Maximum Margin Ranking Algorithms for Information Retrieval

Shivani Agarwal and Michael Collins

Massachusetts Institute of Technology, Cambridge MA 02139, USA
{shivani,mcollins}@csail.mit.edu

Abstract. Machine learning ranking methods are increasingly applied to ranking tasks in information retrieval (IR). However ranking tasks in IR often differ from standard ranking tasks in machine learning, both in terms of problem structure and in terms of the evaluation criteria used to measure performance. Consequently, there has been much interest in recent years in developing ranking algorithms that directly optimize IR ranking measures. Here we propose a family of ranking algorithms that preserve the simplicity of standard pair-wise ranking methods in machine learning, yet show performance comparable to state-of-the-art IR ranking algorithms. Our algorithms optimize variations of the hinge loss used in support vector machines (SVMs); we discuss three variations, and in each case, give simple and efficient stochastic gradient algorithms to solve the resulting optimization problems. Two of these are stochastic gradient projection algorithms, one of which relies on a recent method for $l_{1,\infty}$-norm projections; the third is a stochastic exponentiated gradient algorithm. The algorithms are simple and efficient, have provable convergence properties, and in our preliminary experiments, show performance close to state-of-the-art algorithms that directly optimize IR ranking measures.

1 Introduction

Ranking methods in machine learning have gained considerable popularity in information retrieval (IR) in recent years [1–12]. Although the benefit of using such methods is rarely in question, there has been much debate recently about what types of ranking algorithms are best suited for the domain. In particular, ranking tasks in IR often differ from standard ranking tasks in machine learning in a variety of ways: for example, often in IR, one does not wish to learn a single ranking over all objects (in this case documents), but rather wishes to learn a ranking function that can rank different sets of documents with respect to different queries. Moreover, the ranking performance measures used in IR are usually different from standard pair-wise ranking measures, often focusing on the ranking quality at the top of a retrieved list.

These differences have led to several questions about how to best design ranking algorithms for IR, as well as several worthwhile adaptations and improvements of existing ranking algorithms [4, 7, 13]. For example, Qin et al. [13] argue that loss functions in IR should be defined at the level of queries rather than individual documents or document pairs. There has also been much discussion on pair-wise vs. list-wise ranking algorithms, where the latter employ loss functions that directly take into account the total order in a ranked list [14].

More recently, there has been much interest in algorithms that attempt to directly optimize ranking measures that are popular in IR, such as the normalized discounted cumulative gain (NDCG), mean average precision (MAP), and others [15, 5, 8, 7, 16, 10–12]. For example, Joachims [15] proposed a general method, inspired by large margin methods for structured prediction [17, 18], for optimizing multivariate performance measures that include as special cases the area under the ROC curve and measures related to recall and precision; this was extended by Yue et al. [8] to a support vector method for optimizing the MAP. Large margin structured prediction methods have also been used by Chapelle et al. [16] to optimize the NDCG. Other algorithms that attempt to optimize the NDCG include the LambdaRank algorithm of Burges et al. [5], the AdaRank algorithm of Xu and Li [7], the SoftRank algorithm of Taylor et al. [9], the regression-based algorithm of Cossock and Zhang [11], and the recent algorithm of Chapelle and Wu [12]. These algorithms have shown considerable promise, often resulting in significant improvement in performance over standard pair-wise ranking algorithms, such as RankSVM [19, 2] and RankBoost [20], applied to IR ranking tasks.

Here we propose a family of algorithms that preserve the simplicity of the pair-wise approach, yet exhibit performance comparable to state-of-the-art IR ranking algorithms. Our algorithms are based on optimizing variations of the hinge loss used in support vector machines (SVMs). We start with a pair-wise ranking loss that takes into account the degree of difference in the relevance (with respect to a query) of a pair of documents, and then use this to construct a query-level loss function. We discuss three variations of the query-level loss; in each case, we provide a stochastic gradient algorithm for solving the dual of the corresponding optimization problem. Two of these are stochastic gradient projection algorithms, one of which relies on a recent algorithm for projections onto $l_{1,\infty}$-norm constraints [21]; the third is a stochastic exponentiated gradient algorithm, similar to exponentiated gradient algorithms developed for structured prediction [22]. The resulting algorithms are simple and efficient, have provable convergence properties, and in our preliminary experiments, show performance close to state-of-the-art algorithms that optimize the MAP or NDCG.

The rest of the paper is organized as follows. In Section 2, we describe more formally the problem setting we consider. Section 3 gives our algorithms; this is followed by our experimental results in Section 4. We conclude with a discussion in Section 5.

2 Preliminaries

The problem we consider can be described as follows. There is a query space $\mathcal{Q}$ and a document space $\mathcal{D}$. The learner is given as training examples m queries $q^1, \ldots, q^m \in \mathcal{Q}$, the ith query q^i being associated with n_i documents $d^i_1, \ldots, d^i_{n_i} \in \mathcal{D}$, together with real-valued relevance labels $y^i_j \in \mathbb{R}$ denoting the (human-judged) relevance of document d^i_j to query q^i, and the goal is to learn from these examples a ranking function which, given a new query q, can rank the documents associated with the query such that more relevant documents are ranked higher than less relevant ones.

More formally, as is standard in the use of machine learning methods in IR, we shall assume a query-document feature mapping $\boldsymbol{\phi} : \mathcal{Q} \times \mathcal{D} \rightarrow \mathbb{R}^d$ that maps each query-document pair to a d-dimensional feature vector. The learner then receives labeled training examples of the form $S = (S^1, \ldots, S^m)$, where $S^i = ((\boldsymbol{\phi}_1^i, y_1^i), \ldots, (\boldsymbol{\phi}_{n_i}^i, y_{n_i}^i))$ is the training sample associated with the ith query; here $\boldsymbol{\phi}_j^i \equiv \boldsymbol{\phi}(q^i, d_j^i)$. The goal is to learn a real-valued ranking function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ that ranks accurately documents associated with future queries; f is taken to rank a document d_j associated with a query q higher than a document d_k if $f(\boldsymbol{\phi}(q, d_j)) > f(\boldsymbol{\phi}(q, d_k))$, and lower than d_k otherwise. In this paper, we shall be interested in linear ranking functions $f_{\mathbf{w}} : \mathbb{R}^d \rightarrow \mathbb{R}$ given by

$$f_{\mathbf{w}}(\boldsymbol{\phi}) = \mathbf{w} \cdot \boldsymbol{\phi}$$

for some weight vector $\mathbf{w} \in \mathbb{R}^d$.

Let us consider first the loss of such a function $f_{\mathbf{w}}$ on a pair of documents d_j^i, d_k^i associated with query q^i. In earlier SVM algorithms for ranking [2, 19], the following pair-wise hinge loss was often used:

$$\ell_{\mathsf{H}}(\mathbf{w}, (\boldsymbol{\phi}_j^i, y_j^i), (\boldsymbol{\phi}_k^i, y_k^i)) = \left[1 - z_{jk}^i \mathbf{w} \cdot (\boldsymbol{\phi}_j^i - \boldsymbol{\phi}_k^i)\right]_+ ,$$

where $[a]_+ = \max(0, a)$ and

$$z_{jk}^i = \text{sign}(y_j^i - y_k^i) .$$

These algorithms then consisted of minimizing an l_2-regularized version of the average pair-wise hinge-loss across all document pairs and all queries. In particular, if

$$R_i = \left\{(j, k) \mid y_j^i > y_k^i\right\}$$

denotes the set of 'preference pairs' for the ith query, then these early algorithms learned $\mathbf{w}$ by solving the following optimization problem:

$$\min_{\mathbf{w}} \left[\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{\left(\sum_{i=1}^m |R_i|\right)} \sum_{i=1}^m \sum_{(j,k) \in R_i} \ell_{\mathsf{H}}(\mathbf{w}, (\boldsymbol{\phi}_j^i, y_j^i), (\boldsymbol{\phi}_k^i, y_k^i))\right] , \tag{1}$$

where $C > 0$ denotes an appropriate regularization parameter. The pair-wise hinge loss ℓ_{H} can be seen as a convex upper bound on the following binary mis-ranking error, which simply assigns a constant penalty of 1 to each mis-ranked pair of documents:

$$\ell_{\text{0-1}}(\mathbf{w}, (\boldsymbol{\phi}_j^i, y_j^i), (\boldsymbol{\phi}_k^i, y_k^i)) = \mathbf{1}\left(z_{jk}^i \mathbf{w} \cdot (\boldsymbol{\phi}_j^i - \boldsymbol{\phi}_k^i) < 0\right) ,$$

where $\mathbf{1}(\psi)$ is the indicator function that takes the value 1 if the predicate ψ is true, and 0 otherwise. Thus, early SVM ranking algorithms ignored the possibility of a need to assign different penalties to different mis-ranked pairs, which arises when documents can have multiple relevance levels y_j^i. Recently, Cao et al. [4] addressed this issue by suggesting a modification to the above pair-wise hinge loss; however, the loss they propose relies on certain heuristics in order to set some parameters. Here we use the following simple and intuitive variation of the pair-wise hinge loss, which takes into account the different relevance levels of different documents; this loss was also used recently in the context of standard ranking algorithms in [23]:

$$\ell_{\mathsf{H,rel}}(\mathbf{w}, (\boldsymbol{\phi}_j^i, y_j^i), (\boldsymbol{\phi}_k^i, y_k^i)) = \big[|y_j^i - y_k^i| - z_{jk}^i \mathbf{w} \cdot (\boldsymbol{\phi}_j^i - \boldsymbol{\phi}_k^i) \big]_+ \ .$$

This can be viewed as a convex upper bound on the following relevance-weighted mis-ranking error:

$$\ell_{\mathsf{rel}}(\mathbf{w}, (\boldsymbol{\phi}_j^i, y_j^i), (\boldsymbol{\phi}_k^i, y_k^i)) = |y_j^i - y_k^i| \, \mathbf{1}\left(z_{jk}^i \mathbf{w} \cdot (\boldsymbol{\phi}_j^i - \boldsymbol{\phi}_k^i) < 0 \right) \ .$$

Thus, under ℓ_{rel} (and therefore $\ell_{\mathsf{H,rel}}$), mis-ranking a pair of documents with relevance labels 1 and 5 incurs a larger penalty than mis-ranking a pair of documents with relevance labels 1 and 2.

Our main interest will be not in the above pair-wise loss $\ell_{\mathsf{H,rel}}$ itself, but rather in query-level loss functions derived from it. In particular, we shall be interested in learning algorithms that select $\mathbf{w} \in \mathbb{R}^d$ as follows:

$$\min_{\mathbf{w}} \left[\frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{m} \sum_{i=1}^{m} L(\mathbf{w}, S^i) \right] \ ,$$

where $L(\mathbf{w}, S^i)$ is an appropriate loss term that measures the ranking loss incurred by $f_{\mathbf{w}}$ on the training sample S^i associated with the ith query, and $C > 0$ acts as a regularization parameter.

We note that our approach is different from recent large margin methods aimed at directly optimizing measures such as the MAP or NDCG [8, 16], which also use query-level loss functions based on the hinge loss. In particular, the query-level loss L in these approaches takes the form

$$L(\mathbf{w}, S^i) = \max_{\pi} \left[\varDelta(\pi, \pi_y) - \left(g_{\mathbf{w}}((\boldsymbol{\phi}_1^i, \ldots, \boldsymbol{\phi}_{n_i}^i), \pi_y) - g_{\mathbf{w}}((\boldsymbol{\phi}_1^i, \ldots, \boldsymbol{\phi}_{n_i}^i), \pi) \right) \right]_+ \ ,$$

where π denotes either a permutation of the n_i documents (for NDCG) or a vector of binary assignments to the documents (for MAP, in the case of binary relevance labels y_j^i); π_y denotes a 'true' permutation or binary assignment vector induced by the relevance labels y_j^i; $\varDelta(\pi, \pi_y)$ measures the loss incurred in predicting π instead of π_y (which can be taken to be one minus the NDCG or MAP of π relative to π_y); and $g_{\mathbf{w}}$ is an appropriately defined function that assigns a score to each permutation or binary assignment vector over the documents, and is used to predict such a permutation or assignment vector (rather than directly rank the documents based on the scores $\mathbf{w} \cdot \boldsymbol{\phi}_j^i$ as in our case) via

$$\widehat{\pi} = \arg\max_{\pi} \ \left[g_{\mathbf{w}}((\boldsymbol{\phi}_1^i, \ldots, \boldsymbol{\phi}_{n_i}^i), \pi) \right] \ .$$

See [8, 16] for further details of such approaches.

In contrast, the query-level loss L in our case will be constructed from the simple and intuitive relevance-weighted pair-wise hinge loss $\ell_{\mathsf{H,rel}}$ described above. In the following, we describe three different constructions for L, and in each case give efficient stochastic gradient algorithms to solve the resulting optimization problems.

3 Algorithms

As discussed above, the ranking algorithms we consider learn a linear ranking function $f_{\mathbf{w}} : \mathbb{R}^d \to \mathbb{R}$ by solving an optimization problem of the following form:

$$\min_{\mathbf{w}} \left[\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{m}\sum_{i=1}^{m} L(\mathbf{w}, S^i) \right] ,$$

where $L(\mathbf{w}, S^i)$ denotes a query-level loss function that will be constructed from the relevance-weighted pair-wise hinge loss $\ell_{\mathsf{H,rel}}$ described above.

3.1 Stochastic gradient projection algorithm for average pair-wise loss

The first construction we consider for the query-level loss L is the following **average pair-wise loss:**

$$L_1^{\mathsf{H,rel}}(\mathbf{w}, S^i) = \frac{1}{|R_i|}\sum_{(j,k)\in R_i} \ell_{\mathsf{H,rel}}(\mathbf{w}, (\boldsymbol{\phi}_j^i, y_j^i), (\boldsymbol{\phi}_k^i, y_k^i)) .$$

Notice that in addition to the relevance weighting in $\ell_{\mathsf{H,rel}}$, this differs from the early SVM ranking algorithms described in Section 2 (see Eq. (1)) in that the loss is normalized by query, taking into account different numbers of document pairs for different queries. While one could in principle use stochastic subgradient methods (such as those of [24]) to directly solve the resulting optimization problem, we focus here on a dual version as this will facilitate the development of similar algorithms for the other two loss formulations. Using standard techniques involving the introduction of slack variables, we can write the minimization problem corresponding to the above loss as

$$\min_{\mathbf{w},\boldsymbol{\xi}} \left[\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{m}\sum_{i=1}^{m} \frac{1}{|R_i|} \sum_{(j,k)\in R_i} \xi_{jk}^i \right]$$

subject to

$$\xi_{jk}^i \geq 0 \qquad \forall\, i, j, k$$

$$\xi_{jk}^i \geq (y_j^i - y_k^i) - \mathbf{w}\cdot(\boldsymbol{\phi}_j^i - \boldsymbol{\phi}_k^i) \qquad \forall\, i, j, k.$$

Introducing Lagrange multipliers and taking the Lagrangian dual then results in the following convex quadratic program (QP):

$$\min_{\boldsymbol{\alpha}} \left[\frac{1}{2}\sum_{i=1}^{m} \sum_{(j,k)\in R_i} \sum_{i'=1}^{m} \sum_{(j',k')\in R_{i'}} \alpha_{jk}^i \alpha_{j'k'}^{i'}\, Q_{jk,j'k'}^{i,i'} - \sum_{i=1}^{m} \sum_{(j,k)\in R_i} \alpha_{jk}^i (y_j^i - y_k^i) \right]$$

subject to

$$0 \leq \alpha_{jk}^i \leq \frac{C}{m|R_i|} \qquad \forall\, i, j, k ,$$

where

$$Q_{jk,j'k'}^{i,i'} \;=\; (\boldsymbol{\phi}_j^i - \boldsymbol{\phi}_k^i)\cdot(\boldsymbol{\phi}_{j'}^{i'} - \boldsymbol{\phi}_{k'}^{i'}) .$$

Solving the above QP using standard QP solvers would take $O\big((\sum_{i=1}^m |R_i|)^3 \big)$ time, which is $O(m^3 n^6)$ if $n_i = n$ and $|R_i| = O(n^2)$ for all i. Instead, we use a simple stochastic gradient projection method which starts with some initial values $\boldsymbol{\alpha}^{(1)}$ for $\boldsymbol{\alpha}$, and on each iteration t, randomly selects a single query i and updates the corresponding $|R_i|$ variables ${\alpha^i_{jk}}^{(t)}$ using a gradient and projection step:

$$\begin{aligned} {\boldsymbol{\alpha}^i}^{(t+1)} &\leftarrow \mathcal{P}_{\Omega_i}\Big({\boldsymbol{\alpha}^i}^{(t)} - \eta_t {\nabla^i}^{(t)}\Big)\,, \\ {\boldsymbol{\alpha}^{i'}}^{(t+1)} &\leftarrow {\boldsymbol{\alpha}^{i'}}^{(t)} \;\; \text{for } i' \neq i\,, \end{aligned}$$

where $\eta_t > 0$ is a learning rate; ${\nabla^i}^{(t)} \in \mathbb{R}^{|R_i|}$ is the partial gradient of the objective function in the above QP with respect to $\boldsymbol{\alpha}^i$, evaluated at $\boldsymbol{\alpha}^{(t)}$; $\Omega_i = \{\boldsymbol{\alpha}^i \in \mathbb{R}^{|R_i|} : 0 \leq \alpha^i_{jk} \leq \frac{C}{m|R_i|} \; \forall (j,k) \in R_i\}$ is constraint set for $\boldsymbol{\alpha}^i$ in the above QP; and $\mathcal{P}_{\Omega_i}$ denotes Euclidean projection onto Ω_i. The projection onto the box constraints in Ω_i is straightforward: values of α^i_{jk} outside the interval $[0, \frac{C}{m|R_i|}]$ are simply clipped to the interval. The convergence proof for standard gradient projection methods can be extended to show that if $\eta_t = \frac{\eta_0}{\sqrt{t}}$ for some constant $\eta_0 > 0$, then the above stochastic gradient projection algorithm converges (in expectation) to an optimal solution, and moreover, the number of iterations required to reach a solution whose objective value (in expectation) is within ϵ of the optimal is $O(m^2/\epsilon^2)$; we omit the details for lack of space. An iteration that updates the variables associated with the ith query takes $O(|R_i|)$ time; this leads to a total of $O(m^2 n^2/\epsilon^2)$ time if $n_i = n$ and $|R_i| = O(n^2)$ for all i. On solving the above QP for $\boldsymbol{\alpha}$, the weight vector $\mathbf{w}$ can be recovered from $\boldsymbol{\alpha}$ in a standard manner.

3.2 Stochastic exponentiated gradient algorithm for maximum pair-wise loss

The next construction we consider for L is the following **maximum pair-wise loss:**

$$L_2^{\mathsf{H,rel}}(\mathbf{w}, S^i) = \max_{(j,k)\in R_i} \left[\ell_{\mathsf{H,rel}}(\mathbf{w}, (\boldsymbol{\phi}^i_j, y^i_j), (\boldsymbol{\phi}^i_k, y^i_k))\right] \,.$$

Define the ranking margin of $\mathbf{w}$ on a pair of documents $(j,k) \in R_i$ as $\mathbf{w} \cdot (\boldsymbol{\phi}^i_j - \boldsymbol{\phi}^i_k)$ if $\mathbf{w} \cdot (\boldsymbol{\phi}^i_j - \boldsymbol{\phi}^i_k) < (y^i_j - y^i_k)$, and $(y^i_j - y^i_k)$ otherwise. Then the resulting algorithm in this case will maximize not the average ranking margin over document pairs associated with a query, but rather the minimum ranking margin across all document pairs associated with each query. Again, we can write the corresponding minimization problem as

$$\begin{aligned} &\min_{\mathbf{w},\boldsymbol{\xi}} \left[\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{m}\sum_{i=1}^m \xi^i\right] \\ &\text{subject to} \\ &\xi^i \geq 0 \qquad\qquad\qquad\qquad\qquad\quad \forall\, i \\ &\xi^i \geq (y^i_j - y^i_k) - \mathbf{w}\cdot(\boldsymbol{\phi}^i_j - \boldsymbol{\phi}^i_k) \quad\; \forall\, i,j,k. \end{aligned}$$

Introducing Lagrange multipliers and taking the Lagrangian dual then results in the following convex QP (after an appropriate scaling of variables, and introduction of an additional variable α^i_0 for each i):

$$\min_{\boldsymbol{\alpha}} \left[\frac{1}{2} \sum_{i=1}^{m} \sum_{(j,k)\in R_i} \sum_{i'=1}^{m} \sum_{(j',k')\in R_{i'}} \alpha^i_{jk} \alpha^{i'}_{j'k'} \frac{C}{m} Q^{i,i'}_{jk,j'k'} - \sum_{i=1}^{m} \sum_{(j,k)\in R_i} \alpha^i_{jk} (y^i_j - y^i_k) \right]$$

subject to

$$\alpha^i_0 + \sum_{(j,k)\in R_i} \alpha^i_{jk} = 1 \qquad \forall\, i$$

$$\alpha^i_{jk},\, \alpha^i_0 \geq 0 \qquad\qquad \forall\, i, j, k.$$

The constraints in the above problem force $\boldsymbol{\alpha}^i$ to lie in the simplex Δ_i of distributions over $|R_i| + 1$ elements for each i. This allows us to derive an efficient exponentiated gradient (EG) algorithm in this case which starts with an initial set of distributions $\boldsymbol{\alpha}^{(1)} \in \Delta_1 \times \ldots \times \Delta_m$, and on each iteration t, updates the distribution associated with a single randomly chosen query i using an exponentiated gradient step:

$$\alpha^{i\;(t+1)}_{jk} \leftarrow \frac{\alpha^{i\;(t)}_{jk} \exp(-\eta_0 \nabla^{i\;(t)}_{jk})}{Z^{(t)}} \;; \quad \alpha^{i\,(t+1)}_0 \leftarrow \frac{\alpha^{i\,(t)}_0}{Z^{(t)}} \,,$$

$$\boldsymbol{\alpha}^{i'(t+1)} \leftarrow \boldsymbol{\alpha}^{i'(t)} \text{ for } i' \neq i \,,$$

where $\eta_0 > 0$ is a constant learning rate; $\nabla^{i\;(t)}_{jk}$ is the partial derivative of the objective in the above QP with respect to α^i_{jk}, evaluated at $\boldsymbol{\alpha}^{(t)}$; and $Z^{(t)}$ is chosen to ensure $\boldsymbol{\alpha}^{i\,(t+1)} \in \Delta_i$. The algorithm can actually be implemented in a way that requires only $O(n_i)$ time per iteration rather than $O(|R_i|)$ time using ideas developed for structured prediction problems [22]; we omit the details for lack of space. As in [22], the resulting algorithm can be shown to converge (in expectation) to an optimal solution; if $n_i = n$ for all i, then the time required to reach a solution whose objective value (in expectation) is within ϵ of the optimal is $O\left(\frac{mn}{\epsilon}\left(|A|_\infty D[\boldsymbol{\alpha}^* \| \boldsymbol{\alpha}^{(1)}] + Q(\boldsymbol{\alpha}^{(1)}) - Q(\boldsymbol{\alpha}^*)\right)\right)$, where $\boldsymbol{\alpha}^{(1)}, \boldsymbol{\alpha}^*$ denote the initial and optimal sets of distributions, respectively; $Q(\boldsymbol{\alpha})$ denotes the objective function in the above problem; $D[\boldsymbol{\alpha}^* \| \boldsymbol{\alpha}^{(1)}]$ denotes the sum of the Kullback-Leibler divergences between $\boldsymbol{\alpha}^{i*}$ and $\boldsymbol{\alpha}^{i\,(1)}$ over all i, and $|A|_\infty$ is the largest entry in the matrix whose entries are $A_{(i,j,k),(i',j',k')} = \frac{C}{m} Q^{i,i'}_{jk,j'k'}$.

3.3 Stochastic gradient projection algorithm for maximum-average pair-wise loss

In taking the average or maximum pair-wise ranking loss as above, one does not distinguish ranking errors at the top of the list from ranking errors at the bottom. However, in practice, and in IR in particular, ranking errors at the top of the list are often more costly (for example, in web search, the accuracy of the first few web pages returned by a search engine is paramount). To this end, we consider a hybrid **maximum-average pair-wise loss** construction for L:

$$L^{\mathsf{H,rel}}_3(\mathbf{w}, S^i) = \max_{k:|P_{ik}|>0} \frac{1}{|P_{ik}|} \sum_{j\in P_{ik}} \ell_{\mathsf{H,rel}}(\mathbf{w}, (\boldsymbol{\phi}^i_j, y^i_j), (\boldsymbol{\phi}^i_k, y^i_k)) \,,$$

where $P_{ik} = \left\{ j \mid y^i_j > y^i_k \right\}$ denotes the set of documents that are preferred to d^i_k. To see why this loss term might penalize ranking errors at the top more heavily than ranking errors at the bottom, note that the cost of each 'mis-ranking up' of a document is inversely weighted by the number of documents preferred to it; therefore, 'mis-ranking up' a lower-relevance document by a few positions is less costly than 'mis-ranking up' a higher-relevance document. By minimizing the largest such 'mis-ranking up' cost over all documents, the resulting algorithm should therefore discourage mis-ranking of higher-relevance documents, resulting in good accuracy at the top of the returned ranking. Indeed, the above loss is reminiscent of the l_p-norm based loss studied in [25], where a greater value of p corresponds to a greater push toward more accurate ranking performance at the top of the returned list; the above loss can be viewed as an l_∞ extreme, using the relevance-weighted hinge loss instead of the (binary) exponential loss used in [25]. Introducing slack variables as before, the corresponding minimization problem can be written as

$$\min_{\mathbf{w},\boldsymbol{\xi}} \left[\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{m}\sum_{i=1}^{m} \xi^i \right]$$

subject to

$$\begin{aligned} &\xi^i \geq \frac{1}{|P_{ik}|} \sum_{j \in P_{ik}} \xi^i_{jk} && \forall\, i,k \\ &\xi^i_{jk} \geq 0 && \forall\, i,j,k \\ &\xi^i_{jk} \geq (y^i_j - y^i_k) - \mathbf{w}\cdot(\boldsymbol{\phi}^i_j - \boldsymbol{\phi}^i_k) && \forall\, i,j,k. \end{aligned}$$

Introducing Lagrange multipliers and taking the Lagrangian dual then results in the following convex optimization problem (after an appropriate scaling of variables):

$$\min_{\boldsymbol{\alpha},\boldsymbol{\gamma}} \left[\frac{1}{2} \sum_{i=1}^{m} \sum_{(j,k)\in R_i} \sum_{i'=1}^{m} \sum_{(j',k')\in R_{i'}} \alpha^i_{jk} \alpha^{i'}_{j'k'} \frac{Q^{i,i'}_{jk,j'k'}}{|P_{ik}|\,|P_{i'k'}|} - \sum_{i=1}^{m} \sum_{(j,k)\in R_i} \alpha^i_{jk} \frac{(y^i_j - y^i_k)}{|P_{ik}|} \right]$$

subject to

$$\begin{aligned} &0 \leq \alpha^i_{jk} \leq \gamma^i_k && \forall\, i,j,k \\ &\sum_{k:|P_{ik}|>0} \gamma^i_k = \frac{C}{m} && \forall\, i. \end{aligned}$$

The constraints in this case can be interpreted as a set of constraints on the $l_{1,\infty}$-norm of $\boldsymbol{\alpha}^i$ for each i, together with non-negativity constraints. Quattoni et al. [21] recently developed an efficient algorithm for $l_{1,\infty}$-norm projections; this allows us to use a stochastic gradient projection method similar to that discussed for the average pair-wise loss in Section 3.1. In this case, each projection step consists of a projection onto the $l_{1,\infty}$ constraints using the method of [21], followed by a projection onto the non-negativity constraints (which simply involves setting negative values of α^i_{jk} to zero). The number of iterations to reach an ϵ-optimal solution (in expectation) is $O(m^2/\epsilon^2)$ as before; each iteration now takes $O(|R_i|\log|R_i|)$ time owing to the projection, thus leading to a total of $O(m^2n^2\log n/\epsilon^2)$ time if $n_i = n$ and $|R_i| = O(n^2)$ for all i.

4 Experiments

In preliminary experiments, we evaluated our algorithms on the OHSUMED data set, a benchmark data set for IR ranking algorithms available publicly as part of the LETOR distribution[1] [26] (we used LETOR 3.0). The data set consists of 106 medical queries. Each query is associated with a number of documents, each of which has been judged by human experts as being either definitely relevant to the query (label 2), partially relevant (label 1), or not relevant (label 0); there are a total of 16,140 such query-document pairs with relevance judgments (an average of roughly 152 judged documents per query). Each query-document pair is represented as a vector of 45 features.

There are five folds provided in the data set; each fold consists of a split of the queries into roughly 60% for training, 20% for validation, and 20% for testing. We evaluated our algorithms on these five folds and compared their performance with several other ranking algorithms for which results are available as baselines in the LETOR distribution. The following performance measures were used to evaluate the algorithms:

1. **NDCG@k**: The NDCG@k of a ranking function f on a query q with n documents $d_1, \ldots, d_n$, associated with relevance labels $y_1, \ldots, y_n$, is the NDCG (which is simply NDCG@n) [27] truncated to the top k documents returned by f:

$$\text{NDCG@}k[f] \;=\; \frac{1}{Z_k} \sum_{j=1}^{k} \frac{2^{y_{\sigma(j)}} - 1}{\text{discount}(j)}\,,$$

where $\sigma(j)$ denotes the index of the document ranked at the jth position by f; discount(j) is a discounting factor that discounts the contribution of documents ranked lower in the list; and Z_k is a constant that ensures the maximum NDCG@k over all rankings is 1. In the LETOR evaluation tool, discount(j) is defined to be 1 if $j = 1$, and $\log_2(j)$ otherwise.

2. **Prec@k**: For binary labels, where a label of 1 is considered as relevant and 0 as irrelevant, the Prec@k of a ranking function f on a query q as above is the proportion of relevant documents in the top k documents returned by f:

$$\text{Prec@}k[f] \;=\; \frac{1}{k} \sum_{j=1}^{k} \mathbf{1}(y_{\sigma(j)} = 1)\,,$$

where $\mathbf{1}(\cdot)$ is an indicator function whose value is 1 if its argument is true and 0 otherwise. For the OHSUMED data, relevance labels of 1 and 2 are considered relevant, and 0 irrelevant.

3. **MAP**: The average precision (AP) of a ranking function f on a query q as above is the average Prec@k over all positions k occupied by relevant documents:

$$\text{AP}[f] \;=\; \frac{\sum_{k=1}^{n} \text{Prec@}k[f] \cdot \mathbf{1}(y_{\sigma(j)} = 1)}{\sum_{k=1}^{n} \mathbf{1}(y_{\sigma(j)} = 1)}\,.$$

The mean average precision (MAP) refers to the mean AP across a set of queries.

The results are shown in Figure 1; here the algorithms of Sections 3.1, 3.2, and 3.3 are referred to as RankMM-1, RankMM-2, and RankMM-3, respectively. For comparison, we also show results for the following algorithms (all obtained from the LETOR

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

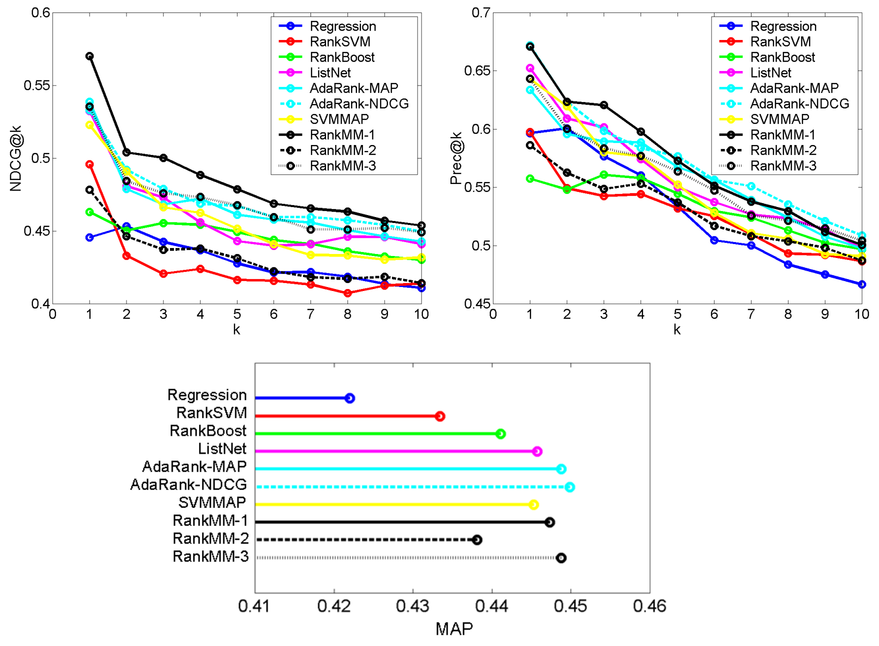

Fig. 1. Results on the OHSUMED data set in terms of (**top left**) NDCG@k; (**top right**) Prec@k; and (**bottom**) MAP. See text for details.

website): regression, RankSVM [2, 19], RankBoost [20], ListNet [14], two versions of AdaRank [7], and SVMMAP [8]. All results shown are averages across the five folds. For each fold, the regularization parameter C, learning rate η_0, and number of iterations T in our algorithms were selected from the ranges $\{0.1, 1, 10, 100, 1000\}$, $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$, and $\{100, 250, 500, 750, 1000\}$, respectively; in particular, for consistency with the other results reported in LETOR, the parameters that gave the highest MAP on the validation set were used in each case for training.

Of the algorithms shown for comparison, regression is a point-wise algorithm that predicts labels of individual documents; RankSVM and RankBoost are pair-wise ranking algorithms; and the remainder are list-wise ranking algorithms, with the last three directly optimizing the MAP or NDCG. Other than RankBoost, which uses thresholded features as weak rankers, all algorithms learn a linear ranking function. As can be seen, the performance of our algorithms, particularly RankMM-1 and RankMM-3, is considerably superior to the standard pair-wise (and point-wise) ranking algorithms[2], and indeed, in many cases, is comparable to the performance of the best algorithms that directly optimize the MAP or NDCG. We note that RankMM-2 appears not to be as suited to IR performance measures; RankMM-3 appears to be particularly suited to MAP. The best overall performance (for this data set) is obtained using RankMM-1.

[2] It was recently found that RankSVM, with proper training, yields better results than those reported in LETOR [28]. Our algorithms show improvement over these new results as well.

5 Conclusion and Open Questions

We have proposed a family of ranking algorithms for IR that employ loss functions derived from a pair-wise ranking loss, yet show performance comparable to a number of algorithms that have been proposed recently for optimizing IR ranking measures such as the MAP and NDCG. Our two best performing algorithms are both stochastic gradient projection algorithms, one of which requires $l_{1,\infty}$-norm projections, for which we use a method of [21]; the third is a stochastic exponentiated gradient (EG) algorithm.

There are several open questions regarding ranking algorithms in IR. The relationships between different ranking methods and performance measures are still not clearly understood; for example, it would be of interest to study statistical convergence properties of these algorithms, as has been done for example in [11]. Another practical issue, given the scale of many IR applications, is efficiency. We have obtained an $O(1/\epsilon^2)$ rate of convergence to an ϵ-optimal solution for our stochastic gradient projection algorithms, and an $O(1/\epsilon)$ rate for the EG algorithm. We expect it may be possible to obtain an $O(1/\epsilon)$ rate for the projection algorithms as well; we leave this to future work.

Finally, a larger scale comparison of different ranking algorithms is required to better understand their respective merits and shortcomings. Unfortunately obtaining appropriate data sets for this purpose that include more than two relevance levels has been a challenge due to their mostly proprietary nature. We hope to evaluate these algorithms on larger scale web search data in the near future.

Acknowledgments. This work was supported in part by the National Science Foundation (NSF) under Grant No. DMS-0732334. Any opinions, findings, and conclusions or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the NSF.

References

1. Herbrich, R., Graepel, T., Bollmann-Sdorra, P., Obermayer, K.: Learning preference relations for information retrieval. In: Proceedings of the ICML-1998 Workshop on Text Categorization and Machine Learning. (1998)
2. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the 8th ACM Conference on Knowledge Discovery and Data Mining. (2002)
3. Burges, C.J.C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: Proceedings of the 22nd International Conference on Machine Learning. (2005)
4. Cao, Y., Xu, J., Liu, T.Y., Li, H., Hunag, Y., Hon, H.W.: Adapting ranking SVM to document retrieval. In: Proceedings of the 29th ACM SIGIR Conference on Research and Development in Information Retrieval. (2006)
5. Burges, C.J.C., Ragno, R., Le, Q.V.: Learning to rank with non-smooth cost functions. In: Advances in Neural Information Processing Systems 19, MIT Press (2007)
6. Tsai, M.F., Liu, T.Y., Qin, T., Chen, H.H., Ma, W.Y.: FRank: A ranking method with fidelity loss. In: Proceedings of the 30th ACM SIGIR Conference on Research and Development in Information Retrieval. (2007)
7. Xu, J., Li, H.: AdaRank: A boosting algorithm for information retrieval. In: Proceedings of the 30th ACM SIGIR Conference on Research and Development in Information Retrieval. (2007)

8. Yue, Y., Finley, T., Radlinski, F., Joachims, T.: A support vector method for optimizing average precision. In: Proceedings of the 30th ACM SIGIR Conference on Research and Development in Information Retrieval. (2007)
9. Taylor, M., Guiver, J., Robertson, S., Minka, T.: Softrank: optimizing non-smooth rank metrics. In: Proceedings of the 1st ACM International Conference on Web Search and Data Mining. (2008)
10. Chakrabarti, S., Khanna, R., Sawant, U., Bhattacharyya, C.: Structured learning for non-smooth ranking losses. In: Proceedings of the 14th ACM Conference on Knowledge Discovery and Data Mining. (2008)
11. Cossock, D., Zhang, T.: Statistical analysis of bayes optimal subset ranking. IEEE Transactions on Information Theory **54**(11) (2008) 5140–5154
12. Chapelle, O., Wu, M.: Gradient descent optimization of smoothed information retrieval metrics. Information Retrieval Journal (To appear) (2010)
13. Qin, T., Zhang, X.D., Tsai, M.F., Wang, D.S., Liu, T.Y., Li, H.: Query-level loss functions for information retrieval. Information Processing and Management **44**(2) (2008) 838–855
14. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: From pairwise approach to listwise approach. In: Proceedings of the 24th International Conference on Machine Learning. (2007)
15. Joachims, T.: A support vector method for multivariate performance measures. In: Proceedings of the 22nd International Conference on Machine Learning. (2005)
16. Chapelle, O., Le, Q., Smola, A.: Large margin optimization of ranking measures. In: Proceedings of the NIPS-2007 Workshop on Machine Learning for Web Search. (2007)
17. Taskar, B., Guestrin, C., Koller, D.: Max-margin markov networks. In: Advances in Neural Information Processing Systems 16, MIT Press (2004)
18. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research (JMLR) **6** (2005) 1453–1484
19. Herbrich, R., Graepel, T., Obermayer, K.: Large margin rank boundaries for ordinal regression. Advances in Large Margin Classifiers (2000) 115–132
20. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research **4** (2003) 933–969
21. Quattoni, A., Carreras, X., Collins, M., Darrell, T.: An efficient projection for $l_{1,\infty}$ regularization. In: Proceedings of the 26th International Conference on Machine Learning. (2009)
22. Collins, M., Globerson, A., Koo, T., Carreras, X., Bartlett, P.: Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. Journal of Machine Learning Research **9** (2008) 1775–1822
23. Agarwal, S., Niyogi, P.: Generalization bounds for ranking algorithms via algorithmic stability. Journal of Machine Learning Research **10** (2009) 441–474
24. Shalev-Shwartz, S., Singer, Y., Srebro, N.: Pegasos: Primal estimated sub-gradient solver for SVM. In: Proceedings of the 24th International Conference on Machine Learning. (2007)
25. Rudin, C.: Ranking with a p-norm push. In: Proceedings of the 19th Annual Conference on Learning Theory. (2006)
26. Liu, T.Y., Xu, J., Qin, T., Xiong, W., Li, H.: LETOR: Benchmark dataset for research on learning to rank for information retrieval. In: Proceedings of the SIGIR-2007 Workshop on Learning to Rank for Information Retrieval. (2007)
27. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems **20**(4) (2002) 422–446
28. Chapelle, O., Keerthi, S.S.: Efficient algorithms for ranking with SVMs. Information Retrieval Journal (To appear) (2010)