MEMR: A Margin Equipped Monotone Retargeting Framework for Ranking

Sreangsu Acharyya *
Dept. of Electrical Engineering
University of Texas Austin.

Joydeep Ghosh
Dept. of Electrical Engineering
University of Texas Austin.

Abstract

We bring to bear the tools of convexity, margins and the newly proposed technique of monotone retargeting upon the task of learning permutations from examples. This leads to novel and efficient algorithms with guaranteed prediction performance in the online setting and on global optimality and the rate of convergence in the batch setting. Monotone retargeting efficiently optimizes over all possible monotone transformations as well as the finite dimensional parameters of the model. As a result we obtain an effective algorithm to learn transitive relationships over items. It captures the inherent combinatorial characteristics of the output space yet it has a computational burden not much more than that of a generalized linear model.

1 INTRODUCTION

Many applications require items to be ordered correctly. Prototypical examples of such applications are information retrieval and recommender systems. In most cases, however, the quality measure that actually defines the *transitive relation* of interest can be accessed only through examples. This lack of direct access to the ordering relation motivates learning the quality measure from the covariates of the items. We distinguish this task from a related and easier one of learning binary pairwise relations where transitivity is not required by the application.

Existing techniques of learning to rank (LETOR) fall under 3 categories: (i) point-wise methods, (ii) pair-wise methods and (iii) list-wise methods. In point-wise methods, higher ranked items are assigned higher target scores. The method ignores the combinatorial structure of the output space and regresses the scores directly. Pair-wise methods capture some structure by trying to classify for a pair whether the first item in the pair out-ranks the second. Their predictions need not be transitive and an *order-reconciliation step* is necessary to enforce it. This is NP hard [8], necessitating approximations and heuristics. Finally, there are list-wise methods that model the full combinatorial structure and need to solve formidable optimization problems. They have to cut corners for scalability. Notable approaches include sampling [25], approximations [2], and resorting to point-wise methods [6] .

An ideal LETOR formulation should (i) capture combinatorial structure like list-wise methods, but with (ii) algorithms as simple as point-wise methods. While this seems too much to ask, the recently proposed monotone retargeting (MR) technique is one way how this may be approached [1]. MR outperforms several state of the art ranking algorithms such as Listnet [6] and RankCosine, even after improving those algorithms for statistical consistency as proposed by Ravikumar et. al. [21].

MR efficiently reduces, the LETOR problem to a generalized linear model (GLM) with no loss in generality. It subsumes *statistically consistent* methods of [21]. The distinguishing characteristic of MR is its *"retargeting" paradigm*, where instead of fitting training scores exactly, it tries to fit any score that captures the desired order. Recall that our task is to retrieve the correct order and not the training scores. In this setting, retrieving the specified training scores are an unnecessary burden. The specified training scores may be particularly difficult to fit for the chosen family of regression function class, but there might exist score assignments that capture the desired order and also simultaneously lie in the range space of the regression function class being used. The MR framework tries to find such score assignments by formulating it as a Bregman divergence minimization problem.

In this paper we push the *retargeting* idea further.

*Authors acknowledge NSF grant IIS-1017614

This is facilitated by (i) a remarkably efficient finite time optimization over the infinite space of all monotonic transformations and (ii) properties of Bregman divergences particularly suited for learning orders.

Let us draw a few analogies from classification. A pointwise approach to a $\{-1, 1\}$ encoded classification problem would try to fit the $\{-1, 1\}$ training scores exactly, possibly enriching the approximating function class till the quality of the fit is acceptable. Most successful classifiers, however, fit values that are discriminable, ignoring, entirely, whether they are close to the training scores of $\{-1, 1\}$ in value.

The MR cost function consists of two parts: a loss and a regularization. Similar to perceptrons, the moment MR predictions retrieve the training ranks, its loss drops to zero. Experience in classification has taught us that losses that continue to be active after training error has dropped to zero yield better accuracy, for example, SVMs, logistic regression and boosting. In our paper we equip MR with such a margin-like property. This can be done in a few different ways. Our intent is not to champion one over another. This paper is not about advocacy, but about exploring how margin may be incorporated into the *"retargeting" paradigm*.

In this paper (i) we introduce large and fixed margin variants of the MR approach. Without margins the MR cost function is degenerate, an aspect that is not developed in the previous work [1]. Unlike the previous approach, we model the requirement of a margin explicitly in this paper. (ii) Unlike [1] we are able to model the notion that ordering errors at the top are worse than those at the bottom. (iii) It was shown that MR cost function is jointly convex *iff* the Bregman divergence chosen is squared Euclidean. We extend the formulation to enable joint convexity to all strongly convex Bregman divergence, not to advocate non-Euclidean divergences but to explore them.

Joint convexity has two important ramifications: one affects ease of evaluation of the technique, the other affects efficiency of training. The initialization independence of the optimum, gained as a result of convexity induced uniqueness, makes comparing different Bregman divergences easier, eliminating the need for multiple initializations during training. (iv) On the other hand for training, joint convexity allows us to replace *exact* coordinate-wise updates that were used in [1] with more efficient gradient updates with guarantees on global optimality. (v) This yields efficient online algorithms with regret bounds over permutations. Finally, (vi) we provide rates of convergence guarantees, an aspect missing from the previous work.

To date many cost functions have been designed to *evaluate* rankings, for example, discounted cumulative gain (DCG), normalized discounted cumulative gain (NDCG) [13], expected reciprocal rank (ERR) [7], mean average precision (MAP) [3]. They are functions of permutations and capture the notion that positional accuracy at the top is more important than at the bottom. They are reasonably easy to compute given a ranking, but to optimize them in training is notoriously intractable. Our formulation, on the other hand, introduces a family of cost functions that have characteristics desired in ranking: dependence on order not on scores and the ability to capture the importance of non-uniform positional accuracy, but at the same time optimized *globally* with ease. These aspects set our work apart from other approaches of learning to rank.

We follow the **notation** used in the MR paper. Vectors are denoted by bold lower case letters, matrices are capitalized. $\boldsymbol{x}^\dagger$ is $\boldsymbol{x}$ transposed and $||\boldsymbol{x}||$ its L_2 norm. Adj-Diff$(\cdot)$ is the adjacent difference operator, and Cum-Sum(Adj-Diff$(\boldsymbol{x})) = \boldsymbol{x}$. $\boldsymbol{x}$ is in *descending order* if $x_i \geq x_j$ when $i > j$. the set of such vectors is $\mathcal{R}\!\downarrow$. $\boldsymbol{x}$ is isotonic with $\boldsymbol{y}$ if $x_i \geq x_j$ implies $y_i \geq y_j$. Δ denotes an unit simplex and Δ_ϵ its subset with members component-wise bounded away from 0 by ϵ. ${\mathbb{R}_+}^d$ is the positive orthant and R_ϵ^d its subset similarly bounded away from 0 by ϵ. Interior is denoted by int .

2 BACKGROUND

We will use **Bregman Divergences** to construct our cost function. Let $\boldsymbol{\phi} : \Theta \mapsto \mathbb{R}$, $\Theta = \operatorname{dom} \boldsymbol{\phi} \subseteq \mathbb{R}^d$ be a strictly convex, closed function, differentiable on $\operatorname{int}\Theta$. The corresponding Bregman divergence $D_\phi\big(\cdot\big|\big|\cdot\big) : \operatorname{dom}(\boldsymbol{\phi}) \times \operatorname{int}(\operatorname{dom}(\boldsymbol{\phi})) \mapsto \mathbb{R}_+$ is defined as $D_\phi\big(\boldsymbol{x}\big|\big|\boldsymbol{y}\big) \triangleq \boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{\phi}(\boldsymbol{y}) - \langle \boldsymbol{x} - \boldsymbol{y}, \nabla\boldsymbol{\phi}(\boldsymbol{y})\rangle$. From strict convexity it follows that $D_\phi\big(\boldsymbol{x}\big|\big|\boldsymbol{y}\big) \geq 0$ and $D_\phi\big(\boldsymbol{x}\big|\big|\boldsymbol{y}\big) = 0$ iff $\boldsymbol{x} = \boldsymbol{y}$. Bregman divergences are (strictly) convex in their first argument, but not necessarily convex in their second.

In this paper we only consider functions $\boldsymbol{\phi}(\cdot) : \mathbb{R}^n \ni \boldsymbol{x} \mapsto \sum_i w_i \phi(x_i)$ that are weighted sums of *identical* scalar convex functions applied to each component, the former referred to as *weighted, identically separable* (**WIS**) or **IS** if the weights are equal. [1] and [21] identify this class to have properties particularly suited for ranking. The MR approach, in concert with Bregman divergences can provide compelling guarantees that includes convergence, parallelizability, statistical consistency, and avoids solving a linear assignment problem in every iteration of their training loop. Many LETOR algorithms [24], [25] fall prey to the latter.

Monotone Retargeting: The ranking problem in-

volves set of queries $\mathcal{Q} = \{q_1, q_i \dots q_{|\mathcal{Q}|}\}$ and a set of training items $\mathcal{V}$. For every query q_i, the elements of $\mathcal{V}_i \subset \mathcal{V}$ are ordered based on their relevance to the query. This ordering is expressed through a rank score vector $\tilde{\boldsymbol{r}}_i \in \mathbb{R}^{|\mathcal{V}_i|}$ whose components $\tilde{r}_{ij}$ correspond to items in $\mathcal{V}_i$. Beyond establishing the order, the actual values are irrelevant. In our formulation, however, one may choose whether to treat these as irrelevant or incorporate them in the *retargeting* step, making the formulation more flexible.

For a query q_i the index j of $\tilde{r}_{ij}$ is local to $\mathcal{V}_i$ and assigned such that $\tilde{r}_{ij}$ are in descending order for any $\mathcal{V}_i$. For every pair $\{q_i, v_{ij}\}$ a feature vector $\mathbb{R}^n \ni \boldsymbol{a}_{ij} = F(q_i, v_{ij})$ is an input to the algorithm, $\boldsymbol{A}_i$ is a matrix whose j^{th} row is ${\boldsymbol{a}_{ij}}^\dagger$. The following formulation seems suitable for ranking:

$$\min_{\boldsymbol{w}, \boldsymbol{\Upsilon}_i \in \mathcal{M}} \sum_i D_i\big(\tilde{\boldsymbol{r}}_i, \boldsymbol{\Upsilon}_i \circ f(\boldsymbol{A}_i, \boldsymbol{w})\big), \tag{1}$$

where $D_i : \mathbb{R}^{|\mathcal{V}_i|} \times \mathbb{R}^{|\mathcal{V}_i|} \mapsto \mathbb{R}_+$ is some distance-like loss function, $f : \mathbb{R}^{|\mathcal{V}_i| \times n} \times \mathbb{R}^n \mapsto \mathbb{R}^{|\mathcal{V}_i|}$ is some parametric form with the parameter $\boldsymbol{w}$ and $\boldsymbol{\Upsilon}_i : \mathbb{R}^{|\mathcal{V}_i|} \mapsto \mathbb{R}^{|\mathcal{V}_i|}$ is a mapping that transforms the components by a scalar, strictly monotonic increasing function Υ_i, and $\mathcal{M}$ is the class of all such functions. Formulation (1) avoids the problem that adversely affects pointwise-methods: solving an unnecessarily hard problem of matching the scores by value.

To avoid working in the space of $\mathcal{M}$ which is infinite dimensional, MR solves a qualitative equivalent

$$\min_{\boldsymbol{w}, \boldsymbol{r} \in \mathcal{R}\downarrow_i} \sum_i D_i(\boldsymbol{r}_i, f(\boldsymbol{A}_i, \boldsymbol{w})) \text{ s.t. } \mathcal{R}\downarrow_i = \{\boldsymbol{r} | {}^{\exists \boldsymbol{M} \in \mathcal{M}}_{\boldsymbol{M}(\tilde{\boldsymbol{r}}_i) = \boldsymbol{r}}\}. \tag{2}$$

Let us take a closer look at the constraint set used in formulation (2): Instead of considering all strictly increasing monotonic transforms $\boldsymbol{\Upsilon}_i$ of the right argument, MR considers all inverse monotonic transformations of the left argument. This, remarkably, is a finite dimensional optimization problem because $\mathcal{R}\downarrow_i$, the set of all vectors isotonic with $\tilde{\boldsymbol{r}}_i$ is a finitely characterizable convex cone. Motivated by convexity, MR chooses $D_i(\cdot, \cdot)$ to be a Bregman divergence $D_\phi\big(\cdot \big|\big| \cdot\big)$ and $f(\boldsymbol{A}_i, \boldsymbol{w})$ to be $(\nabla\phi)^{-1}(\boldsymbol{A}_i \boldsymbol{w})$ to obtain[1]

$$\min_{\beta_i, \boldsymbol{w}, \boldsymbol{r}_i \in \mathcal{R}\downarrow_i \cap \mathcal{S}_i} \sum_{i=1}^{|\mathcal{Q}|} \frac{1}{|\mathcal{V}_i|} D_{\boldsymbol{\phi}}\big(\boldsymbol{r}_i \big|\big| (\nabla\phi)^{-1}\left(\boldsymbol{A}_i \boldsymbol{w} + \beta_i \mathbf{1}\right)\big) + \frac{C}{2}||\boldsymbol{w}||^2. \tag{3}$$

[1]We take a shortcut of writing $D_{\boldsymbol{\phi}}\big(\cdot \big|\big| (\nabla\phi)^{-1}(\cdot)\big)$ instead of $D_{\phi_i}(\cdot, (\nabla\phi)^{-1})$ where ϕ_i indicates a separable convex function of an input dimension d_i built from component-wise sum of scalar function $\phi(\cdot)$.

$$\mathbb{P}_i^{t+1} = \operatorname*{Argmin}_{\pi} D_{\boldsymbol{\phi}}\big(\boldsymbol{r}_i^t \big|\big| (\nabla\phi)^{-1}\left(\pi \boldsymbol{A}_i \boldsymbol{w}^t + \beta_i^t\right)\big) \ \ \forall i \tag{4}$$

$$\boldsymbol{r}_i^{t+1} = \operatorname*{Argmin}_{\boldsymbol{r} \in \mathcal{R}\downarrow_i \cap \mathcal{S}_i} D_{\boldsymbol{\phi}}\big(\boldsymbol{r} \big|\big| (\nabla\phi)^{-1}\left(\mathbb{P}_i^{t+1} \boldsymbol{A}_i \boldsymbol{w}^t + \beta_i^t\right)\big) \ \ \forall i \tag{5}$$

$$\boldsymbol{w}^{t+1}, \{\beta_i^{t+1}\} = \tag{6}$$

$$\operatorname*{Argmin}_{\boldsymbol{w}, \{\beta_i\}} \sum_{i=1}^{|\mathcal{Q}|} D_{\boldsymbol{\phi}}\big(\boldsymbol{r}_i^{t+1} \big|\big| (\nabla\phi)^{-1}\left(\mathbb{P}_i^{t+1} \boldsymbol{A}_i \boldsymbol{w} + \beta_i^t\right)\big) \frac{C}{2}||\boldsymbol{w}||^2 \tag{7}$$

Figure 1: Updates of Monotone Retargeting

where $(\nabla\phi)^{-1}$ is the inverse of the gradient mapping, $\mathcal{S}_i$ is a convenient convex set excluding $\mathbf{0}$, that is necessary only for technical reasons.

In practice, even if $\mathcal{V}_i$ is totally ordered, it is common to have a part of that information erased by quantization in the scores $\tilde{\boldsymbol{r}}$. MR deals with this by optimizing over block diagonal permutation matrices $\mathbb{P}_i$ that permute contiguous blocks of indices that correspond to items whose relative order have been erased. The model is trained by iterating over the updates (4), (5) and (7) shown in Figure 1. It has been shown that these *exact coordinate-wise minimizations* updates converge to a local minimum(*or global for square loss* [1]) of function (3). Update (4) is accomplished by sorting. This turns out to be so because of special properties of separable Bregman divergences (see [1] for details). Update (5) uses the exponentiated gradient algorithm [15] and (7) is the same problem as estimating the parameters of a generalized linear model [19]. A quasi-Newton method (LBFGS [17]) was used to solve (7). In the rest of the paper the block diagonal permutation matrices $\mathbb{P}_i$ will be suppressed. Our extensions continue to be effective for partial order via updates that correspond to (4), but this is not elaborated further for brevity.

3 FORMULATION

The rest of the paper describes our contribution. Its prominent features are: (i) formulation of fixed and large margin aspects, (ii) *joint* convexity of the cost function in the targets $\boldsymbol{r}$ and the parameters $\boldsymbol{w}$, which yields (iii) guarantees on performance in the online setting and super-linear convergence in the batch setting.

Since there are multiple moving parts in our formulation, it is easy to get lost in the details. To preempt that we lay out the flow of our arguments. We explain the formulation by modifying the cost function (3) suc-

cessively. We conclude each subsection with summary of what has been achieved in the subsection so far.

Convexity: We equip the cost function with strong and joint convexity, aspects missing in the original work. We pick a *matching* form of the regularizer so that it adds no extra computational burden and quantify the amount of regularization that is sufficient to guarantee joint convexity. It may not be surprising that regularization extends convexity properties to MR losses other than squared Euclidean. What is surprising, however, is that this convexity applies jointly to $\boldsymbol{r}$ and $\boldsymbol{w}$ although the regularizers themselves are separated. Strong joint convexity and smoothness thus gained lead to the performance and convergence guarantees. This is the topic of section 3.1.

Margins: Second we plug a loophole in the MR cost function by ensuring margins between all adjacent target scores $r_{i,j}, r_{i,j+1}$. Without this, the cost function (3) is degenerate: one can achieve zero loss by setting $\boldsymbol{w}, \beta = \mathbf{0}$. We provide different ways of ensuring this: (i) directly by setting constraints, and (ii) indirectly by rewarding margins. Since both the constraints and the rewards are linear, this does not disrupt joint convexity. The key is to optimize the modified cost function efficiently. This is the topic of section 3.3.

3.1 Convexity, Smoothness and Optimization

MR ensures joint convexity *only* if squared Euclidean distance is used. We incorporate joint convexity into the cost function (3). This benefits us in two ways: (i) it removes initialization dependence of the training method and (ii) as we shall see, allows for a more efficient method of training, both online and batch with excellent convergence rates. We know that strong convexity together with smooth gradients (and Hessians for second order methods) admit efficient minimization: gradient descent achieves linear rate of convergence, quasi-Newton (truncated-Newton) achieves superlinear rates. We examine conditions under which our ranking formulations have these properties.

3.1.1 Joint Convexity

Let $\phi(\cdot)$ be s strongly convex [5]. Consider the term:

$$F_i(\boldsymbol{r}_i, \boldsymbol{w}) = \frac{1}{|\mathcal{V}_i|}\Big(D_{\boldsymbol{\phi}}\big(\boldsymbol{r}_i\Big|\Big|(\nabla\phi)^{-1}(\boldsymbol{A}_i\boldsymbol{w})\big) + \underbrace{C_{ri}D_\phi\big(\boldsymbol{r}_i\Big|\Big|\boldsymbol{q}_i\big) + \frac{C_{wi}}{2}||\boldsymbol{w}||^2_{A_i}}_{\text{Regularization terms}}\Big) \quad (8)$$

using which we modify cost function (3) to

$$F(\{\boldsymbol{r}_i\}, \boldsymbol{w}) = \sum_i^{|\mathcal{Q}|} F_i(\boldsymbol{r}_i, \boldsymbol{w}) + \frac{C}{2}||\boldsymbol{w}||^2. \quad (9)$$

The β terms of equation (3) may be absorbed into $\boldsymbol{A}_i$ by augmenting the features by vectors of ones, so no generality is lost in equation (9).

Let us pause to take note of the extra terms in the cost function (9). There is a term regularizing $\boldsymbol{w}$ towards 0 and another regularizing $\boldsymbol{r}_i$ towards $\boldsymbol{q}_i$. Vector $\boldsymbol{q}_i$ is a "center" of regularization for the targets $\boldsymbol{r}_i$. If C_{ri} are nonzero we set these to $\tilde{\boldsymbol{r}}_i$ when training scores are available, otherwise we use $\boldsymbol{q}_i = \text{Argmin}_{\boldsymbol{x}}\, \boldsymbol{\phi}_i(\boldsymbol{x})$ when only ordering is available (this corresponds to $\mathbf{0}$ for square loss and uniform distribution for KL loss). This allows one to bias the targets towards the training scores when C_{ri} is high and focus on order otherwise.

Proposition 1. *Let ϕ be s strongly convex with L Lipschitz continuous gradients, and σ_i be the smallest singular value of A_i, then the cost function* (9) *is jointly convex if* $\sum \frac{\sigma_i(C_{wi}+1/L)}{|\mathcal{V}_i|} + \frac{C}{2} - \frac{4\mathcal{Q}(\sum \frac{1}{|\mathcal{V}_i|})^2}{s\sum \frac{1+C_{ri}}{|\mathcal{V}_i|}} \geq 0$

Proof. $\sum_{i=1}^{|\mathcal{Q}|} \frac{1}{|\mathcal{V}_i|}\begin{bmatrix} (1+C_{ri})H_\phi & -I \\ -I & {A_i}^\dagger(H_\psi + C_{wi})A_i + \frac{C|\mathcal{V}_i|}{2|\mathcal{Q}|}I \end{bmatrix}$, is the Hessian of the cost function (9) where ψ is the Legendre conjugate of ϕ and H_ϕ, H_ψ the corresponding diagonal. Recall that $\boldsymbol{\phi}(\cdot)$ and consequently $\boldsymbol{\psi}(\cdot)$ are separable. The smallest eigenvalue of the Hessian may be bounded as the value of the following optimization problem:

$$\min \langle \boldsymbol{y}, \boldsymbol{y}\rangle \left(\sum_i \frac{\sigma_i}{|\mathcal{V}_i|}(C_{wi} + \frac{1}{L}) + \frac{C}{2}\right) - 2\langle \boldsymbol{x}, \boldsymbol{y}\rangle \sum_i \frac{|\mathcal{Q}|}{|\mathcal{V}_i|} + \langle \boldsymbol{x}, \boldsymbol{x}\rangle \sum_i \frac{s}{|\mathcal{V}_i|}(1 + C_{ri}) \quad s.t.\ \langle \boldsymbol{x}, \boldsymbol{x}\rangle + \langle \boldsymbol{y}, \boldsymbol{y}\rangle = 1 \quad (10)$$

where σ_i is the smallest singular value of A_i. Invoking Cauchy-Schwarz inequality and treating the expression as a quadratic function in $\sqrt{\langle \boldsymbol{x}, \boldsymbol{x}\rangle}$ we can see that convexity is implied by $\sum \frac{\sigma_i(C_{wi}+1/L)}{|\mathcal{V}_i|} + \frac{C}{2} - \frac{4(\sum \frac{\mathcal{Q}}{|\mathcal{V}_i|})^2}{s\sum \frac{1+C_{ri}}{|\mathcal{V}_i|}} \geq 0$ □

Corollary 1. *The cost function* (9) *is jointly convex if $C \geq \frac{8\mathcal{Q}}{s(1+C_r)}(\sum \frac{1}{|\mathcal{V}_i|})$, if $C_{ri} = C_r\ \forall_i$.*

Corollary 1 gives practitioners an easy thumb rule to ensure joint convexity.

These additional regularization terms do not come at an extra computational burden. Estimating $\boldsymbol{r}, \boldsymbol{w}$ remain just as easy. We show that the result of the additional terms are that the $\boldsymbol{r}_i$ updates (5) need to be computed with respect to the *deflected* predicted score $(\nabla\phi)^{-1}(\alpha A\boldsymbol{w} + (1-\alpha)\nabla\phi(\boldsymbol{q}_i))$, as opposed to the predicted score $(\nabla\phi)^{-1}(A\boldsymbol{w})$.

Lemma 1. *Let $\alpha_i = \frac{1}{1+C_{\phi i}}$, then* $\underset{\boldsymbol{r}_i \in \mathcal{R}\downarrow_i \cap \mathcal{S}_i}{\text{Argmin}}\ F_i(\boldsymbol{r}_i, \boldsymbol{w})$ $= \underset{\boldsymbol{r}_i \in \mathcal{R}\downarrow_i \cap \mathcal{S}_i}{\text{Argmin}}\ D_{\boldsymbol{\phi}}\big(\boldsymbol{r}_i\Big|\Big|(\nabla\phi)^{-1}(\alpha_i\boldsymbol{A}_i\boldsymbol{w} + (1-\alpha_i)\nabla\phi(\boldsymbol{q}_i))\big)$.

Proof. Use $\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\boldsymbol{\pi}}\left[D_\phi(\boldsymbol{x}\middle\|\boldsymbol{s})\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\boldsymbol{\pi}}\left[D_\phi(\boldsymbol{x}\middle\|\boldsymbol{\mu})\right] + D_\phi(\boldsymbol{\mu}\middle\|\boldsymbol{s})$ [4]. □

3.1.2 Marginal Strong Convexity and Smoothness

Recall our motivations for pursuing joint convexity: (i) initialization independence of the training and (ii) more efficient training algorithms. In light of Proposition 1 and Corollary 1, the reader should be convinced of the former. In this section we explore how joint convexity may be exploited to provide an efficient optimization algorithm for training, as well as guarantees of convergence rates. Previous work on MR [1] come with no guarantees on rates of convergence.

The MR cost function was minimized in [1] using *exact* coordinate-wise minimizations. This can be expensive for the $\boldsymbol{w}, \beta$ updates (7) because they are iterative in nature. Further since a single $\boldsymbol{w}, \beta$ update is equivalent to solving a generalized linear model (GLM), it is clear that the MR procedure would be slower than solving for a GLM because typically multiple iterations of GLM update are required for convergence.

Here we will replace exact coordinate-wise minimizations over $\boldsymbol{r}, \boldsymbol{w}$ by inexact gradient descent updates that satisfy any of the standard "sufficient descent" criteria [5] (for example Armijo's criteria) used in gradient based methods. Joint convexity will play a crucial role in making this possible.

Joint convexity of $F(\{\boldsymbol{r}_i\}, \boldsymbol{w})$ allows us to work with the marginal function

$$G(\boldsymbol{w}) = \min_{\{\boldsymbol{r}_i\}} F(\{\boldsymbol{r}_i\}, \boldsymbol{w}) \tag{11}$$

without losing convexity. This luxury is not available in MR. The marginal function is guaranteed to be convex when the joint function is convex [23]. Recall convexity is always preserved under pointwise *maximization*, however if the function is *jointly convex* it is also preserved under pointwise minimization as in equation (11).

The gradient $\nabla G(\boldsymbol{w})$ of the marginal is obtained as

$$\nabla G(\boldsymbol{w}) = \sum_i^{|\mathcal{Q}|} G_i(\boldsymbol{w}) = \sum_i^{|\mathcal{Q}|} \nabla F_i(\{\boldsymbol{r}_i^*\}, \boldsymbol{w}) \tag{12}$$

where $\boldsymbol{r}_i^* = \text{Argmin}_{\boldsymbol{r}_i \in \mathcal{R}\downarrow} F_i(\boldsymbol{r}_i, \boldsymbol{w})$.

Now we can make a few observations: for a choice of a closed form of $\phi(\cdot)$ we know ∇F_i in closed form. Hence the moment we are able to compute $\boldsymbol{r}_i^*$ we can also compute the gradient of the function $G(\boldsymbol{w})$ and hence minimize it using any gradient based minimization methods. Also observe that this *gradient computation trivially parallelizes because the $\boldsymbol{r}_i$s are all independent* and can be computed simultaneously. We shall show that $\boldsymbol{r}_i^*$ can be computed very efficiently in not only finite time but also linear in the number of training points per query. This is covered in Section 3.5.

If in addition to just convexity of the marginal function $G(\boldsymbol{w})$ we also had strong convexity, not only would it facilitate super-linear convergence of quasi-Newton methods, but it will also guarantee logarithmic regret in the online setting [11]. With these motivations in mind we investigate the conditions for strong convexity of $G(\boldsymbol{w})$. We do so by examining the Hessian $\nabla^2 G(\boldsymbol{w})$. Note however that $G(\boldsymbol{w})$ is not obtained in closed form but by equation (11), which we now need to differentiate twice to find the Hessian.

Differentiating Twice Under the Minimization Sign: A prominent role is played in the analysis by the ability to differentiate under the minimization sign. We do not know the function $G(w)$ in closed form but are able to compute its Hessian in terms of $\boldsymbol{r}_i^*$. Using assumptions of continuous second order differentiability and the shorthand $F_i^* = F_i(\boldsymbol{r}_i^*, \boldsymbol{w})$ we obtain

$$\nabla^2 G_i(\boldsymbol{w}) = \nabla^2_{\boldsymbol{w}} F_i^* \; - \nabla_{\boldsymbol{w},\boldsymbol{r}_i}\nabla F_i^{*\dagger} (\nabla^2_{\boldsymbol{r}_i} F_i^*)^{-1} \nabla_{\boldsymbol{w},\boldsymbol{r}_i}\nabla F_i^* =$$
$$\frac{\boldsymbol{A}_i}{|\mathcal{V}_i|}^\dagger \left[H_\psi + C_{wi} - \frac{1}{1+C_{ri}}(H_\phi)^{-1}\right] \boldsymbol{A}_i + \frac{C}{|\mathcal{Q}|} I \tag{13}$$

by differentiation twice under the min operator. Expression (13) will be useful because it allows to determine when is $G(\boldsymbol{w})$ strongly convex (see Lemma 2) and also because it gives us a way to compute the Hessian that is important for Newton methods that we employ.

Lemma 2. *If ϕ is s strongly convex with L-Lipschitz continuous gradient and σ_i is the principal singular value of A_i, then $G(\boldsymbol{w})$ is C strongly convex if* $\sum_i \left(\frac{\sigma_i}{L} + \sigma_i C_{wi} - \frac{\sigma_i}{s(1+C_{ri})}\right) > 0$.

Strong convexity and Lipschitz continuity of the gradient ensures that a gradient descent method will have linear rate of convergence [5]. Lemma 2 gives the practitioner a way to choose C_{wi} and C_{ri} appropriately.

Can the convergence rates be pushed further ? Can we obtain locally quadratic convergence ? We answer in the affirmative in the next section.

3.1.3 Lipschitz Continuity of Hessian

In order to enjoy local quadratic convergence, quasi-Newton methods require that the objective function (i) be twice differentiable, (ii) be strongly convex and (iii) have Lipschitz continuous Hessians [5]. The first two have already been established, now we explore

the third. Observe from equation (13) that we only need to be concerned about the sensitivity of the term $\left[H_\psi + C_{wi} - \frac{1}{1+C_{\phi_i}}(H_\phi)^{-1}\right]$ to variations in $\boldsymbol{w}$. We make the notation more precise about dependency on $\boldsymbol{w}$. Let $\boldsymbol{r}_i^*(\boldsymbol{w}) = \text{Argmin}_{\boldsymbol{r}_i \in \mathcal{R}\downarrow} F_i(\boldsymbol{r}_i, \boldsymbol{w})$ and the parenthesis indicate where the Hessians are evaluated in the expression: $\left[H_\psi(\boldsymbol{w}) + C_{wi} - \frac{1}{1+C_{\phi_i}}(H_\phi(\boldsymbol{r}_i^*(\boldsymbol{w})))^{-1}\right]$.

Lemma 3. *Let $\psi(\cdot)$ be the Legendre conjugate of $\phi(\cdot)$ that defines the cost function $G(\boldsymbol{w})$ in equation (11). Then if $\psi(\cdot)$ has a Lipschitz continuous Hessian then $G(\boldsymbol{w})$ has a Lipschitz continuous Hessian.*

Proof. $\left[H_\psi(\boldsymbol{w}) + C_{wi} - \frac{1}{1+C_{\phi_i}}(H_\phi(\boldsymbol{r}_i^*(\boldsymbol{w})))^{-1}\right] = \left[H_\psi(\boldsymbol{w}) + C_{wi} - \frac{1}{1+C_{\phi_i}} H_\psi(\nabla\phi(\boldsymbol{r}_i^*(\boldsymbol{w})))\right]$ using Legendre duality. Further, the vector $\nabla\phi(\boldsymbol{r}_i^*(\boldsymbol{w}))$ turns out to be the Euclidean projection of the vector $\boldsymbol{A}_i\boldsymbol{w}$ on the set $\mathcal{R}\downarrow_i$ (see Proposition2). Since projection is a non-expansive operator, $H_\psi(\nabla\phi(\boldsymbol{r}_i^*(\boldsymbol{w})))$ is Lipschitz continuous in $\boldsymbol{w}$. □

3.1.4 Summary: Impact on Optimization

Let us take stock of what have we achieved so far. Lemmas 1 through 3 led to quantitative guarantees on rate of convergence in the batch setting. They allow selecting the regularization parameters C_{ϕ_i}, C_{wi} based on desired convergence performance. The paper [1] could not provide any such quantitative guarantees, because their cost function was not proven to be jointly convex. Note that the nested minimization in the gradient computation trivially parallelizes. We shall see that each parallel task completes in finite time (Section 3.6). *Batch gradient descent on the marginal with* (12) *evaluated in parallel* converges linearly as a result of strong marginal convexity and smoothness [5]. *Stochastic gradient descent by sampling an index from* (12) also has linear rate of convergence (in an expected sense) [20]. *Quasi-Newton (and truncated Newton) methods with parallel evaluation of gradients* use the gradient computation (12) (and explicit Hessian (13) which has a simple diagonal structure) have superlinear convergence [5].

3.2 Online Algorithm for Learning Permutations

In this section we propose an online model for learning to rank where we have a varying set of items that need to be ordered in each round. The adversary, at round t provides the feature matrix $\boldsymbol{A}_t$ of d_t items that it has ranked, but that order is not revealed till the learner responds with a "scoring vector" $\boldsymbol{w}_t$. The learner is then charged a cost of $G_t(\boldsymbol{w}_t)$ as defined in (11) according to any twice differentiable σ strongly convex function ϕ_t with L Lipschitz continuous gradient. The order and the function ϕ_t is then revealed for the learner to use. The objective is to minimize the cumulative loss $\sum_t G_t(\boldsymbol{w}_t)$.

For the t^{th} gradient update we use the t^{th} term of the gradient (12) with a learning rate of $\frac{1}{\sigma t}$ as

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{1}{\sigma t} F_t(\{\boldsymbol{r}_t^*\}, \boldsymbol{w})$$

where $\boldsymbol{r}_t^* = \text{Argmin}_{\boldsymbol{r}_t \in \mathcal{R}\downarrow \cap \mathcal{S}_t} F_i(\boldsymbol{r}_t, \boldsymbol{w})$ and F_t is defined in (9).

Theorem 1. *[11] The online gradient algorithm applied in an online setting to a s strongly function that has L Lipschitz continuous gradients has regret $\mathcal{O}(\frac{L^2}{\sigma}\log T)$.*

Neither the algorithm nor the bound is new, what is novel though is that the ranking problem of such combinatorial nature can be transformed into a form, without loss in generality, that this algorithm can exploit.

Summary: This concludes what we have to say about the implications joint convexity of the cost function we propose. One can see that it leads to quantitative guarantees on rate of convergence in the batch setting and performance guarantees in the online and the adversarial setting. Now we turn our attention the next topic of this paper: large margins.

3.3 Margins

Performant classification loss functions such as hinge loss [22], logistic loss and exponential loss [9] continue to be active even after training error has fallen to zero. For MR such a margin like property is not only beneficial but also essential because otherwise the cost function is degenerate as may be verified by setting $\boldsymbol{w}, \beta = \boldsymbol{0}$. The necessity of this margin property is not mentioned in [1]. Here we take an explicit approach.

By controlling the margin we can also model the notion that errors at the top of the list are more severe than at the bottom. We achieve this by adding linear inequalities and terms. Therefore the properties of strong convexity and Lipschitz continuity of the gradient established in Section 3.1 continue to hold.

We incorporate the margin property in two alternative ways. We augment the cost function (9) by introducing a fixed margin (14) and alternatively a large margin variant (15). In addition to enforcing order in the target vector $\boldsymbol{r}_i$ it enforces (for the fixed margin formulation) or encourages (for the large margin formulation) a gap between the target values of adjacently ordered items $r_{i,j}$, $r_{i,j+1}$. In the formulations (14), (15), the components of $\boldsymbol{t}_i$ denote the gap between the adjacent

targets. In (14) the gaps are pre-specified. It is natural to specify a comparatively higher gap at the top. In (15) the gaps are not specified explicitly, but a reward $\boldsymbol{c}_i$ is awarded per unit gap.

The **fixed margin formulations** is posed in terms of positive pre-prescribed margins $t_{i,j}$ as follows:

$$\min_{\boldsymbol{r}_i,\boldsymbol{w}} \sum_{i=1}^{|\mathcal{Q}|} F_i(\boldsymbol{r}_i,\boldsymbol{w})$$
$$r_{i,j+1} - r_{i,j} \geq t_{i,j} \quad \forall j \in [0, d_i - 1], \forall i \in [1, |\mathcal{Q}|]$$
$$r_{i,0} \geq t_{i,0} \quad \forall i \in [1, |\mathcal{Q}|] \quad (14)$$

The **large margin formulations** are posed in terms of a vector of *rewards* $\boldsymbol{c}_i$ associated with the vector of gaps $\boldsymbol{t}_i > \boldsymbol{0}$ as follows: for every query $q_i i \in \mathcal{Q}$, solve:

$$\min_{\boldsymbol{r}_i,\boldsymbol{w},\boldsymbol{t}_i} \sum_{i=1}^{|\mathcal{Q}|} F_i(\boldsymbol{r}_i,\boldsymbol{w}) - \langle \boldsymbol{c}_i, \boldsymbol{t}_i \rangle$$
$$r_{i,j+1} - r_{i,j} \geq t_{i,j} \geq 0 \quad \forall j \in [0, d_i - 1], \forall i \in [1, |\mathcal{Q}|]$$
$$r_{i,0} \geq t_{i,0} \quad \forall i \in [1, |\mathcal{Q}|], \quad (15)$$

Note that the $\boldsymbol{r}_i$ optimization is a Bregman projection problem. Furthermore, the $\boldsymbol{r}_i'$s are independent and therefore can be projected in parallel. Readers familiar with generalized linear models (GLM) will recognize that the optimization over $\boldsymbol{w}$ is penalized maximum likelihood parameter estimation for GLMs. Since this procedure is standard, we focus on $\boldsymbol{r}$ and $\boldsymbol{t}$ only in the interest of space.

3.4 Bregman Projection on $\mathcal{R}\downarrow_t$

Both the formulations (14) and (15) involve Bregman projections on $\mathcal{R}\downarrow_t$. Elements of $\mathcal{R}\downarrow_{\boldsymbol{t}} \subset \mathbb{R}^n$ are not only sorted but also have separation between adjacent components, given by the vector $\boldsymbol{t}$. In this section we reduce it to a square Euclidean projection on $\text{Argmin}_{\boldsymbol{y} \in \mathcal{R}\downarrow_{\boldsymbol{t}}}$, hence removing the need to solve a non-linear optimization problem. It is quite remarkable that this is possible. For the reduction to hold we need additional assumptions of strong convexity and/or Lipschitz continuity. Consider the problem:

$$\min_{\boldsymbol{r}} D_\phi\big(\boldsymbol{r} \big|\big| (\nabla\phi)^{-1}(A\boldsymbol{w})\big) \text{ s.t. Adj-Diff}(\boldsymbol{r}) \leq \boldsymbol{t}. \quad (16)$$

If $\boldsymbol{t} = \boldsymbol{0}$ this is $\min_{\boldsymbol{r} \in \mathcal{R}\downarrow} D_\phi\big(\boldsymbol{r} \big|\big| (\nabla\phi)^{-1}(A\boldsymbol{w})\big)$. When $\boldsymbol{t}$ is component-wise strictly positive it imposes strict margin between adjacent components of $\boldsymbol{r}$.

Proposition 2. *Let $\phi(\cdot)$ be s strongly convex, then*

$$(\nabla\phi)^{-1}(\boldsymbol{z}^*) = \text{Argmin}_{\boldsymbol{r}}\, D_\phi\big(\boldsymbol{r} \big|\big| (\nabla\phi)^{-1}(A\boldsymbol{w})\big) + \langle \boldsymbol{v}, \boldsymbol{r} \rangle$$
$$\textit{s.t.}\ \text{Adj-Diff}(\boldsymbol{r}) \leq \boldsymbol{t} \quad (17)$$

where $\boldsymbol{z}^* = \text{Argmin}_{\boldsymbol{z}} ||\boldsymbol{z} - A\boldsymbol{w}|| + \langle \boldsymbol{v}, \boldsymbol{r} \rangle$ *s.t.* $\text{Adj-Diff}(\boldsymbol{z}) \leq s\boldsymbol{t}$.

Proof. For the moment let us ignore the term $\langle \boldsymbol{v}, \boldsymbol{r} \rangle$. Let the set of points satisfying the KKT conditions for (16) be $\mathcal{A} = \left\{ \begin{matrix} \boldsymbol{r} \\ \boldsymbol{\lambda} \end{matrix} \middle| \begin{matrix} \nabla\phi(\boldsymbol{r}) = A\boldsymbol{w} - \text{Adj-Diff}(\boldsymbol{\lambda}) \\ \text{Adj-Diff}(\boldsymbol{r}) \leq \boldsymbol{t} \end{matrix} \right\}$, let us denote the KKT points of the optimization problem

$$\min_{\boldsymbol{z}} ||\boldsymbol{z} - A\boldsymbol{w}|| \text{ s.t. Adj-Diff}(\boldsymbol{z}) \leq s\boldsymbol{t} \text{ by } \mathcal{B} =$$

$$\left\{ \begin{matrix} \boldsymbol{z} \\ \boldsymbol{\lambda} \end{matrix} \middle| \begin{matrix} \boldsymbol{z} = A\boldsymbol{w} - \text{Adj-Diff}(\boldsymbol{\lambda}) \\ \text{Adj-Diff}(\boldsymbol{z}) \leq s\boldsymbol{t} \end{matrix} \right\} = \left\{ \begin{matrix} \nabla\phi(\boldsymbol{r}) \\ \boldsymbol{\lambda} \end{matrix} \middle| \begin{matrix} \nabla\phi(\boldsymbol{r}) = A\boldsymbol{w} - \text{Adj-Diff}(\boldsymbol{\lambda}) \\ \text{Adj-Diff}(\nabla\phi(\boldsymbol{r})) \leq s\boldsymbol{t} \end{matrix} \right\}.$$

From $r_{j+1} - r_j \geq t_j$ and strong convexity we have $\nabla\phi(r_{j+1}) - \nabla\phi(r_j) \geq st_j$ thus $\mathcal{A} \subset \mathcal{B}$. Complementary slackness conditions are also verified thus $\mathcal{A}, \mathcal{B}$ are unique minimizers. The term $\langle \boldsymbol{v}, \boldsymbol{r} \rangle$ maintains the relation between $\mathcal{A}$ and $\mathcal{B}$ proving that the minima of the two problems coincide. □

Proposition 3. *Let $\phi(\cdot)$ be strictly convex, $\boldsymbol{t} \leq 0$ and $\nabla\phi(\cdot)$ $\frac{1}{L}$ Lipschitz continuous, then minimizer $\boldsymbol{z}^*$ of (17) is*

$$\boldsymbol{z}^* = \text{Argmin}_{\boldsymbol{z}} ||\boldsymbol{z} - A\boldsymbol{w}|| + \langle \boldsymbol{v}, \boldsymbol{r} \rangle \textit{ s.t. } \text{Adj-Diff}(\boldsymbol{z}) \leq L\boldsymbol{t}.$$

Proof. Define $\mathcal{A}$ and $\mathcal{B}$ as before. From $\nabla\phi(r_{j+1}) - \nabla\phi(r_j) \geq Lt_j$ and Lipschitz continuity we have $r_{j+1} - r_j \geq t_j$ therefore $\mathcal{B} \subset \mathcal{A}$, but $\mathcal{A}$ and $\mathcal{B}$ are unique minimizers. Therefore the proposition holds. □

The implications: of the propositions are, of course, that, for the optimization over $\boldsymbol{r}$, one only needs to implement the square loss variants of (14) and (15) because they are in correspondence with other Bregman divergences as long as the convex function is strongly convex or its gradient is Lipschitz continuous.

The final piece is to show that the reduced quadratic program (QP) is efficiently solvable. This is critical because it is required for the numerical evaluating the gradient (and Hessian) of $G(\boldsymbol{w})$ where we cannot afford the expense of a generic QP solver. We now show how the QP can be solved in linear time.

3.5 Pool Adjacent Violators Algorithm

The pool adjacent violators algorithm [10] solves

$$\min_{\boldsymbol{z}} ||\boldsymbol{z} - A\boldsymbol{w}|| \quad \text{s.t.} \quad \text{Adj-Diff}_*(\boldsymbol{z}) \leq \boldsymbol{0} \quad (18)$$

called the isotonic regression problem. PAV is essentially a block coordinate ascent of the dual of (18). It runs in *finite time* and a straight-forward implementation scales as $\mathcal{O}(d^2)$ in the dimensions. Subsequently [10] observed that if implemented carefully it remarkably has complexity that is linear in d.

The nonlinear optimization problems (14) and (15) from (18). Fortunately, by a series of non-linear and linear change of variables one can reduce these problems to variations of the isotonic regression problem.

3.6 Decomposing the Margin Formulation

For a fixed $\boldsymbol{w}$, a plausible way to optimize (15) is to fix $\boldsymbol{t}_i$ and optimize $\boldsymbol{r}_i$ and alternate, keeping $\boldsymbol{w}$ fixed. One may update $\boldsymbol{w}$ once $\boldsymbol{t}_i$ and $\boldsymbol{r}_i$ converge. This clearly fails because the constraints couple $\boldsymbol{r}_i$ and $\boldsymbol{t}_i$. However, we show that an affine transformation can not only correctly decompose the problem, but also that it separates out the problem out into versions of isotonic regression problems: namely isotonic regression with a lower-bound on the smallest r. Thus it adds another (scalar) constraint to the system Adj-Diff$(\boldsymbol{r}) \leq -\boldsymbol{t}$, where Adj-Diff is the adjacent-difference operator.

Because of the reduction properties shown in Propositions 2 and 3 to estimate $\boldsymbol{r}_i$ in (15) one only needs to consider the problem of the form:

$$\min_{\boldsymbol{r}_i,\boldsymbol{t}_i} \frac{1}{2}||\boldsymbol{r}_i - \boldsymbol{y}_i||^2 - \langle \boldsymbol{c}_i, \boldsymbol{t}_i \rangle \text{ s.t. Adj-Diff}(\boldsymbol{r}_i) \leq -\boldsymbol{t}_i, \;\; \boldsymbol{t}_i > 0.$$

Substituting $\boldsymbol{t}_i = -\text{Adj-Diff}(\boldsymbol{d}_i)$, $\boldsymbol{z}_i = \boldsymbol{r}_i - \boldsymbol{d}_i$ we obtain

$$\frac{1}{2}||\boldsymbol{z}_i + \boldsymbol{d}_i - \boldsymbol{y}_i||^2 + \langle \boldsymbol{c}_i, \text{Adj-Diff}(\boldsymbol{d}_i) \rangle$$
$$\text{s.t.} \quad \text{Adj-Diff}(\boldsymbol{z}_i) \leq 0, \quad \text{Adj-Diff}(\boldsymbol{d}_i) \leq 0. \quad (19)$$

The variables $\boldsymbol{z}_i$ and $\boldsymbol{d}_i$ are completely decoupled, the constraints are the ordering constraints, and if either $\boldsymbol{z}_i$ or $\boldsymbol{d}_i$ fixed, the formulation reduces to an isotonic problem in the other (for $\boldsymbol{d}_i$ some simple algebraic manipulation is necessary to expose the PAV form). Thus, one may alternate over $\boldsymbol{z}_i$ and $\boldsymbol{d}_i$ as follows:

$$\boldsymbol{z}_i^{t+1} = PAV(\boldsymbol{y}_i - \boldsymbol{d}_i^t) \quad (20)$$
$$\boldsymbol{d}_i^{t+1} = PAV(\boldsymbol{y}_i - \boldsymbol{z}_i^{t+1} - \text{Adj-Diff}^{\dagger}(\boldsymbol{c})) \quad (21)$$

and obtain the large margin solution by recovering $\boldsymbol{r}_i, \boldsymbol{t}_i$ from converged $\boldsymbol{z}_i$ and $\boldsymbol{d}_i$.

Problem (14) can be decomposed similarly using propositions 2, 3 and the exact same affine transformation $\boldsymbol{t}_i = -\text{Adj-Diff}(\boldsymbol{d}_i)$ and $\boldsymbol{z}_i = \boldsymbol{r}_i - \boldsymbol{d}_i$. Here however $\boldsymbol{d}_i$ is immediately determined, so no iteration over the variables $\boldsymbol{z}_i$ and $\boldsymbol{d}_i$ is necessary and solving $\boldsymbol{z}_i = PAV(\boldsymbol{y}_i - \boldsymbol{d}_i)$ is sufficient to recover the optimal $\boldsymbol{r}_i$. *Since this requires a single instance of PAV, it is obvious that this converges in finite time, linear in the number of items.*

4 EXPERIMENTS

We evaluated the ranking performance of the proposed margin equipped monotone retargeting (MEMR) approach on the benchmark LETOR 4.0 datasets [18] as well as the OHSUMED dataset [12]. Each of these datasets are pre-partitioned into five-fold validation sets for easy comparison across algorithms. We focus on the variants that use Sqr-loss and KL-divergence because these are strongly convex Bregman divergences. We compare the performance of MEMR against the following strong baselines (i) The MR algorithm as reported in [1] (Recall that the MR algorithm has been shown to outperform many of the current state of the art techniques [1]), (ii) NDCG consistent generalized linear models that also use different Bregman divergences [21] and (iii) max-margin based pairwise learning to rank method RankSVM as implemented by SVMPerf [14] (Note RankSVM as implemented by SVMPerf is a factor of 20 faster than its original implementation in SVMLight). MEMR is implemented in C++ as a minimization method on the function $G(\boldsymbol{w})$. PAV algorithm is used to compute the gradient, and the Hessian. We tried two strategies (i) quasi-Newton using LBFGS [17] and (ii) Trust region truncated Newton (TRON) [16]. While both were an order of magnitude faster than our baselines the latter gave the fastest convergence. The CPU timings of serial implementations on a 2.8 Ghz Intel Quad core processor are reported in Table 1. We parallelized the LBFGS based implementation. The timings and corresponding speedups are shown in Table 2. We found that overprovisioning of threads (8 threads on a quad-core) was necessary to reach full speedup supported by the hardware.

	Sqr.MEMR LBFGS	Sqr.MEMR TRON	Sqr MR	RankSVM
MQ'07	0.166s	**0.101s**	26.396s	17.187s
	KL.MEMR LBFGS	KL.MEMR TRON	KL MR	-
MQ'07	0.326s	**0.199s**	54.15s	

Table 1: CPU time of MEMR and Baselines

HyperThreads	1	2	3	4	8
Sqr.MEMR LBFGS,ms	166	91	72	59	46
Speedup	1	1.8	2.3	2.8	3.6

Table 2: MEMR speedup with parallelism

In our experiments the fixed margin constraints (see equation (14)) were set using different non-increasing functions of the rank. In Figure 2 we show the effect of margins set to different constant values. In Figure 3 we show the effect of margins set by different polynomially decaying functions. The regularization parameter C was selected on the basis of maximum NDCG on

MQ 2007: Mean NDCG (non-truncated)			
	SQ	KL	Hinge
MEMR	0.7491	**0.7564**	-
MR	0.7398	0.6978	-
NDCG consistent GLM [21]	0.7344	0.7399	-
RankSVM	-	-	0.6528

Table 3: Test NDCG on MQ2007 Dataset

OHSUMED: Mean NDCG (non-truncated)			
	SQ	KL	Hinge
MEMR	0.7115	**0.7146**	-
MR	0.6878	0.6997	-
NDCG consistent GLM [21]	0.6892	0.6947	-
RankSVM	-	-	0.6571

Table 4: Test NDCG on OHSUMED Dataset.

the validation set.Figure 4 shows the behavior of the same margin function but for the loss measured by KL divergence.

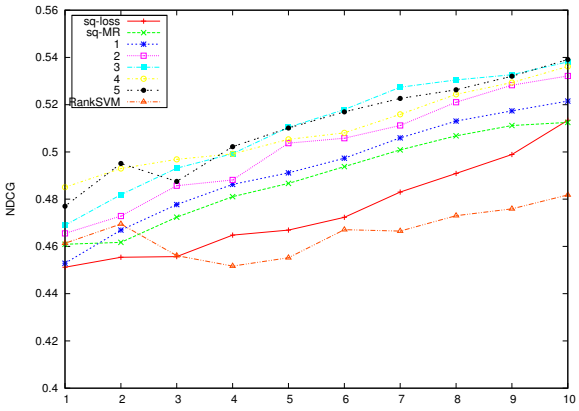

Figure 2: Truncated NDCG@N obtained on MQ2007 using Sqr-loss MEMR with margin between adjacent targets set to $\{0.0625e-3, 0.125e-3, 0.25e-3, .5e-3, 1e-3\}$ respectively showing improved rank quality as margins increase. The plot labeled "Sqr-Loss" represents pointwise NDCG consistent Sqr loss proposed by [21]. Plot labeled "Sqr-MR" corresponds to MR [1] with Sqr-loss.Performance of RankSVM is also shown

5 CONCLUSION

In this paper we presented a margin based monotone retargeting framework for learning to rank. Pointwise ranking methods search for optimal parameters of a regression function to fit the training scores that were specified to define the correct ranking order. MEMR on the other hand searches not only for optimal parameters of a regression function but also over all order-

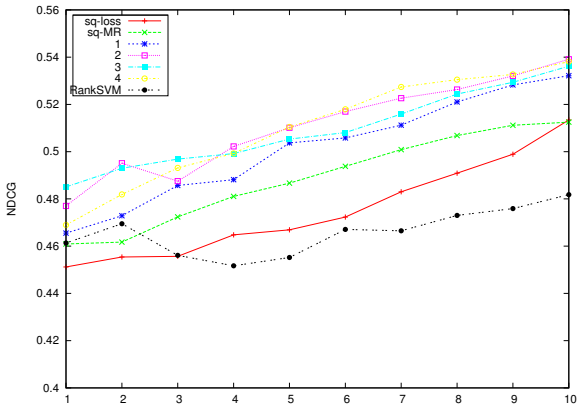

Figure 3: Truncated NDCG@N obtained on MQ2007 using Sqr-loss MEMR with margin between adjacent targets set by function $\frac{C}{\sqrt{\cdot}}$ on the rank associated with the target. Plots shown for values of $C \in \{0.0625e-3, 0.125e-3, 0.25e-3, .5e-3\}$. The baselines are the same as in Figure 2.

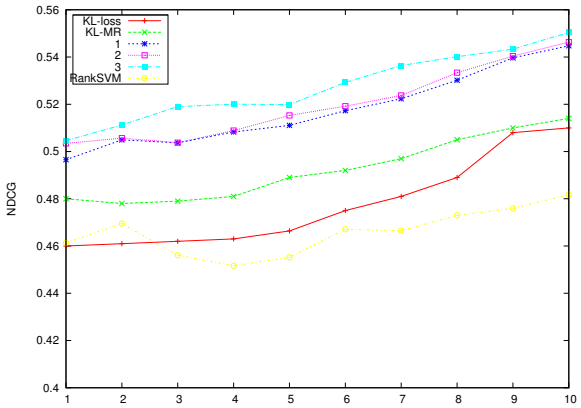

Figure 4: Truncated NDCG@N obtained on MQ2007 using KL-loss MEMR with margin between adjacent targets set by the function $\frac{C}{\sqrt{\cdot}}$ for values of $C \in \{1e-1, 2e-1, 3e-1, 4e-1\}$. The plot labeled "KL-Loss" corresponds KL loss minimizing NDCG consistent GLM [21].

preserving transformations of the training score vectors such that its adjacent components are well separated. The separation property leads to state of the art performance as compared to MR and other max-margin based ranking formulations. Moreover its joint convexity and second order smoothness properties permit efficient algorithms that lead to running times that are a small fraction of competing algorithms, giving almost the best of both worlds: ranking accuracy better than pairwise methods and running times comparable to simple pointwise methods.

References

[1] S. Acharyya, O. Koyejo, and J. Ghosh. Learning to rank with Bregman divergences and monotone retargeting. In *Uncertainty in Artificial Intelligence, UAI*, 2012.

[2] Nir Ailon and Mehryar Mohri. An efficient reduction of ranking to classification. In *Conference on Learning Theory, COLT 2008*, pages 87–98, 2008.

[3] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[4] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

[5] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *24th international conference on Machine learning*, ICML'07, 2007.

[7] Olivier Chapelle, Donald Metlzer, Ya Zhang, and Pierre Grinspan. Expected reciprocal rank for graded relevance. In *18th ACM conference on Information and knowledge management*, CIKM '09, pages 621–630, 2009.

[8] William. Cohen, Robert Schapire, and Yoram Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.

[9] Michael Collins, Robert Schapire, and Yoram Singer. Logistic regression, adaboost and Bregman distances. In Nicolo Cesa-Bianchi and Sally Goldman, editors, *COLT*, pages 158–169, 2000.

[10] S.J. Grotzinger and C. Witzgall. Projections onto order simplexes. *Applied Mathematics and Optimization*, 12:247–270, 1984.

[11] Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69:169–192, 2007.

[12] William Hersh, Chris Buckley, T. J. Leone, and David Hickam. OHSUMED: an interactive retrieval evaluation and new large test collection for research. SIGIR '94, pages 192–201, 1994.

[13] Kalervo Järvelin and Jaana Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *23rd ACM SIGIR conference on Research and development in information retrieval*, SIGIR '00, pages 41–48, 2000.

[14] T. Joachims. Training linear SVMs in linear time. In *KDD'06 Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006.

[15] Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132, 1995.

[16] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust region newton method for logistic regression. *J. Mach. Learn. Res.*, 9:627–650, June 2008.

[17] Dong C. Liu, Jorge Nocedal, Dong C. Liu, and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[18] Tie-Yan Liu, Jun Xu, Tao Qin, Wenying Xiong, and Hang Li. LETOR: Benchmark dataset for research on learning to rank for information retrieval. In *SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, 2007.

[19] C. E. McCulloch and S. R. Searle. *Generalized Linear and Mixed Models*. John Wiley & Sons, 2001.

[20] Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *ICML*, 2012.

[21] Pradeep Ravikumar, Ambuj Tewari, and Eunho Yang. On NDCG consistency of listwise ranking methods. In *Proceedings of 14th International Conference on Artificial Intelligence and Statistics*, AISTATS, 2011.

[22] Mark Reid and Robert C Williamson. Information, divergence and risk for binary experiments. *Journal of Machine Learning Research*, 12:731–817, 2011.

[23] R T. Rockafellar. *Convex Analysis (Princeton Landmarks in Mathematics and Physics)*. Princeton University Press, December 1996.

[24] Markus Weimer, Alexandros Karatzoglou, Quoc V. Le, and Alex J. Smola. CoFi RanK - maximum margin matrix factorization for collaborat ive ranking. In *NIPS*, 2007.

[25] Jason Weston and John Blitzer. Latent structured ranking. In *Uncertainty in Artificial Intelligence, UAI 2012*, 2012.