Multi-objective ranking with directions of preferences

Michinari Momma
michi@amazon.com
Amazon

Chaosheng Dong
chaosd@amazon.com
Amazon

Yetian Chen
yetichen@amazon.com
Amazon

ABSTRACT

Recently, gradient based multi-objective optimization methods have been developed to find models that are aligned with preference directions (MOO-PD) in machine learning community. Most of the methods are tuned and tested with multi-task learning problems in computer vision tasks with deep neural networks. While MOO-PD is useful in building a model with user specified MOO criteria, there is no existing work in the learning-to-rank (LTR) applications with gradient boosted ranking trees (GBRT), which is a popular method in LTR especially in production systems. Hence, there is no evidence demonstrating that existing MOO-PD methods work well for LTR. In this paper, we apply several MOO-PD methods such as the Exact Pareto Optimal search, etc. to LTR. Further, to quantify model performance on MOO-PD, we propose a novel model evaluation metric, which is referred to as the maximum weighted loss. Through experiments, we reveal common challenges with MOO-PD methods, and propose a smoothing technique to address the challenges. The revised algorithms are shown to significantly improve the empirical performance on both public and proprietary datasets, indicating that we now have a realistic way to build MOO-PD models in GBRT, which may benefit many application use cases in practice.

KEYWORDS

learning to rank, multi-objective optimization, multi-task learning, gradient boosted ranking tree

ACM Reference Format:
Michinari Momma, Chaosheng Dong, and Yetian Chen. 2022. Multi-objective ranking with directions of preferences. In *Proceedings of ACM SIGIR Workshop on eCommerce (SIGIR eCom'22).* ACM, New York, NY, USA, 5 pages.

1 INTRODUCTION

Learning to Rank (LTR) [17] is a machine learning framework to learn a *scoring function* to rank items according to some criteria such as relevance to a given query. Applications of LTR are diverse and include web search ranking [7], product search [31], recommender systems [14], question answering [1], machine translation [11], etc. As LTR is a prominent framework in information retrieval (IR), characteristics in IR problems such as multi-dimensionality of relevance concept [2, 22] are naturally inherited to LTR – the multi-objective optimization in LTR (MOO-LTR)[30, 32, 33] has emerged as an important topic for research and applications in production systems. For example, in web search, there are multiple metrics, such as click through rate, relevance, dwell-time, etc., all of which need to be simultaneously optimized. Further, in product search, while the primary objectives would be purchase, other factors such as relevance, surfacing popular brands, etc., are factors needed to be optimized in the ranking model. Typically, in MOO-LTR, these objectivees tend to conflict to each other since there is limited space for ranking, making optimality in each objective not achievable.

In the presence of conflicts, the multi-objective optimization (MOO) problem is characterized by *Pareto optimality* (PO) where there is no model that is superior to others (i.e., dominate) in all objectives. In practice, discovery / understanding of *trade-off* needs to be done in model selection of MOO-LTR models for online testing and production deployment. A set of Pareto optimal models is referred to as *Pareto Frontier* (PF). The rich history of MOO research has introduced several ways for specifying a trade-off [21], e.g., setting priorities on objectives (Linear Scalarization, Chebyshev Scalarization, etc.) or constraining them (e.g., $\epsilon-$Constraint method), that could lead to a PO model.

Recently, building models over the entire PF has become a popular research topic in the application of multi-task learning (MTL) [15, 16, 19, 23, 25, 29]. Typically, concepts/methods of MOO are applied to MTL problems with conflicting tasks, which generate PF of models over multiple objectives where a task is optimized based on an objective associated with the task. A popular line of search is to find a model on PF with a vector of preference [15, 19, 23, 25]. Specifically, given a positive vector $\mathbf{r} \in \mathbb{R}_+^K$ representing preference across objectives, we want a model that minimizes each loss function according to the preference. In the loss function space, this direction is expressed as a ray of vector $\mathbf{r}^{-1}$ whose component is inverse of that in $\mathbf{r}$, which is referred to as "$\mathbf{r}^{-1}$-ray". The problem in MOO with the specified $\mathbf{r}$ is thus denoted as MOO with preference direction (MOO-PD).

In this paper, we investigate application of various MOO-PD methods to the MOO-LTR problems. We note that this extension is not a trivial task for two main reasons: model types and applications. First, MOO-PD methods are typically tested on computer vision problems in deep learning frameworks. However, in LTR, many production systems leverage the gradient boosted ranking model (GBRT). Second, the ranking dataset / features are typically structured. Hence, it is unknown if existing methods work well in MOO-LTR, or require modifications to achieve Pareto optimality that is a requirement to deploy MOO-PD models in production systems.

Contributions. Our contributions are summarized as follows; (1) We apply various MOO-PD methods to MOO-LTR problems with GBRT, and evaluate how the model satisfies the preference information. (2) We define a metric called maximum weighted loss, to evaluate the performance of MOO-PD models. (3) We investigate how to improve existing MOO-PD methods so that the models built by them can achieve PO.

SIGIR eCom'22, July 15, 2022, Madrid, Spain

2 MOO METHODS IN GBRT AND EVALUATION METRIC

In this section, we review MOO in GBRT and several existing MOO methods. GBRT is based on the Gradient Boosted Tree [12], where gradient vector is used as the target label to build a decision tree, given task dependent loss function and gradient. In GBRT, we use ranking loss function such as RankNet [8], LambdaRank [6], etc., [4, 5, 26, 28]. Thus, applying MOO in GBRT has been focusing on how to generate gradient, which is done by considering multiple cost functions. Given model scores $\mathbf{s} \in \mathbb{R}^M$ for all queries and items, K-objectives and associated cost functions c_k, in MOO-LTR, the cost is a vector valued function

$$\mathbf{c}(\mathbf{s}) = [c_1(\mathbf{s}), \cdots, c_K(\mathbf{s})]^T. \tag{1}$$

Formally, a model that generates scores $\mathbf{s}$ is said to **dominate** another model with $\hat{\mathbf{s}}$ if and only if $c_i(\mathbf{s}) \leq c_i(\hat{\mathbf{s}})$, $\forall i \in I$ and $c_j(\mathbf{s}) < c_j(\hat{\mathbf{s}})$, $\exists j \in I$, where $I \in [M]$. A model with score $\hat{\mathbf{s}}$ is said to be **Pareto optimal** if there is no other model that dominates the model with $\hat{\mathbf{s}}$. A model with $\hat{\mathbf{s}}$ is said to be **weak Pareto optimal** when there is no other model $\mathbf{s}$ s.t., $c_i(\mathbf{s}) < c_i(\hat{\mathbf{s}})$, $\forall i \in I$, which implies equality in determining dominance is allowed to be weak PO while PO mandates strict inequality in costs.

MOO methods are characterized by types of information they use to build a PO model on PF that is a $K-1$ dimensional manifold [13]. The MOO-LTR loss function in (1) gives rise to K score-gradients, $\nabla_{\mathbf{s}} c_k$ for $k \in [K]$. However, for training the GBRT based scoring function, the t^{th} decision tree requires exactly one score-gradient as labels, not K score-gradients. Therefore, the constituent trees require scalar labels. We combine the K score-gradients as

$$\boldsymbol{\lambda} = \sum_{k=1}^{K} \alpha_k \nabla_{\mathbf{s}} c_k, \quad \text{s.t.} \sum_{k=1}^{K} \alpha_k = 1, \quad \boldsymbol{\alpha} \in \mathbb{R}_+^K, \tag{2}$$

and use $\boldsymbol{\lambda} \in \mathbb{R}^M$ as the labels for training the trees in GBRT. The combination coefficients $\boldsymbol{\alpha} \in \mathbb{R}_+^K$ have $K-1$ free parameters, and are derived from the trade-off specification.

2.1 MOO methods in GBRT

2.1.1 ***Linear Scalarization (LS).*** In LS, the MOO cost in (1) is converted to a scalar cost

$$g_{\mathbf{r}}^{\text{LS}}(\mathbf{s}) = \sum_{k=1}^{K} r_k c_k(\mathbf{s}), \tag{3}$$

where $\mathbf{r} \in \mathbb{R}_+^K$ represents preferences/weights given to the costs. The coefficient vector in LS is

$$\boldsymbol{\alpha} = \mathbf{r}/\|\mathbf{r}\|_1, \tag{4}$$

and remains constant throughout the iterations. The preference vector $\mathbf{r}$ is an element in the dual space [18] of the objective space. It represents a hyperplane in the objective space. Let $\mathbf{s}_{\mathbf{r}}^* = \arg\min_{\mathbf{s}} g_{\mathbf{r}}(\mathbf{s})$ be the PO solution corresponding to $\mathbf{r}$. Then the hyperplane of $\mathbf{r}$ passing through the point $\mathbf{c}(\mathbf{s}_{\mathbf{r}}^*)$ has to be both a tangent and a support to the PF. Although LS provides a simple combination strategy (4), there are several limitations. If any of the costs is a non-convex function, LS can not guarantee to reach all points in the PF by varying the preferences [3]. Moreover, a single preference specification can have non-unique PO solutions.

2.1.2 ***Stochastic Label Aggregation (SLA).*** In SLA, one gradient is randomly chosen for a query:

$$\alpha_k = \begin{cases} 1, & \text{if } k = \overline{K}, \\ 0, & \text{otherwise,} \end{cases} \quad \text{for } k \in [K] \tag{5}$$

where $\overline{K}$ is a categorical random variable over the K indices with $\mathbf{r}/\|\mathbf{r}\|_1$ as its parameters of the distribution. The expected cost of SLA is the same as the cost of LS in (3) [9]. Thus, SLA can be seen as a special type of LS.

2.1.3 ***Chebyshev Scalarization (CS).*** In CS, the cost $\mathbf{c}(\mathbf{s})$ in (1) is scalarized to

$$g_{\mathbf{r}}^{\text{CS}}(\mathbf{s}) = \max_{k \in [K]} r_k c_k(\mathbf{s}). \tag{6}$$

To generate gradient, only the gradient of maximum relative objective value is chosen:

$$\alpha_k = \begin{cases} 1, \text{ if } k = k^*, \\ 0, \text{ otherwise,} \end{cases} \quad \text{s.t. } k^* = \arg\max_{k \in [K]} r_k c_k(\mathbf{s}). \tag{7}$$

As seen in (6), CS solves for ℓ_∞-norm of $\mathbf{c}$ weighted by $\mathbf{r}$. Geometrically, it forms a hyper-rectangular level set along with $\mathbf{r}^{-1}$, which is used to explore PF. With CS, PO points in the PF can be reached by varying the preferences, even when the objectives are non-convex. Moreover, apart from some corner cases [21] of weak PO, the CS method guarantees that the final solution is unique. The weakness in CS is non-smoothness optimization with ℓ_∞-norm minimization, which causes oscillations of solutions.

2.1.4 ***Exact Pareto Optimal Search (EPO).*** In EPO [19, 20], the trade-off specification is the same as that of CS. Therefore, most properties of CS are inherited. However, to overcome the limitations of CS, its gradient combination is designed to avoid oscillations around the $\mathbf{r}^{-1}$ ray. The coefficients are obtained by solving a K dimensional quadratic program:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}_+^K} \|\mathrm{C}^T \mathrm{C} \boldsymbol{\alpha} - \mathbf{a}\|_2^2, \quad \text{s.t.} \sum_{k=1}^{K} \alpha_k = 1, \tag{8}$$

where $\mathrm{C} \in \mathbb{R}^{M \times K}$ is the matrix with K gradients in its column, and $\mathbf{a}$ is an *anchor direction* in the objective space that determines the first order change in cost vector: $\mathbf{c}^{t+1} - \mathbf{c}^t \approx \delta \mathbf{c} = \mathrm{C}^T \mathrm{C} \boldsymbol{\alpha}$ from Taylor series expansion of $\mathbf{c}(\mathbf{s}^t - \mathrm{C}\boldsymbol{\alpha})$. Here, $\mathbf{a}$ is determined by

$$\mathbf{a} = \begin{cases} \mathbf{c}^t - \frac{\langle \mathbf{c}^t, \mathbf{r}^{-1} \rangle}{\|\mathbf{r}^{-1}\|_2} \mathbf{r}^{-1}, & \text{if } \mathbf{c}^t \text{ is far,} \\ \mathbf{r}^{-1}, & \text{otherwise.} \end{cases} \tag{9}$$

When $\mathbf{c}^t$ is far (w.r.t. cosine distance) from $\mathbf{r}^{-1}$ ray, the anchor is orthogonal to the $\mathbf{r}^{-1}$ ray and directed towards it. When $\mathbf{c}^t$ is near $\mathbf{r}^{-1}$ ray, we move the cost along the $\mathbf{r}^{-1}$ ray to avoid oscillations.

2.2 Evaluation metric on MOO-PD

To quantify the performance on MOO-PD, we propose to use objective function of CS (i.e., (6)), which captures alignment with the $\mathbf{r}^{-1}$-ray, and refer to it as the **maximum weighted loss (MWL)**. Figure 1 illustrates a prototypical case with 3 models. In terms of MWL, M_1 and M_3 are the same, although M_3 dominates M_1. As a tiebreaker, we use the volume of intersection between negative orthant (VNO) pivoted by each model and $\mathbb{R}_+^K$ (shaded area in Figure

1). Note that VNO should always be used as a tiebreaker when the difference in MWL is insignificant. Compared with M_2, M_1 should be preferable as it has a smaller MWL and thus is better aligned with $\mathbf{r}^{-1}$-ray, although neither one dominates the other.

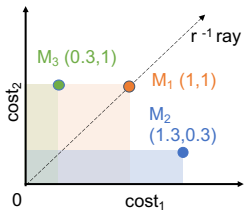


Figure 1: Illustration of MWL metric. M_1 and M_3 are aligned with preference direction. M_2 is the worst. For M_1 and M_3, we need to use the negative orthant area as a tiebreaker, to identify M_3 is best overall.

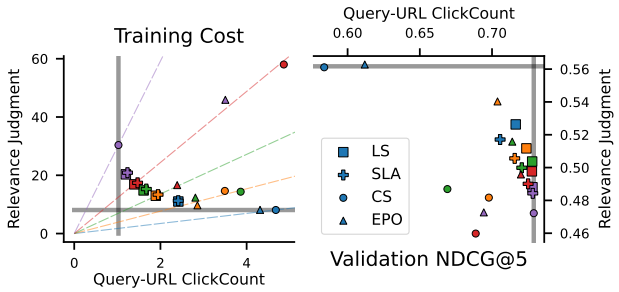


Figure 2: Quality Score vs. Relevance Judgment for existing MOO methods. Same color is used for $\mathbf{r}^{-1}$-ray and the corresponding models to distinguish results from different rays.

3 EXPERIMENTS ON MICROSOFT DATASET

Dataset: We test MOO methods on Microsoft Learning to Rank web search dataset (MSLR-WEB30K) [27] with 30,000 queries. Each query-url pair is represented by a 136 dimensional feature vector. The labels are in the form of a relevance judgment (Rel) of five grades. To construct multiple labels, we follow Momma et al. [24], and use four of its 136 features, viz., *Query-URL Click Count* (Click), *URL Dwell Time* (Dwell), *Quality Score* (QS1) and *Quality Score2* (QS2)[1], as additional relevance labels, and remove them from the feature list to avoid data leakage.

Experimental setting For bi-objective experiments, we select four pairs of labels: (Click, Rel), (QS1, Rel), (QS1, QS2), (Click, Dwell) as objectives. Five $\mathbf{r}^{-1}$ rays are generated in the cost space to be equally distributed over the region between the two baseline cost vectors that are the costs for single objective models. While training cost is used to measure alignment with the ray, test NDCG@5 is used to compare the ranking performance. For three objective case, we choose (QS1, QS2, Rel) and (Click, Dwell, Rel), and generate equi-distributed 25 preference directions. We tune the hyperparameters, i.e., number-of-trees and learning rate, of GBRT model by optimizing Rel. We choose 600 trees and 0.25 learning rate, after validating on the grid of $\{300, 600, 900, 1200\}$ trees and $\{0.05, 0.15, 0.25, 0.35\}$ learning rates, using NDCG@5. We use two folds (1 and 2) available in [27], and aggregate results. For significance test, we collect pairs of observations from different methods for each ray, and conduct paired t-Test with 0.05 as significance level.

3.1 Initial Experiment

As an initial experiment to apply existing methods in their original form, we run LS, SLA, CS and EPO on (Click, Rel) pair (Figure 2). While the simplest baseline, LS, performs well and seems to achieve PO, others are inferior to it. For SLA, it is dominated by LS for most of the cases in both cost and NDCG results (cf. square and plus markers of the same colors in Figure 2). Further, performance of CS and EPO is unstable and inferior to LS. To understand this, we plot cost function curves in Figure 3. Clearly, LS is the only method that has a smooth behavior. As SLA samples one label per query, and would change over different queries, there are non-smooth changes, and this disruption seems to cause inferior performance in terms of PO. For CS, the behavior is expected, as it chooses one label that maximizes the weighted cost, which causes oscillations. However, for EPO, though it uses *fixed* $\mathbf{r}^{-1}$ as anchor when cost is close to the $\mathbf{r}^{-1}$-ray, it easily deviates from it, thus requiring correction steps to move the model closer to $\mathbf{r}^{-1}$ again. Clearly, the mechanism of EPO, as a smoother version of CS, is broken and oscillation occurs, resulting in performance degradations. Same issues are seen for all pairs (and all datasets in this paper), although page limitation does not allow us to show.

[1] As the two quality scores in this dataset represent badness scores, we linearly convert them to be treated as goodness scores.

3.2 Remedy by smoothing

To mitigate the issues, there are couple of approaches: (1) Formulating each problem with Augmented Lagrangian, which has been successfully applied to solving ϵ-constraint problem [25]; (2) Simply taking moving average to force smoothing $\boldsymbol{\alpha}$:

$$\boldsymbol{\alpha}^{t+1} = \nu\boldsymbol{\alpha}^{t} + (1-\nu)\boldsymbol{\alpha}^{t-1},\ 0 < \nu < 1, \tag{10}$$

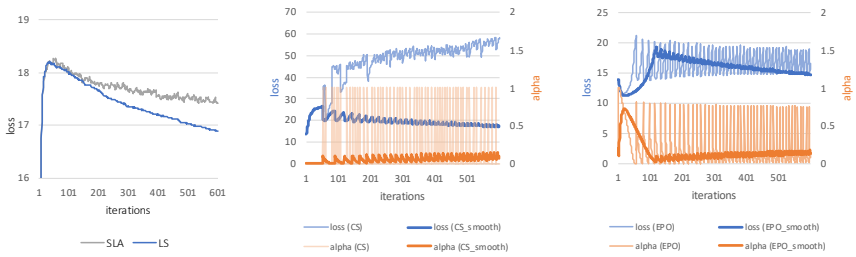
for each iteration t. We test (2) as a quick way to address the issue, and set $\nu = 0.1$ throughout this paper. Cost curves of the smoothed versions are shown in Figure 3 as bold colors. The cost/NDCG result with smoothed $\boldsymbol{\alpha}$ is shown in Figure 4. The improvement is evident when comparing the original methods (smaller markers) to the smoothed versions (larger markers). After the smoothing, the models follow similar PF curves as that of LS. To quantify the improvement, we compute MWL for test loss as the primary metric, and hypervolume indicator (HI) as a secondary metric to check the PO, and averaged over all runs (Table 1). Note LS is a smoother version of SLA, since LS uses fixed $\boldsymbol{\alpha}$ whereas SLA samples from it. For all models, the smoothed version shows improvements on HI and MWL. Particularly, improvements on the metrics on CS are remarkable, making it the single best performer in MWL with competitive performance in HI with EPO.

4 EXPERIMENTS ON OTHER DATASETS

We further validate our analysis by the Yahoo Dataset and one proprietary E-commerce Dataset.

4.1 Yahoo Dataset

We experiment on the Yahoo Learning to Rank (YLTR) [10] challenge dataset with 36K queries. Each query-url pair is represented by 700 features. Although these features are engineered (not learnt),



(a) cost curve for SLA/LS (b) loss curve for CS (c) loss curve for EPO

Figure 3: Cost curves for (a) SLA/LS, (b) CS and (c) EPO for (Click, Rel). For CS and EPO, we also show α associated with cost of Rel. We use light color for the original methods and bold for smoothed versions.

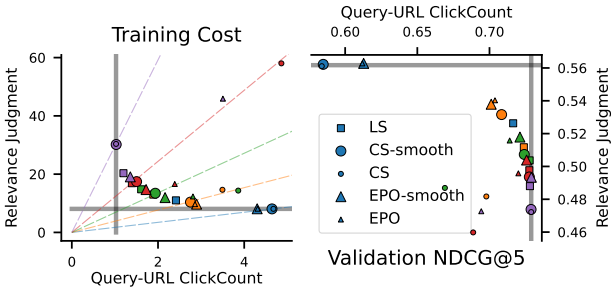


Figure 4: Improvement by smoothing α Smaller (larger) markers indicate original (smoothed) methods.

Table 1: Metrics on MLSR dataset. "orig" refers to SLA and non-smooth (original) versions of CS / EPO. "smt" refers to LS and smoother version of CS / EPO. Bold numbers means statistical significance between orig and smt. Red number refers to a single winner (i.e., significance vs. all others).

	MWL (test)			HI (train cost)			HI (test NDCG)		
	orig	smt	gain%	orig	smt	gain%	orig	smt	gain%
2-obj									
SLA/LS	2.24	**2.09**	-6.7%	3.51	**3.55**	1.03%	0.93	**0.96**	2.20%
CS	5.14	**1.97**	-61.7%	3.41	**3.55**	4.21%	0.95	**0.97**	2.00%
EPO	2.57	**2.02**	-21.2%	3.51	**3.56**	1.70%	0.95	**0.97**	1.70%
3-obj									
SLA/LS	2.01	**1.86**	-7.5%	6.37	**6.52**	2.40%	0.79	**0.84**	7.09%
CS	10.5	**1.75**	-83.3%	6.06	**6.57**	8.57%	0.81	**0.89**	9.80%
EPO	2.65	**1.90**	-28.3%	6.45	**6.61**	2.40%	0.87	**0.88**	1.69%

their descriptions, however, are not publicly released. Hence, we select additional 5 labels, which have 5+ levels of values and least correlation vs. each other. Note as we see cost vanishing behavior coming from NDCG computation within LambdaRank due to low granularity, we use RankNet cost [8], which is the pairwise cost without NDCG factors. For tuning the model hyperparameters, we follow a similar strategy as for MSLR-WEB30K, and select 600 trees and 0.25 for the learning rate. The results are summarized in Table 2. Similar to MSLR, we see significant gains in MWL and HI via smoothing except for EPO. EPO shows significantly worse performance, which even smoothing did not help. CS (smooth) is the single best model in this case as well.

4.2 E-commerce Dataset

We test MOO methods on E-commerce dataset. This dataset is proprietary at the time of writing, and have similarity with dataset used

Table 2: Evaluation metrics on Yahoo dataset.

	MWL (test)			HI (train cost)			HI (test NDCG)		
	orig	smt.	gain%	orig	smt.	gain%	orig	smt.	gain%
2-obj									
SLA/LS	91.7	**86.2**	-6.00%	3.28	**3.32**	1.15%	0.94	**0.95**	1.85%
CS	87.3	**81.0**	-7.28%	3.28	**3.34**	1.86%	0.95	**0.96**	0.91%
EPO	107.3	107.4	0.07%	3.16	3.16	-0.01%	0.87	0.87	-0.05%
3-obj									
SLA/LS	70.60	**66.72**	-5.49%	6.14	**6.23**	1.42%	0.84	**0.88**	5.17%
CS	72.88	**60.46**	-17.04%	6.24	**6.40**	2.54%	0.88	**0.91**	3.28%
EPO	82.05	**82.00**	-0.06%	5.91	5.90	-0.14%	0.73	0.73	-0.15%

in [25, 31]. The data is collected in 2021 for a particular shopping site. We sample ~70K queries for training and ~30K queries for test, and repeat two times for gaining power of the statistical test. We create 4 labels, consisting of (historical) purchase, relevance score, brandedness and shipping speed. The results are summarized in Table 3. In this dataset as well, performance of all model is significantly boosted by smoothing, and CS (smooth) is the best model.

Table 3: Evaluation metrics on E-commerce dataset.

	MWL (test)			HI (train cost)			HI (test NDCG)		
	orig	smt.	gain%	orig	smt.	gain%	orig	smt.	gain%
2-obj									
SLA/LS	1.72	**1.69**	-2.24%	3.29	**3.30**	0.34%	0.94	**0.95**	1.14%
CS	18.0	**1.55**	-91.4%	2.74	**3.34**	22.2%	0.88	**0.98**	10.2%
EPO	4.13	**1.83**	-55.7%	3.27	**3.30**	1.07%	0.93	**0.95**	2.25%
3-obj									
SLA/LS	1.34	**1.28**	-4.47%	7.04	**7.09**	0.66%	0.86	**0.89**	3.30%
CS	21.3	**1.12**	-94.7%	5.39	**7.13**	32.1%	0.80	**0.93**	15.4%
EPO	2.53	**1.30**	-48.5%	7.11	**7.15**	0.49%	0.90	**0.91**	1.54%

5 CONCLUSIONS

In this paper, we applied existing multi-objective optimization methods with preference directions (MOO-PD) to learning to rank (LTR) problems. Although MOO-PD methods are developed and validated in multi-task learning scenarios on mainly computer vision tasks, their applications to LTR problems remains a novel research topic. Indeed, simply applying the methods failed due to high sensitivity of non-smooth optimization steps to the performance in LTR. To address the issue, we proposed to use exponential moving averaging of the gradient coefficient. We verified that the issues exist in all three LTR datasets, and the remedy worked effectively for all cases. In future work, we would further explore Augmented Lagrangian for smoothing and verify the smoothing method with other MOO methods on LTR datasets.

REFERENCES

[1] Arvind Agarwal, Hema Raghavan, Karthik Subbian, Prem Melville, Richard D. Lawrence, David C. Gondek, and James Fan. 2012. Learning to Rank for Robust Question Answering. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management* (Maui, Hawaii, USA) *(CIKM '12)*. Association for Computing Machinery, New York, NY, USA, 833–842. https://doi.org/10.1145/2396761.2396867

[2] Pia Borlund. 2003. The Concept of Relevance in IR. *J. Am. Soc. Inf. Sci. Technol.* 54, 10 (Aug. 2003), 913–925. https://doi.org/10.1002/asi.10286

[3] Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press. https://doi.org/10.1017/CBO9780511804441

[4] Sebastian Bruch. 2021. An Alternative Cross Entropy Loss for Learning-to-Rank. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 118–126. https://doi.org/10.1145/3442381.3449794

[5] Sebastian Bruch, Masrour Zoghi, Michael Bendersky, and Marc Najork. 2019. Revisiting Approximate Metric Optimization in the Age of Deep Neural Networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 1241–1244. https://doi.org/10.1145/3331184.3331347

[6] C. Burges. 2010. From RankNet to LambdaRank to LambdaMART: An Overview.

[7] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *Proceedings of the 22nd International Conference on Machine Learning* (Bonn, Germany) *(ICML '05)*. Association for Computing Machinery, New York, NY, USA, 89–96. https://doi.org/10.1145/1102351.1102363

[8] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. 89–96.

[9] David Carmel, Elad Haramaty, Arnon Lazerson, and Liane Lewin-Eytan. 2020. Multi-Objective Ranking Optimization for Product Search Using Stochastic Label Aggregation. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 373–383. https://doi.org/10.1145/3366423.3380122

[10] Olivier Chapelle and Yi Chang. 2011. Yahoo! Learning to Rank Challenge Overview. In *Proceedings of the Learning to Rank Challenge (Proceedings of Machine Learning Research, Vol. 14)*, Olivier Chapelle, Yi Chang, and Tie-Yan Liu (Eds.). PMLR, Haifa, Israel, 1–24. https://proceedings.mlr.press/v14/chapelle11a.html

[11] Huadong Chen, Shujian Huang, David Chiang, Xinyu Dai, and Jiajun Chen. 2017. Top-Rank Enhanced Listwise Optimization for Statistical Machine Translation. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Association for Computational Linguistics, Vancouver, Canada, 90–99. https://doi.org/10.18653/v1/K17-1011

[12] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[13] Claus Hillermeier. 2001. *Nonlinear Multiobjective Optimization*. Birkhäuser Basel. https://doi.org/10.1007/978-3-0348-8280-4

[14] Alexandros Karatzoglou, Linas Baltrunas, and Yue Shi. 2013. Learning to Rank for Recommender Systems. In *Proceedings of the 7th ACM Conference on Recommender Systems* (Hong Kong, China) *(RecSys '13)*. Association for Computing Machinery, New York, NY, USA, 493–494. https://doi.org/10.1145/2507157.2508063

[15] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. 2019. Pareto Multi-Task Learning. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/685bfde03eb646c27ed565881917c71c-Paper.pdf

[16] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. 2021. Conflict-Averse Gradient Descent for Multi-task Learning. *Advances in Neural Information Processing Systems* 34 (2021).

[17] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. *Found. Trends Inf. Retr.* 3, 3 (mar 2009), 225–331. https://doi.org/10.1561/1500000016

[18] David G. Luenberger. 1997. *Optimization by Vector Space Methods* (1st ed.). John Wiley & Sons, Inc., USA.

[19] Debabrata Mahapatra and Vaibhav Rajan. 2020. Multi-Task Learning with User Preferences: Gradient Descent with Controlled Ascent in Pareto Optimization. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 6597–6607. http://proceedings.mlr.press/v119/mahapatra20a.html

[20] Debabrata Mahapatra and Vaibhav Rajan. 2021. Exact Pareto Optimal Search for Multi-Task Learning: Touring the Pareto Front. arXiv:2108.00597 [cs.LG]

[21] Kaisa Miettinen. 1998. *Nonlinear multiobjective optimization*. International series in operations research and management science, Vol. 12. Kluwer.

[22] S. Mizzaro. 1998. How many Relevances in Information Retrieval? *Interacting With Computers* 10, 3 (1998), 305–322.

[23] Michinari Momma, Chaosheng Dong, and Jia Liu. 2022. A multi-objective / multi-task learning framework induced by Pareto stationarity. In *Proceedings of the 39th International Conference on Machine Learning*.

[24] Michinari Momma, Alireza Bagheri Garakani, Nanxun Ma, and Yi Sun. 2020. Multi-objective relevance ranking via constrained optimization. In *The Web Conference 2020*.

[25] Michinari Momma, Alireza Bagheri Garakani, and Yi Sun. 2019. Multi-objective Relevance Ranking. In *eCOM@ SIGIR*.

[26] Przemyslaw Pobrotyn and Radoslaw Bialobrzeski. 2021. NeuralNDCG: Direct Optimisation of a Ranking Metric via Differentiable Relaxation of Sorting. *CoRR* abs/2102.07831 (2021). arXiv:2102.07831 https://arxiv.org/abs/2102.07831

[27] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013). http://arxiv.org/abs/1306.2597

[28] Tao Qin, Tie-Yan Liu, and Hang Li. 2010. A General Approximation Framework for Direct Optimization of Information Retrieval Measures. *Inf. Retr.* 13, 4 (Aug. 2010), 375–397. https://doi.org/10.1007/s10791-009-9124-x

[29] Michael Ruchte and Josif Grabocka. 2021. Scalable Pareto Front Approximation for Deep Multi-Objective Learning. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*.

[30] Ashudeep Singh and Thorsten Joachims. 2019. *Policy Learning for Fairness in Ranking*. Curran Associates Inc., Red Hook, NY, USA.

[31] Daria Sorokina and Erick Cantu-Paz. 2016. Amazon Search: The Joy of Ranking Products. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 459–460. https://doi.org/10.1145/2911451.2926725

[32] Krysta M. Svore, Maksims N. Volkovs, and Christopher J.C. Burges. 2011. Learning to Rank with Multiple Objective Functions. In *Proceedings of the 20th International Conference on World Wide Web* (Hyderabad, India) *(WWW '11)*. Association for Computing Machinery, New York, NY, USA, 367–376. https://doi.org/10.1145/1963405.1963459

[33] Joost van Doorn, Daan Odijk, Diederik M. Roijers, and Maarten de Rijke. 2016. Balancing Relevance Criteria through Multi-Objective Optimization. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 769–772. https://doi.org/10.1145/2911451.2914708