} \sum_t \ell^2(w, (q_t^i, d_t^1, d_t^2))$$

where γ_{NDCG} and γ_{MAP} are constant specified by the properties of IR measures: $\gamma_{NDCG} = \frac{G(K-1)D(1)}{N_n}$ and $\gamma_{MAP} = \frac{1}{m}$, $G(K) = 2^K - 1, D(K) = \frac{1}{\log_2(1+K)}$, $N_n = \max_\pi \sum_{r=1}^m G(l(\pi(r)))D(r)$, $l(r)$ is the corresponding labels as K-level ratings, π is rank list, m is the number of relevant documents.

Sketch Proof. Using the $essential\ loss$ idea defined in [11], from Theorem 1 of [11] we could see the $essential\ loss$ is an upper bound of measure-based ranking errors; besides, the $essential\ loss$ is the lower bound of the sum of pairwise squared hinge loss, using the properties of squared hinge loss, which is non-negative, non-increasing and satisfy $\ell^2(0) = 1$.

The above lemma indicates that if we could prove bounds for the online cumulative squared hinge loss compared to the best ranking model with all data beforehand, we could obtain the cumulative IR measures bounds. Fortunately there are strong theoretical loss bounds for the proposed online learning to ranking algorithms. The following shows the theorem of such loss bounds for the proposed SOLAR algorithms.

Theorem 1. *For the SOLAR-I algorithm with Q queries, for any rank model u, suppose $R = \max_{i,t} \|\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))\|$, the cumulative squared hinge loss is bounded by*

$$\sum_{i=1}^{Q}\sum_{t=1}^{T_i} \ell_t^2(w_t) \leq (R^2 + \frac{1}{2C})(\|u\|^2 + 2C\sum_{i=1}^{Q}\sum_{t=1}^{T_i} \ell_t^2(u)) \quad (10)$$

The proof for Theorem 1 can be found in Appendix A. By combining the results of Lemma 1 and Theorem 1, we can easily derive the cumulative IR measure bound of the SOLAR-I algorithm.

Theorem 2. *For the SOLAR-I algorithm with Q queries, for any ranking model u, the NDCG and MAP performances are respectively bounded by*

$$NDCG \geq 1 - \frac{\gamma_{NDCG}}{Q}(R^2 + \frac{1}{2C})(\|u\|^2 + 2C\sum_{i}^{Q}\sum_{t=1}^{T_i} \ell_t^2(u))$$

$$MAP \geq 1 - \frac{\gamma_{MAP}}{Q}(R^2 + \frac{1}{2C})(\|u\|^2 + 2C\sum_{i}^{Q}\sum_{t=1}^{T_i} \ell_t^2(u))$$

The analysis of the SOLAR-II algorithm would be much more complex. Let us denote by $\mathcal{M}(M = |\mathcal{M}|)$ the set of example indices for which the algorithm makes a mistake, and by $\mathcal{U}(U = |\mathcal{U}|)$ the set of example indices for which there is an update but not a mistake. Let $X_A = \sum_{(q_t^i, d_t^1, d_t^2) \in \mathcal{M} \cup \mathcal{U}} (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))(\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))^T$. The theorem below give the squared hinge loss bound.

Theorem 3. *For the SOLAR-II algorithm with Q queries, Let $\chi_t = (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))^T \Sigma_t (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))$ of examples in $\mathcal{M} \cup \mathcal{U}$ at time t, K and k is the maximum and minimum value of χ_t, respectively. Σ_T be the final covariance matrix and u_T be the final mean vector. For any ranking model u, the squared hinge loss is bounded by*

$$\sum_{i=1}^{Q}\sum_{t=1}^{T_i} \ell_t^2(w_t) \leq \frac{K+\gamma}{k+\gamma}(a + \sum_{i=1}^{Q}\sum_{t=1}^{T_i} \ell_t(u)) + (K+\gamma)(\log\det(\Sigma_T^{-1}) - \frac{a^2}{\gamma^2 u^T \Sigma_T^{-1} u})$$

where $a = \sqrt{\gamma\|u\|^2 + u^t X_A u}\sqrt{\log(\det(I + \frac{1}{\gamma}X_A)) + U}$

The proof for Theorem 3 can be found in Appendix B. Now, by combining the Lemma 1 and Theorem 3, we can derive the cumulative IR measure bound achieved by the proposed SOLAR-II algorithm.

Theorem 4. *For the SOLAR-II algorithm with Q queries, for any ranking model u, the NDCG and MAP performances are respectively bounded by*

$$NDCG \geq 1 - \frac{\gamma_{NDCG}(K+\gamma)}{Q(k+\gamma)}(a + \sum_{i}^{Q}\sum_{t=1}^{T_i} \ell_t(u)) - \frac{\gamma_{NDCG} b}{Q}$$

$$MAP \geq 1 - \frac{\gamma_{MAP}(K+\gamma)}{Q(k+\gamma)}(a + \sum_{i}^{Q}\sum_{t=1}^{T_i} \ell_t(u)) - \frac{\gamma_{MAP} b}{Q}$$

where $b = (K+\gamma)(\log\det(\Sigma_T^{-1}) - \frac{a^2}{\gamma^2 u^T \Sigma_T^{-1} u})$

The above theorems show that our online algorithm is no much worse than that of the best ranking model u with all data beforehand.

5 Experiments

We conduct extensive experiments to evaluate the efficacy of our algorithms in two major aspects: (i) to examine the learning efficacy of the proposed SOLAR algorithms for online learning to rank tasks; (ii) to directly compare the proposed SOLAR algorithms with the state-of-the-art batch learning to rank algorithms. Besides, we also show an application of our algorithms for transfer learning to rank tasks to demonstrate the importance of capturing changing search intention timely in real web applications. The results are in the supplemental file due to space limitation.

5.1 Experimental Testbed and Metrics

We adopt the popular benchmark testbed for learning to rank: LETOR[1] [31]. To make a comprehensive comparison, we perform experiments on all the available datasets in LETOR3.0 and LETOR4.0. The statistics are shown in Table 1. For performance evaluation metrics, we adopt the standard IR measures, including "MAP", "NDCG@1", "NDCG@5", and "NDCG@10".

Table 1: LETOR datasets used in the experiments.

Dataset	#Queries	#features	avg#Docs/query
OHSUMED	106	45	152.26
MQ2007	1692	46	41.14
MQ2008	784	46	19.40
HP2003	150	64	984.04
HP2004	75	64	992.12
NP2003	75	64	991.04
NP2004	75	64	984.45
TD2003	50	64	981.16
TD2004	50	64	988.61

5.2 Evaluation of Online Rank Performance

This experiment evaluates the online learning performance of the proposed algorithms for online learning to rank tasks by comparing them with the existing "Prank" algorithm [15], a Perceptron-based pointwise online learning to rank algorithm, and a recently proposed "Committee Perceptron (Com-P)" algorithm [17], which explores the ensemble learning for Perceptron. We evaluate the performance in terms of both online cumulative NDCG and MAP measures. As it is an online learning task, the parameter C of SOLAR-I is fixed to 10^{-5} and the parameter γ of SOLAR-II is fixed to 10^4 for all the datasets, as suggested by [17], we set the number of experts in "Com-P" to 20. All experiments were conducted over 10 random permutations of each dataset, and all results were averaged over the 10 runs.

[1] http://research.microsoft.com/en-us/um/beijing/projects/letor/

Table 2 give the results of NDCG on all the datasets, where the best results were bolded. Several observations can be drawn as follows.

First of all, among all the algorithms, we found that both SOLAR-I and SOLAR-II achieve significantly better performance than Prank, which proves the efficacy of the proposed pairwise algorithms. Second, we found that Prank (pointwise) performs extremely poor on several datasets (HP2003, HP2004, NP2003, NP2004, TD2003, TD2004). By looking into the details, we found that it is likely because Prank (pointwise), as a pointwise algorithm, is highly sensitive to the imbalance of training data, and the above datasets are indeed highly imbalanced in which very few documents are labeled as relevant among about 1000 documents per query. By contrast, the pairwise algorithm performs much better. This observation further validates the importance of the proposed pairwise SOLAR algorithms that are insensitive to imbalance issue. Last, by comparing the two SOLAR algorithms, we found SOLAR-II outperforms SOLAR-I in most cases, validating the efficacy of exploiting second-order information.

5.3 Batch v.s. Online Learning

5.3.1 Comparison of ranking performance

This experiment aims to directly compare the proposed algorithms with the state-of-the-art batch algorithms in a standard learning to rank setting. We choose four of the most popular and cutting-edge batch algorithms that cover both pairwise and listwise approaches, including RankSVM [20], AdaRank [39], RankBoost [18], and ListNet [5]. For comparison, we follow the standard setting: each dataset is divided into 3 parts: 60% for training, 20% for validation to select the best parameters, and 20% for testing. We use the training data to learn the ranking model by the proposed SOLAR algorithms, the validation data to select the best parameters, and use the test data to evaluate performance. For SOLAR-I, we choose the best parameter C from $[10^{-3.5}, 10^{-6.5}]$ via grid search on the validation set; and similarly for SOLAR-II, we choose the best parameter γ from $[10^3, 10^6]$. Following [31], we adopt 5 division versions of all the datasets, and report the average performance. The results are shown in Table 3, where the best performances were bolded[2]. Several observations can drawn from the results.

[2] Results of the baseline algorithms are taken from LETOR.

Table 2: Evaluation of NDCG performance of online learning to rank algorithms.

Algorithm	OHSUMED			MQ2007			MQ2008		
	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10
Prank(Pointwise)	0.2689	0.2253	0.2221	0.2439	0.2748	0.3039	0.2369	0.3352	0.4036
Prank(Pairwise)	0.4456	0.3953	0.3904	0.2777	0.3010	0.3294	0.2834	0.3823	0.4403
Com-P	0.4327	0.3993	0.3934	0.3640	0.3828	0.4135	0.3378	0.4415	0.4885
SOLAR-I	0.5060	0.4479	0.4337	0.3760	0.3973	0.4271	0.3490	0.4584	0.5022
SOLAR-II	**0.5352**	**0.4635**	**0.4461**	**0.3897**	**0.4095**	**0.4383**	**0.3594**	**0.4680**	**0.5107**
Algorithm	**HP2003**			**HP2004**			**NP2003**		
	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10
Prank(Pointwise)	0.0033	0.0047	0.0050	0.0053	0.0083	0.0088	0.0033	0.0051	0.0075
Prank(Pairwise)	0.5267	0.6491	0.6745	0.5107	0.6438	0.6717	0.4033	0.5926	0.6255
Com-P	0.6487	0.7744	0.7884	**0.5640**	**0.7163**	0.7392	0.5227	0.7146	0.7417
SOLAR-I	0.6993	0.7796	0.7917	0.5347	0.7072	0.7335	0.5527	0.7486	0.7792
SOLAR-II	**0.7020**	**0.7959**	**0.8079**	0.5413	0.7146	**0.7419**	**0.5693**	**0.7621**	**0.7895**
Algorithm	**NP2004**			**TD2003**			**TD2004**		
	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10
Prank(Pointwise)	0.0080	0.0100	0.0100	0.0040	0.0063	0.0056	0.0040	0.0018	0.0025
Prank(Pairwise)	0.4213	0.6039	0.6290	0.1920	0.1707	0.1737	0.2773	0.2235	0.2071
Com-P	0.4867	0.6989	0.7226	**0.3300**	0.2717	0.2635	**0.3427**	**0.2988**	0.2794
SOLAR-I	0.5613	0.7649	**0.7869**	0.2160	0.2968	0.2916	0.2533	0.2750	0.2625
SOLAR-II	**0.5627**	**0.7667**	0.7858	0.2960	**0.3251**	**0.3245**	0.2893	0.2874	**0.2806**

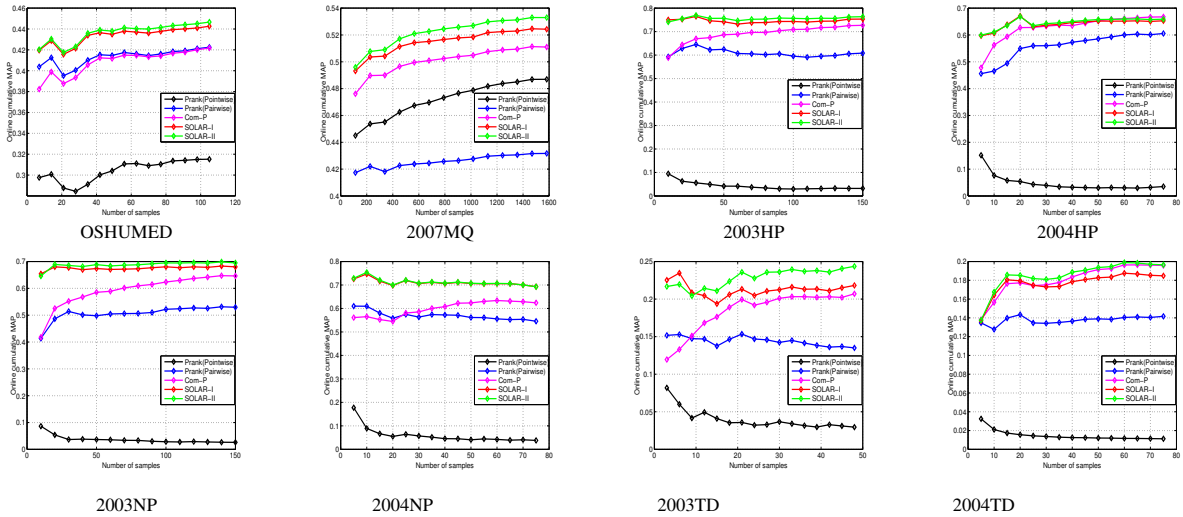

Figure 2: Evaluation of MAP performances of Online Learning to Rank algorithms

First of all, we found that no single algorithm beats all the others on all the datasets. Second, on all the datasets, we found that the SOLAR algorithms are generally achieve comparable to the state-of-the-art batch algorithms. On some datasets, e.g., "MQ2008", "MQ2007" "HP2003", "TD2003", the proposed online algorithms can even achieve best performances in terms of MAP. This encouraging result proves the efficacy of the proposed algorithms as an efficient and scalable online solution to train ranking models. Second, among the two proposed online algorithms, SOLAR-II still outperforms SOLAR-I in most cases, which again shows the importance of exploiting second-order information.

5.3.2 Scalability Evaluation

This experiment aims to examine the scalability of the proposed SOLAR algorithms. We compare it with RankSVM [20], a widely used and efficient batch algorithm. For implementation, we adopt the code from [9] [3], which is known to be the fastest implementation. Figure 3 illustrates the scalability evaluation on "2008MQ" dataset. From the results, we observe that SOLAR is much faster (e.g., 100+ times faster on this dataset)and significantly more scalable than RankSVM.

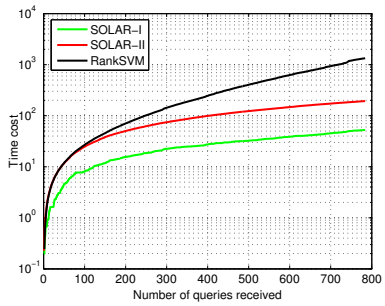

Figure 3: Scalability Evaluation on "2008MQ"

[3] http://olivier.chapelle.cc/primal/

Table 3: Evaluation of NDCG of Online vs Batch Learning to Rank algorithms.

Algorithm	OHSUMED			MQ2007			MQ2008		
	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10
RankSVM	0.4958	0.4164	0.4140	0.4096	0.4142	0.4438	0.3626	0.4695	0.2279
AdaRank-NDCG	0.5330	0.4673	0.4496	0.3876	0.4102	0.4369	0.3826	**0.4821**	0.2307
RankBoost	0.4632	0.4494	0.4302	**0.4134**	**0.4183**	**0.4464**	**0.3856**	0.4666	0.2255
ListNet	0.5326	0.4432	0.4410	0.4002	0.4170	0.4440	0.3754	0.4747	0.2303
SOLAR-I	0.5111	0.4668	**0.4497**	0.3886	0.4101	0.4361	0.3677	0.4634	0.5086
SOLAR-II	**0.5397**	**0.4690**	0.4490	0.4104	0.4149	0.4435	0.3720	0.4771	**0.5171**

Algorithm	HP2003			HP2004			NP2003		
	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10
RankSVM	0.6933	0.7954	0.8077	0.5733	0.7512	0.7687	0.5800	0.7823	0.8003
AdaRank-NDCG	0.7133	0.8006	0.8050	0.5867	**0.7920**	**0.8057**	0.5600	0.7447	0.7672
RankBoost	0.6667	0.8034	0.8171	0.5067	0.7211	0.7428	**0.6000**	0.7818	**0.8068**
ListNet	**0.7200**	**0.8298**	**0.8372**	**0.6000**	0.7694	0.7845	0.5667	**0.7843**	0.8018
SOLAR-I	0.7067	0.8036	0.8056	0.5467	0.7325	0.7544	0.5800	0.7664	0.7935
SOLAR-II	0.7000	0.8068	0.8137	0.5733	0.7394	0.7640	0.5667	0.7691	0.7917

Algorithm	NP2004			TD2003			TD2004		
	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10	NDCG@1	NDCG@5	NDCG@10
RankSVM	0.5067	0.7957	0.8062	0.3200	0.3621	0.3461	0.4133	0.3240	0.3078
AdaRank-NDCG	0.5067	0.7122	0.7384	0.3600	0.2939	0.3036	0.4267	0.3514	0.3163
RankBoost	0.4267	0.6512	0.6914	0.2800	0.3149	0.3122	**0.5067**	**0.3878**	**0.3504**
ListNet	0.5333	**0.7965**	**0.8128**	**0.4000**	0.3393	**0.3484**	0.3600	0.3325	0.3175
SOLAR-I	**0.5733**	0.7814	0.7976	0.2600	0.3060	0.3071	0.3600	0.3119	0.3049
SOLAR-II	**0.5733**	0.7830	0.8013	0.3000	**0.3652**	0.3462	0.3333	0.3167	0.3056

6 Conclusions and Future Work

This paper presented SOLAR — a new framework of Scalable Online Learning Algorithms for Ranking. SOLAR overcomes the limitations of traditional batch learning to rank for real-world online applications. Our empirical results concluded that SOLAR algorithms share competitive efficacy as the state-of-the-art batch algorithms, but enjoy salient properties which are critical to many applications. Our future work include (i) extending our techniques to the framework of listwise learning to rank; (ii) modifying the framework to handle learning to ranking with ties; and (iii) conducting more in-depth analysis and comparisons to other types of online learning to rank algorithms in diverse settings, e.g., partial feedback [41, 22].

Appendix Proof of Theorem 1

Proof. Let $\Delta_t = \|w_t - u\|^2 - \|w_{t+1} - u\|^2$, then

$$\sum_{t=1}^{T} \Delta_t = \|u\|^2 - \|w_{T+1} - u\|^2 \leq \|u\|^2$$

Further, Δ_t can be expressed as:

$$\begin{aligned} \Delta_t &= -2\lambda_t y_t (w_t - u) \cdot (\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)) \\ &\quad -\lambda_t \|\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))\|^2 \\ &\geq \lambda_t (2\ell_t(w_t) - \lambda_t - 2\ell_t(u)). \end{aligned}$$

We thus have

$$\begin{aligned} \|u\|^2 &\geq \sum_{t=1}^{T} (2\lambda_t \ell_t(w_t) - \lambda_t^2 \|\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))\|^2 - 2\lambda_t \ell_t(u)) \\ &\geq \sum_{t=1}^{T} (2\lambda_t \ell_t(w_t) - \lambda_t^2 \|\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))\|^2 - 2\lambda_t \ell_t(u) \\ &\quad -(\frac{\lambda_t}{\sqrt{2C}} - \sqrt{2C}\ell_t(u))^2) \\ &\geq \sum_{t=1}^{T} (2\lambda_t \ell_t(w_t) - \lambda_t^2 (\|\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2)\|^2 + \frac{1}{2C}) - 2C\ell_t(u)^2) \\ &= \sum_{t=1}^{T} (\frac{\ell_t(w_t)^2}{\|\phi(q_t^i, d_t^1) - \phi(q_t^i, d_t^2))\|^2 + \frac{1}{2C}} - 2C\ell_t(u)^2) \end{aligned}$$

Combining the above concludes the theorem. □

Appendix B: Proof of Theorem 3

Proof. Using the Cauchy-Schwarz inequality, we have $u_T^T \Sigma_T^{-1} u_T \geq \frac{(u^T \Sigma_T^{-1} u_T)^2}{u^T \Sigma_T^{-1} u}$. Notice that some inequalities could be easily obtained by extending the Lemma3, Lemma 4 and Theorem 2 of [14] to the pairwise setting as follows:

$$u^T \Sigma_T^{-1} u_T \geq \frac{M + U - \sum_{t \in \mathcal{M} \cup \mathcal{U}} \ell_t(u)}{\gamma},$$

$$\sum_{t \in \mathcal{M} \cup \mathcal{U}} \frac{\chi_t}{r(\chi_t + \gamma)} \leq \log(\det(\Sigma_T^{-1}))$$

$$u_T^T \Sigma_T^{-1} u_T = \sum_{t \in \mathcal{M} \cup \mathcal{U}} \frac{\chi_t}{r(\chi_t + \gamma)} + \sum_{t \in \mathcal{M} \cup \mathcal{U}} \frac{1 - \ell_t^2(w_t)}{\chi_t + \gamma},$$

$$M + U \leq a + \sum_{t \in \mathcal{M} \cup \mathcal{U}} \ell_t(u)$$

where $a = \sqrt{\gamma \|u\|^2 + u^t X_A u} \sqrt{\log(\det(I + \frac{1}{\gamma} X_A)) + U}$.

We thus have

$$\begin{aligned} \sum_{t \in \mathcal{M} \cup \mathcal{U}} \frac{\ell_t^2(w_t)}{\chi_t + \gamma} &\leq \sum_{t \in \mathcal{M} \cup \mathcal{U}} \frac{\chi_t}{r(\chi_t + \gamma)} + \sum_{t \in \mathcal{M} \cup \mathcal{U}} \frac{1}{\chi_t + \gamma} \\ &\quad - \frac{(M + U - \sum_{t \in \mathcal{M} \cup \mathcal{U}} \ell_t(u))^2}{r^2 u^T \Sigma_T^{-1} u} \\ &\leq \log(\det(\Sigma_T^{-1})) + \sum_{t \in \mathcal{M} \cup \mathcal{U}} \frac{1}{\chi_t + \gamma} - \frac{a^2}{r^2 u^T \Sigma_T^{-1} u} \\ &\leq \log(\det(\Sigma_T^{-1})) - \frac{a^2}{r^2 u^T \Sigma_T^{-1} u} + \frac{M + U}{k + \gamma} \\ &\leq \log(\det(\Sigma_T^{-1})) - \frac{a^2}{r^2 u^T \Sigma_T^{-1} u} + \frac{a + \sum_{t \in \mathcal{M} \cup \mathcal{U}} \ell_t(u)}{k + \gamma} \end{aligned}$$

Combining the above, we achieve the final result:

$$\begin{aligned} \sum_{i=1}^{Q} \sum_{t=1}^{T_i} \ell_t^2(w_t) &\leq \frac{K + \gamma}{k + \gamma} (a + \sum_{i=1}^{Q} \sum_{t=1}^{T_i} \ell_t(u)) \\ &\quad + (K + \gamma)(\log \det(\Sigma_T^{-1}) - \frac{a^2}{\gamma^2 u^T \Sigma_T^{-1} u}) \end{aligned}$$

□

Acknowledgments

This work was supported by Singapore MOE tier 1 research grant (C220/MSS14C003) and the National Nature Science Foundation of China (61428207).

References

[1] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval - the concepts and technology behind search, Second edition*. Pearson Education Ltd., Harlow, England, 2011.

[2] C. J. C. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In *NIPS*, pages 193–200, 2006.

[3] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender. Learning to rank using gradient descent. In *ICML*, pages 89–96, 2005.

[4] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking svm to document retrieval. In *SIGIR*, pages 186–193, 2006.

[5] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *ICML*, pages 129–136, 2007.

[6] N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order perceptron algorithm. *SIAM J. Comput.*, 34(3):640–668, 2005.

[7] O. Chapelle and Y. Chang. Yahoo! learning to rank challenge overview. In *Yahoo! Learning to Rank Challenge*, pages 1–24, 2011.

[8] O. Chapelle, Y. Chang, and T.-Y. Liu. Future directions in learning to rank. *Journal of Machine Learning Research - Proceedings Track*, 14:91–100, 2011.

[9] O. Chapelle and S. S. Keerthi. Efficient algorithms for ranking with svms. *Inf. Retr.*, 13(3):201–215, 2010.

[10] G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *J. Mach. Learn. Res.*, 11:1109–1135, Mar. 2010.

[11] W. Chen, T.-Y. Liu, Y. Lan, Z. Ma, and H. Li. Ranking measures and loss functions in learning to rank. In *NIPS*, pages 315–323, 2009.

[12] W. S. Cooper, F. C. Gey, and D. P. Dabney. Probabilistic retrieval based on staged logistic regression. In *SIGIR'98*, pages 198–210. ACM, 1992.

[13] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[14] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. In *NIPS*, pages 414–422, 2009.

[15] K. Crammer and Y. Singer. Pranking with ranking. In *NIPS*, pages 641–647, 2001.

[16] M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. In *ICML*, pages 264–271, 2008.

[17] J. L. Elsas, V. R. Carvalho, and J. G. Carbonell. Fast learning of document ranking functions with the committee perceptron. In *WSDM*, pages 55–64, 2008.

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

[19] F. C. Gey. Inferring probability of relevance using the method of logistic regression. In *In Proceedings of ACM SIGIR'94*, pages 222–231. Springer-Verlag, 1994.

[20] R. Herbrich, T. Graepel, and K. Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, pages 115–132, 2000.

[21] K. Hofmann. Fast and reliable online learning to rank for information retrieval. Phd thesis, University of Amsterdam, Amsterdam, 05/2013 2013.

[22] K. Hofmann, A. Schuth, S. Whiteson, and M. de Rijke. Reusing historical interaction data for faster online learning to rank for ir. In *Proceedings of the sixth ACM international conference on Web search and data mining*, WSDM, pages 183–192, Rome, Italy, 2013.

[23] K. Hofmann, S. Whiteson, and M. Rijke. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Inf. Retr.*, 16(1):63–90, Feb. 2013.

[24] S. C. Hoi, J. Wang, and P. Zhao. Libol: A library for online learning algorithms. *The Journal of Machine Learning Research*, 15(1):495–499, 2014.

[25] K. Järvelin and J. Kekäläinen. Ir evaluation methods for retrieving highly relevant documents. In *SIGIR*, pages 41–48, 2000.

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

[27] H. Li. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 7(3):1–121, 2014.

[28] P. Li, C. J. C. Burges, and Q. Wu. Mcrank: Learning to rank using multiple classification and gradient boosting. In *NIPS*, 2007.

[29] T.-Y. Liu. *Learning to Rank for Information Retrieval*. Springer, 2011.

[30] R. Nallapati. Discriminative models for information retrieval. In *SIGIR'04*, pages 64–71, Sheffield, United Kingdom, 2004.

[31] T. Qin, T.-Y. Liu, J. Xu, and H. Li. Letor: A benchmark collection for research on learning to rank for information retrieval. *Inf. Retr.*, 13(4):346–374, 2010.

[32] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psych. Rev.*, 7:551–585, 1958.

[33] M. Surdeanu, M. Ciaramita, and H. Zaragoza. Learning to rank answers on large online qa collections. In *ACL*, pages 719–727, 2008.

[34] M. Taylor, J. Guiver, S. Robertson, and T. Minka. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the international conference on Web search and web data mining*, WSDM, pages 77–86, Palo Alto, California, USA, 2008. ACM.

[35] M.-F. Tsai, T.-Y. Liu, T. Qin, H.-H. Chen, and W.-Y. Ma. Frank: a ranking method with fidelity loss. In *SIGIR'07*, pages 383–390, Amsterdam, The Netherlands, 2007.

[36] E. Tsivtsivadze, K. Hoffman, and T. Heskes. Large scale co-regularized ranking. In J. Fürnkranz and E. Hüllermeier, editors, *ECAI Workshop on Preference Learning*, 2012.

[37] H. Valizadegan, R. Jin, R. Zhang, and J. Mao. Learning to rank by optimizing ndcg measure. In *NIPS*, pages 1883–1891, 2009.

[38] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li. Listwise approach to learning to rank: theory and algorithm. In *ICML'08*, pages 1192–1199, Helsinki, Finland, 2008.

[39] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *SIGIR*, pages 391–398, 2007.

[40] J. Xu, T.-Y. Liu, M. Lu, H. Li, and W.-Y. Ma. Directly optimizing evaluation measures in learning to rank. In *SIGIR'08*, pages 107–114, Singapore, Singapore, 2008. ACM.

[41] Y. Yue, J. Broder, R. Kleinberg, and T. Joachims. The k-armed dueling bandits problem. *J. Comput. Syst. Sci.*, 78(5):1538–1556, 2012.

[42] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *SIGIR'07*, pages 271–278, Amsterdam, The Netherlands, 2007. ACM.

[43] Z. Zheng, K. Chen, G. Sun, and H. Zha. A regression framework for learning ranking functions using relative relevance judgments. In *SIGIR'07*, pages 287–294, Amsterdam, The Netherlands, 2007.