

A general approximation framework for direct optimization of information retrieval measures

Tao Qin · Tie-Yan Liu · Hang Li

Received: 29 April 2009 / Accepted: 1 December 2009 / Published online: 30 December 2009
© Springer Science+Business Media, LLC 2009

Abstract Recently direct optimization of information retrieval (IR) measures has become a new trend in learning to rank. In this paper, we propose a general framework for direct optimization of IR measures, which enjoys several theoretical advantages. The general framework, which can be used to optimize most IR measures, addresses the task by approximating the IR measures and optimizing the approximated surrogate functions. Theoretical analysis shows that a high approximation accuracy can be achieved by the framework. We take average precision (AP) and normalized discounted cumulated gains (NDCG) as examples to demonstrate how to realize the proposed framework. Experiments on benchmark datasets show that the algorithms deduced from our framework are very effective when compared to existing methods. The empirical results also agree well with the theoretical results obtained in the paper.

Keywords Learning to rank · Direct optimization of IR measures · Position function approximation · Truncation function approximation · Accuracy analysis

1 Introduction

In this paper, we consider the direction optimization of IR measures in learning to rank. This has been regarded as one of the most important directions for the area (Xu et al. 2008).

Several methods that directly optimize IR measures have been developed. In general, they can be grouped into two categories. The methods in the first category introduce upper

T. Qin (✉) · T.-Y. Liu · H. Li
Microsoft Research Asia, Beijing, China
e-mail: taoqin@microsoft.com

T.-Y. Liu
e-mail: tyliu@microsoft.com

H. Li
e-mail: hangli@microsoft.com

bounds of the IR measures and try to optimize the upper bounds as surrogate objective functions (Chapelle et al. 2007; Xu and Li 2007; Yue et al. 2007). The methods in the other category approximate the IR measures using some smooth functions and conduct optimization on the surrogate objective functions (Guiver and Snelson 2008; Taylor et al. 2008).

Previous studies have shown that the approach of directly optimizing IR measures can achieve high performances when compared to the other approaches (Chapelle et al. 2007; Taylor et al. 2008; Xu and Li 2007; Xu et al. 2008; Yue et al. 2007). This is mainly because IR measures are explicitly considered in the direct optimization approach. However, there are still some open problems regarding the approach, as shown below.

First, although there seems to be some relationship between the surrogate functions and the corresponding IR measures, the relationship has not been sufficiently studied. This is a critical issue, because it is necessary to know whether optimizing the surrogate functions can indeed optimize the corresponding IR measures.

Second, some of the proposed surrogate functions are not easy to optimize. Complicated techniques have to be employed for the optimization. For example, both SVM^{map} (Yue et al. 2007) and SVM^{ndcg} (Chapelle et al. 2007) use Structured SVM to optimize the surrogate objective functions. However, the optimization technologies (e.g., the construction of the joint feature map and the way of finding the most violated constraints) are measure-specific, and thus it is not trivial to extend them to new measures.

In this work, we propose a general direct optimization framework, which can effectively address the aforementioned problems. The framework can accurately approximate any position-based IR measure, and then transform the optimization of an IR measure to that of an approximated surrogate function.

The key idea of our proposed framework is as follows. The difficulty in directly optimizing IR measures lies in that the measures are position based, and thus non-continuous and non-differentiable with respect to the score outputted by the ranking function. If we can accurately approximate the positions of documents by a continuous and differentiable function of the scores of the documents, then we will be able to approximate any position based IR measure. Our theoretical analysis demonstrates that highly accurate approximation of a position based IR measure can be obtained and thus high test performance in ranking can be achieved.

Taking average precision (AP) and normalized discounted cumulated gains (NDCG) as examples, we show that it is easy to derive learning algorithms (ApproxAP and ApproxNDCG) to optimize the surrogate functions in the proposed framework. Experimental results show that the derived algorithms can outperform existing algorithms.

The main contributions of this work include two aspects:

1. We set up a general framework for direct optimization, which is applicable to any position based IR measure, theoretically justifiable, and empirically effective;
2. We show that it is easy to derive algorithms to optimize position based IR measures within the framework. Two effective algorithms are proposed as examples to optimize two popular IR measures, AP and NDCG.

The remainder of this paper is as follows. We start with a review on existing methods in Sect. 2. Section 3 sets up a general framework to approximate and optimize IR measures, and shows two examples of using this framework. Theoretical analysis on approximation accuracy is given in Sect. 4. Experimental results are presented in Sect. 5. We conclude the paper and discuss future directions in the last section.

2 Related work

2.1 Learning to rank for information retrieval

The key problem for document retrieval is ranking, specifically, how to create the ranking model (function) that can sort documents based on their relevance to the given query. It is a common practice in IR to tune the parameters of a ranking model using some labeled data and a performance measure. For example, the state-of-the-art methods of BM25 (Robertson and Hull 2000) and LMIR (Language Models for Information Retrieval) (Zhai and Lafferty 2001) all have parameters to tune. As the ranking models become more sophisticated (more features are used) and more labeled data become available, how to tune or train ranking models turns out to be a challenging issue.

The learning to rank technology can successfully leverage multiple features for ranking, and can automatically tune the parameters in ranking models based on a large volume of training data. This technology has been gaining increasing attention from both the research community and the industry in the past several years. The setting of learning to rank, when applied to document retrieval and web search, is as follows. Assume that there is a corpus of documents. In training, a number of queries are provided; each query is associated with a set of documents with relevance judgments. Each query-document pair is represented by a feature vector. A ranking function is then created using the training data, such that the model can precisely predict the ranked lists in the training data by appropriately combining the features. In retrieval (i.e., testing), given a new query, the ranking function is used to create a ranked list for the documents associated with the query.

Many learning to rank methods have been proposed and applied to different IR applications.

One approach in previous work takes document pairs as instances and reduces the problem of ranking to that of classification on the orders of document pairs. It then applies existing classification techniques to ranking. The methods include Ranking SVM (Herbrich et al. 1999; Joachims 2002), RankBoost (Freund et al. 2003), RankNet (Burges et al. 2005). Ranking SVM solves the problem of pairwise classification using Support Vector Machines, RankBoost using the boosting techniques, and RankNet using Neural Networks. See also Tsai et al. (2007), Zheng et al. (2007) for other pairwise methods.

Another approach regards ranking lists as instances and conducts learning on the lists of documents. For instance, Cao et al. proposed using a permutation probability model in the rank learning and employing a listwise ranking algorithm called ListNet (Cao et al. 2007). In their recent work (Xia et al. 2008), they further studied the properties of the related algorithms and derived a new algorithm based on Maximum Likelihood Estimation called ListMLE. See also Qin et al. (2008c), Volkovs and Zemel (2009) for other listwise methods.

2.2 Direct optimization of IR measures

Recently, a new approach, direct optimization of IR measures, has attracted much attention in learning to rank. The basic idea of the direct optimization approach is to find an optimal ranking function by directly maximizing some IR measures such as AP (Voorhees and Harman 2005) and NDCG (Järvelin and Kekäläinen 2002) on the training set. This new approach seems more straightforward and appealing, because what is used in evaluation is exactly an IR measure. There are two major categories of algorithms for direct optimization of IR measures.

One group of algorithms tries to optimize objective functions that are bounds of the IR measures. For example, SVM^{map} (Yue et al. 2007) optimizes an upper bound of $(1 - AP)$ in the predicted rankings. Specifically, a joint feature map is constructed for each possible ranking, and Structured SVM is used to iteratively optimize the most violated constraint (the way of finding the most violated constraint depends on the property of AP). The idea of SVM^{map} is further extended to optimize other IR evaluation measures, and the corresponding algorithms include SVM^{ndcg} (Chapelle et al. 2007) and SVM^{nrr} (Chakrabarti et al. 2008). In these new algorithms, different joint feature maps and different ways of finding the most-violated constraints are proposed. AdaRank (Xu and Li 2007) minimizes an exponential loss function which can upper bound either $(1 - AP)$ or $(1 - NDCG)$ using boosting methods. It repeatedly constructs weak rankers on the basis of re-weighted training queries and finally linearly combines the weak rankers for making ranking predictions. Two sub methods have been proposed in (Xu and Li 2007). AdaRank.MAP utilizes AP to measure the goodness of a weak ranker, and AdaRank.NDCG utilizes NDCG to measure the goodness of a weak ranker.

Another group of algorithms manages to smooth the IR measures with easy-to-optimize functions. For example, SoftRank (Guiver and Snellson 2008; Taylor et al. 2008) introduces randomness to the ranking scores of the documents, so as to smooth NDCG. It assumes the ranking score of a document to be governed by a Gaussian distribution, and then derives a rank distribution of the document in an iterative manner. Based on the rank distributions of all the documents associated with a query, SoftRank computes the expectation of NDCG as the objective function for learning to rank. The gradient descent method is used to learn the ranking function.

According to previous studies, direct optimization algorithms can achieve higher performances when compared to the other approaches (Chapelle et al. 2007; Taylor et al. 2008; Xu and Li 2007; Xu et al. 2008; Yue et al. 2007). However, there are still some open issues as shown below.

1. The relationships between the surrogate functions and the corresponding IR measures have not been sufficiently studied. Therefore, it is unknown whether optimizing the surrogate functions can indeed optimize the corresponding IR measures.
2. Some of the proposed surrogate functions are not easy to optimize. Existing methods (e.g., the Structured SVM series) have to employ complicated, measure-specific techniques in the optimization. It is not trivial to extend them to new measures.

3 A general approximation framework

In this section, we propose a general framework for direct optimization of IR measures. The framework is applicable to any position based IR measure, and is theoretically justifiable.

In the framework, we take the approach of approximating the IR measures. The framework consists of four steps:

1. *Reformulating an IR measure from ‘indexed by positions’ to ‘indexed by documents’.* The newly formulated IR measure then contains a position function and optionally a truncation function. Both functions are non-continuous and non-differentiable.
2. *Approximating the position function with a logistic function of ranking scores of documents.*

3. *Approximating the truncation function with a logistic function of positions of documents.*
4. *Applying a global optimization technique to optimize the approximated measure (surrogate function).*

For ease of description, we give some notations here. Suppose that \mathcal{X} is a set of documents associated with a query, and x is an element in \mathcal{X} . A ranking model f outputs a score s_x for each x :

$$s_x = f(x; \theta), \quad x \in \mathcal{X}$$

where θ denotes the parameter of f . A ranked list π can be obtained by sorting the documents in the descending order of their scores. We use $\pi(x)$ to denote the position of document x in the ranked list π . Given the relevance label $r(x)$ of each document x , an IR measure can be used to evaluate the goodness of π . Note that different f 's will generate different π 's and thus achieve different ranking performances in terms of the IR measure. Further, we use $\mathbf{1}\{A\}$ to denote an indicator function: $\mathbf{1}\{A\} = 1$ if A is true, and $\mathbf{1}\{A\} = 0$ for other cases.

We first give a brief introduction to several popular IR measures used in learning to rank, and then take some measures as examples to introduce the four steps of the framework.

3.1 Review on IR measures

To evaluate the effectiveness of a ranking model, many IR measures have been proposed. Here we give a brief introduction to several popular ones which are widely used in learning to rank. See also (Moffat and Zobel 2008) for other measures.

Precision@ k (Voorhees et al. 2005) is a measure for evaluating top k positions of a ranked list using two levels (relevant and irrelevant) of relevance judgment:

$$\text{Pre@}k = \frac{1}{k} \sum_{j=1}^k r_j, \tag{1}$$

where k denotes the truncation position, r_j equals one if the document in the j th position is relevant and zero otherwise.

Average Precision (AP) (Voorhees et al. 2005) is defined on the basis of Precision:

$$\text{AP} = \frac{1}{|D_+|} \sum_j r_j \times \text{Pre@}j, \tag{2}$$

where $|D_+|$ denotes the number of relevant documents with respect to the query. Given the ranked list for a query, we can compute an AP for this query. Then MAP is defined as the mean of AP over a set of queries.

While Precision@ k and AP consider only two levels of relevance judgments, Normalized Discounted Cumulated Gain (NDCG) (Järvelin and Kekäläinen 2002) is designed for multiple levels of relevance judgments. NDCG@ k evaluates top k positions of a ranked list using multiple levels (labels) of relevance judgment. It is defined as below,

$$\text{NDCG@}k = N_k^{-1} \sum_{j=1}^k \frac{2^{r_j} - 1}{\log_2(1 + j)}. \tag{3}$$

where N_k is a constant depending on a query to make the maximum value of NDCG@ k of the query is 1.

By considering all the n documents associated with a query, we can attain $NDCG@n$, which is referred to as $NDCG$ for short in this paper if without confusion.

$$NDCG = NDCG@n = N_n^{-1} \sum_{j=1}^n \frac{2^j - 1}{\log_2(1 + j)}. \tag{4}$$

3.2 Step 1: Measure reformulation

Most of the IR measures, for example, $Precision@k$, AP and $NDCG$ are position based. Specifically, the summations in the definitions of IR measures are taken over positions, as can be seen in (1)–(4). Unfortunately, the position of a document may change during the training process, which makes the optimization of the IR measures difficult. To deal with the problem, we reformulate IR measures using the indexes of documents.

When indexed by documents, $Precision@k$ in (1) can be re-written as below,

$$Pre@k = \frac{1}{k} \sum_{x \in \mathcal{X}} r(x) \mathbf{1}\{\pi(x) \leq k\}, \tag{5}$$

where $r(x)$ equals one for relevant documents and zero for irrelevant documents, and $\mathbf{1}\{\pi(x) \leq k\}$ is a truncation function indicating whether document x is ranked at top k positions.

With documents as indexes, AP in (2) can be re-written as follows,

$$AP = \frac{1}{|D_+|} \sum_{y \in \mathcal{X}} r(y) \times Pre@ \pi(y). \tag{6}$$

Combining (5) and (6) yields

$$\begin{aligned} AP &= \frac{1}{|D_+|} \sum_{y \in \mathcal{X}} r(y) \times \frac{1}{\pi(y)} \sum_{x \in \mathcal{X}} r(x) \mathbf{1}\{\pi(x) \leq \pi(y)\} \\ &= \frac{1}{|D_+|} \sum_{y \in \mathcal{X}} \left(\frac{r(y)}{\pi(y)} + \sum_{x \in \mathcal{X}, x \neq y} r(y)r(x) \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)} \right) \end{aligned} \tag{7}$$

where $\mathbf{1}\{\pi(x) < \pi(y)\}$ is also a truncation function indicating whether document x is ranked before document y .

Similarly, when indexed by documents, (3) of $NDCG@k$ can be re-written as the following,

$$NDCG@k = N_k^{-1} \sum_{x \in \mathcal{X}} \frac{2^{r(x)} - 1}{\log_2(1 + \pi(x))} \mathbf{1}\{\pi(x) \leq k\}. \tag{8}$$

Here $r(x)$ is an integer. For example, $r(x) = 0$ means that document x is irrelevant to the query, and $r(x) = 4$ means that the document is highly relevant to the query.

Note that $NDCG$ (more accurately $NDCG@n$) does not need a truncation function,

$$NDCG = N_n^{-1} \sum_{x \in \mathcal{X}} \frac{2^{r(x)} - 1}{\log_2(1 + \pi(x))}. \tag{9}$$

The reformulated IR measures [e.g., (5), (7)–(9)] contain two kinds of functions: position function $\pi(x)$ and truncation functions $\mathbf{1}\{\pi(x) < \pi(y)\}$ and $\mathbf{1}\{\pi(x) \leq k\}$. Both of

them are non-continuous and non-differentiable. We will discuss how to approximate them separately in the next two subsections.

3.3 Step 2: Position function approximation

The position function can be represented as a function of ranking scores,

$$\pi(x) = 1 + \sum_{y \in \mathcal{X}, y \neq x} \mathbf{1}\{s_{x,y} < 0\}, \tag{10}$$

where $s_{x,y} = s_x - s_y$.

That is, positions can be regarded as outputs of functions of ranking scores. Due to the indicator function in it, the position function is non-continuous and non-differentiable.

We propose approximating the indicator function $\mathbf{1}\{s_{x,y} < 0\}$ using a logistic function (which is continuous and differentiable):

$$\frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})}, \tag{11}$$

where $\alpha > 0$ is a scaling constant.

In this way, the position function is correspondingly approximated and becomes continuous and differentiable (denoted as $\hat{\pi}(x)$), as shown below.

$$\hat{\pi}(x) = 1 + \sum_{y \in \mathcal{X}, y \neq x} \frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})} \tag{12}$$

Table 1 shows an example of the above position approximation process. We can see that the approximation is very accurate in this case.

Now we can obtain the approximation of NDCG by simply replacing $\pi(x)$ in (9) with $\hat{\pi}(x)$,

$$\widehat{\text{NDCG}} = N_n^{-1} \sum_{x \in \mathcal{X}} \frac{2^{r(x)} - 1}{\log_2(1 + \hat{\pi}(x))}. \tag{13}$$

3.4 Step 3: Truncation function approximation

As can be seen in Sect. 3.2, some measures have truncation functions in their definitions, such as Precision@ k , AP, and NDCG@ k . These measures need further approximations on the truncation functions. We will introduce in this subsection how it can be achieved. Some other measures including NDCG do not have truncation functions; In this case, the techniques introduced below can be skipped.

Table 1 Examples of position approximation

Document	s_x	$\pi(x)$	$\hat{\pi}(x)$ ($\alpha = 100$)
x_1	4.20074	2	2.00118
x_2	3.12378	4	4.00000
x_3	4.40918	1	1.00000
x_4	1.55258	5	5.00000
x_5	4.13330	3	2.99882

To approximate the truncation function $\mathbf{1}\{\pi(x) < \pi(y)\}$ in (7), a simple way is to use the logistic function once again,

$$\mathbf{1}\{\pi(x) < \pi(y)\} \approx \frac{\exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))}{1 + \exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))}, \tag{14}$$

in which $\beta > 0$ is a scaling constant.

Thus, we obtain the approximation of AP as follows,

$$\widehat{\text{AP}} = \frac{1}{|D_+|} \sum_y \left(\frac{r(y)}{\hat{\pi}(y)} + \sum_{x \neq y} \frac{r(y)r(x)}{\hat{\pi}(y)} \frac{\exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))}{1 + \exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))} \right). \tag{15}$$

Similarly, we can also approximate the truncation function $\mathbf{1}\{\pi(x) \leq k\}$ in (8) as

$$\mathbf{1}\{\pi(x) \leq k\} \approx \mathbf{1}\{\hat{\pi}(x) \leq k\} \approx \frac{\exp(\beta(k + 0.5 - \hat{\pi}(x)))}{1 + \exp(\beta(k + 0.5 - \hat{\pi}(x)))}. \tag{16}$$

Here we add 0.5 to change \leq to $<$. Note that we use 0.5 instead of other values such as 0.1 and 0.8 because 0.5 is in the middle of range [0, 1] and it will achieve the smallest approximation error in the worse case. That is,

$$\gamma^* = \arg \min_{\gamma} \left\{ \max_{\hat{\pi}(x)} \left| \mathbf{1}\{\hat{\pi}(x) \leq k\} - \frac{\exp(\beta(k + \gamma - \hat{\pi}(x)))}{1 + \exp(\beta(k + \gamma - \hat{\pi}(x)))} \right| \right\} = 0.5. \tag{17}$$

With (16), one can approximate measures like Precision@ k and NDCG@ k . Here we omit the details.

3.5 Step 4: Surrogate function optimization

With the aforementioned approximation technique, the surrogate objective functions (e.g., AP and NDCG) become continuous and differentiable with respect to the parameter θ in the ranking model, and many optimization algorithms can be used to maximize them. Measure specific optimization techniques are no longer needed.

However, considering that the original IR measures contain a lot of local optima, the approximations of them will also contain local optima. Therefore, one should better choose those global optimization methods such as random restart (Hu et al. 1994) and simulated annealing (Kirkpatrick et al. 1983) in order to avoid being trapped to local optima. Note that there are also some alternative ways of dealing with the issue of local optimum. For example, one can use a robust but likely less effective learning to rank method (e.g., Ranking SVM) to obtain an initial guess of the ranking model. Then use it as the starting point for the optimization of the approximated IR measure.

In this work, we choose to use the random restart technology (Hu et al. 1994) as an example. That is, we first use a gradient descent method to find a local optimum of the objective function given a certain initial value of the ranking model, and then we randomly re-initialize the model parameters and do another round of optimization. We repeat this for several times. Finally we regard the best local optimum as a global optimum.

Since AP and NDCG are widely used for evaluation in learning to rank, we take them as examples to show how to perform the optimization, and call the corresponding algorithms ApproxAP and ApproxNDCG respectively. The details about the derivation of gradients of AP and NDCG can be found in Appendix sections “[Gradient of ApproxNDCG](#)” and

“Gradient of ApproxAP”. According to the derivations, the complexity of computing the gradient is $O(n^2)$, where n is the number of documents associated with a query. The training process is shown in Algorithm 1.

Algorithm 1. ApproxAP (ApproxNDCG)

Input:

- 1: m training queries, their associated documents and relevance judgments.
- 2: Number of random restarts K ;
- 3: Stop threshold δ ;
- 4: Learning rate η .

Training:

- 5: Set iteration number $t = 0$;
- 6: For $k = 1: K$ Do {
- 7: Randomly initialize the parameter θ_t of the ranking model $f(x; \theta)$
- 8: Do {
- 9: Set $\theta = \theta_t$;
- 10: Shuffle the m training queries;
- 11: For $i = 1$ to m Do {
- 12: Feed i -th training query (after shuffle) to the learning system;
- 13: Compute the gradient $\Delta\theta$ of $\widehat{\text{AP}}$ ($\widehat{\text{NDCG}}$) with respect to θ using (38) [using (35)];
- 14: Update parameter $\theta = \theta + \eta \times \Delta\theta$;
- 15: }
- 16: Set $t = t + 1$, $\theta_t = \theta$.
- 17: } While ($\|\theta_t - \theta_{t-1}\| > \delta$)
- 18: Set $\omega_k = \theta_t$.
- 19: }

Output:

- 20: Compute the objectives ($\widehat{\text{AP}}$ or $\widehat{\text{NDCG}}$) of the K parameters $\{\omega_1, \omega_2, \dots, \omega_K\}$.
 - 21: Output the parameter ω_k with the maximal objective.
-

3.6 Comparison with previous methods

SoftRank (Taylor et al. 2008) also approximates the IR measure using a smooth function. When comparing the proposed framework to SoftRank, we can find the following differences.

1. Our framework approximates IR measures by approximating document positions, while SoftRank smooths NDCG by smoothing the document scores.
2. The gradient of our approach can be computed in a complexity of $O(n^2)$ while a complexity of $O(n^3)$ is needed for SoftRank, where n is the number of documents for a query. That is, the computation complexity of our approach is much lower than that of SoftRank. In our experiments, we observed that our method took less than 0.5 second to compute the gradients of all the training queries on the OHSUMED dataset and SoftRank took about 10 seconds.

3. We propose a general framework, which can be used to optimize any position based measures, while SoftRank only focuses on NDCG, and some extensive efforts are needed to generalize it to other measures.
4. Our approach has a solid theoretical justification on the accuracy of the approximation (see next section), while there is not yet such justification for SoftRank.

4 Theoretical analysis of the framework

As mentioned in Sect. 1, the relationships between the surrogate objective functions and the corresponding IR measures are not clear for the previous methods. In contrast, the relation between the surrogate functions obtained by our framework and the IR measures can be well justified. In this section, we will study this issue.

4.1 Accuracy of position function approximation

The approximation of positions is a basic component in our framework. In order to approximate an IR measure, we need to approximate positions first; in order to analyze the accuracy of approximation of IR measures, we need to analyze the accuracy of approximation of positions first.

Note that if $s_{x,y} = 0$ (i.e., document x and y have the same score), there will be no unique ranked list by sorting. This would bring uncertainty to IR measures. For the sake of clarity, in this paper, we assume that

$$\delta = \min_{x,y \in \mathcal{X}, x \neq y} |s_{x,y}| > 0 \quad (18)$$

The following theorem shows that the position approximation in (12) can achieve very high accuracy. The proof can be found in the appendix.

Theorem 1 *Given a document collection \mathcal{X} with n documents in it, for $\forall \alpha > 0$, (12) can approximate the true position with the following accuracy,*

$$|\hat{\pi}(x) - \pi(x)| < \frac{n-1}{\exp(\delta_x \alpha) + 1}, \quad (19)$$

where $\delta_x = \min_{y \in \mathcal{X}, y \neq x} |s_{x,y}|$.

This theorem tells us that when δ_x and α are large, the approximation will be very accurate. For example,

$$\lim_{\delta_x \alpha \rightarrow \infty} \hat{\pi}(x) = \pi(x).$$

A corollary of Theorem 1 is given below:

Corollary 1 *Given a document collection \mathcal{X} with n documents in it, for $\forall \alpha > 0$, (12) can approximate the true position with an accuracy as below.*

$$\varepsilon \triangleq \max_{x \in \mathcal{X}} |\hat{\pi}(x) - \pi(x)| < \frac{n-1}{\exp(\delta \alpha) + 1} \quad (20)$$

For the example in Table 1, we have an accurate approximation:

$$0.00118 = \varepsilon < \frac{5 - 1}{\exp(0.06744 * 100) + 1} \approx 0.00471. \tag{21}$$

One may argue that the bounds in (19) and (20) are not very meaningful when n is large. Note that the denominator is in the exponential order of parameter α and the numerator is in the linear order of n . Due to this difference in the orders, for a given n , by selecting a not-too-large α , one can always make the bound tight enough. For example in (21), if n increases from 5 to 1000, we can still get a very tight bound by simply increasing α from 100 to 200:

$$\frac{1000 - 1}{\exp(0.06744 * 200) + 1} \approx 0.001386.$$

4.2 Accuracy of IR measure approximation

The following theorems quantify the errors in the approximations of MAP and NDCG. The proof can be found in the Appendix.

Theorem 2 *If the error ε of position approximation in (20) is smaller than 0.5, then we have*

$$|\widehat{AP} - AP| < \frac{1}{1 + \exp(\beta(1 - 2\varepsilon))} \sum_{i=1}^{|D_+|} \frac{1}{i - \varepsilon} + \sum_{i=1}^{|D_+|} \frac{2\varepsilon}{i \cdot (i - \varepsilon)} \tag{22}$$

The theorem indicates that when ε is small and β is large, the approximation of AP can be very accurate. In the extreme case, we have $\lim_{\varepsilon \rightarrow 0, \beta \rightarrow \infty} \widehat{AP} = AP$. For the example in Table 1, if setting $\beta = 100, |D_+| = 1$, we have $|\widehat{AP} - AP| < 0.0024$. That is, the AP approximation is very accurate in this case.

Theorem 3 *The approximation error of \widehat{NDCG} can be bounded as*

$$|\widehat{NDCG} - NDCG| < \frac{\varepsilon}{2 \ln 2}. \tag{23}$$

This theorem indicates that when ε is small, the approximation of $NDCG$ can be very accurate. In the extreme case, we have $\lim_{\varepsilon \rightarrow 0} \widehat{NDCG} = NDCG$. For the example in Table 1, we have $|\widehat{NDCG} - NDCG| < \frac{\varepsilon}{2 \ln 2} \approx 0.00085$. That is, the $NDCG$ approximation is very accurate in this case.

From these two examples (AP and $NDCG$), one can see that the surrogate functions obtained by the proposed framework can be very accurate approximations to the IR measures.

4.3 Justification of accurate approximation

As shown in the previous subsection, the surrogate objective function we obtained can be very close to the original IR measure. One may argue whether such an accurate approximation really has benefit for learning the ranking model. To answer such questions, we have the following discussions.

If an algorithm can directly optimize an IR measure on the training set, then the learned ranking model will definitely be the optimal model in terms of the IR measure on the training set. Note that in statistical machine learning, the training performance is computed as an average on the training set, while the test performance is measured as an expectation on the entire instance space (Vapnik 1998). If the training set is extremely large, the training performance will converge to the test performance (i.e., the average will converge to the expectation when the number of samples is infinite). Therefore, directly optimizing the IR measure on a extremely large training set can guarantee the optimal test performance in terms of the same IR measure.

Furthermore, it is easy to understand that if the surrogate measures are very close to the IR measures (i.e., the approximations are very accurate), the optimization of the surrogate measures will also lead to high performances of IR measures on training set. Again, if the training set is very large, the optimization of the surrogate measures will also have high test performances. This intuitively justifies the necessity of accurately approximating the IR measures.

One possible issue with regards to the accurate approximation of IR measures is that the more accurate the approximation is, the more complex the surrogate function will be. This is because the IR measure itself is very complex (as a function of the ranking model), and contains a lot of local optima. In this case, the optimization of the surrogate function is likely to be trapped into some local optimum and the model learned may not have the desired good performance. This is why we propose using global optimization techniques in our framework.

Due to space restrictions, we have only given some high level discussions here. More details can be found in Qin et al. (2008a).

5 Experimental results

We conducted a set of experiments to test the effectiveness of the proposed framework.

5.1 Datasets

We used LETOR datasets (Liu et al. 2007) in our experiments. LETOR is a benchmark collection for the research on learning to rank for information retrieval. It has been widely used in research community for research on learning to rank (Duh and Kirchhoff 2008; Guiver and Snelson 2008; Qin et al. 2008b; Xu et al. 2008; Zhou et al. 2008). The first version of LETOR was released in April 2007 and used in the SIGIR 2007 workshop on learning to rank for information retrieval (<http://www.research.microsoft.com/users/LR4IR-2007/>). At the end of 2007, the second version of LETOR was released, which was later used in the SIGIR 2008 workshop on learning to rank for IR (<http://www.research.microsoft.com/users/LR4IR-2008/>). The third version of LETOR, namely LETOR 3.0, was released in December 2008.¹

We used three datasets in LETOR 3.0 to test our algorithms: TD2003, TD2004 and OHSUMED. The statistics of the three datasets are shown in Table 2. The datasets can be downloaded from LEOTR website (<http://www.research.microsoft.com/~letor>). The TD2003 and TD2004 datasets were used to test ApproxAP and the OHSUMED dataset was

¹ The latest version when the paper was submitted.

Table 2 Datasets

Datasets	# Query	Relevance levels	# (Docs per query)
TD2003	50	2	~1,000
TD2004	75	2	~1,000
OHSUMED	106	3	~150

used to test ApproxNDCG, since the first two datasets contain two-level relevance judgments and the third one contains three-level relevance judgments.

We note that most baseline algorithms in LETOR used linear ranking models. For fair comparison, we also used linear ranking model for ApproxAP and ApproxNDCG in the experiments, although our algorithms can also make use of other kinds of ranking models.

5.2 On the approximation of IR measures

We first evaluated the accuracy of the approximations of AP and NDCG.

As seen in Sect. 3.4, there are two parameters in \widehat{AP} , α and β . We first fixed $\beta = 10$ and set three different values for α . Then, we applied the ApproxAP algorithm to the TD2004 dataset with these three different parameters. Figure 1a shows the error ρ in the training process defined as

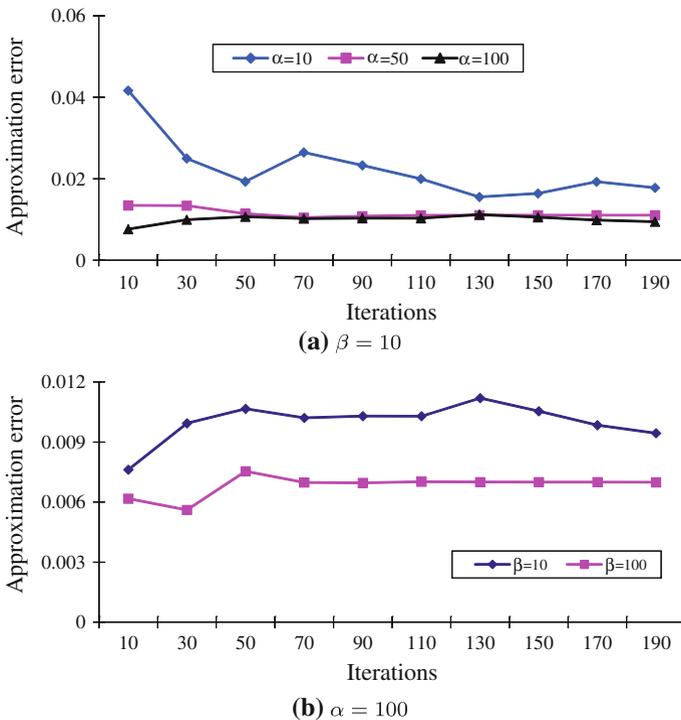


Fig. 1 Accuracy of AP approximation on TD2004 dataset. This is training curve over fold 1. The x-axis is the number of iterations t in Algorithm 1. **a** $\beta = 10$, **b** $\alpha = 100$

$$\rho = \frac{1}{|Q|} \sum_{q \in Q} |\widehat{AP}(q) - AP(q)|,$$

in which $\widehat{AP}(q)$ and $AP(q)$ mean the values of \widehat{AP} and AP respectively over a query q , Q is the training query set, and $|Q|$ is the number of queries in the training set.

We can see that for all the three α values, the approximation accuracy is very high, which is more than 95%. Furthermore, as the increase of α , the approximation becomes more accurate: the accuracy is higher than 98% when $\alpha = 100$.

We then fixed $\alpha = 100$ and tried different values of β . Figure 1b shows the error ρ with respect to different β values. As can be seen, when β increases, the accuracy of the approximation also improves.

Figure 2 shows the error $\rho = \frac{1}{|Q|} \sum_{q \in Q} |\widehat{NDCG}(q) - NDCG(q)|$ with regards to different α values. We can get similar observations.

All these results verify the correctness of the discussions in Sect. 4.2, and indicate that the approximation of IR measures using our proposed method can achieve high accuracy.

5.3 On the performance of ApproxAP

We adopted the five fold cross validation as suggested in LETOR for both TD2003 and TD2004 datasets. For each fold, we used the training set to learn the ranking model, use the validation set to select hyper parameters α and β in the ApproxAP algorithm, and use the test set to report the ranking performance. The detailed process is as follows:

- (a) we first chose a set of α values $\{50, 100, 150, 200, 250, 300\}$ and a set of β values $\{1, 10, 20, 50, 100\}$.
- (b) we set $\delta = 0.001$, $\eta = 0.01$, $K = 10$ in Algorithm 1. That is, we made 10 random restarts.
- (c) for each combination of α and β , we learned a ranking model with 10 random restarts to avoid local optima. We learned 30 models in total.
- (d) we tested the performance of each model on the validation set and selected the model with the highest MAP as the final model;
- (e) we tested the performance of the final model on the test set.

As baselines, we used AdaRank.MAP and SVM^{map}, which directly optimize AP. We also compared with Ranking SVM and ListNet, two state-of-the-art algorithms that do not

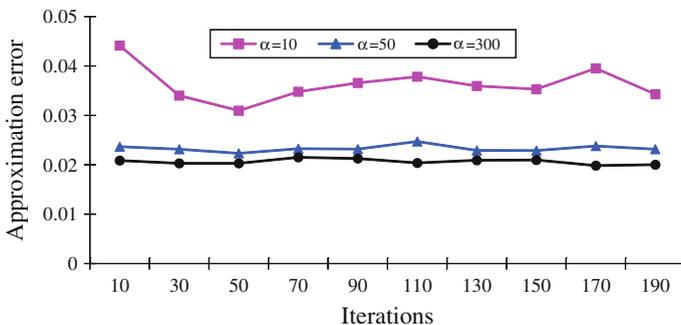


Fig. 2 Accuracy of NDCG@n approximation on OHSUMED dataset. This is training curve over fold 1. The x-axis is the number of iterations t in Algorithm 1

Table 3 Ranking accuracy in terms of MAP

Algorithm	TD2003	TD2004
AdaRank.MAP	0.2283	0.2189
SVM ^{map}	0.2445	0.2049
Ranking SVM	0.2628	0.2237
ListNet	0.2753	0.2231
ApproxAP	0.2834	0.2224

belong to the approach of direct optimization. We cited the results of AdaRank.MAP, SVM^{map}, Ranking SVM, and ListNet directly from LETOR official website (<http://www.research.microsoft.com/~letor>). According to the information in the website, the hyper parameters of these algorithms have been carefully tuned and the validation set has been used for model selection. In this regard, the experimental settings for our methods and these baselines are the same, which ensures a fair comparison among them.

As can be seen from Table 3, ApproxAP performs better than AdaRank.MAP and SVM^{map} on both datasets. For example on TD2003, ApproxAP gets more than 15% relative improvement over SVM^{map} and more than 20% relative improvement over AdaRank.MAP. Since ApproxAP only uses a simple gradient method for the optimization (as compared to the structured SVM and Boosting used in the two baselines), the current result clearly shows the advantage of using the proposed method for direct optimization, and we foresee that with the use of more advanced optimization techniques, the performance of ApproxAP could be further improved.

Furthermore, ApproxAP is better than Ranking SVM and ListNet on TD2003 and gets similar result as Ranking SVM and ListNet on TD2004. We also find that AdaRank.MAP and SVM^{map} are not as good as Ranking SVM and ListNet. We hypothesize the reason as follows. AdaRank.MAP and SVM^{map} optimize the upper bound of AP and it is not clear whether the bound is tight. If the bound is very loose, optimization of the bound cannot always lead to the optimization of AP, and so they may not perform well on some datasets. This is in accordance with the discussions in He and Liu (2008).

5.4 On the performance of ApproxNDCG

We used a similar strategy to select the hyper parameters α for ApproxNDCG to that for ApproxAP. We chose the same set of α values {50, 100, 150, 200, 250, 300} and the same value of δ, η and K . But we used NDCG@ n for model selection instead of MAP on the validation set.

We compared ApproxNDCG with AdaRank.NDCG and SoftRank, which directly optimize NDCG. We also compared with Ranking SVM and ListNet. We cited the results of AdaRank.NDCG, Ranking SVM and ListNet from LETOR official website (<http://www.research.microsoft.com/~letor>). Again, according to the information in the

Table 4 Ranking accuracy in terms of NDCG on OHSUMED dataset

Algorithm	@1	@3	@5	@10
AdaRank.NDCG	0.5330	0.4790	0.4673	0.4496
SoftRank	0.5229	0.4732	0.4580	0.4539
Ranking SVM	0.4958	0.4207	0.4164	0.4140
ListNet	0.5326	0.4732	0.4432	0.4410
ApproxNDCG	0.5771	0.5037	0.4794	0.4620

website, the hyper parameters of these algorithms have been carefully tuned and the validation set has been used for model selection. There is a hyper parameter σ in SoftRank. We tuned the parameter and used validation set to select the best value. That is, the same experimental strategy was applied to all the algorithms here for fair comparisons.

Table 4 shows average NDCG at positions 1, 3, 5 and 10 for the five algorithms on the OHSUMED dataset. For position 1, ApproxNDCG gets 0.08 NDCG gain over Ranking SVM, which is about 16% relative improvement; it also gets more than 0.04 NDCG gain over AdaRank.NDCG and SoftRank, which is about 8% relative improvement. Improvements can also be observed at other positions. Overall, ApproxNDCG achieves the highest accuracy. The performances in terms of NDCG@ n of AdaRank.NDCG, SoftRank, Ranking SVM, ListNet and ApproxNDCG are 0.6640, 0.6623, 0.6457, 0.6600 and 0.6680 respectively. Again ApproxNDCG is the best one. Note that since NDCG@ n considers all the documents in a ranked list, the difference of it of different algorithms is not so large as NDCG value at top positions. Overall, ApproxNDCG is the best of the compared algorithms. This verifies the effectiveness of our proposed method.

5.5 Discussions

In this sub section, we will make some deep investigations on the algorithms derived from our proposed framework.

5.5.1 Approximation accuracy versus optimization feasibility

As mentioned in Sect. 4.3, the larger the hyper-parameters (α and β) are, the more accurate the approximations of IR measures are, and the more difficult the optimization of the surrogate functions is.

Table 5 shows the training performance in terms of NDCG@5 of ApproxNDCG on fold 1 of the OHSUMED dataset with respect to different α and K values. As can be seen, with only one random restart ($K = 1$), $\alpha = 300$ does not get better training performance than $\alpha = 50$ and $\alpha = 100$. This is because a larger value of α makes the objective function more difficult to maximize. As we increase the number of random restarts to 100, we see that $\alpha = 300$ gets the best training accuracy, $\alpha = 100$ the second, and $\alpha = 50$ the third.

From this table, we conclude that larger value of α indeed makes the objective more difficult to maximize; to learn a better ranking model for large α , more random restarts are needed (or generally, more effective global optimization methods are needed). We got similar observations for ApproxAP. The details are omitted here.

5.5.2 Comparison with SoftRank

In Sect. 3.6, we have performed some analysis on the comparison with SoftRank, which belongs to the same sub category of the direct optimization approach as our methods. Here we make some experimental studies, including training performance and time complexity.

Table 5 Training performance (NDCG@5) of ApproxNDCG on fold 1 of OHSUMED dataset

K	1	100
$\alpha = 50$	0.4818	0.4828
$\alpha = 100$	0.4849	0.4862
$\alpha = 300$	0.4793	0.5073

K is the number of random restarts in Algorithm 1

Table 6 Running time per iteration on fold 1 of OHSUMED dataset

Algorithm	Time (seconds)
ApproxNDCG	<0.5
SoftRank	~ 10

After tuning the hyper parameter of SoftRank, it achieved its best training performance in terms of NDCG@5 on fold 1 of OHSUMED as 0.4940. Comparing the results in Table 5, we see that the best training accuracy of ApproxNDCG is better than that of SoftRank. From Tables 4 and 5, we get that ApproxNDCG achieved better ranking accuracy than SoftRank on both training and testing sets.

Furthermore, we also logged the running time in the experiments. Table 6 shows the information of ApproxNDCG and SoftRank on fold 1 of the OHSUMED dataset. As can be seen, ApproxNDCG is much faster than SoftRank.

6 Conclusions and future work

In this paper, we have set up a general framework to approximate position based IR measures. The key part of the framework is to approximate the positions of documents by logistic functions of their scores. There are several advantages of this framework: (1) the way of approximating position based measures is simple yet general; (2) many existing techniques can be directly applied to the optimization and the optimization process itself is measure independent; (3) it is easy to conduct analysis on the accuracy of the approach and high approximation accuracy can be achieved by setting appropriate parameters.

We have taken AP and NDCG as examples to show how to approximate IR measures within the proposed framework, how to analyze the accuracy of the approximation, and how to derive effective learning algorithms to optimize the approximated functions. Experiments on public benchmark datasets have verified the correctness of the theoretical analysis and have proved the effectiveness of our algorithms.

There are still some issues that need to be further studied.

1. The approximated measures are not convex, and there may be many local optima in training. We have used random restart strategy to find a good solution. We plan to study other global optimization methods to further improve the performance of the proposed algorithms.
2. We have used linear ranking models in the experiments. Our algorithms can be directly applied for other functions such as neural networks. We will conduct experiments to test our algorithms with other function classes.

Appendix: Approximation accuracy analysis

In the appendix, we give proofs of the major theorems in this paper.

(I) Proof of Theorem 1

Proof Note that

$$\begin{aligned}
 |\hat{\pi}(x) - \pi(x)| &= \sum_{y \in X, y \neq x} \left(\frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})} - \mathbf{1}\{s_{x,y} < 0\} \right) \\
 &\leq \sum_{y \in X, y \neq x} \left| \frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})} - \mathbf{1}\{s_{x,y} < 0\} \right|.
 \end{aligned}
 \tag{24}$$

If we can prove that for any document $y \in \mathcal{X}$,

$$\left| \frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})} - \mathbf{1}\{s_{x,y} < 0\} \right| < \frac{1}{\exp(\delta_x \alpha) + 1},
 \tag{25}$$

then we can have

$$|\hat{\pi}(x) - \pi(x)| < \sum_{y \in X, y \neq x} \frac{1}{\exp(\delta_x \alpha) + 1} = \frac{n - 1}{\exp(\delta_x \alpha) + 1}.
 \tag{26}$$

Now we prove the inequality (25). We consider $s_{x,y} > 0$ and $s_{x,y} < 0$ separately.

– For $s_{x,y} > 0$, from (18) we have

$$1 + \exp(\alpha s_{x,y}) > 1 + \exp(\delta_x \alpha).$$

Then,

$$\frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})} = \frac{1}{1 + \exp(\alpha s_{x,y})} < \frac{1}{1 + \exp(\delta_x \alpha)}.$$

Note that $\mathbf{1}\{s_{x,y} < 0\} = 0$ when $s_{x,y} > 0$. Hence, for $s_{x,y} > 0$,

$$\left| \frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})} - \mathbf{1}\{s_{x,y} < 0\} \right| < \frac{1}{1 + \exp(\delta_x \alpha)}.$$

– For $s_{x,y} < 0$, from (18) we have

$$1 + \exp(-\alpha s_{x,y}) > 1 + \exp(\delta_x \alpha).$$

Note that $\mathbf{1}\{s_{x,y} < 0\} = 1$ when $s_{x,y} < 0$. Hence, for $s_{x,y} < 0$,

$$\left| \frac{\exp(-\alpha s_{x,y})}{1 + \exp(-\alpha s_{x,y})} - \mathbf{1}\{s_{x,y} < 0\} \right| = \frac{1}{1 + \exp(-\alpha s_{x,y})} < \frac{1}{1 + \exp(\delta_x \alpha)}.$$

Combining the two cases we end up with (25). According to (26), Theorem 1 is correct.

(II) Proof of Theorem 2

We prove Theorem 2 about the accuracy of precision approximation.

Proof For simplicity, we denote

$$\hat{\mathbf{1}}\{\pi(x) < \pi(y)\} = \frac{\exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))}{1 + \exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))}.$$

From (7) and (15), we have

$$\begin{aligned}
 |\widehat{\text{MAP}} - \text{MAP}| &= \left| \sum_y \frac{r(y)}{|D_+|} \left(\frac{1}{\hat{\pi}(y)} - \frac{1}{\pi(y)} \right) \right. \\
 &\quad \left. + \sum_y \sum_{x \neq y} \frac{r(y)r(x)}{|D_+|} \left(\frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\hat{\pi}(y)} - \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)} \right) \right| \\
 &\leq \sum_y \sum_{x \neq y} \frac{r(y)r(x)}{|D_+|} \left| \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\hat{\pi}(y)} - \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)} \right| \\
 &\quad + \sum_y \frac{r(y)}{|D_+|} \left| \frac{1}{\hat{\pi}(y)} - \frac{1}{\pi(y)} \right|
 \end{aligned} \tag{27}$$

Now we consider $\left| \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\hat{\pi}(y)} - \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)} \right|$ and $\left| \frac{1}{\hat{\pi}(y)} - \frac{1}{\pi(y)} \right|$ respectively.

$$\begin{aligned}
 &\left| \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\hat{\pi}(y)} - \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)} \right| \\
 &= \left| \frac{\pi(y)\mathbf{1}\{\pi(x) < \pi(y)\} - \hat{\pi}(y)\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)\hat{\pi}(y)} \right| \\
 &= \left| \frac{\pi(y)(\mathbf{1}\{\pi(x) < \pi(y)\} - \mathbf{1}\{\pi(x) < \pi(y)\}) + (\pi(y) - \hat{\pi}(y))\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)\hat{\pi}(y)} \right| \\
 &\leq \left| \frac{\pi(y)(\mathbf{1}\{\pi(x) < \pi(y)\} - \mathbf{1}\{\pi(x) < \pi(y)\})}{\pi(y)\hat{\pi}(y)} \right| \\
 &\quad + \left| \frac{(\pi(y) - \hat{\pi}(y))\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)\hat{\pi}(y)} \right| \\
 &\leq \left| \frac{\mathbf{1}\{\pi(x) < \pi(y)\} - \mathbf{1}\{\pi(x) < \pi(y)\}}{\hat{\pi}(y)} \right| + \frac{\varepsilon}{\hat{\pi}(y)\pi(y)}
 \end{aligned} \tag{28}$$

Similar to the derivation of (25), we can get

$$\left| \mathbf{1}\{\pi(x) < \pi(y)\} - \mathbf{1}\{\pi(x) < \pi(y)\} \right| < \frac{1}{1 + \exp(\beta(1 - 2\varepsilon))}. \tag{29}$$

Combining (28) and (29), we get

$$\begin{aligned}
 &\sum_y \sum_{x \neq y} \frac{r(y)r(x)}{|D_+|} \left| \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\hat{\pi}(y)} - \frac{\mathbf{1}\{\pi(x) < \pi(y)\}}{\pi(y)} \right| \\
 &\leq \sum_y \sum_{x \neq y} \frac{r(y)r(x)}{|D_+|} \left(\frac{1}{\hat{\pi}(y)(1 + \exp(\beta(1 - 2\varepsilon)))} + \frac{\varepsilon}{\hat{\pi}(y)\pi(y)} \right) \\
 &< \sum_y r(y) \left(\frac{1}{\hat{\pi}(y)(1 + \exp(\beta(1 - 2\varepsilon)))} + \frac{\varepsilon}{\hat{\pi}(y)\pi(y)} \right) \\
 &< \frac{1}{1 + \exp(\beta(1 - 2\varepsilon))} \sum_{i=1}^{|D_+|} \frac{1}{i - \varepsilon} + \varepsilon \sum_{i=1}^{|D_+|} \frac{1}{i \cdot (i - \varepsilon)}
 \end{aligned} \tag{30}$$

$$\left| \frac{1}{\hat{\pi}(y)} - \frac{1}{\pi(y)} \right| = \left| \frac{\pi(y) - \hat{\pi}(y)}{\hat{\pi}(y)\pi(y)} \right| < \frac{\varepsilon}{\hat{\pi}(y)\pi(y)} \tag{31}$$

Then

$$\begin{aligned} \sum_y \frac{r(y)}{|D_+|} \left| \frac{1}{\hat{\pi}(y)} - \frac{1}{\pi(y)} \right| &< \sum_y \frac{r(y)}{|D_+|} \frac{\varepsilon}{\hat{\pi}(y)\pi(y)} \\ &\leq \frac{\varepsilon}{|D_+|} \sum_{i=1}^{|D_+|} \frac{1}{i \cdot (i - \varepsilon)} \end{aligned} \tag{32}$$

Substitute (30) and (32) into (27), we get

$$|\widehat{\text{MAP}} - \text{MAP}| \frac{1}{1 + \exp(\beta(1 - 2\varepsilon))} \sum_{i=1}^{|D_+|} \frac{1}{i - \varepsilon} + \varepsilon \frac{1 + |D_+|}{|D_+|} \sum_{i=1}^{|D_+|} \frac{1}{i \cdot (i - \varepsilon)}.$$

Since $|D_+| \geq 1$, hence

$$|\widehat{\text{MAP}} - \text{MAP}| < \frac{1}{1 + \exp(\beta(1 - 2\varepsilon))} \sum_{i=1}^{|D_+|} \frac{1}{i - \varepsilon} + 2\varepsilon \sum_{i=1}^{|D_+|} \frac{1}{i \cdot (i - \varepsilon)}.$$

Proof of Theorem 3

Proof From (9) and (13), we obtain

$$\begin{aligned} &|\widehat{\text{NDCG}} - \text{NDCG}| \\ &\leq N_n^{-1} \sum_{x \in \mathcal{X}} \frac{2^{r(x)} - 1}{\log_2(1 + \pi(x))} \left| \frac{\log_2(1 + \hat{\pi}(x)) - \log_2(1 + \pi(x))}{\log_2(1 + \hat{\pi}(x))} \right|. \end{aligned} \tag{33}$$

Since $\frac{\partial \log_2(1+t)}{\partial t} = \frac{1}{(1+t) \ln 2}$ and $\pi(x) \geq 1, \hat{\pi}(x) \geq 1$, we have

$$|\log_2(1 + \hat{\pi}(x)) - \log_2(1 + \pi(x))| < \frac{1}{2 \ln 2} |\hat{\pi}(x) - \pi(x)| \leq \frac{\varepsilon}{2 \ln 2}.$$

Considering that $\log_2(1 + \hat{\pi}(x)) > 1$, we have

$$\left| \frac{\log_2(1 + \hat{\pi}(x)) - \log_2(1 + \pi(x))}{\log_2(1 + \hat{\pi}(x))} \right| < \frac{\varepsilon}{2 \ln 2}. \tag{34}$$

Then (33) becomes

$$\begin{aligned} |\widehat{\text{NDCG}} - \text{NDCG}| &< N_n^{-1} \sum_{x \in \mathcal{X}} \frac{2^{r(x)} - 1}{\log_2(1 + \pi(x))} \frac{\varepsilon}{2 \ln 2} \\ &= \frac{\varepsilon}{2 \ln 2} \text{NDCG}. \end{aligned}$$

According to the definition of NDCG, we always have $\text{NDCG} \leq 1$. Hence,

$$|\widehat{\text{NDCG}} - \text{NDCG}| < \frac{\varepsilon}{2 \ln 2}.$$

Appendix: Gradient derivation

(I) Gradient of ApproxNDCG

We show how to derive the gradient for ApproxNDCG.

According to the chain rule, we obtain

$$\Delta\theta = \frac{\partial \widehat{\text{NDCG}}}{\partial \theta} = N_n^{-1} \sum_x \frac{\partial_{\frac{2^{r(x)}-1}{\log_2(1+\hat{\pi}(x))}} \partial \hat{\pi}(x)}{\partial \hat{\pi}(x)} \frac{\partial \hat{\pi}(x)}{\partial \theta}. \tag{35}$$

Further,

$$\begin{aligned} \frac{\partial \hat{\pi}(x)}{\partial \theta} &= -\alpha \sum_{y \neq x} \frac{\exp(\alpha s_{xy})}{(1 + \exp(\alpha s_{xy}))^2} \frac{\partial s_{xy}}{\partial \theta} \\ &= -\alpha \sum_{y \neq x} \frac{\exp(\alpha s_{xy})}{(1 + \exp(\alpha s_{xy}))^2} \left(\frac{\partial f(x; \theta)}{\partial \theta} - \frac{\partial f(y; \theta)}{\partial \theta} \right) \end{aligned} \tag{36}$$

$$\frac{\partial_{\frac{2^{r(x)}-1}{\log_2(1+\hat{\pi}(x))}}}{\partial \hat{\pi}(x)} = -\frac{2^{r(x)} - 1}{(\log_2(1 + \hat{\pi}(x)))^2} \frac{\ln 2}{(1 + \hat{\pi}(x))} \tag{37}$$

Substituting (36) and (37) into (35), we get the gradient for ApproxNDCG.

Note that $\frac{\partial f(x; \theta)}{\partial \theta}$ in (36) depends on the specific form of the ranking model f . For example, for linear function, we have $\frac{\partial f(x; \theta)}{\partial \theta} = x$.

(II) Gradient of ApproxAP

We next show how to derive the gradient for ApproxAP.

According to the chain rule, we obtain

$$\frac{\partial \widehat{\text{AP}}}{\partial \theta} = \frac{-1}{|D_+|} \sum_y \frac{r(y)}{\hat{\pi}^2(y)} \frac{\partial \hat{\pi}(y)}{\partial \theta} + \frac{1}{|D_+|} \sum_y \sum_{x \neq y} r(y)r(x) \frac{\partial J(\theta)}{\partial \theta}, \tag{38}$$

where

$$J(\theta) = \frac{1}{\hat{\pi}(y)} \frac{\exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))}{1 + \exp(\beta(\hat{\pi}(y) - \hat{\pi}(x)))}.$$

Again by the chain rule, we have

$$\frac{\partial J(\theta)}{\partial \theta} = \frac{\partial J(\theta)}{\partial \hat{\pi}(y)} \frac{\partial \hat{\pi}(y)}{\partial \theta} + \frac{\partial J(\theta)}{\partial \hat{\pi}(x)} \frac{\partial \hat{\pi}(x)}{\partial \theta} \tag{39}$$

Now we consider $\frac{\partial J(\theta)}{\partial \hat{\pi}(x)}$ and $\frac{\partial J(\theta)}{\partial \hat{\pi}(y)}$.

$$\frac{\partial J(\theta)}{\partial \hat{\pi}(x)} = \frac{-1}{\hat{\pi}(y)} \frac{\beta \exp(\beta(\hat{\pi}(x) - \hat{\pi}(y)))}{(1 + \exp(\beta(\hat{\pi}(x) - \hat{\pi}(y))))^2}. \tag{40}$$

$$\begin{aligned} \frac{\partial J(\theta)}{\partial \hat{\pi}(y)} &= \frac{-1}{\hat{\pi}^2(y)} \frac{1}{1 + \exp(\beta(\hat{\pi}(x) - \hat{\pi}(y)))} \\ &\quad + \frac{1}{\hat{\pi}(y)} \frac{\beta \exp(\beta(\hat{\pi}(x) - \hat{\pi}(y)))}{(1 + \exp(\beta(\hat{\pi}(x) - \hat{\pi}(y))))^2}. \end{aligned} \tag{41}$$

Substituting (36), (40) and (41) into (39), and then substituting (39) into (38), we get the gradient for ApproxAP.

References

- Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., et al. (2005). Learning to rank using gradient descent. In *ICML '05: Proceedings of the 22nd international conference on Machine learning* (pp. 89–96). New York, NY, USA: ACM Press.
- Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., & Li, H. (2007). Learning to rank: From pairwise approach to listwise approach. In *ICML '07: Proceedings of the 24th international conference on Machine learning* (pp. 129–136). New York, NY, USA: ACM Press.
- Chakrabarti, S., Khanna, R., Sawant, U., & Bhattacharyya, C. (2008). Structured learning for non-smooth ranking losses. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 88–96). New York, NY, USA: ACM.
- Chapelle, O., Le, Q., & Smola, A. (2007). Large margin optimization of ranking measures. In *NIPS2007 workshop on machine learning for Web search*.
- Duh, K., & Kirchhoff, K. (2008). Learning to rank with partially-labeled data. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 251–258). New York, NY, USA: ACM.
- Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4, 933–969.
- Guiver, J., & Snelson, E. (2008). Learning to rank with softrank and gaussian processes. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 259–266). New York, NY, USA: ACM.
- He, Y., & Liu, T.-Y. (2008). *Are algorithms directly optimizing ir measures really direct?* Technical Report MSR-TR-2008-154, Microsoft Corporation.
- Herbrich, R., Graepel, T., & Obermayer K. (1999). Support vector learning for ordinal regression. In *ICANN1999* (pp. 97–102).
- Hu, X., Shonkwiler, R., & Spruill, M. (1994). Random restarts in global optimization. Technical Report, School of Mathematics, Georgia Institute of Technology, Atlanta.
- Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4), 422–446.
- Joachims T. (2002). Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the 8th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 133–142). New York, NY, USA: ACM Press.
- Kirkpatrick, S., Gelatt, C., & Vecchi, M. (1983). Optimization by simulated annealing. *Science* 220(4598), 671–680.
- Liu, T.-Y., Xu, J., Qin, T., Xiong, W.-Y., & Li, H. (2007). Letor: benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of SIGIR 2007 workshop on learning to rank for information retrieval*.
- Moffat, A., & Zobel, J. (2008). Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems*, 27(1), 1–27.
- Qin, T., Liu, T.-Y., & Li, H. (2008a). *A general approximation framework for direct optimization of information retrieval measures*. Technical Report MSR-TR-2008-164, Microsoft Corporation.
- Qin, T., Liu, T.-Y., Zhang, X.-D., Wang, D.-S., Xiong, W.-Y., & Li, H. (2008b). Learning to rank relational objects and its application to web search. In *WWW '08: Proceeding of the 17th international conference on World Wide Web* (pp. 407–416). New York, NY, USA: ACM.
- Qin, T., Zhang, X.-D., Tsai, M.-F., Wang, D.-S., Liu, T.-Y., & Li, H. (2008c). Query-level loss functions for information retrieval. *Information Processing & Management*, 44(2), 838–855.
- Robertson S. E., & Hull D. A. (2000). The TREC-9 filtering track final report. In *TREC* (pp. 25–40).
- Taylor, M., Guiver, J., Robertson, S., & Minka, T. (2008). Softrank: optimizing non-smooth rank metrics. In *WSDM '08: Proceedings of the international conference on Web search and web data mining* (pp. 77–86). New York, NY, USA: ACM.
- Tsai, M.-F., Liu, T.-Y., Qin, T., Chen, H.-H., & Ma, W.-Y. (2007). Frank: a ranking method with fidelity loss. In *SIGIR '07* (pp. 383–390). New York, NY, USA: ACM Press.
- Vapnik, V. (1998). *Statistical learning theory*. New York: Wiley.

- Volkovs, M. N., & Zemel, R. S. (2009). Boltzrank: Learning to maximize expected ranking gain. In *ICML '09: Proceedings of the 26th annual international conference on machine learning* (pp. 1089–1096). New York, NY, USA: ACM.
- Voorhees, E., & Harman, D. (2005). *TREC: Experiment and evaluation in information retrieval*. Cambridge: MIT Press.
- Xia, F., Liu, T.-Y., Wang, J., Zhang, W., & Li, H. (2008). Listwise approach to learning to rank: Theory and algorithm. In *ICML '08: Proceedings of the 25th international conference on machine learning* (pp. 1192–1199). New York, NY, USA: ACM.
- Xu, J., & Li, H. (2007). Adarank: A boosting algorithm for information retrieval. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 391–398). New York, NY, USA: ACM Press.
- Xu, J., Liu, T.-Y., Lu, M., Li, H., & Ma, W.-Y. (2008). Directly optimizing evaluation measures in learning to rank. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 107–114). New York, NY, USA: ACM.
- Yue, Y., Finley, T., Radlinski, F., & Joachims, T. (2007). A support vector method for optimizing average precision. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 271–278). New York, NY, USA: ACM Press.
- Zhai, C., & Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 334–342). New York, NY, USA: ACM Press.
- Zheng, Z., Chen, K., Sun, G., & Zha, H. (2007). A regression framework for learning ranking functions using relative relevance judgments. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 287–294). New York, NY, USA: ACM.
- Zhou, K., Xue, G.-R., Zha, H., & Yu, Y. (2008). Learning to rank with ties. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 275–282). New York, NY, USA: ACM.