Multi-Label Learning to Rank through Multi-Objective Optimization

Debabrata Mahapatra
debabrata@u.nus.edu
National University of Singapore
Singapore

Chaosheng Dong*
chaosd@amazon.com
Amazon
Seattle, WA, USA

Yetian Chen
yetichen@amazon.com
Amazon
Seattle, WA, USA

Michinari Momma*
michi@amazon.com
Amazon
Seattle, WA, USA

ABSTRACT

Learning to Rank (LTR) technique is ubiquitous in Information Retrieval systems, especially in search ranking applications. The relevance labels used to train ranking models are often noisy measurements of human behavior, such as product ratings in product searches. This results in non-unique ground truth rankings and ambiguity. To address this, Multi-Label LTR (MLLTR) is used to train models using multiple relevance criteria, capturing conflicting but important goals, such as product quality and purchase likelihood for improved revenue in product searches. This research leverages Multi-Objective Optimization (MOO) in MLLTR and employs modern MOO algorithms to solve the problem. A general framework is proposed to combine label information to characterize trade-offs among goals, and allows for the use of gradient-based MOO algorithms. We test the proposed framework on four publicly available LTR datasets and one E-commerce dataset to show its efficacy.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Multi-task learning**; • **Information systems** → **Learning to rank**.

KEYWORDS

Learning to Rank, Multi-Objective Optimization

ACM Reference Format:
Debabrata Mahapatra, Chaosheng Dong, Yetian Chen, and Michinari Momma. 2023. Multi-Label Learning to Rank through Multi-Objective Optimization. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3580305.3599870

1 INTRODUCTION

The field of Learning to Rank (LTR) has seen significant growth in recent years due to the availability of large amounts of labeled data for query-item relevance, either obtained through manual labeling or user behavior tracking. LTR aims to train a scoring function that assigns a relevance score to each retrieved item in order to rank them in the final results. Initially, LTR relied on a single criterion for relevance, but this uni-dimensional approach has been criticized for its limitations, such as subjectivity and noise in relevance articulation and an inability to account for multiple goals. To address these limitations, Multi-Label Learning to Rank (MLLTR) was introduced, which utilizes a multi-dimensional approach to relevance [35]. However, this multi-dimensional aspect also presents a major challenge: different relevance criteria can sometimes be in conflict. For instance, in web search, the goals of displaying items that the user is familiar with based on their view/click history and increasing the number of serendipitous items in the top results can be at odds. Given this conflict, it is often difficult to find a scoring function that optimizes for all relevance criteria simultaneously, requiring a compromise between them.

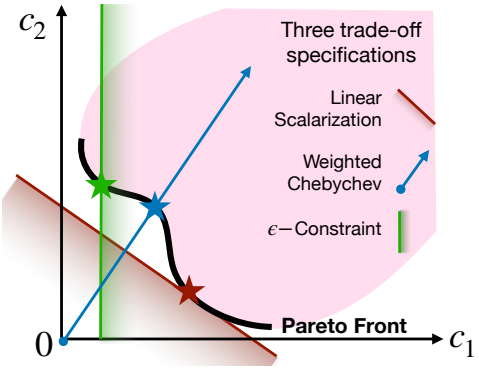


Figure 1: Three trade-off specifications investigated in our paper for Multi-Label Learning to Rank: Linear Scalarization, Weighted Chebyshev, and $\epsilon-$Constraint. These are used to prioritize one objective/criterion over another with different levels/types of specificity.

Multi-Objective Optimization (MOO) is a fascinating area that deals with the trade-offs between multiple objectives. One of the most important concepts in MOO is the Pareto Frontier (PF), which is a set of non-dominated solutions that represent the trade-offs between the objectives. The history of MOO research is rich, with numerous methods developed to specify the trade-off between

*Corresponding author

This work is licensed under a Creative Commons Attribution-NonCommercial International 4.0 License.

KDD '23, August 6–10, 2023, Long Beach, CA, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0103-0/23/08.
https://doi.org/10.1145/3580305.3599870

objectives. These methods include linear scalarization, weighted Chebyshev, and the $\epsilon-$constraint method, among others. Despite these many methods, recent studies on MLLTR have mainly focused on approximating the PF, rather than finding a unique non-dominated solution that represents a particular trade-off. This is because the individual solutions on the PF may not necessarily correspond to a specific trade-off [6]. At best, they may only correspond to one type of trade-off [27]. Thus, there is a need for more research to develop methods that can accurately capture the trade-offs between objectives, leading to a better understanding of the PF and the trade-offs represented by its solutions.

Approximating the entire PF without considering trade-off specifications may seem attractive, but it is not practical for MLLTR. The final result presented to the user is merely a single ranked list of items based on a single non-dominated scoring function, which does not take into account potential trade-offs among different objectives. In contrast, our approach combines principles of MOO and MLLTR to not only approximate the PF, but also to identify scoring functions associated with different types of trade-off specifications that would be more applicable in real world deployment.

1.1 Our Contributions

We summarize our contributions below:

- We formulate the MLLTR problem as a MOO problem and develop a comprehensive framework that allows for the integration of any first-order gradient-based MOO algorithm.
- We investigate three forms of trade-off specifications, as depicted in Figure 1, and evaluate various MOO techniques to obtain their non-dominated solutions. Our analysis delves into the pros and cons of each method, enabling a decision on the most appropriate method to employ in a given scenario. Additionally, we present a practical example of how a reference model deployed in the production system can be updated by exploring the PF around it.
- We uncover a hindrance in training models using a majority of MOO methods, which prevents them from converging to the PF due to oscillations in the cost function. To address this issue, we introduce a smoothing technique as a solution.
- We evaluate the effectiveness of the proposed MLLTR framework on five industrial datasets - four open-source datasets and one production dataset from E-commerce. The comparison of MOO methods is based on their ability to adhere to the trade-off specification and accurately approximate the PF. The incorporation of a smoothing technique results in a significant improvement in their empirical performance on these datasets. Moreover, one of the MOO algorithms has been deployed in production. This demonstrates that our MLLTR framework offers a practical solution for building MLLTR models that can benefit various industrial production systems.

1.2 Related Work

Several studies have integrated multiple relevance criteria into information retrieval systems, including web search and recommendation [35, 8, 9, 14, 38, 36], and product search [19, 15]. These traditional methods can be grouped into three categories: model aggregation, where individually trained models are combined to produce the final ranking; label aggregation, where the relevance labels are combined to create a single ranking model; and Linear Scalarization (LS), where a weight is assigned to each relevance criterion, collapsing the utility into a scalar function. The state-of-the-art Stochastic Label Aggregation (SLA) method [6] has been shown to be equivalent to LS. For recommendation applications, [18] proposed a framework for MOO-based MLLTR that guarantees finding non-dominated solutions, but does not consider trade-offs. In product search applications, [27, 26] proposed multiple relevance criteria and developed an $\epsilon-$Constraint MOO algorithm that enables trade-off specification as upper bounds for all objectives except one.

Recently, various gradient-based MOO algorithms have been developed for Multi-Task Learning (MTL) applications [31, 17] to approximate the PF. [23] introduced an EPO algorithm that guarantees to find solutions that correspond to trade-off specifications defined by objective priorities. [25] developed the WC-MGDA algorithm that offers the same guarantees and can improve over arbitrary reference models. Meanwhile, [13] proposed the DBGD algorithm, an $\epsilon-$Constraint type method that allows for trade-off specification as upper bounds of all objectives except one. In our MLLTR framework, we facilitate trade-off specification through various MOO methods, including classic methods such as LS and modern ones like EPO.

Another related research area in LTR also examines a multi-task learning approach. However, it uses only a single relevance label and incorporates auxiliary objectives to ensure that ranking results meet specific requirements, such as scale calibration [40], fairness [32, 28, 21], and diversity [16]. Conversely, our study focuses on employing sophisticated MOO algorithms to train ranking models using multiple labels, not just one relevance label.

2 BACKGROUND

2.1 Learning to Rank

Let $\mathbb{Q}$ be the set of all possible queries and $\mathbb{D}$ be the set of all documents or items. For a given query $q \in \mathbb{Q}$, let $D^q = \{d_i\}_{i=1}^{n_q} \subset \mathbb{D}$ be the subset of n_q matched items. Let a query-item pair (q, d_i) be represented by a p-dimensional feature vector $\mathbf{x}_i^q \in \mathbb{R}^p$. The goal of LTR is to learn a parametric scoring function $f_{\boldsymbol{\theta}} : \mathbb{R}^p \to \mathbb{R}$ that can assign a score s_i^q to each (q, d_i) pair from its corresponding vector representation, i.e., $\mathbf{x}_i^q \mapsto s_i^q$. The items can then be ranked in descending order of scores.

For a (q, d_i) pair, we denote the relevance label as $y_i^q \in \mathbb{Y}$, The training dataset for LTR consists of several queries: $\mathcal{D}_{\text{LTR}} = \left\{ \left\{ \left(\mathbf{x}_i^q, y_i^q \right) \right\}_{i=1}^{n_q} \right\}_{q=1}^{m}$, where m is the number of queries and n_q is the number of data points in each query group.

For a query q, let the output of a scoring function $f_{\boldsymbol{\theta}}$ for all the matched items in D^q be represented by a score vector $\mathbf{s}^q \in \mathbb{R}^{n_q}$. Similarly, let the corresponding relevance labels be denoted by the vector $\mathbf{y}^q \in \mathbb{R}^{n_q}$. The training cost is given by

$$c(\boldsymbol{\theta}) = \frac{1}{m} \sum_{q=1}^{m} \ell(\mathbf{s}^q, \mathbf{y}^q), \quad \text{where } s_i^q = f_{\boldsymbol{\theta}}(\mathbf{x}_i^q) \tag{1}$$

for all $i \in [n_q] = \{1, 2, \cdots, n_q\}$, and the per-query loss $\ell(\mathbf{s}^q, \mathbf{y}^q)$ quantifies the extent to which the ordering of scores disagrees with that of the relevance labels.

In the pair-wise approach of LambdaMART cost [4], the event that one item d_i is more relevant than another d_j w.r.t. q, denoted by $d_i \rhd_q d_j$, is probabilistically modeled as $\mathrm{P}(d_i \rhd_q d_j) = \frac{1}{1+e^{-\sigma(s_i^q - s_j^q)}}$, where σ controls the spread of the Sigmoid function. The per-query loss ℓ in (1) is constructed from the log-likelihood ($\ell\ell$) of $\boldsymbol{\theta}$ given the (presumably independent) observations in the training data:

$$\begin{aligned}\ell(\mathbf{s}^q, \mathbf{y}^q) &= -\ell\ell(\boldsymbol{\theta}|\mathcal{D}^q) \\ &= \sum_{(i,j)\in I^q} |\Delta NDCG(i,j)| \cdot \log\left(1 + e^{-\sigma(s_i^q - s_j^q)}\right), \quad (2)\end{aligned}$$

where $\mathcal{D}^q = \{(\mathbf{x}_i^q, y_i^q)\}_{i=1}^{n_q}$ is data pertaining to the matched items D^q, $I^q = \left\{(i,j) \in [n_q]^2 \,\middle|\, y_i^q > y_j^q\right\}$ consists of item pairs having a strict relevance order, and $\Delta NDCG(i,j)$ is the change of the NDCG value when two items i and j swap their rank positions [4].

The scoring function is modeled by GBM [12] with N decision trees: $f_{\boldsymbol{\theta}}^N(\mathbf{x}) = T_{\theta^0}(\mathbf{x}) - \sum_{t=1}^{N-1} \eta_t T_{\theta^t}(\mathbf{x})$, where η_t is the learning rate, T_{θ^t} is the tth tree, and the full model parameter is $\boldsymbol{\theta} = \{\theta^t\}_{t=0}^{N-1}$. On the tth iteration, the tree T_{θ^t} is learnt from the following training data:

$$\mathcal{D}_{T_{\theta^t}} = \left\{\left\{\left(\mathbf{x}_i^q, \partial c/\partial s_i^q\right)\right\}_{i=1}^{n_q}\right\}_{q=1}^{m}, \quad (3)$$

where the labels are gradients of cost w.r.t. the scores. In other words, instead of updating $f_{\boldsymbol{\theta}}$ in the parameter space, it is updated in the function space of trees: $f_{\boldsymbol{\theta}}^{t+1} = f_{\boldsymbol{\theta}}^t - \eta_t T_{\theta^t}$. The function space update of GBM suffices to treat the cost as a function of scores rather than the parameters $\boldsymbol{\theta}$. Henceforth, we consider the cost $c : \mathbb{R}^M \to \mathbb{R}$ as a function of $\mathbf{s}$, and rewrite (1) as

$$c(\mathbf{s}) = \frac{1}{m}\sum_{q=1}^{m} \ell(\mathbf{s}^q, \mathbf{y}^q). \quad (4)$$

2.2 LTR from Multiple Relevance Labels

In MLLTR, different relevance criteria are measured, providing multiple labels for each query-item pair. The goal of MLLTR is still the same as that of LTR: to learn a scoring function $f_{\boldsymbol{\theta}}$ that assigns a scalar value to each (q, d_i) pair.

The labels for (q, d_i) are $y_{ik}^q \in \mathbb{Y}_k$ for $k = 1, \cdots, K$, where K is the number of relevance criteria. Similar to LTR, each label set $\mathbb{Y}_k$ could be either discrete or continuous, endowed with a total ordering relation. The training dataset for MLLTR is denoted by

$$\mathcal{D}_{\text{MLLTR}} = \left\{\left\{\left(\mathbf{x}_i^q, y_{i1}^q, \cdots, y_{iK}^q\right)\right\}_{i=1}^{n_q}\right\}_{q=1}^{m}. \quad (5)$$

Each relevance criterion has a training cost. Therefore, in MLLTR, the cost is a vector valued function: $\mathbf{c}(\mathbf{s}) = [c_1(\mathbf{s}), \cdots, c_K(\mathbf{s})]^T$, naturally making it an MOO problem.

2.3 Multi-Objective Optimization

In MOO, the cost function $\mathbf{c} : \mathbb{R}^M \to \mathbb{R}^K$ is a mapping from the *solution space* $\mathbb{R}^M$ to the *objective space* $\mathbb{R}^K$.

We use $\mathbb{R}_+^K := \left\{\mathbf{c} \in \mathbb{R}^K \,\middle|\, c_k \geq 0,\ \forall\, k \in [K]\right\}$, the cone of positive orthant, to define a partial ordering relation. For any two points $\mathbf{c}^1, \mathbf{c}^2 \in \mathbb{R}^K$, we write $\mathbf{c}^1 \succcurlyeq \mathbf{c}^2$, if $\mathbf{c}^1$ lies in the positive cone pivoted at $\mathbf{c}^2$, i.e., $\mathbf{c}^1 \in \left\{\mathbf{c}^2 + \mathbf{c} \,\middle|\, \mathbf{c} \in \mathbb{R}_+^K\right\}$. In other words, $\mathbf{c}^1 \succcurlyeq \mathbf{c}^2 \iff \mathbf{c}^1 - \mathbf{c}^2 \in \mathbb{R}_+^K$, making $c_k^1 \geq c_k^2$, $\forall k \in [K]$. We define $\mathbf{c}^1 \succ \mathbf{c}^2$ when there is at least one k for which $c_k^1 > c_k^2$, i.e., $\mathbf{c}^1 \neq \mathbf{c}^2$.

For minimization, a solution $\mathbf{s} \in \mathbb{R}^M$ is said to be non-dominated or *Pareto optimal*, if there exists no other solution $\mathbf{s}' \in \mathbb{R}^M$ such that $\mathbf{c}(\mathbf{s}) \succ \mathbf{c}(\mathbf{s}')$. We call the set of all non-dominated solutions the Pareto optimal set. The image of this Pareto set under the function $\mathbf{c}$ is the *Pareto Frontier* (PF), which can be a $K-1$-dimensional manifold if connected [11, 22].

3 A FRAMEWORK FOR MULTI-LABEL LTR

Multi-Gradient Combination: The MLLTR cost function gives rise to K score-gradients, $\nabla_{\mathbf{s}} c_k$ for $k \in [K]$. However, for training the GBM based scoring function, the tth decision tree requires exactly one score-gradient as labels in its training data (3), not K score-gradients. Although the cost is upgraded to become a vector valued function in MLLTR, the scoring function remains a scalar valued function. We combine the K score-gradients as

$$\boldsymbol{\lambda} = \sum_{k=1}^{K} \alpha_k \nabla_{\mathbf{s}} c_k, \quad \text{s.t.} \sum_{k=1}^{K} \alpha_k = 1, \quad \boldsymbol{\alpha} \in \mathbb{R}_+^K, \quad (6)$$

where $\boldsymbol{\lambda} \in \mathbb{R}^M$ are the labels for training the trees in GBM and $\boldsymbol{\alpha}$ are combination coefficients.

3.1 Linear Scalarization Based Methods

Linear Scalarization (LS): The MOO cost is converted to a scalar cost $g_{\mathbf{r}}^{\text{LS}}(\mathbf{s}) = \sum_{k=1}^{K} r_k c_k(\mathbf{s})$, where $\mathbf{r} \in \mathbb{R}_+^K$ represents preferences/priorities given to the costs.

Gradient Combination: It remains static throughout the iterations

$$\boldsymbol{\alpha} = \mathbf{r}/\|\mathbf{r}\|_1. \quad (7)$$

Although LS is simple, specifying trade-offs by elements in the dual space has limitations. If any of the costs is a non-convex function, i.e., the range $\mathbb{O}$ becomes a non-convex set, LS can not guarantee to reach all points in the PF by varying the preferences [2], as illustrated in Figure2a.

Stochastic Label Aggregation (SLA): One gradient is randomly chosen following the distribution:

$$\alpha_k = \begin{cases} 1, & \text{if } k = \overline{K}, \\ 0, & \text{otherwise,} \end{cases} \quad \text{for } k \in [K] \quad (8)$$

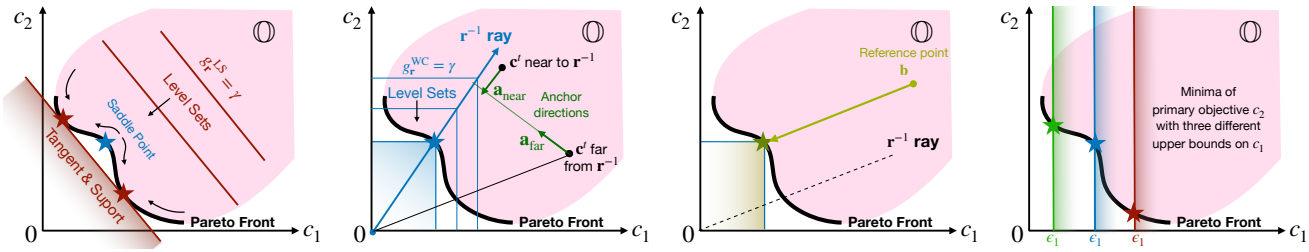
where $\overline{K}$ is a categorical random variable over the K indices with $\mathbf{r}/\|\mathbf{r}\|_1$ as its probability distribution. In other words, the $\overline{K}^{\text{th}}$ label is used for training. The expected cost of SLA is the same as that of LS [6]. Thus, **SLA can be seen as a special type of LS.**

3.2 Preference Direction Based Methods

3.2.1 Weighted Chebyshev (WC):. In WC, the vector valued cost is scalarized to

$$g_{\mathbf{r}}^{\text{WC}}(\mathbf{s}) = \max_{k\in[K]} r_k c_k(\mathbf{s}). \quad (9)$$

In general, the solution $\mathbf{s}_{\mathbf{r}}^* = \min_{\mathbf{s}} g_{\mathbf{r}}^{\text{WC}}(\mathbf{s})$ satisfies $r_1 c_1(\mathbf{s}_{\mathbf{r}}^*) = r_2 c_2(\mathbf{s}_{\mathbf{r}}^*) = \cdots = r_K c_K(\mathbf{s}_{\mathbf{r}}^*)$ [24], which can be deduced by analyzing the level sets, illustrated in Figure2b. This makes the trade-off specification between the objectives stricter than the penalty approach in the LS.



(a) LS & SLA (b) WC & EPO (c) WC-MGDA (d) ϵ−Constraint methods

Figure 2: Illustration of trade-off specifications. (2a) **shows how LS can have non-unique Pareto optimal points.** (2b) **shows WC can attain the blue optimum by minimizing $g_{\mathbf{r}}^{WC}$ and illustrates how EPO works.** (2c) **shows WC-MGDA can find Pareto optima better than the arbitrary reference point b.** (2d) **shows ϵ−Constraint can find different Pareto optima by constraining the cost C_1.**

Gradient Combination: Only the gradient of maximum relative objective value is chosen:

$$\alpha_k = \begin{cases} 1, & \text{if } k = k^*, \\ 0, & \text{otherwise,} \end{cases} \quad \text{s.t. } k^* = \arg\max_{k \in [K]} r_k c_k(\mathbf{s}). \tag{10}$$

The objective vector value is proportional to the $\mathbf{r}^{-1}$ ray as illustrated in Figure2b. This trade-off specification guarantees that Pareto optimal points in the PF can be reached by varying the preferences, even when the objectives are non-convex. However, in practice, the strict trade-off requirement hinders the progress in cost value reduction. When optimizing with a step size (i.e., learning rate), the iterate $\mathbf{c}^t$ (cost at t^{th} iteration) oscillates around $\mathbf{r}^{-1}$ ray.

3.2.2 Exact Pareto Optimal Search (EPO):. In EPO [23, 22], the trade-off specification is the same as that of WC. Therefore, most properties of WC are inherited. However, to overcome the limitations of WC, its gradient combination is designed to avoid oscillations around the $\mathbf{r}^{-1}$ ray.

Gradient Combination: The coefficients are obtained by solving a quadratic program:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}_+^K} \|\mathrm{C}^T \mathrm{C} \boldsymbol{\alpha} - \mathbf{a}\|_2^2, \quad \text{s.t.} \sum_{k=1}^{K} \alpha_k = 1, \tag{11}$$

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

$$\mathbf{a} = \begin{cases} \mathbf{c}^t - \frac{\langle \mathbf{c}^t, \mathbf{r}^{-1} \rangle}{\|\mathbf{r}^{-1}\|_2} \mathbf{r}^{-1}, & \text{if } \mathbf{c}^t \text{ is far,} \\ \mathbf{r}^{-1}, & \text{otherwise.} \end{cases} \tag{12}$$

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 directs towards it, as illustrated in Figure2b. On the other hand, when $\mathbf{c}^t$ is near $\mathbf{r}^{-1}$ ray, we move the cost along the $\mathbf{r}^{-1}$ ray avoiding oscillations.

3.2.3 Weighted Chebyshev MGDA (WC-MGDA):. In WC-MGDA algorithm [25], the trade-off specification is similar to that of WC method, but the SOCP formulations are designed to avoid the shortcomings of WC, i.e., through the preferences over the objectives. WC-MGDA aims to build models that are closer or better than the reference model.

Gradient Combination: The coefficients are obtained by:

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}_+^K, \boldsymbol{x} \in R^n, \gamma} \boldsymbol{\alpha}^\top (\mathbf{r} \odot (\boldsymbol{l}(\boldsymbol{x}) - \boldsymbol{b})) - u\gamma \tag{13}$$

$$\text{s.t.} \sum_{k=1}^{K} \alpha_k = 1,, \ \|\boldsymbol{G}_{\mathbf{r}} \boldsymbol{\alpha}\|_2 \leq \gamma, \tag{14}$$

where $\boldsymbol{b}$ is the loss of the reference model, and $\boldsymbol{G}_{\mathbf{r}} \equiv \text{diag}(\sqrt{\mathbf{r}}) \boldsymbol{G} \text{diag}(\sqrt{\mathbf{r}})$. Here, $\boldsymbol{G} = (\sqrt{\mathrm{C}^T \mathrm{C}})$.

WC-MGDA jointly solves WC and MGDA to ensure achieving both preference alignment and Pareto Optimality. While the WC problem tries to find solutions by minimizing weighted ℓ_∞, the norm minimization ensures Pareto Optimality.

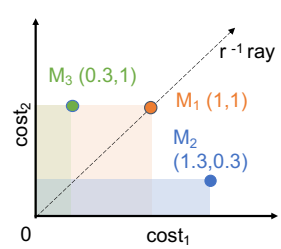


Figure 3: Illustration of the Maximum Weighted Loss (MWL) metric. M_1 is better than M_3. However, between M_1 and M_3, we need to use the negative orthant area for tie breaker.

3.2.4 Evaluation Metric for Preference Direction Based MLLTR:. To quantify the performance on preference based MLLTR, we use the objective function of WC (9), which exactly captures alignment with the $\mathbf{r}^{-1}$-ray and is referred to as maximum weighted loss (MWL). Figure 3 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, and better than M_2. Between M_1 and M_3, we use the volume of intersection between the negative orthant (VNO) pivoted by each model and $\mathbb{R}_+^K$ (color shaded area in Figure 3) as a tiebreaker. Note, VNO should always be used as a tie breaker when the difference in MWL is insignificant in our paper. For example, MWL are 1 for both M_1 and M_3. VNO for M_1 is 1 while VNO for M_3 is $1 \times 0.3 = 0.3$. Thus, M_3 is better than M_1 due to the VNO of M_3 is smaller even if MWL are the same for both models.

Table 1: Description of when to use which MOO method for MLLTR

MOO Method	Suitable When	Type of Trade-off specification
LS/SLA	total utility / cost is known	weights on linear combination of costs
WC/EPO	ratio between objectives is given as preference direction	preference direction (ratio between objectives)
WC-MGDA	a reference model is known (pretrained model), preference direction from the reference model	preference direction (ratio between objectives)
EC-AL/EC-DBGD	hard constraints on objectives are known	constraints (upper bounds) for secondary criteria

Algorithm 1 MLLTR by Gradient Boosted Decision Trees

Input: $\mathcal{D}_{\text{MLLTR}}$ from (5), N, learning rate(s) η
Parameter: GBM configurations, *combinator*
Output: scoring function $f_{\boldsymbol{\theta}}$

Set $f_{\boldsymbol{\theta}}^0 = T_{\theta^0}$ ▷ Usually, this is set to 0
for $t \leftarrow 1$ to $N-1$ **do**
  Get $\mathbf{s}$: $s_i^q = f_{\boldsymbol{\theta}}^t(\mathbf{x}_i^q)$ for all (q, d_i) pair, $\mathbf{c}^t$ and gradients C^t
  $\boldsymbol{\alpha}^t$ = GetCoefficients($\mathbf{c}^t, \mathrm{C}^t$, *combinator*)
  Smooth $\boldsymbol{\alpha}^t$ using (19)
  Prepare data $\mathcal{D}_{T_\theta^t}$ as (3) but with labels $\boldsymbol{\lambda} = \mathrm{C}^t\boldsymbol{\alpha}^t$
  Fit the tree $\mathbf{T}_{\theta^t}$ to $\mathcal{D}_{T_\theta^t}$
  Update Scoring function: $f_{\boldsymbol{\theta}}^t = f_{\boldsymbol{\theta}}^{t-1} - \eta_t \mathbf{T}_{\theta^t}$
  if $\|\mathbf{c}^t - \mathbf{c}^{t-1}\| \approx 0$ **then** Break
return $f_{\boldsymbol{\theta}}^t$

3.3 ϵ−Constraint (EC) Methods

3.3.1 ϵ−Constraint Augmented Lagrangian (EC-AL). In this method, the MOO problem is transformed into

$$\min_{\mathbf{s}\in\mathbf{R}^M} c_{k_p}(\mathbf{s}) \quad \text{s.t. } c_k(\mathbf{s}) \le \epsilon_k, \text{ for } k \in [K] - \{k_p\}, \tag{15}$$

where one cost k_p is treated as the primary cost and the rest $K-1$ costs are restricted to satisfy an upper bounded constraint given by the ϵ_k.

Gradient Combination: [27] proposed an augmented Lagrangian form of (15) as

$$\max_{\boldsymbol{\alpha}} \min_{\mathbf{s}} \mathcal{L}(\mathbf{s}, \boldsymbol{\alpha}) = c_{k_p}(\mathbf{s}) + \sum_{k\in[K]_p} \alpha_k (c_k(\mathbf{s}) - \epsilon_k), \tag{16}$$

where $[K]_p = [K] - \{k_p\}$. At iteration t, $\boldsymbol{\alpha}$ is decided according to a proximal update strategy

$$\alpha_k^t = \begin{cases} \mu(c_k^t - \epsilon_k) + \alpha_k^{t-1}, & \text{if } c_k^t - \epsilon_k \ge 0, \\ 0, & \text{otherwise.} \end{cases} \tag{17}$$

for $k \in [K] - \{k_p\}$, where $\mu > 0$ is a positive value, and α_k^{t-1} is the coefficient of the previous iteration. Coefficient of a secondary objective is non-zero only when its constraint is violated.

3.3.2 ϵ-Constraint Dynamic Barrier Gradient Descent(EC-DBGD). The trade-off specification is the same as that of EC-AL, i.e., through the upper bounds on secondary objectives. The coefficients are obtained by solving the following convex quadratic program [13]:

$$\min_{\boldsymbol{\alpha}, \alpha_{k_p}=1} \frac{1}{2} \|\mathrm{C}\boldsymbol{\alpha}\|_2^2 - \sum_{k\in[K]_p} \alpha_k \phi_k(\mathbf{s}), \tag{18}$$

where ϕ_k is a control function associated with constraint $c_{k_p}(\mathbf{s})$ for $k \in [K]_p$.

Algorithm 2 Multi-Gradient Coefficient from MOO

function GetCoefficients($\mathbf{c}$, C, *combinator*)
  Do some action.
  if *combinator* = LS **then** $\boldsymbol{\alpha}$ from (7)
  else if *combinator* = SLA **then** $\boldsymbol{\alpha}$ from (8)
  else if *combinator* = WC **then** $\boldsymbol{\alpha}$ from (10)
  else if *combinator* = EPO **then** $\boldsymbol{\alpha}$ from (11)
  else if *combinator* = WC-MGDA **then** $\boldsymbol{\alpha}$ from (13)
  else if *combinator* = EC-AL **then** $\boldsymbol{\alpha}$ from (17)
  else if *combinator* = EC-DBGD **then** $\boldsymbol{\alpha}$ from (18)
  return $\boldsymbol{\alpha}$

We illustrate three types of trade-off specifications in Figure 2, summarize the training of scoring function in algorithm 1, the MOO methods in algorithm 2, and when to use which method in table 1.

3.4 Non-Smooth Trajectory and Remedy by Moving Average (MA)

All the MOO methods discussed previously are first order methods, where the final search direction is formulated by adaptively combining the objective gradients. However, the step size is kept fixed (or heuristically decreased) in every iteration, instead of adapting it to the ever changing search direction. This causes the iterates to exhibit oscillatory behavior in their cost functions, as empirically verified in section 4.2 and 4.3. Note, although theoretically step size selection techniques such as *Line Search* methods [37] can adapt the step size, they cannot be used in practice due to high computational cost: the objective function needs to be computed several times in every iteration. Moreover, it is non-trivial to extend these methods, primarily developed for single objective, to multi-objective setup.

Not all MOO method exhibit non-smooth trajectory though. In LS, it does not happen because the search direction do not change drastically as it is the gradient of a fixed objective function. In EC-AL, there is no change in the coefficients of primary cost, and change for secondary costs are smoothed by the coefficients of previous iteration.

To mitigate this issue of non-smooth trajectory MOO methods, we propose a simple yet effective technique. We apply a moving average (MA) filter to $\boldsymbol{\alpha}$ between consecutive iterations:

$$\boldsymbol{\alpha}^t \leftarrow \nu\boldsymbol{\alpha}^t + (1-\nu)\boldsymbol{\alpha}^{t-1}, \tag{19}$$

where $\nu \in (0, 1)$ is a smoothing factor. Note that this is different from the momentum based first order methods [30] for single objective, where MA is applied on its gradient. It is not applicable to multi-objective case, because the search direction can change significantly between consecutive iterations due to change in the $\boldsymbol{\alpha}$, even when the objective gradients do not change significantly. Therefore, we smoothen the coefficients rather than the gradients.

Table 2: Details of the Learning-to-Rank datasets

Datasets	# queries (train/test)	# features	# labels	# bi-objectives	# tri-objectives	Label selection from
MSLR-WEB30k	20K/10K	136	5	10	6	feature description
Yahoo	20K/7K	519	6	15	10	descriptive analytics
ISTELA	20K/6.6K	220	5	10	10	descriptive analytics
ISTELA-S	20K/6.6K	220	5	10	10	descriptive analytics
E-commerce	500K/100K	29	5	8	6	feature description

4 EXPERIMENTS

We evaluate the effectiveness of our MLLTR framework by examining the compliance with the trade-off specification and accuracy in approximating the PF. The performance improvement provided by the MA method, as described in Section 3.4, is also assessed for each MOO method. Lastly, we demonstrate how extended preference-based methods can be used to explore the PF around a reference objective vector, thereby updating production models.

4.1 Datasets and Experimental Settings

We test our MLLTR framework using five datasets: the Microsoft LETOR dataset (MSLR-WEB30K) [29], Yahoo! LETOR dataset [7], Istella LETOR dataset [10], Istella-S [20], and a proprietary E-commerce dataset that has similarities to the datasets used in [33, 27], but was collected in 2021. Details of these datasets can be found in Table 2.

For all datasets, we use some features as additional labels in addition to the original relevance label and remove them from the feature list to prevent data leakage. The extra labels for MSLR-WEB30K are *Query-URL Click Count* (Click), *URL Dwell Time* (Dwell), *Quality Score* (QS), and *Quality Score2* (QS2). Unlike the MSLR dataset, feature descriptions for Yahoo! and two Istella datasets are not publicly available. Thus, we selected extra labels for them through a descriptive analysis of the features (for details, see Appendix A). The labels for the E-commerce dataset include a binary target indicating whether an item was purchased or not, historical purchases, relevance score between the query and product, brandedness scores of the product, and delivery speed of the product.

4.1.1 Trade-off Specification. For LS and preference-based methods, we set $\mathbf{r}$ as follows. For bi-objective cases, we generated 5 $\mathbf{r}^{-1}$ rays that are equally distributed in the region between the two "baseline cost" vectors. A baseline cost vector was obtained by training a model for only one objective and computing costs for all objectives. For the tri-objective case, we generated 25 preference directions using PESA [34, 22], which samples equi-distributed points on the convex hull of baseline cost vectors. For the ϵ-Constraint methods, we set the $\boldsymbol{\epsilon}$ of the secondary objectives as follows. In a bi-objective case, the baseline cost vector corresponding to the primary objective has a sub-optimal cost value for the non-primary objective. We set 5 upper bounds by dividing this sub-optimal cost into 5 levels. Similarly, in a tri-objective case, the two sub-optimal cost values in the baseline cost vector were divided into 5 levels each, resulting in 25 pairs of upper bounds for the two non-primary objectives.

4.1.2 Hyperparameter Tuning. For each dataset, we fine-tuned hyperparameters (i.e., number of trees and learning rate) of the GBM model by conducting LTR using main relevance judgments as the single label. We selected the best configuration of hyperparameters (according to the NDCG@5), which was 600 trees and a learning rate of 0.25, after evaluating the grid of hyperparameters $\{300, 600, 900, 1200\} \times \{0.05, 0.15, 0.25, 0.35\}$ for number of trees and learning rates, respectively.

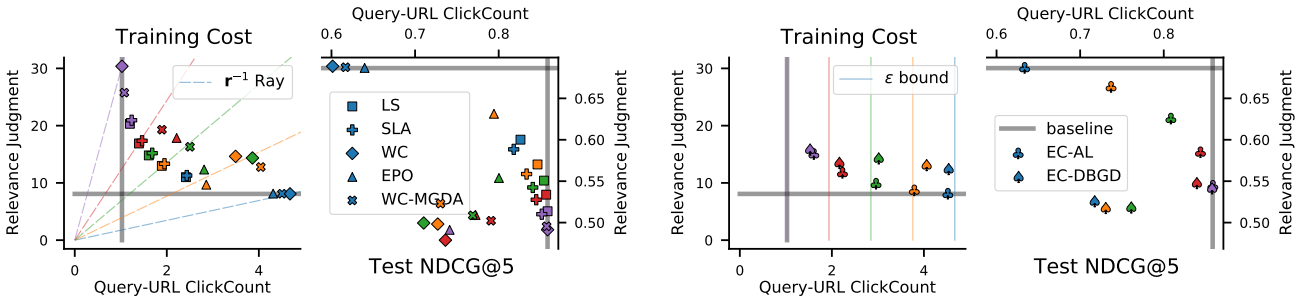
4.2 Qualitative Evaluation

As a preliminary experiment, we applied existing methods in their original form to the (Click, Rel) pair on the MLSR dataset, including linear weighting methods (LS, SLA), preference-based methods (WC, EPO, WC-MGDA), and EC methods (EC-AL, EC-DBGD).

4.2.1 Compliance with Trade-off Specification. To visualize the compliance, we compute the cost vectors on training dataset. To assess the ranking performance, we compute the NDCG@5 metric on validation data. Figure 4 shows the result.

In Figure 4a, we observe that the final cost vectors of LS and SLA are very similar for each preference specification, which is expected as they are known to be probabilistically equivalent [6]. However, the solutions generated by LS slightly dominate those of SLA in the cost space, which translates to better performance in the NDCG space. The solutions produced by preference-based methods (WC, EPO, and WC-MGDA) have a closer alignment with the preference rays than LS and SLA. This is particularly evident for extreme preference rays near the baselines, where the blue and purple square points of LS are farther from the corresponding preference rays compared to WC, EPO, and WC-MGDA. This indicates that if the trade-off is specified not as a utility but as a ratio between objectives, LS should not be used. Nevertheless, if one considers the frontier of solutions, LS dominates every other method in the non-extreme regions of the PF. Similarly, in Figure 4b, for the ϵ-Constraint methods, the solutions of both EC-AL and EC-DBGD generally comply with the constraint specifications. However, the frontier of EC-AL is superior to that of EC-DBGD in most cases.

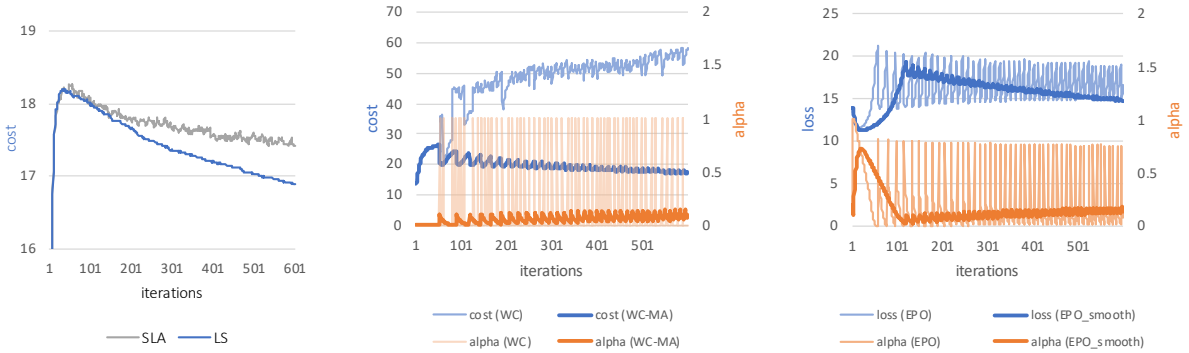
4.2.2 Issue of Non-Smooth Trajectory. Keeping the aspect of trade-off compliance aside, a surprising observation is the solutions from the simpler baselines such as LS and EC-AL seems to dominate the other methods. To understand this, we plot cost curve for several models in Figure 5. LS is the only method that has smooth behavior in the figure. SLA is a stochastic version of LA and non-smooth changes are visible, which causes inferior dominance. For WC, the oscillation is expected, as it chooses only one label that have maximum weighted cost. Although EPO does achieve lower values of the cost as compared to WC, it still has oscillations. We observed the same issues exists in WC-MGDA and EC-DBGD, across all multi-label experiments and datasets.



(a) LS/SLA and preference based MOO methods.

(b) ϵ-Constraint methods.

Figure 4: Initial results of bi-objective experiments on MSLR-WEB30K [29] dataset. Colored lines and points represent different trade-off specifications and the corresponding solutions, respectively.

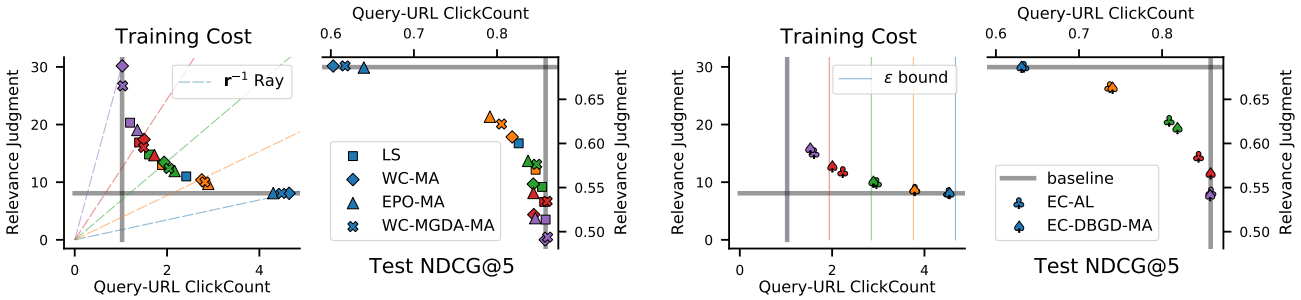


(a) cost curve for SLA/LS.

(b) cost curve for WC.

(c) cost curve for EPO.

Figure 5: Cost curves for (a) SLA/LS, (b) WC and (c) EPO for (Click, Rel). For WC and EPO, we also show α for Rel. We use light color for the original methods and dark for smoothed versions.



(a) preference based methods with moving average.

(b) ϵ-Constraint methods with moving average

Figure 6: Improved results with moving average. Most of the models are close to the Pareto Front.

4.2.3 Improvement with Moving Average. We have consistently used a smoothing factor of $\nu = 0.1$, as specified in (19). The smoothed cost curves are displayed in Figure 5 using dark colors for the WC and EPO methods. The improved results in the cost/NDCG space with smoothed $\boldsymbol{\alpha}$ can be seen in Figure 6, presenting a noticeable improvement when compared to the results in Figure 4. The smoothing has effectively prevented the solutions of the preference based methods from being dominated by the LS method, while at the same time aligning them with the specified preference rays. Similarly, the solution frontier of EC-DBGD is now not dominated by that of EC-AL.

Table 3: Metrics on MSLR for bi-objective and tri-objective experiments. "orig" refers to SLA and original versions of WC / EPO / WC-MGDA / EC-DBGD, including EC-AL. "ma" refers to LS and moving average version of them. Bold numbers mean statistical significance between orig and ma. Red number refers to a single winner (significance vs. all others) for each type.

(a) MSLR dataset (2-obj)

	MWL (test)			HVI (train cost)			HVI (test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	2.24	**2.09**	-6.7%	3.51	**3.55**	1.0%	0.93	**0.96**	2.2%
WC	5.08	**1.97**	-61.7%	3.40	**3.55**	4.5%	0.95	**0.96**	2.1%
EPO	2.55	**2.02**	-20.7%	3.51	**3.56**	1.4%	0.95	**0.97**	1.6%
WC-MGDA	2.02	**1.93**	-4.6%	3.53	**3.57**	1.0%	0.96	**0.97**	0.9%
EC method									
EC-AL	–	–	–	3.52	–	–	0.97	–	–
EC-DBGD	–	–	–	3.47	**3.52**	1.5%	0.95	**0.97**	1.8%

(b) MSLR dataset (3-obj)

	MWL (test)			HVI(train cost)			HVI(test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	2.01	**1.86**	-7.5%	6.37	**6.52**	2.4%	0.79	**0.84**	7.0%
WC	10.3	**1.75**	-83.0%	6.00	**6.57**	9.4%	0.81	**0.89**	9.4%
EPO	2.46	**1.90**	-23.0%	6.45	**6.61**	2.4%	0.87	**0.88**	1.7%
WC-MGDA	1.88	**1.74**	-7.6%	6.54	**6.63**	1.4%	0.88	**0.90**	2.0%
EC method									
EC-AL	–	–	–	6.51	–	–	0.88	–	–
EC-DBGD	–	–	–	6.41	**6.50**	1.4%	0.84	**0.87**	4.1%

Table 4: Metrics on Yahoo! dataset for bi-objective and tri-objective experiments.

(a) Yahoo! dataset (2-obj)

	MWL (test)			HVI(train cost)			HVI(test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	91.7	**86.2**	-6.0%	3.28	**3.32**	1.2%	0.94	**0.95**	1.9%
WC	87.3	**81.0**	-7.3%	3.28	**3.34**	1.9%	0.95	**0.96**	0.9%
EPO	107.3	107.4	0.1%	3.16	3.16	0.0%	0.87	0.87	-0.1%
WC-MGDA	85.4	**80.2**	-6.1%	3.30	**3.35**	1.4%	0.95	**0.96**	0.7%
EC method									
EC-AL	–	–	–	3.30	–	–	0.95	–	–
EC-DBGD	–	–	–	3.29	**3.31**	0.7%	0.95	**0.95**	0.4%

(b) Yahoo! dataset (3-obj)

	MWL (test)			HVI(train cost)			HVI(test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	70.6	**66.7**	-5.5%	6.14	**6.23**	1.4%	0.84	**0.88**	5.2%
WC	72.9	**60.5**	-17.0%	6.24	**6.40**	2.5%	0.88	**0.91**	3.3%
EPO	82.1	**82.0**	-0.1%	5.91	5.90	-0.1%	0.73	0.73	-0.2%
WC-MGDA	66.4	**60.3**	-9.3%	6.31	**6.41**	1.7%	0.89	**0.91**	1.9%
EC method									
EC-AL	–	–	–	6.38	–	–	0.89	–	–
EC-DBGD	–	–	–	6.34	**6.39**	0.7%	0.89	**0.90**	0.3%

Table 5: Metrics on E-commerce dataset for bi-objective and tri-objective experiments.

(a) E-commerce dataset (2-obj)

	MWL (test)			HVI (train cost)			HVI (test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	2.14	**2.14**	-1.7%	2.89	**2.90**	0.6%	0.94	**0.95**	1.27%
WC	15.7	**2.05**	-86.9%	2.65	**2.94**	11.0%	0.93	**0.98**	5.39%
EPO	6.15	**2.22**	-63.9%	2.85	**2.90**	1.7%	0.94	**0.96**	2.18%
WC-MGDA	5.93	**2.03**	-65.9%	2.86	**2.96**	3.3%	0.97	**0.98**	1.03%
EC method									
EC-AL	–	–	–	2.82	–	–	0.97	–	–
EC-DBGD	–	–	–	2.88	**2.93**	1.5%	0.97	**0.98**	0.7%

(b) E-commerce dataset (3-obj)

	MWL (test)			HVI(train cost)			HVI(test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	1.45	**1.42**	-2.2%	5.14	**5.21**	1.4%	0.87	**0.90**	1.41%
WC	18.4	**1.35**	-92.7%	4.07	**5.27**	29.4%	0.86	**0.93**	8.41%
EPO	1.91	**1.47**	-23.1%	5.10	**5.16**	1.2%	0.91	**0.93**	1.73%
WC-MGDA	5.83	**1.34**	-76.9%	4.99	**5.31**	6.3%	0.92	**0.94**	2.08%
EC method									
EC-AL	–	–	–	5.00	–	–	0.93	–	–
EC-DBGD	–	–	–	523	**5.35**	2.2%	0.93	**0.95**	2.7%

Table 6: Metrics on smaller Istella LETOR dataset for bi-objective and tri-objective experiments.

(a) Istella-S LETOR dataset (2-obj)

	MWL (test)			HVI (train cost)			HVI (test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	2.24	**1.88**	-16.2%	3.88	3.89	0.5%	0.89	**0.92**	3.2%
WC	8.66	**1.64**	-81.1%	3.75	3.87	3.3%	0.85	**0.91**	7.2%
EPO	4.48	**1.72**	-61.6%	3.84	3.89	1.5%	0.89	**0.92**	3.8%
WC-MGDA	2.21	**1.60**	-27.6%	3.85	3.88	0.9%	0.89	**0.92**	3.3%
EC method									
EC-AL	-	-	-	3.84	-	-	0.87	-	-
EC-DBGD	-	-	-	3.83	**3.87**	1.0%	0.87	**0.89**	2.6%

(b) Istella-S LETOR dataset (3-obj)

	MWL (test)			HVI (train cost)			HVI (test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	2.89	**2.39**	-17.3%	7.58	7.66	1.0%	0.72	**0.78**	8.9%
WC	40.17	**1.94**	-95.2%	6.22	7.56	21.6%	0.58	**0.78**	34.3%
EPO	2.43	**1.97**	-18.7%	7.59	7.67	1.2%	0.77	**0.80**	3.8%
WC-MGDA	2.57	**1.90**	-26.2%	7.50	7.59	1.1%	0.73	**0.79**	7.6%
EC method									
EC-AL	-	-	-	7.62	-	-	0.75	-	-
EC-DBGD	-	-	-	7.57	**7.66**	1.1%	0.72	**0.77**	6.0%

4.3 Quantitative Evaluation

We quantify the improvement of employing MA over the vanilla MOO method on two metrics: MWL (defined in section 3.2.4) for preference based methods, and *Hypervolume Indicator* (HVI) [1] for all methods. The MWL quantification (lower is better) combines two aspects of an MOO method: 1) the cost vector's alignment with

Table 7: Metrics on Istella LETOR dataset dataset for bi-objective and tri-objective experiments.

(a) Istella LETOR dataset (2-obj)

	MWL (test)			HVI (train cost)			HVI (test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	6.26	**4.79**	-23.4%	3.90	**3.93**	0.6%	0.86	**0.90**	4.4%
WC	61.05	**3.93**	-93.6%	3.59	**3.90**	8.7%	0.80	**0.90**	11.5%
EPO	31.72	**4.31**	-86.4%	3.87	**3.92**	1.2%	0.88	**0.90**	2.9%
WC-MGDA	25.67	**3.82**	-85.1%	3.86	**3.91**	1.3%	0.86	**0.90**	5.3%
EC method									
EC-AL	–	–	–	3.84	-	-	0.83	-	-
EC-DBGD	–	–	–	3.86	**3.89**	0.9%	0.84	**0.85**	2.1%

(b) Istella LETOR dataset (3-obj)

	MWL (test)			HVI (train cost)			HVI (test NDCG)		
	orig	ma	gain%	orig	ma	gain%	orig	ma	gain%
Preference based									
SLA/LS	2.79	**2.36**	-15.5%	7.91	7.93	0.2%	0.68	**0.73**	7.8%
WC	36.64	**2.05**	-94.4%	7.79	7.92	1.7%	0.58	**0.73**	25.7%
EPO	2.90	**2.18**	-24.8%	7.87	7.91	0.5%	0.73	**0.77**	5.6%
WC-MGDA	3.76	**2.03**	-46.1%	7.91	7.92	0.1%	0.68	**0.74**	8.1%
EC method									
EC-AL	-	-	-	7.88	-	-	0.66	-	-
EC-DBGD	-	-	-	7.89	**7.90**	0.0%	0.68	**0.69**	0.5%

the preference ray and 2) its closeness to the **0** cost vector, which is a utopia solution. Whereas, the HVI[1] metric (higher is better) quantifies how closely the entire PF is approximated using all the solutions generated from equi-distributed trade-off specifications. We conduct paired t-test (significance level 0.05) on repeated randomized experiments to establish the statistical significance of the improvement due to MA. The t-test pairing for MWL metric is done by multi-indexing an observation from the Cartesian product of 3 sets (variables): 1) the set of all relevance label tuples (bi-objective and tri-objective cases), 2) the set of all preferences, and 3) random seed. Whereas, the t-test pairing for HVI metric is done by 2 variables: 1) the set of all relevance label tuples (bi-objective and tri-objective cases), 2) random seed.

We report in Table 3, 4, 5, 6 and 7 bi-objective and tri-objective results on MSLR, Yahoo!, E-commerce, Istella-S and Istella LETOR dataset, respectively. The effect of smoothing is clear. For all cases, the gain due to smoothing is significant for all metrics. Notably, for bi-objective cases, it benefits WC significantly – helping it to become 2nd best model behind WC-MGDA. WC-MGDA worked well even without MA. When it failed for E-commerce dataset, MA helped a lot and made it the best model for all metrics. Overall, WC-MGDA showed best performance in MWL and competitive performance in HVIs. For EC methods, EC-DBDA with MA works at least as competitive as EC-AL. However, WC-MGDA / EC-DBDA requires extra computation of the generating gradient matrix while WC / EC-AL does not. Hence, users can choose either method based on the cost-efficiency trade-off. Moreover, results from the tri-objective cases further support our conclusion.

4.4 Exploring PF around a Reference Model

We used the LS model with an early stopping point of 50 as the reference model and generated equi-distributed preferences. We then applied the WC and WC-MGDA methods to this setting. It is worth noting that it is simple to modify the WC method to handle a reference model by subtracting the cost of the reference model. The comparison between WC and WC-MGDA can be seen in Figure 7. Table 8 shows the results of the reference point-based methods (WC-MGDA and WC) in terms of MWL and HVI. As MA demonstrated better performance, we only used it in the methods. It is evident that WC-MGDA outperforms WC, which is consistent with the visualization in Figure 7. This use case enables us to automatically update production models using a fresh dataset, ultimately leading to improved performance across all objectives.

[1]Note, when computing HVI on cost, we scale each cost by the worst performance of single objective methods, so the HVI is not influenced by different scales of costs.

Table 8: Metrics on preference with reference points.

dataset	E-commerce			MSLR		
metric	MWL	HVI(cost)	HVI(ndcg)	MWL	HVI(tr)	HVI(ndcg)
WC-MA	-7.1e-2	8.9e-3	8.0e-4	-1.6e-1	5.1e-2	3.9e-3
WC-MGDA-MA	**-8.9e-2**	**1.3e-2**	**9.3e-4**	**-1.9e-1**	**6.2e-2**	**4.4e-3**
gain (%)	-27	50	16	-19	21	14

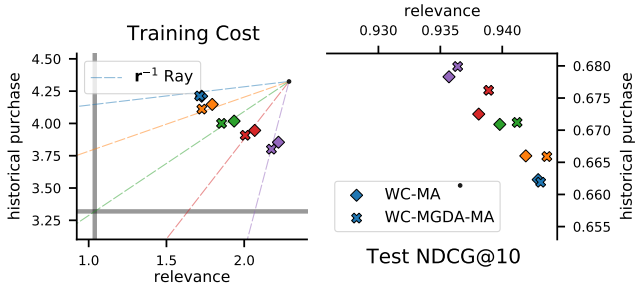


Figure 7: Exploring PF from a reference model (black dot) on E-commerce dataset.

5 CONCLUSION AND FUTURE WORK

We present a comprehensive framework for Multi-Label Learning to Rank that integrates any first-order gradient-based MOO algorithm to train a ranking model. Our framework incorporates three distinct trade-off specifications and implements a systematic approach to preserve the relative ranking quality with regards to various relevance criteria. Through a thorough evaluation of multiple state-of-the-art MOO algorithms, we demonstrate the efficacy of our framework by testing it on four publicly available datasets and one E-commerce dataset.

Our framework for MLLTR can be enhanced in several ways as further research. Firstly, the current pairwise cost can be extended to list-wise cost to improve performance. Secondly, one can investigate the use of non-convex surrogates, which have been shown to approximate the NDCG metric more effectively than list-wise costs (as per [3]). Thirdly, more MOO algorithms can be integrated to further refine and optimize our framework. Moreover, we will also study the online and offline performance for our proposed algorithms in production search engine [39]. We will make the source code public available.

REFERENCES

[1] Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. 2009. Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point. In *Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms* (FOGA '09). Association for Computing Machinery, Orlando, Florida, USA, 87–102.

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

[3] Sebastian Bruch. 2021. An alternative cross entropy loss for learning-to-rank. In *Proceedings of the Web Conference 2021* (WWW '21). Association for Computing Machinery, Ljubljana, Slovenia, 118–126.

[4] C. Burges. 2010. From ranknet to lambdarank to lambdamart: an overview. In.

[5] 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.

[6] 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* (WWW '20). Association for Computing Machinery, Taipei, Taiwan, 373–383.

[7] 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). Olivier Chapelle, Yi Chang, and Tie-Yan Liu, (Eds.) Vol. 14. PMLR, Haifa, Israel, 1–24.

[8] Na Dai, Milad Shokouhi, and Brian D. Davison. 2011. Learning to rank for freshness and relevance. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR '11). Association for Computing Machinery, Beijing, China, 95–104.

[9] Onkar Dalal, Srinivasan H. Sengemedu, and Subhajit Sanyal. 2012. Multi-objective ranking of comments on web. In *Proceedings of the 21st International Conference on World Wide Web* (WWW '12). Association for Computing Machinery, Lyon, France, 419–428.

[10] Domenico Dato, Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, Raffaele Perego, Nicola Tonellotto, and Rossano Venturini. 2016. Fast ranking with additive ensembles of oblivious and non-oblivious regression trees. *ACM Transactions on Information Systems (TOIS)*.

[11] Chaosheng Dong and Bo Zeng. 2020. Expert learning through generalized inverse multiobjective optimization: models, insights, and algorithms. In *International Conference on Machine Learning*.

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

[13] Chengyue Gong, Xingchao Liu, and qiang liu. 2021. Automatic and harmless regularization with constrained and lexicographic optimization: a dynamic barrier approach. In *Advances in Neural Information Processing Systems*.

[14] Changsung Kang, Xuanhui Wang, Yi Chang, and Belle Tseng. 2012. Learning to rank with multi-aspect relevance for vertical search. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining* (WSDM '12). Association for Computing Machinery, Seattle, Washington, USA, 453–462.

[15] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On application of learning to rank for e-commerce search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR '17). Association for Computing Machinery, Shinjuku, Tokyo, Japan, 475–484.

[16] Chang Li, Haoyun Feng, and Maarten de Rijke. 2020. Cascading hybrid bandits: online learning to rank for relevance and diversity. In *Fourteenth ACM Conference on Recommender Systems*.

[17] 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.

[18] Xiao Lin, Hongjie Chen, Changhua Pei, Fei Sun, Xuanji Xiao, Hanxiao Sun, Yongfeng Zhang, Wenwu Ou, and Peng Jiang. 2019. A pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems* (RecSys '19). Association for Computing Machinery, Copenhagen, Denmark, 20–28.

[19] Bo Long, Jiang Bian, Anlei Dong, and Yi Chang. 2012. Enhancing product search by best-selling prediction in e-commerce. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management* (CIKM '12). Association for Computing Machinery, Maui, Hawaii, USA, 2479–2482.

[20] Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, Raffaele Perego, Fabrizio Silvestri, and Salvatore Trani. 2016. Post-learning optimization of tree ensembles for efficient ranking. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*.

[21] Debabrata Mahapatra, Chaosheng Dong, and Michinari Momma. 2023. Querywise fair learning to rank through multi-objective optimization. In *Proceedings of the 29th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[22] Debabrata Mahapatra and Vaibhav Rajan. 2021. Exact pareto optimal search for multi-task learning: touring the pareto front. (2021). arXiv: 2108.00597 [cs.LG].

[23] 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*.

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

[25] 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*.

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

[27] Michinari Momma, Alireza Bagheri Garakani, and Yi Sun. 2019. Multi-objective relevance ranking. In *eCOM@ SIGIR*.

[28] Marco Morik, Ashudeep Singh, Jessica Hong, and Thorsten Joachims. 2020. Controlling fairness and bias in dynamic learning-to-rank. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*.

[29] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 datasets. *CoRR*, abs/1306.2597.

[30] Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747. arXiv: 1609.04747.

[31] Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*. S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, (Eds.) Vol. 31. Curran Associates, Inc.

[32] Ashudeep Singh and Thorsten Joachims. 2019. Policy learning for fairness in ranking. *Advances in Neural Information Processing Systems*, 32.

[33] 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* (SIGIR '16). Association for Computing Machinery, Pisa, Italy, 459–460.

[34] Bogdana Stanojević and Fred Glover. 2020. A new approach to generate pattern-efficient sets of non-dominated vectors for multi-objective optimization. *Information Sciences*, 530, 22–42.

[35] 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* (WWW '11). Association for Computing Machinery, Hyderabad, India, 367–376.

[36] 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*, 769–772.

[37] 2019. *Painless stochastic gradient: interpolation, line-search, and convergence rates. Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, 14 pages.

[38] Lidan Wang, Paul N. Bennett, and Kevyn Collins-Thompson. 2012. Robust ranking models via risk-sensitive optimization. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR '12). Association for Computing Machinery, Portland, Oregon, USA, 761–770.

[39] Xiaojie Wang, Ruoyuan Gao, Anoop Jain, Graham Edge, and Sachin Ahuja. 2023. How well do offline metrics predict online performance of product ranking models? In *Proceedings of the 46th International ACM SIGIR conference on Research and Development in Information Retrieval*.

[40] Le Yan, Zhen Qin, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2022. Scale calibration of deep ranking models. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

A EXPERIMENTS

A.1 Datasets and Experimental Settings

A.1.1 Yahoo! Learning to Rank Dataset. We experimented on Yahoo! Learning to Rank [7] challenge dataset with 36K queries. Each query-url pair is represented by 700 features. Although these features are engineered (not learnt), their descriptions, however, are not publicly released. Therefore, we selected several labels to use as additional objectives. Specifically, we selected features that have more than 5 levels of values, then chose the ones that were least correlated among each other. The selected labels are ['0', '18', '22', '39', '92']. Details can be seen in Figure 9. In total, we selected 5 objectives including the original relevance label, and created 15 bi-objectives and 10 tri-objective cases. The generation of preference and constraints follows the same strategy explained in MSLR-WEB30K. Note as we saw cost vanishing behavior coming from NDCG computation within LambdaRank due to low granularity, we use RankNet cost [5], which is the pairwise cost without NDCG factors. For tuning the model hyperparameters, we followed a similar strategy as in MSLR-WEB30K, and selected 600 trees and 0.25 learning rate. We used the original training and test data for our experiment.

A.1.2 Istella-S and Istella LETOR Datesets. We also ran experiments on Istella-S LETOR dataset [20] and Istella LETOR dataset [10], where the former is a smaller sample of the later dataset. Each query-url pair is represented by 200 features. Although these features are engineered (not learnt), their descriptions, however, are not publicly released. Therefore, we selected several labels to use as additional objectives. Specifically, we selected features that have more than 5 levels of values, then chose the ones that were least correlated among each other. The selected labels are ['0', '11', '194', '203', '214']. Details can be seen in Figure 10. In total, we selected 5 objectives including the original relevance label, and created 4 bi-objectives cases. The generation of preference and constraints follows the same strategy explained in MSLR-WEB30K. For tuning the model hyperparameters, we followed a similar strategy as in MSLR-WEB30K, and selected 1000 trees and 0.05 learning rate. We used the original training and test data for our experiment.

Experiment results: Figure 8 shows the results of one run for Istella LETOR dataset. We plot the preference based methods and ϵ−Constraint method separately to avoid overcrowding the figures. Moreover, both their respective trade-off specifications are different, further justifying our decision of separate figures.

Among the preference based methods, first we observe that the performance of LS and SLA are similar, from both the aspects of dominance and preference compliance w.r.t. the other MOO methods. This is expected as they are probabilistically equivalent [6]: expected cost of SLA is same as LS. However, in practice, the solutions of LS slightly dominate that of SLA in the cost space, which translates to a significant dominance in the NDCG space.

Next, we observe a similar relation among WC-MGDA, EPO and CS. The solutions of WC-MGDA either, in most cases, dominate or stay non-dominated w.r.t. to the solutions of EPO Search and CS. The dominance is apparent in the cost space. Solutions from all methods have stricter compliance to the preferences than that of LS and SLA.

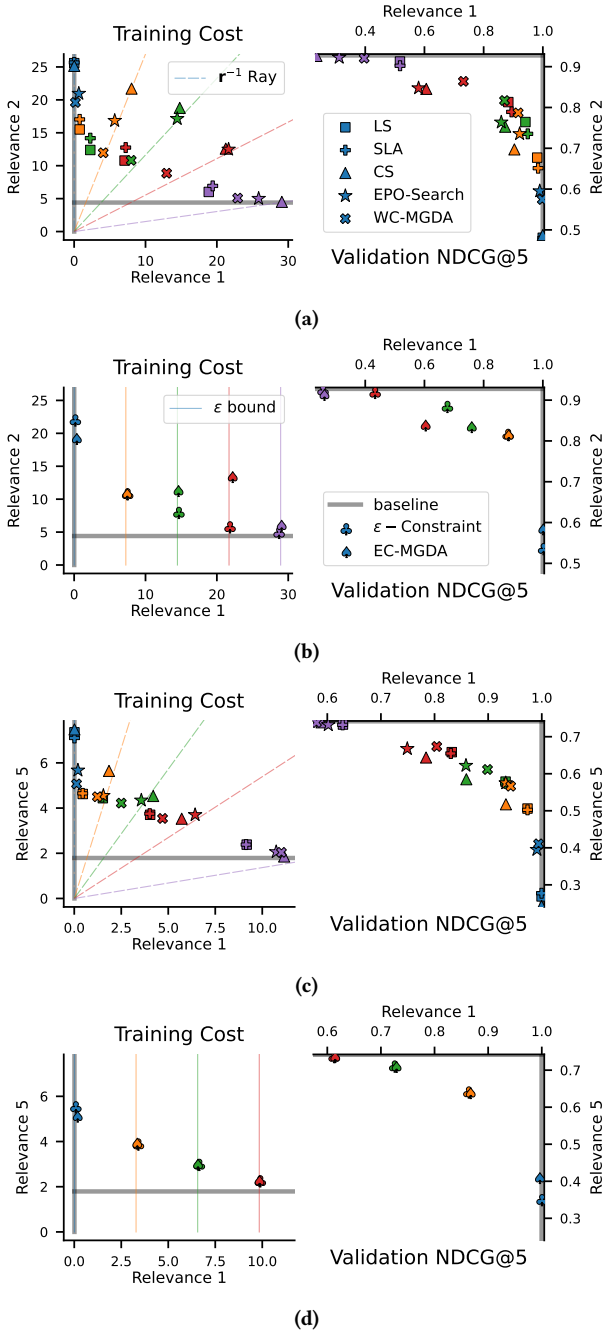


Figure 8: Results from four bi-objective experiments on Istella LETOR dataset. Figures 8a and 8c compare among the preference based methods, viz., LS, SLA, CS, and EPO Search. Figures 8b and 8d show results for ϵ−Constraint. For each experiment, we show the training costs and validation NDCG@5 for corresponding labels. Colored lines and points represent different trade-off specifications and the corresponding solutions, respectively.

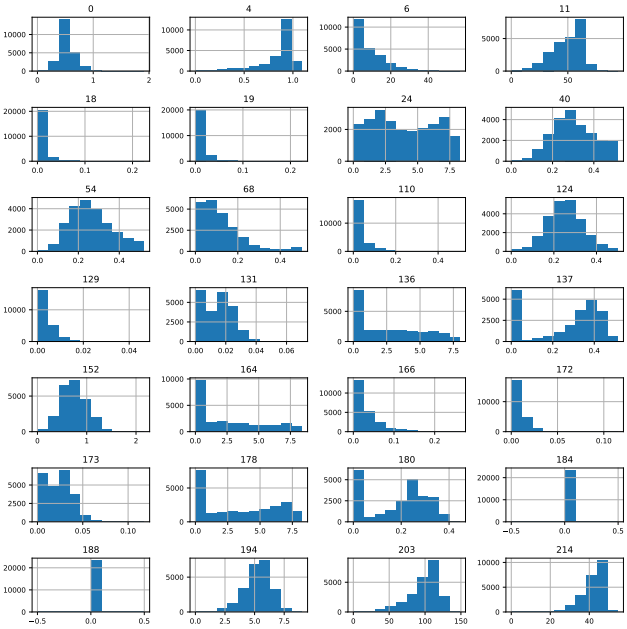


Figure 10: Objectives selection for Istella LETOR dataset. We plot the histogram of the standard deviation of the feature per query group. We then select the features that have the most diverisified values per query group.

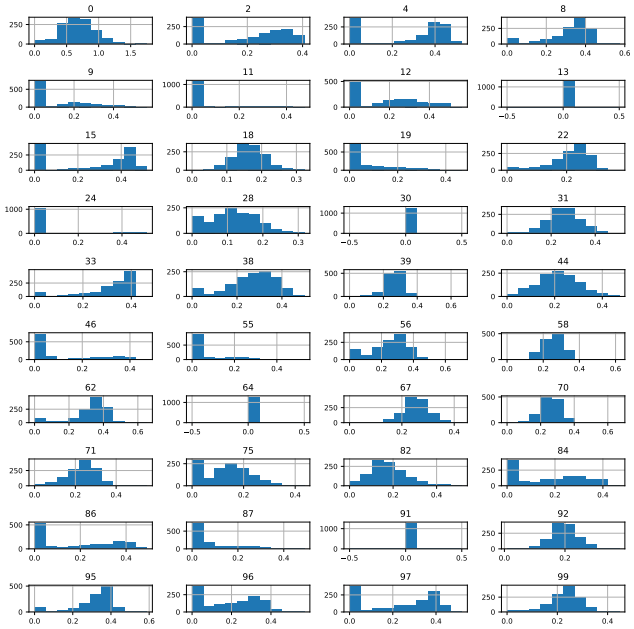


Figure 9: Objectives selection for Yahoo! Learning to Rank dataset. We plot the histogram of the standard deviation of the feature per query group. We then select the features that have the most diverisified values per query group.

For $\epsilon-$Constraint method, an equidistributed trade-off specification (the upper bounds) in the cost space does not lead to an equidistributed solution frontier in the NDCG space. However, this discrepancy is problem dependent, and should be attributed to the loose approximation of RankNet or LambdaRank [4] cost to NDCG metric, but not to the $\epsilon-$Constraint method. The MOO method complies to the given trade-off specifications in the cost space.