Chapter 1
Efficient Learning of Sparse Ranking Functions

Mark Stevens, Samy Bengio, Yoram Singer

Abstract Algorithms for learning to rank can be inefficient when they employ risk functions that use structural information. We describe and analyze a learning algorithm that efficiently learns a ranking function using a domination loss. This loss is designed for problems in which we need to rank a small number of positive examples over a vast number of negative examples. In that context, we propose an efficient coordinate descent approach that scales *linearly* with the number of examples. We then present an extension that incorporates regularization thus extending Vapnik's notion of regularized empirical risk minimization to ranking learning. We also discuss an extension to the case of multi-values feedback. Experiments performed on several benchmark datasets and large scale Google internal dataset demonstrate the effectiveness of learning algorithm in constructing compact models while retaining the empirical performance accuracy.

1.1 Introduction

The past decade's proliferation of search engines and online advertisements has underscored the need for accurate yet efficiently computable ranking functions. Moreover, the emergence of personalized search and targeted advertisement further emphasizes the need for efficient algorithms that can generate a plethora of ranking functions which are used in tandem at serving time. The focus of this paper is the derivation of such an efficient learning algorithm that yields compact ranking functions while achieving competitive accuracy. Before embarking with a description of our approach we would like to make connections to existing methods that influenced our line of research as well as briefly describe alternative methods for learning to rank. Due to the space constraints and the voluminous amount of work on this subject, we clearly cannot give a comprehensive overview. The home pages of two

Google Research, e-mail: {singer,stevensm,bengio}@google.com

recent workshops at NIPS'09, "Learning to Rank" and "Learning with Orderings", are good sources of information on different learning to rank methods and analyses.

The roots of learning to rank go back to the early days of information retrieval (IR), such as the classical IR described by Gerard Salton [10]. One of the early papers to cast the ranking task as a learning problem is "Learning to Order Things" [2]. While the learning algorithm presented in this paper is rather naive as it encompassed the notion of near-perfect ranking "experts", it laid some of the foundations later used by more effective algorithms such as RankSVM [6], RankBoost [4], and PAMIR [5]. These three algorithms, and many other algorithms, reduced the ranking problem into preference learning over pairs. This reduction enabled the usage of existing learning tools with matching generalization analysis that stem from Valadimir Vapnik's work [13, 14, 15]. However, the reduction to pairs of instances may result in poor ranking accuracy when the ranking objective is not closely related to the pairs' preference objective. Moreover, the usage of pairs of instances can impose computational burdens in large ranking tasks. The deficiency of preference based approaches sparked research that tackles non-linear and often non-convex ranking loss functions, see for instance [16, 1, 7]. These more recent approaches resulted in improved results. However, they are typically computationally expensive, may converge to a local optimum [1], or are tailored for a specific setting [7]. Moreover, most learning to rank algorithms do not include a natural mechanism for controlling the compactness of the ranking function.

In this paper we use a loss function called the domination loss [3]. To make this loss applicable to different settings, we extend and generalize the loss by incorporating the notion of margin [14, 15] over pairs of instances and enable the usage of multi-valued feedback. We devise a simple yet effective coordinate descent algorithm that is guaranteed to converge to the unique optimal solution, see for instance [12, 8] for related convergence proofs. Although the domination loss is expressed in terms of ordering relations over pairs, by using a bound optimization technique we are able to decompose each coordinate descent step so that the resulting update scales *linearly* with the number of instances that we rank. Furthermore, we show how to incorporate an ℓ_1 regularization term into the objective and the descent process. This term promotes sparse ranking functions. We present empirical results which demonstrate the effectiveness of our approach in building compact ranking functions whose performance matches the state-of-the-art results.

1.2 Problem Setting

We start by establishing the notation used throughout the paper. Vectors are denoted in bold face, e.g, $\mathbf{x}$, and are assumed to be in column orientation. The transpose of a vector is denoted $\mathbf{x}^\dagger$. The (vector) expectation of a set of vectors $\{\mathbf{x}_i\}$ with respect to a discrete distribution $\mathbf{p}$ is denoted, $\mathbb{E}_{\mathbf{p}}[\mathbf{x}] = \sum_j p_j \mathbf{x}_j$. We observe a set of instances (e.g. images) where each instance is represented as a vector in $\mathbb{R}^n$. The i^{th} instance, denoted $\mathbf{x}_i$, is associated with a quality feedback, denoted $\tau_i \in \mathbb{R}$. We describe our derivation for a single query as the extension to multiple queries is straightforward.

Given the feedback set for instances observed for a query, we wish to learn a *ranking* function for that query, $f : \mathbb{R}^n \to \mathbb{R}$, that is consistent with the feedback as often as possible. Concretely, we would like the function f to induce an ordering such that $\tau_i > \tau_j \Rightarrow f(\mathbf{x}_i) > f(\mathbf{x}_j)$. We use the domination loss as a surrogate convex loss function to promote the ordering requirements. For an instance $\mathbf{x}_i$ we denote by $\mathscr{D}(i)$ the set of instances that should be ranked below it according to the feedback, $\mathscr{D}(i) = \left\{ j \,|\, \tau_i > \tau_j \right\}$. The *combinatorial* domination loss is one if there exists $j \in \mathscr{D}(i)$ such that $f(\mathbf{x}_j) > f(\mathbf{x}_i)$. That is, the requirement that the i^{th} instance dominates all the instances in $\mathscr{D}(i)$ is violated. To alleviate the intractability problems that arise when using a combinatorial loss, we use the following convex relaxation for a single instance,

$$\ell_{\mathscr{D}}(\mathbf{x}_i; f) = \log\left(1 + \sum_{j \in \mathscr{D}(i)} e^{f(\mathbf{x}_j) - f(\mathbf{x}_i) + \Delta(i,j)}\right) \ . \tag{1.1}$$

Here, $\Delta(i, j) \in \mathbb{R}_+$ denotes a margin requirement between the i^{th} and the j^{th} instances. This function enables us to express richer relaxation requirements which are often necessary in retrieval applications. For example, for the query `dog` we would like an image with a single nicely captured dog to attain a large margin over irrelevant images. In contrast, an image with multiple animals, including dogs, should attain only a modest margin over the irrelevant images.

1.3 An Efficient Coordinate Descent Algorithm

In this section we focus on a special case in which $\tau_i \in \{-1, +1\}$. That is, each instance is either positive (good, relevant) or negative (bad, irrelevant). In the next section we discuss generalizations. In this restricted case, the set $\{j \text{ s.t. } \exists i : \mathbf{x}_j \in \mathscr{D}(i)\}$ simply amounts to the set of negatively labeled instances and does not depend on the index i. For brevity we drop the dependency on i and simply denote it $\mathscr{D}$. We further simplify the learning setting and assume that $\Delta(i, j) = 0$. Again, we discuss relaxations of this assumption in the next section. The ranking function f is restricted to the class of linear functions, $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x}$. In this base ranking setting the empirical loss with respect to $\mathbf{w}$ distills to, $\sum_i \log\left(1 + \sum_{j \in \mathscr{D}} e^{\mathbf{w} \cdot \left(\mathbf{x}_j - \mathbf{x}_i\right)}\right)$. For brevity let us focus on a single domination-loss term, $\ell_{\mathscr{D}}(\mathbf{x}_i; \mathbf{w})$. Fixing the query q and performing simple algebraic manipulations we get, $\ell_{\mathscr{D}}(\mathbf{x}_i; \mathbf{w}) = \log(\frac{\sum_{j \in \bar{\mathscr{D}}(i)} e^{\mathbf{w} \cdot \mathbf{x}_j}}{e^{\mathbf{w} \cdot \mathbf{x}_i}}) = \log(\sum_{j \in \bar{\mathscr{D}}(i)} e^{\mathbf{w} \cdot \mathbf{x}_j}) - \mathbf{w} \cdot \mathbf{x}_i$, where $\bar{\mathscr{D}}(i) = \mathscr{D} \cup \{i\}$ consists of all the negatively labeled instances and the i^{th} relevant instance.

There is no closed form solution for the optimum even when we restrict ourselves to a single coordinate of $\mathbf{w}$. We therefore use a bound optimization technique [9] by constructing a quadratic upper bound on the domination loss. This technique

```
initialize:   $Z = m, \ \forall i : \rho_i = 0, \ \forall r : B_r = \max_j x_{j,r}^2$
while not converged do
  for $r \in \{1, ..., n\}$ do
    $\nu_r = 0; \mu_r = 0$
    for $j \in \mathscr{D}$ do
      $\mu_r \leftarrow \mu_r + e^{\rho_j} x_{jr}$
    end for
    for each $i \notin D$ do
      $\nu_r \leftarrow \nu_r + (Z x_{ir} - \mu_r)/(Z + e^{\rho_i})$
    end for
    $\delta_r \leftarrow \nu_r/(mB_r)$ ; $w_r \leftarrow w_r + \delta_r$
    for each $j \in D$ do
      $Z \leftarrow Z - e^{\rho_j}$ ; $\rho_j \leftarrow \rho_j + \delta_r x_{j,r}$ ; $Z \leftarrow Z + e^{\rho_j}$
    end for
    for each $i \notin D$ do
      $\rho_i \leftarrow \rho_i + \delta_r x_{i,r}$
    end for
  end for
end while
```

Fig. 1.1 Coordinate descent algorithm for ranking-learning with the domination loss.

was used for instance in the context of boosting-style algorithms for regression. To construct the upper bound we need to calculate the gradient and the Hessian of the (simplified) domination loss. The gradient amounts to

$$\nabla_{\mathbf{w}} \ell_{\mathscr{D}}(\mathbf{x}_i; \mathbf{w}) = \frac{\sum_{j \in \bar{\mathscr{D}}(i)} e^{\mathbf{w} \cdot \mathbf{x}_j} \mathbf{x}_j}{\sum_{j \in \bar{\mathscr{D}}(i)} e^{\mathbf{w} \cdot \mathbf{x}_j}} - \mathbf{x}_i = \mathbb{E}_{\mathbf{p}_i}[\mathbf{x}] - \mathbf{x}_i \ ,$$

where $\mathbf{p}_i$ is the distribution induced by $\mathbf{w}$ whose r^{th} component is $p_r = e^{\mathbf{w} \cdot \mathbf{x}_r}/\bar{Z}$ and $\bar{Z}$ is a normalization constant $\bar{Z} = \sum_{j \in \bar{\mathscr{D}}(i)} e^{\mathbf{w} \cdot \mathbf{x}_j}$. Using the above the Hessian is,

$$H_i(\mathbf{w}) = \mathbb{E}_{\mathbf{p}_i}\left[\mathbf{x}\mathbf{x}^\dagger\right] - \mathbb{E}_{\mathbf{p}_i}[\mathbf{x}]\left(\mathbb{E}_{\mathbf{p}_i}[\mathbf{x}]\right)^\dagger \ .$$

Recalling the mean value theorem, the loss at $\mathbf{w} + \delta$ can now be written as,

$$\begin{aligned} \ell_{\mathscr{D}}(\mathbf{x}_i; \mathbf{w} + \delta) &= \ell_{\mathscr{D}}(\mathbf{x}_i; \mathbf{w}) + \nabla_{\mathbf{w}} \cdot \delta + \frac{1}{2} \delta^\dagger H_i(\mathbf{w} + \alpha\delta)\, \delta \\ &= \ell_{\mathscr{D}}(\mathbf{x}_i; \mathbf{w}) + (\mathbb{E}_{\mathbf{p}_i}[\mathbf{x}] - \mathbf{x}_i) \cdot \delta + \frac{1}{2} \mathbb{E}_{\tilde{\mathbf{p}}_i}\left[(\delta \cdot \mathbf{x})^2\right] - \frac{1}{2}(\delta \cdot \mathbb{E}_{\tilde{\mathbf{p}}_i}[\mathbf{x}])^2 \ , \end{aligned}$$

where $\tilde{\mathbf{p}}_i$ is the distribution induced at $\mathbf{w} + \alpha\delta$ for an unknown $\alpha \in [0, 1]$. We now derive an update which minimizes the bound along a single coordinate, denoted r, of $\mathbf{w}$. Let $\mathbf{e}_r$ denote the vector whose components are 0 except for the r^{th} component which is 1, then,

$$\ell_{\mathscr{D}}(\mathbf{x}_i; \mathbf{w} + \delta \mathbf{e}_r) = \kappa(i) + \delta\left(\mathbb{E}_{\bar{\mathscr{D}}(i)}[\mathbf{p}, \mathbf{x}_r] - x_{i,r}\right) + \frac{1}{2}\delta^2(\mathbb{E}_{\bar{\mathscr{D}}(i)}[\tilde{\mathbf{p}}, \mathbf{x}_r^2] - \mathbb{E}^2_{\bar{\mathscr{D}}(i)}[\tilde{\mathbf{p}}, \mathbf{x}_r]) \ ,$$

where $\mathbb{E}_{\bar{\mathscr{D}}(i)}[\mathbf{p}, \mathbf{x}_r] = \sum_{j \in \bar{\mathscr{D}}(i)} p_j x_{j,r}$. where $\kappa(i)$ is a constant that does not depend on δ. Since $\tilde{\mathbf{p}}_j$ is not known we need to bound the loss further. Let B_r denote the maximum value of the square of the r^{th} feature over the instances retrieved, namely $B_r = \max_{j \in \bar{\mathscr{D}}} x_{j,r}^2$. We now can bound the multiplier of $\frac{1}{2}\delta^2$ by $\sum_{j \in \bar{\mathscr{D}}(i)} \tilde{p}_j x_{j,r}^2 \leq B_r$. Let m denote the number of relevant training elements for the current query. The bound of the domination loss restricted to coordinate r is, $L(\mathbf{w} + \delta \mathbf{e}_r) \leq \kappa + \delta \sum_{i \notin \mathscr{D}} \left(\sum_{j \in \bar{\mathscr{D}}(i)} p_j x_{j,r} - x_{i,r} \right) + \frac{1}{2} B_r \delta^2 m$. The last term is a simple quadratic term in δ and thus the optimal step $\delta^\star$ along coordinate r can be trivially computed. Alas, our derivation of an efficient coordinate descent update does not end here as the term multiplying δ in the bound above depends on the pairs $i \notin \mathscr{D}$ and $j \in \bar{\mathscr{D}}(i)$. We now exploit the fact that the sole instance in $\bar{\mathscr{D}}(i)$ which depends on i is $\mathbf{x}_i$ itself and rewrite the gradient, ν_r, with respect to r,

$$\begin{aligned}
\nu_r + \sum_{i \notin \mathscr{D}} x_{i,r} &= \sum_{i \notin \mathscr{D}} \sum_{j \in \bar{\mathscr{D}}(i)} p_j x_{j,r} \;=\; \sum_{i \notin \mathscr{D}} \left(\sum_{j \in \mathscr{D}} p_j x_{j,r} + p_i x_{i,r} \right) \\
&= \sum_{i \notin \mathscr{D}} \sum_{j \in \mathscr{D}} \frac{e^{\mathbf{w} \cdot \mathbf{x}_j}}{Z + e^{\mathbf{w} \cdot \mathbf{x}_i}} x_{j,r} \;+\; \sum_{i \notin \mathscr{D}} \frac{e^{\mathbf{w} \cdot \mathbf{x}_i}}{Z + e^{\mathbf{w} \cdot \mathbf{x}_i}} x_{i,r} \\
&= \left(\sum_{i \notin \mathscr{D}} \frac{1}{Z + e^{\mathbf{w} \cdot \mathbf{x}_i}} \right) \left(\sum_{j \in \mathscr{D}} e^{\mathbf{w} \cdot \mathbf{x}_j} x_{j,r} \right) \;+\; \sum_{i \notin \mathscr{D}} \frac{e^{\mathbf{w} \cdot \mathbf{x}_i} x_{i,r}}{Z + e^{\mathbf{w} \cdot \mathbf{x}_i}} \;,
\end{aligned}$$

where $Z = \sum_{j \in \mathscr{D}} e^{\mathbf{w} \cdot \mathbf{x}_j}$. The latter expression can be computed in time linear in the size (number of instances) of each query, and *not* the number of comparable pairs in each query. Finally, to alleviate the dependency in the dimension of the instances, due to the products $\mathbf{w} \cdot \mathbf{x}_i$, on each iteration, we introduce the following variables $\rho_i = \mathbf{w} \cdot \mathbf{x}_i$ and $\mu_r = \sum_{j \in \mathscr{D}} e^{\rho_j} x_{j,r}$. Then, the change to the r^{th} coordinate of $\mathbf{w}$ is,

$$\delta^\star = \frac{1}{m B_r} \sum_{i \notin \mathscr{D}} \frac{Z x_{i,r} - \mu_r}{Z + e^{\rho_i}} \quad . \tag{1.2}$$

To recap we provide the pseudo-code of the algorithm in Fig. 1.1.

1.4 Extensions

In this section we describe a few generalizations and extensions of the base coordinate descent algorithm described in the previous section. Concretely, we show a generalization of the algorithm to multi-valued feedback, we describe the addition of margin values over pairs of elements, and we define the incorporation of regularization throughout the course of the algorithm.

We start with the extension to multi-valued feedback. In the more general case, the feedback can take one of K predefined values. We can assume without loss of generality that the feedback set is $\{1, \ldots, K\}$. We thus can divide the instances retrieved for a query into k disjoint sets denoted $G(1),\ldots,G(K)$. The set $G(K)$ consists of all the top ranked instances and analogously $G(1)$ contains all the bottom ranked elements. The set of instances dominated by any instance in $G(r)$ is denoted by $\mathscr{D}(r) = \{j \in G(l) | l < r\}$. The form of the bound for $\ell_{\mathscr{D}}(\mathbf{x}_i;\mathbf{w})$ with multi-valued feedback remains intact. When summing over all dominating x_i, however, we break down the summation by group k, yielding,

$$L(\mathbf{w}+\delta\mathbf{e}_r) \leq \kappa + \textstyle\sum_{k>1}\sum_{i\in G(k)} \left(\sum_{j\in\bar{\mathscr{D}}(i)} p_j x_{j,r} - x_{i,r}\right)\delta \;+\; \frac{1}{2}B_r \sum_{k>1} m(k)\delta^2 \qquad (1.3)$$

where $m(k)$ is the number of elements in group k. The quadratic term can still be bounded by $mB_r/2$, where we redefine m to be number of dominating elements, $\sum_{k>1} m(k)$. Again, efficient computation requires decomposing the linear multiplier. Using the same argument as earlier we get,

$$\sum_{k>1}\sum_{i\in G(k)}\sum_{j\in\bar{\mathscr{D}}(i)} p_j x_{j,r} = \sum_{k>1}\sum_{i\in G(k)} \frac{1}{Z_k + e^{\mathbf{w}\cdot\mathbf{x}_i}} \sum_{j\in\mathscr{D}(k)} e^{\mathbf{w}\cdot\mathbf{x}_j} x_{j,r} + \sum_{k>1}\sum_{i\in G(k)} \frac{e^{\mathbf{w}\cdot\mathbf{x}_i} x_{i,r}}{Z_k + e^{\mathbf{w}\cdot\mathbf{x}_i}} \;,$$

where $Z_k = \sum_{j\in\mathscr{D}(k)} e^{\mathbf{w}\cdot\mathbf{x}_j}$. The gradient for the multi-valued feedback amounts to,

$$\sum_{k>1}\sum_{i\in G(k)} \left(\sum_{j\in\bar{\mathscr{D}}(i)} p_j x_{j,r} - x_{i,r}\right) = \sum_{k>1}\sum_{i\in G(k)} \frac{\mu_{k,r} - Z_k x_{i,r}}{Z_k + e^{\mathbf{w}\cdot\mathbf{x}_i}} \;,$$

and $\mu_{k,r} = \sum_{j\in\mathscr{D}(k)} e^{\mathbf{w}\cdot\mathbf{x}_j} x_{j,r}$. Note that Z and μ can be constructed recursively in linear time as follows, $Z_k = \sum_{j\in G(k-1)} e^{\mathbf{w}\cdot\mathbf{x}_j} + Z_{k-1}$ and $\mu_{k,r} = \sum_{j\in G(k-1)} e^{\mathbf{w}\cdot\mathbf{x}_j} x_{j,r} + \mu_{k-1,r}$. To recap, each generalized update of δ is computed as follows,

$$\delta = \frac{1}{mB_r} \sum_{k>1}\sum_{i\in G(k)} \frac{Z_k x_{i,r} - \mu_{k,r}}{Z_k + e^{\mathbf{w}\cdot\mathbf{x}_i}} \;.$$

Next we discuss the infusion of margin requirements which distills to using the following loss, $\sum_i \log\left(1 + \sum_{j\in\bar{\mathscr{D}}(i)} e^{\mathbf{w}\cdot(\mathbf{x}_j - \mathbf{x}_i) + \Delta(i,j)}\right)$. When Δ is of general form, the problem is no longer decomposable and we were not able to devise an efficient extension. However, when the margin requirement can be expressed as a sum of two functions, $\Delta(i,j) = s(i) - s(j)$, it is possible to extend the efficient coordinate descent procedure. Adding a separable margin, each term $e^{\mathbf{w}\cdot\mathbf{x}_j}$ is replaced with $e^{\mathbf{w}\cdot\mathbf{x}_j \pm s(j)}$. Concretely, we obtain the following gradient,

$$\nabla_{\mathbf{w}}\, \ell_{\mathscr{D}}(\mathbf{x}_i;\mathbf{w}) = \frac{\sum_{j\in\mathscr{D}} e^{\mathbf{w}\cdot\mathbf{x}_j - s(j)}\mathbf{x}_j + e^{\mathbf{w}\cdot\mathbf{x}_i + s(i)}\mathbf{x}_i}{\sum_{j\in\mathscr{D}} e^{\mathbf{w}\cdot\mathbf{x}_j - s(j)} + e^{\mathbf{w}\cdot\mathbf{x}_i + s(i)}} - \mathbf{x}_i \;.$$

The rest of the derivation is identical to that for the zero-margin case, and yields

$$\delta = \frac{1}{mB_r}\sum_i \frac{Z x_{i,r} - \mu_r}{Z + e^{\mathbf{w}\cdot\mathbf{x}_i + s(i)}} \quad \text{where} \quad \mu_r = \sum_{j\in\mathscr{D}(k)} e^{\mathbf{w}\cdot\mathbf{x}_j - s(j)} x_{j,r} \;\text{ and }\; Z = \sum_{j\in\mathscr{D}(k)} e^{\mathbf{w}\cdot\mathbf{x}_j - s(j)} \;.$$

We conclude this section by showing how to incorporate a regularization term and perform feature selection. We use the ℓ_1 norm of the weight vector $\mathbf{w}$ as the means for regularizing the weights. We would like to note though that closed form extensions can be derived for other ℓ_p norms, in particular the ℓ_2^2 norm. We focus on the ℓ_1 norm since it promotes sparse solutions. Adding an ℓ_1 penalty to the quadratic bound and performing a coordinate descent step on the penalized bound amounts to the following (scalar) optimization problem,

$$\min_{\delta} \; g_r\delta + \frac{\beta_r}{2}\delta^2 + \lambda \left\|w_r + \delta\right\|_1 \;\; , \tag{1.4}$$

where g_r and β_r are the expressions appearing in Eq. (1.3). To find the optimal solution, denoted $\delta^\star$, of the above equation we need the following lemmas. The proofs of the lemmas employ routine analysis tools. For brevity we omit the subscript r in the following lemmas.

Lemma 1. *If $\beta w - g > 0$ then $w + \delta^\star \geq 0$ and if $\beta w - g < 0$ then $w + \delta^\star \leq 0$.*

Proof. Let us consider without loss of generality the case where $\beta w - g > 0$. Suppose that $w + \delta^\star < 0$, then, Eq. (1.4) reduces to, $0 = g + \beta\delta^\star - \lambda$. Combining the above equation with the bound on $\beta w - g$, and recalling that $\lambda \geq 0$ and $\beta > 0$, we obtain that $\beta(w + \delta^\star) > \lambda$ and thus $w + \delta^\star > 0$. We thus get a contradiction, hence $w + \delta^\star \geq 0$. The symmetric case follows similarly. □

Lemma 2. *The optimal solution $\delta^\star$ equals $-w$ iff $|g - \beta w| \leq \lambda$.*

Proof. Without loss of generality, let us consider the case where $\beta w - g > 0$. We can then use Lemma 1 to simplify Eq. (1.4). Substituting $w + \delta$ for $\|w + \delta\|_1$ and adding the constraint $w + \delta \geq 0$ we then get, $\delta^\star = \arg\min_\delta \; g\delta + \frac{\beta}{2}\delta^2 + \lambda(w + \delta)$ s.t. $w + \delta \geq 0$. If the inequality constraint holds with a strict inequality, then $\delta^\star = -(\lambda + g)/\beta > -w$. If, however, the minimum $(\lambda + g)/\beta \geq w$ then the optimum must at the constraint boundary, namely $\delta^\star = -w$. Therefore, if $\lambda \geq \beta w - g \geq 0$ then $\delta^\star = -w$, whereas if $\beta w - g > \lambda$, then $\delta^\star = -(\lambda + g)/\beta$. The symmetric case where $g - \beta w \leq 0$ follows similarly. □

Equipped with the above lemmas, the update amounts to the following two-step procedure. Given the current value of w_r and g_r we check the condition stated in Lemma 2. If it is satisfied we set the new value of w_r to 0. Otherwise, we set $\delta^\star = -(g + \lambda)/\beta$ if $\beta w_r - g_r > 0$ and $\delta^\star = (-g + \lambda)/\beta$ if $\beta w_r - g_r < 0$.

As we discuss in the experiments, the combination of the robust loss, the coordinate descent procedure, and the sparsity inducing regularization often yields compact models consisting of about 700 non-zero weights out of ten thousand features for representing images and even sparser models for documents.

Algorithm	Domination Rank			AdaRank		RankBoost
Measure	Multi-valued	Base	Margin	NDCG	MAP	
NDCG 10	3.62	3.38	2.75	4.12	3.62	3.38
Precision 10	3.25	3.12	2.38	4.00	4.38	3.12
NDCG 5	3.12	3.38	3.62	4.12	3.38	3.38

Table 1.1 Results for LETOR dataset. Average rank (lower is better) over eight datasets is shown.

Algorithm	PAMIR	Domination Rank			
		Base-ℓ_2^2	Base-ℓ_1	Multi-valued	Margin
Avg. Precision	0.051	0.050	0.050	0.050	0.050
Precision @ 10	0.080	0.073	0.073	0.073	0.073
Domination Error	0.964	0.964	0.963	0.964	0.964
All Pairs Error	0.234	0.213	0.237	0.235	0.236
% Zero Weights	3.4	1.4	94.3	93.4	93.2

Table 1.2 Results for the large image dataset.

1.5 Experiments

We evaluated and compared the algorithm and its extension on various datasets. We first evaluated the algorithm on the Microsoft's LETOR collection, which are of modest size. On this dataset we compared the algorithm to RankSVM [6], AdaRank [16], and RankBoost [4]. These algorithms take different approaches to the ranking problem. To compare all algorithms we used three evaluation criteria: NDCG5, NDCG10, and Precision at 10. The results are provided in Table 1.1. Note that RankSVM is not presented in the table, since there are published results for RankSVM on only a subset of the datasets, on which its performance was average. We tested our algorithm with and without margin requirements using three tier feedback. While the performance of our algorithm was often better than AdaRank and Rankboost, the results were not conclusive and all the versions we tested exhibited similar performance. We believe that the lack of ability to discriminate between the algorithms is largely due to the modest size of the LETOR collection and the tacit overfitting of the test sets due to repeated experiments that have been conducted on the LETOR collection. We therefore focused on experiments with larger datasets and compared our approach to a fast online algorithm called PAMIR [5] which can handle large ranking problems. One particular aspect that we tested is the ability of our algorithm to yield compact yet accurate models.

Image Ranking Experiments. The first *large* ranking experiment we conducted is an image ranking task. This image dataset consists of about 2.3 million training images and 0.4 million test images. The dataset is a subset of Google's image search repository and is not publicly available. The evaluation included 1,000 different queries and the results represent the average over these queries. The dataset is similar to the Corel image dataset used in [5], albeit it is much larger. We used the feature extraction scheme described in [5], which yielded a 10,000 dimension vec-

	PAMIR	Domination Rank
Avg. Precision	0.705	0.670
Precision @ 10	0.915	0.918
Domination Error	0.974	0.976
All Pairs Error	0.014	0.024
% Zero Weights	78.1	98.8

Table 1.3 Results for the RCV1 collection.

tor representation for each image with an average density (non-zero features) of 2%. The results are given in Table 1.2. We used both ℓ_2^2 and ℓ_1 regularization in order to check the algorithm's performance with compact models and handle the extremely noisy labels. For each image in our dataset we have real-valued feedback which is based on the number of times the image was selected by a user when returned as a result for its associated query. As reported in Table 1.2 we used the user feedback information in two ways: first to construct three-valued feedback (relevant, somewhat relevant, and irrelevant), and second as the means to impose margin constraints. We defined the margin to be the scaled difference between the user counts of the two images. The scaling factor was chosen using cross validation. It is apparent from the table that all the variants attain comparable performance. However, the ℓ_1 version of the domination loss yielded vastly sparse models, which renders the algorithm usable for very large ranking tasks. Disappointingly, despite our careful design, neither the multi-valued feedback nor the margin requirements resulted in improved performance on the image dataset.

Document Ranking Experiments. The second large dataset we experimented with is the Reuters RCV1 dataset. This dataset consists of roughly 800,000 news articles spanning a one-year period. Each document is associated with one or more of 103 topics. Most documents are labeled with at least two topics and many by three or more. We view each topic as a ranking task across the entire collection. From each article, we extracted the raw text as input. We used unigram features as performed in [11]. We randomly partitioned the dataset, placing half of the documents in the training set and splitting the remainder evenly between a validation set and a test set. Of the 103 topics in the dataset, two had too few topics to provide meaningful results. We thus excluded these topics and report results averaged over the remaining 101 topics. For each of the 101 topics, we learned a ranking function and used it to score all the test instances. The results were averaged across topics and are provided in Table 1.3. Here again we see that the ℓ_2 penalized PAMIR, which is a pure dual algorithm, and the ℓ_1 penalized coordinate descent algorithm achieve similar performance with the exception of the number of mis-ordered pairs, which is closely related to the loss PAMIR employs. The main advantage again is the sparsity of the resulting models. Our approach uses fewer than 2% of the original features whereas PAMIR uses a large portion. (PAMIR does not use all of the features despite the ℓ_2 regularization since some topics consists of a small number of relevant documents.)

1.6 Conclusions

We derived in this paper an efficient coordinate descent algorithm for the task of learning to rank. Our construction is tightly based on the domination loss first proposed in [3]. We described a convex relaxation of that loss with an associated update that scales linearly with the number of training instances. We also derived several extensions of the basic algorithm, including the ability to handle multiple valued feedback, margin requirements over pairs of instances, and ℓ_1 regularization. Furthermore, the algorithm's efficiency is retained for these extensions. Experiments with several datasets show that by using ℓ_1 regularization, the resulting ranking models are trained considerably faster and yield significantly more compact models, yet they attain performance competitive with some of the state-of-the-art learning to rank approaches.

References

1. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: from pairwise approach to listwise approach. Proc. of the 24th Intl. Conference on Machine Learning, 129–136 (2007)
2. Cohen, W.W., Schapire, R.E., Singer, Y.: Learning to order things. Journal of Artificial Intelligence Research **10**, 243–270 (1999)
3. Dekel, O., Manning, C., Singer, Y.: Log-linear models for label ranking. In: Advances in Neural Information Processing Systems 16 (2004)
4. Freund, Y., Iyer, R., E.Schapire, R., Singer, Y.: An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research **54**, 933–969 (2003)
5. Grangier, D., Bengio, S.: A discriminative kernel-based model to rank images from text queries. IEEE Trans. on Pattern Analysis and Machine Intelligence **30**(8), 1371–1384 (2008)
6. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD) (2002)
7. Joachims, T.: A support vector method for multivariate performance measures. In: Proceedings of the International Conference on Machine Learning (ICML) (2005)
8. Luo1, Z., Tseng, P.: On the convergence of the coordinate descent method for convex differentiable minimization. Journal of Optimization Theory and Applications **72**(1), 7–35 (1992)
9. Roweis, S.T., Salakhutdinov, R.: Adaptive overrelaxed bound optimization methods. In: Twentieth International Conference on Machine Learning, pp. 664–671 (2003)
10. Salton, G.: Automatic text processing: the transformation, analysis and retrieval of information by computer. Addison-Wesley (1989)
11. Singhal, A., Buckley, C., Mitra, M.: Pivoted document length normalization. In: Research and Development in Information Retrieval, pp. 21–29 (1996)
12. Tseng, P., Yun, S.: A coordinate gradient descent method for nonsmooth separable minimization. Mathematical Programming Series B **117**, 387–423 (2007)
13. Vapnik, V.N.: Estimation of Dependences Based on Empirical Data. Springer-Verlag (1982)
14. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer (1995)
15. Vapnik, V.N.: Statistical Learning Theory. Wiley (1998)
16. Xu, J., Li, H.: Adarank: a boosting algorithm for information retrieval. Proc. of the 30th Annual Intl. Conf. on Research and Development in Information Retrieval, 391–398 (2007)