Regression Compatible Listwise Objectives for Calibrated Ranking

Aijun Bai, Rolf Jagerman, Zhen Qin, Pratyush Kar, Bing-Rong Lin,
Xuanhui Wang, Michael Bendersky, Marc Najork
{aijunbai,jagerman,zhenqin,pratk,bingrong,xuanhui,bemike,najork}@google.com

ABSTRACT

As Learning-to-Rank (LTR) approaches primarily seek to improve ranking quality, their output scores are not scale-calibrated by design – for example, adding a constant to the score of each item on the list will not affect the list ordering. This fundamentally limits LTR usage in score-sensitive applications. Though a simple multi-objective approach that combines a regression and a ranking objective can effectively learn scale-calibrated scores, we argue that the two objectives can be inherently conflicting, which makes the trade-off far from ideal for both of them. In this paper, we propose a novel regression compatible ranking (RCR) approach to achieve a better trade-off. The advantage of the proposed approach is that the regression and ranking components are well aligned which brings new opportunities for harmonious regression and ranking. Theoretically, we show that the two components share the same minimizer at global minima while the regression component ensures scale calibration. Empirically, we show that the proposed approach performs well on both regression and ranking metrics on several public LTR datasets, and significantly improves the Pareto frontiers in the context of multi-objective optimization. Furthermore, we evaluated the proposed approach on YouTube Search and found that it not only improved the ranking quality of the production pCTR model, but also brought gains to the click prediction accuracy.

CCS CONCEPTS

• **Information systems → Learning to rank**.

KEYWORDS

Learning-to-Rank, Regression, Calibration

1 INTRODUCTION

Ranking is the central part of many real-world applications, such as web search, online advertising, and recommendation systems [19]. Learning-to-Rank (LTR) aims to automatically construct a ranker from training data such that it can rank unseen objects correctly. It is therefore required that a ranker performs well on ranking metrics, such as *Normalized Discounted Cumulative Gain* (NDCG). It is usually the case that a ranking-centric pairwise or listwise approach, such as RankNet [3] or ListNet [33], achieves better ranking quality than a regression approach that adopts a pointwise formulation.

On the other hand, modern systems in these applications have multiple stages and downstream stages consume the scores from previous ones. For example, a deep neural network model provides a single signal to a lightweight interpretable model that fuses it with other signals [18]. It is expected that improving ranking quality at any stage can improve the quality of the whole system. In this setting, it is desired that the ranking scores are well calibrated and the distribution remains stable so that each stage can be improved independently. Take the application of online advertising as an example. In online advertising, the pCTR (predicted Click-Through Rate) model is required to be well calibrated because it affects the downstream auction and pricing models [6, 17, 34], though the final ranking of ads is the one that matters most for the performance. This suggests that we want the ranker to perform well not only on ranking metrics, but also on regression metrics in terms of calibrating ranker output scores to some external scale. It is in this sense that we use regression, calibration, and scale calibration interchangeably in the rest of this paper. Popular regression metrics include MSE for real-valued targets and the logistic loss (LogLoss) for probability targets. It is worth noting that there is a different field called uncertainty calibration [7, 16, 24], where "calibration" is performed to produce confidence intervals.

Unsurprisingly, capable ranking approaches, such as RankNet and ListNet, perform poorly on regression metrics, due to the fact that their loss functions are invariant to rank-preserving score transformations (more specifically score translations), and tend to learn scores that are not scale-calibrated to regression targets. Furthermore, these approaches suffer from training instability in the sense that the learned scores diverge indefinitely in continuous training or re-training [34]. These factors strongly limit their usage in score-sensitive applications. As a result, practitioners have no choice but to fall back to regression-only approaches even if they are sub-optimal in terms of ranking metrics – which are essentially the business objectives that should have been optimized.

It has been shown in the literature that a standard multi-objective approach effectively learns scale-calibrated scores for ranking [17, 29, 34]. However, we argue that in this standard multi-objective setting, the regression and ranking objectives are inherently conflicting and thus the best trade-off might not be ideal for either of them. In this paper, we propose a novel regression compatible ranking (RCR) approach which can be seen as a hybrid loss formulation where the pointwise regression component takes care of individual score approximation and the listwise ranking component takes care of in-list score distribution in a way that is compatible with the regression component. The advantage of the proposed approach is that the regression and ranking components are well aligned, which brings new opportunities for harmonious regression and ranking. Theoretically, we show that the two components share the same minimizer at global minima while the regression component ensures scale calibration. We conduct our experiments on several public LTR datasets, and show that the proposed approach achieves either the best or competitive result in terms of both regression and ranking metrics, and significantly improves the Pareto frontiers in the context of multi-objective optimization. Furthermore, we evaluated the proposed approach on YouTube Search and found it not only improved the ranking capability of the production pCTR model but also brought gains to the click prediction accuracy.

The rest of the paper is organized as follows. In Section 2, we review the related work. In Section 3, we formally lay out the

problem. We present the main approach in Section 4. Section 5 and Section 6 report our experiments on public LTR datasets and the proprietary YouTube Search dataset respectively. We conclude this paper in Section 7.

2 RELATED WORK

Classification problems differ from regression ones in terms of calibration. In regression problems, the regression losses such as MSE and SigmoidCE are usually calibrated by definition because the goal is to regress the predictions to the original labels [34]. In contrast, the goal of classification is to assign a class label to an instance and the model outputs do not necessarily have a probabilistic interpretation such as how likely an instance could be positive and thus are not calibrated. Post-processing techniques are usually used. For example, the output of an SVM model does not have probabilistic interpretation. Platt [25] proposed a parametric approach to train a logistic model on the outputs of an SVM model to address this issue. Later on, isotonic regression [20, 37] and binning method such as histogram binning and Bayesian binning [21, 36] are proposed for classification problems. Our work differs from these as we are working on ranking problems, not classification ones.

There is a long history of the study of Learning-to-Rank (LTR) in the past [19]. The general set up is that a scoring function is trained to score and sort a list of objects given a context. The accuracy is evaluated based on ranking metrics that only care about the order of the objects, but not the scale of the scores. Existing works include designing more effective loss function [4, 10, 33, 38], learning from biased interaction data [9, 13, 31, 35], and different underlying models, from support vector machines [12], to gradient boosted decision trees [14, 32], to neural networks [1, 2, 22, 23, 27, 28]. However, almost none of the existing works have studied the calibration issue of the ranking model outputs, which limits their applicability in many applications where a calibrated output is necessary.

To the best of our knowledge, few works have studied the ranking calibration problem. Similar to classification problems, post-processing methods can be used for calibrating ranking model outputs. For example, Tagami et al. [30] used the pairwise squared hinge loss to train an LTR model for ads ranking, and then used Platt-scaling [25] to convert the ranking scores into probabilities. Recently, Chaudhuri et al. [6] compared different post-processing methods to calibrate the outputs of an ordinal regression model, including Platt-scaling and isotonic regression. Our proposed method does not rely on a post-processing step.

Another class of approaches is based on multi-objective setting where ranking loss is calibrated by a regression loss during the training time, without an additional post-processing step. Sculley [29] is an early work that combines regression and ranking. It has been used in concrete application [17, 34]. In particular, Yan et al. [34] used the multi-objective formulation in deep models to prevent models from diverging during training and achieve output calibration at the same time. The shortcoming of such an approach is that ranking accuracy can be traded for calibration because the two objectives are not designed to be compatible. Our proposed method does not sacrifice ranking accuracy to achieve calibration.

3 BACKGROUND

Learning-to-Rank (LTR) concerns the problem of learning a model to rank a list of objects given a context. Throughout the paper, we use *query* to represent the context and *documents* to represent the objects. In the so-called *score-and-sort* setting, a ranker is learned to score each document, and a ranked list is formed by sorting documents according to the scores.

More formally, let $q \in Q$ be a query and $x \in \mathcal{X}$ be a document, a score function is defined as $s(q, x; \boldsymbol{\theta}) : Q \times \mathcal{X} \to \mathbb{R}$, where Q is the query space, $\mathcal{X}$ is the document space, and $\boldsymbol{\theta}$ is the parameters of the score function s. Without loss of generality, in this paper, we assume each query is associated with the same number of documents.

A typical LTR dataset D consists of examples represented as tuples $(q, x, y) \in D$ where q, x and y are query, document and label respectively. Let $\mathbf{q} = \{q|(q, x, y) \in D\}$ be the query set induced by D. Given model parameters $\boldsymbol{\theta}$, the loss function is formalized as

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{|\mathbf{q}|} \sum_{q \in \mathbf{q}} \mathcal{L}_{query}(\boldsymbol{\theta}; q), \tag{1}$$

where $\mathcal{L}_{query}(\boldsymbol{\theta}; q)$ is the loss function for a single query. Depending on how $\mathcal{L}_{query}$ is defined, LTR techniques can be roughly divided into three categories.

3.1 The Pointwise Approach

In the pointwise approach, it is assumed that the query loss $\mathcal{L}_{query}$ can be represented as a weighted sum of losses over documents sharing the same query, as in:

$$\mathcal{L}_{query}(\boldsymbol{\theta}; q) = \frac{1}{C} \sum_{(q,x,y) \in D} \mathcal{L}_{point}(\boldsymbol{\theta}; q, x, y), \tag{2}$$

where $\mathcal{L}_{point}(\boldsymbol{\theta}; q, x, y)$ is the loss for a single document x with label y given query q, and $C = |\{(x, y)|(q, x, y) \in D\}|$ is a normalizing factor. Depending on the dataset, different definitions of $\mathcal{L}_{point}$ can be utilized. For example, in logistic ranking, $\mathcal{L}_{point}$ is typically defined as the Sigmoid Cross Entropy loss (denoted by SigmoidCE):

$$\begin{aligned} \mathcal{L}_{point}(\boldsymbol{\theta}; q, x, y) &= \text{SigmoidCE}(s, y) \\ &= -y \log \sigma(s) - (1 - y) \log(1 - \sigma(s)), \end{aligned} \tag{3}$$

where $s = s(q, x; \boldsymbol{\theta})$ is the predicted score of query-document pair (q, x) and $\sigma(s) = (1 + \exp(-s))^{-1}$ is the sigmoid function.

In real-valued regression ranking, Mean Squared Error (MSE) is typically used to define $\mathcal{L}_{point}$ as:

$$\begin{aligned} \mathcal{L}_{point}(\boldsymbol{\theta}; q, x, y) &= \text{MSE}(s, y) \\ &= (y - s)^2, \end{aligned} \tag{4}$$

where $s = s(q, x; \boldsymbol{\theta})$ is the predicted score.

It has been shown that both SigmoidCE and MSE are scale-calibrated in the sense that they achieve global minima when

$$\sigma(s) \to \mathbb{E}[y|q, x] \tag{5}$$

for SigmoidCE, and

$$s \to \mathbb{E}[y|q, x] \tag{6}$$

for MSE [34].

3.2 The Pairwise Approach

In the pairwise approach, it is assumed that the query loss $\mathcal{L}_{query}$ can be represented as a weighted sum of losses over all document-document pairs sharing the same query, as in:

$$\mathcal{L}_{query}(\boldsymbol{\theta}; q) = \frac{1}{C} \sum_{\substack{(q,x_1,y_1) \in D \\ (q,x_2,y_2) \in D}} \mathcal{L}_{pair}(\boldsymbol{\theta}; q, x_1, y_1, x_2, y_2), \tag{7}$$

where $\mathcal{L}_{pair}(\boldsymbol{\theta}; q, x_1, y_1, x_2, y_2)$ is the loss for document pair (x_1, x_2), and $C = |\{(x_1, y_1, x_2, y_2)|(q, x_1, y_1) \in D \wedge (q, x_2, y_2) \in D\}|$.

The fundamental RankNet approach represents $\mathcal{L}_{pair}$ as a pairwise Logistic loss (denoted by PairwiseLogistic) [3]:

$$\begin{aligned}\mathcal{L}_{pair}(\boldsymbol{\theta}; q, x_1, y_1, x_2, y_2) &= \text{PairwiseLogistic}(s_1, s_2, y_1, y_2) \\ &= -\mathbb{I}(y_2 > y_1) \log \sigma(s_2 - s_1),\end{aligned} \tag{8}$$

where $s_1 = s(q, x_1; \boldsymbol{\theta})$ and $s_2 = s(q, x_2; \boldsymbol{\theta})$ are the predicted scores for documents x_1 and x_2, $\mathbb{I}$ is the indicator function, and σ is the sigmoid function. PairwiseLogistic achieves global minima when

$$\sigma(s_2 - s_1) \to \mathbb{E}[\mathbb{I}(y_2 > y_1)|q, x_1, x_2], \tag{9}$$

which indicates that the loss function mainly considers the score differences. As for the values of the scores, they could be arbitrarily worse with respect to regression metrics [29].

3.3 The Listwise Approach

In the listwise approach, the query loss $\mathcal{L}_{query}$ is attributed to the whole list of documents sharing the same query:

$$\mathcal{L}_{query}(\boldsymbol{\theta}; q) = \mathcal{L}_{list}(\boldsymbol{\theta}; q, \{(q, x, y) \in D\}). \tag{10}$$

The popular ListNet approach uses the Softmax based Cross Entropy loss (denoted by SoftmaxCE) to represent $\mathcal{L}_{list}$ as [33]:

$$\begin{aligned}\mathcal{L}_{list}(\boldsymbol{\theta}; q) &= \text{SoftmaxCE}(s_{1:N}, y_{1:N}) \\ &= -\frac{1}{C} \sum_{i=1}^{N} y_i \log \frac{\exp(s_i)}{\sum_{j=1}^{N} \exp(s_j)},\end{aligned} \tag{11}$$

where N is the list size, $s_i = s(q, x_i, \boldsymbol{\theta})$ is the predicted score, and $C = \sum_{j=1}^{N} y_j$. The global minima is achieved at [33]:

$$\frac{\exp(s_i)}{\sum_{j=1}^{N} \exp(s_j)} \to \frac{\mathbb{E}[y_i|q, x_i]}{\sum_{j=1}^{N} \mathbb{E}[y_j|q, x_j]}. \tag{12}$$

Similar to PairwiseLogistic, SoftmaxCE gives scores that could be arbitrarily worse with respect to regression metrics.

4 REGRESSION COMPATIBLE RANKING

In this section, we first give the motivation, then formally propose the approach to regression compatible ranking (RCR).

4.1 Motivation

It has been shown in the literature that a standard multi-objective approach effectively learns scale-calibrated scores for ranking [17, 29, 34]. Take logistic ranking as an example, they define the multi-objective loss as a weighted sum of SigmoidCE and SoftmaxCE losses:

$$\begin{aligned}\mathcal{L}_{query}^{MultiObj}(\boldsymbol{\theta}; q) = (1 - \alpha) \cdot \sum_{i=1}^{N} \text{SigmoidCE}(s_i, y_i) \\ + \alpha \cdot \text{SoftmaxCE}(s_{1:N}, y_{1:N}),\end{aligned} \tag{13}$$

where $\alpha \in [0, 1]$ is the trade-off weight. For simplicity, we refer to this method as SigmoidCE + SoftmaxCE. It can be seen that SigmoidCE + SoftmaxCE is no longer translation-invariant, and has been shown effective for calibrated ranking. Let's take a deeper look on what scores are being learned following this simple multi-objective formalization.

Given query q, let $P_i = \mathbb{E}[y_i|q, x_i]$ be the ground truth click probability further conditioned on document x_i. Recall that, SigmoidCE achieves global minima when $\sigma(s_i) \to P_i$, which means we have the following pointwise learning objective for SigmoidCE:

$$s_i \to \log P_i - \log(1 - P_i). \tag{14}$$

On the other hand, SoftmaxCE achieves global minima when

$$\frac{\exp(s_i)}{\sum_{j=1}^{N} \exp(s_j)} \to \frac{P_i}{\sum_{j=1}^{N} P_j}, \tag{15}$$

or equivalently:

$$s_i \to \log P_i - \log \sum_{j=1}^{N} P_j + \log \sum_{j=1}^{N} \exp(s_j), \tag{16}$$

where the log-$\sum$-exp term is an unknown constant and has no effects on the value or gradients of the final SoftmaxCE loss.

In the context of stochastic gradient descent, Equations 14 and 16 indicate that the gradients generated from the SigmoidCE and SoftmaxCE components are *pushing the scores to significantly different targets*. This reveals the fact that the two losses in a standard multi-objective setting are inherently conflicting and will fail to find a minimizer that will be ideal for both. How can we resolve this conflict?

Noticing that since $\sigma(s_i)$ is pointwisely approaching to P_i, if we replace the ground truth probabilities P_i in the right side of Equation 16 with the empirical approximations $\sigma(s_i)$ and drop the constant term, we are constructing some virtual logits:

$$s_i' \leftarrow \log \sigma(s_i) - \log \sum_{j=1}^{N} \sigma(s_j). \tag{17}$$

If we further apply SoftmaxCE loss on the new logits s_i', we are establishing the following novel listwise learning objective:

$$\frac{\exp(s_i')}{\sum_{j=1}^{N} \exp(s_j')} \to \frac{P_i}{\sum_{j=1}^{N} P_j}, \tag{18}$$

which is equivalent to

$$\frac{\sigma(s_i)}{\sum_{j=1}^{N} \sigma(s_j)} \to \frac{P_i}{\sum_{j=1}^{N} P_j}. \tag{19}$$

It is easy to see that Equation 14 implies Equation 19 automatically, which means, as pointwise regression and listwise ranking objectives, they are well aligned in the sense that they achieve global minima simultaneously. This example on logistic ranking motivates our approach to regression compatible ranking for logistic and regression ranking tasks.

4.2 The Main Approach

Inspired by the above motivating example, we firstly define a novel Listwise Cross Entropy loss (ListCE) as follows.

Definition 1. *Let N be the list size, $s_{1:N}$ be the predicted scores, and $y_{1:N}$ be the labels. Let $T(s) : \mathbb{R} \to \mathbb{R}^+$ be a non-decreasing transformation on scores. The Listwise Cross Entropy loss with transformation T is defined as:*

$$\text{ListCE}(T, s_{1:N}, y_{1:N}) = -\frac{1}{C} \sum_{i=1}^{N} y_i \log \frac{T(s_i)}{\sum_{j=1}^{N} T(s_j)}, \tag{20}$$

where $C = \sum_{j=1}^{N} y_j$ is a normalizing factor.

For the scope of this paper, we interchangeably use ListCE with transformation T, ListCE(T), or even ListCE when there is no ambiguity. We immediately have the following propositions.

Proposition 2. ListCE(exp) *reduces to* SoftmaxCE.

Proposition 3. ListCE(T) *achieves global minima when*

$$\frac{T(s_i)}{\sum_{j=1}^{N} T(s_j)} \rightarrow \frac{\mathbb{E}[y_i|q, x_i]}{\sum_{j=1}^{N} \mathbb{E}[y_j|q, x_j]}. \tag{21}$$

With the help of the ListCE loss, we apply a hybrid formulation to achieve regression compatible ranking. In particular, we selectively choose a pointwise regression loss which will be learning meaningful and scale-calibrated scores for the task of interest. We further choose a ListCE loss with a specific transformation to match the regression loss such that they achieve global minima simultaneously. It is possible to develop a hybrid formulation where we match the pointwise loss with a pairwise loss, however, in this paper, we mainly focus on listwise losses since 1) listwise losses have been shown consistently effective in popular ranking benchmarks [28], and 2) a listwise loss with list size 2 reduces to a pairwise loss. For tasks with different properties of labels, we should be able to find the most suitable transformations. In this paper, we mainly focus on logistic and regression ranking tasks in general and leave the optimal selection of transformations to future work.

4.3 RCR for Logistic Ranking Tasks

In logistic ranking, all labels are binarized or within the range of $[0, 1]$. A natural pointwise objective is the SigmoidCE loss. With SigmoidCE as the pointwise component, it is then required to use the sigmoid function as the transformation such that they can be optimized simultaneously without conflict.

Definition 4. *The RCR loss for a single query in a logistic ranking task is defined as:*

$$\mathcal{L}_{query}^{Compatible}(\boldsymbol{\theta}; q) = (1-\alpha) \cdot \sum_{i=1}^{N} \text{SigmoidCE}(s_i, y_i) + \alpha \cdot \text{ListCE}(\sigma, s_{1:N}, y_{1:N}), \tag{22}$$

where σ is the sigmoid function.

For simplicity, we refer to this method as SigmoidCE+ListCE(σ). We have the following proposition:

Proposition 5. SigmoidCE + ListCE(σ) *achieves global minima when*

$$\sigma(s_i) \rightarrow \mathbb{E}[y_i|q, x_i]. \tag{23}$$

4.4 RCR for Regression Ranking Tasks

The same idea can be extended to real-valued regression ranking. In this paper, we assume all labels are greater than or equal to zero, and leave regression compatible ranking with negative labels to future work.

For regression ranking tasks, it is natural to use MSE as the pointwise component to learn the labels directly – which further implies the use of the ListCE loss with the identity transformation as the listwise component. However, this implementation is troublesome since the raw logits from the ranker could be negative and invalidate the cross entropy computation. To overcome this limitation, we choose the softplus function as the transformation before applying the MSE and ListCE losses. In experiments, we also tried using the exp function as the transformation but found it is not numerically stable in the hybrid formulation.

Table 1: In logistic ranking, given labels ($[0.4, 0.4, 0.5]$) and predictions (after sigmoid transformations), comparison of different losses for different rankers.

	Predictions	SigmoidCE	SoftmaxCE	ListCE(σ)
Ranker 1	[0.4, 0.4, 0.4]	**2.060**	**1.099**	1.099
Ranker 2	[0.2, 0.2, 0.3]	2.336	1.105	**1.097**
Ranker 3	[0.1, 0.1, 0.2]	2.885	1.135	1.120
Ranker 4	[0.4, 0.4, 0.6]	**2.060**	1.135	**1.097**

Definition 6. *The RCR loss for a single query in a regression ranking task is defined as:*

$$\mathcal{L}_{query}^{Compatible}(\boldsymbol{\theta}; q) = (1-\alpha) \cdot \sum_{i=1}^{N} \text{MSE}(\text{softplus}(s_i), y_i) + \alpha \cdot \text{ListCE}(\text{softplus}, s_{1:N}, y_{1:N}), \tag{24}$$

where $\text{softplus}(s) = \log(1 + \exp(s))$ *is the softplus function.*

Throughout the paper, we refer to this method as MSE(softplus) + ListCE(softplus), where MSE(softplus) stands for applying MSE after a softplus transformation. We have the following proposition:

Proposition 7. MSE(softplus)+ListCE(softplus) *achieves global minima when*

$$\text{softplus}(s_i) \rightarrow \mathbb{E}[y_i|q, x_i]. \tag{25}$$

4.5 Discussion

The proposed RCR approach can be seen as a hybrid loss formulation where the pointwise objective takes care of individual score approximation and the listwise objective takes care of in-list score distribution. More importantly, they share the same minimizer at global minima. Let's consider the following example to see how they benefit both regression and ranking.

Take logistic ranking as an example. Suppose we have a list of items where the ground truth click probabilities are $[0.4, 0.4, 0.5]$. Table 1 summarizes their predictions (after sigmoid transformations) and different losses on the raw scores. From the results, it can be seen that Rankers 1 and 3 achieve the best regression accuracy, however Ranker 1 gets the ranking completely wrong. On the other hand, Rankers 2, 3 and 4 get the perfect ranking, however the predictions of Rankers 2 and 3 have poor accuracy in terms of regression. If we use the SigmoidCE loss in training, it cannot differentiate Ranker 1 from Ranker 4; if we use ListCE in training, it cannot differentiate Ranker 2 from Ranker 4; if we use SoftmaxCE in training, it cannot differentiate Ranker 3 from Ranker 4; even worse, it would prefer Ranker 1! Relatively speaking, Ranker 4 is the best ranker among the four as it achieves the best metrics in both regression and ranking simultaneously. However, it is hard to tell the best ranker using a standalone regression or ranking loss we considered in this example. Fortunately, if we choose the proposed SigmoidCE+ListCE(σ) loss, for any $\alpha \in (0, 1)$, it would clearly prefer Ranker 4 – which is indeed the best. It is worth noting that the standard multi-objective approach (i.e. SigmoidCE + SoftmaxCE) counter-intuitively prefers Ranker 1 for any $\alpha \in [0, 1]$.

From this example, we can see that, in the proposed hybrid formulation, the pointwise regression component would shape the predictions to be more individually accurate, while the listwise ranking component would shape the predictions to be more consistently distributed. From the regression perspective, the added ranking constraint compatibly promotes scores that are correctly ranked. From the ranking perspective, the added regression constraint compatibly ensures scale calibration. Since they share the same minimizer at global minima, the combined approach would achieve optimal predictions for both at the same time. We suspect that the proposed approach will be widely applicable for many domains where there is a list structure – which is typical in applications such as web search, recommendation systems, etc.

5 EXPERIMENTS ON PUBLIC LTR DATASETS

To validate the proposed approach, we conduct our experiments on several public LTR datasets in this section.

5.1 Experiment Setup

5.1.1 Datasets. We extensively compare our methods with baselines on three datasets: Web30K [26], Yahoo [5], and Istella [8]. These datasets have graded relevance labels and thus are suitable for real-valued regression ranking. To study logistic ranking, we simply binarize them by treating non-zero labels as 1s.

Web30K is a public dataset where the 31531 queries are split into training, validation, and test partitions with 18919, 6306, and 6306 queries respectively. There are on average about 119 candidate documents associated with each query. Each document is represented by 136 numerical features and graded with a 5-level relevance label. The percentages of documents with relevance label equal to 0, 1, 2, 3, and 4 are about 51.4%, 32.5%, 13.4%, 1.9%, and 0.8%. When being binarized, the percentages for 0 and 1 are 51.4% and 48.6%.

Yahoo LTR challenge dataset consists of 29921 queries, with 19944, 2994 and 6983 queries for training, validation, and test respectively. There are 700 numerical features extracted for each query-document pair. The average number of documents per query is 24, but some queries have more than 100 documents. The labels are numerically graded. The distribution over 0, 1, 2, 3, and 4 is 21.9%, 50.2%, 22.3%, 3.9%, and 1.7%. In binarized form, the distribution over 0 and 1 is 21.9% and 78.1%.

Istella LETOR dataset is composed of 33018 queries, with 20901, 2318, and 9799 queries respectively in training, validation, and test partitions. The candidate list to each query is with on average 316 documents, and each document is represented by 220 numerical features. The graded relevance labels also vary from 0 to 4 but with a more skewed distribution: 96.3% for 0s, 0.8% for 1s, 1.3% for 2s, 0.9% for 3s, and 0.7% for 4s. With binarization, this distribution becomes 96.3% for 0s and 3.7% for 1s.

5.1.2 Metrics. In our experiments, we are interested in both regression and ranking performance. For ranking performance, we adopt the popular NDCG@10 [11] as the main metric. Higher NDCG@10 values indicate better ranking. More formally, given a list of labels $y_{1:N}$ and a list of output scores $s_{1:N}$, NDCG@k is defined as:

$$\text{NDCG@}k(s_{1:N}, y_{1:N}) = \frac{\text{DCG@}k(s_{1:N}, y_{1:N})}{\text{DCG@}k(y_{1:N}, y_{1:N})}, \tag{26}$$

where DCG@k is the so-called *Discounted Cumulative Gain* up to position k metric defined as:

$$\text{DCG@}k(s_{1:N}, y_{1:N}) = \sum_{i=1}^{N} \mathbb{I}(\pi(s_i) \le k) \frac{2^{y_i} - 1}{\log_2(\pi(s_i) + 1)}, \tag{27}$$

where $\pi(s_i)$ is the 1-based rank of score s_i in the descendingly sorted list of $s_{i:N}$.

For regression performance, we mainly look at LogLoss and MSE for logistic and regression ranking tasks respectively, which are defined as:

$$\text{LogLoss}(\hat{y}_{1:N}, y_{1:N}) = -\frac{1}{N} \sum_{i=1}^{N} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i), \tag{28}$$

and,

$$\text{MSE}(\hat{y}_{1:N}, y_{1:N}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \tag{29}$$

where N is the total data size, y_i is the label and $\hat{y}_i$ is the predicted label. Note that for LogLoss, $\hat{y}_i = \sigma(s_i)$ is the predicted probability after sigmoid transformation. Lower LogLoss or MSE values indicate better regression performance.

5.1.3 Methods. To validate in logistic ranking tasks, we mainly compare the proposed SigmoidCE + ListCE(σ) method with SigmoidCE, SoftmaxCE and SigmoidCE + SoftmaxCE. We also compare with ListCE(σ) – which is equivalent to $\alpha = 1$ in SigmoidCE + ListCE(σ). For real-valued regression ranking tasks, we compare MSE(softplus) + ListCE(softplus) with MSE, SoftmaxCE and MSE + SoftmaxCE. Additionally, we compare with MSE(softplus) and ListCE(softplus) which are equivalent to $\alpha = 0$ and $\alpha = 1$ in MSE(softplus) + ListCE(softplus).

We conduct our experiments using the TF-Ranking library [23]. In all experiments, we fix the ranker architecture to be a 3-layer Dense Neural Network (DNN) whose hidden layer dimensions are 1024, 512 and 256. The fraction of neuron units dropped out in training is set to be 0.5. We also apply a batch normalization layer after each dense layer. Additionally, we apply a simple $\log1\text{p}(x) = \text{sign}(x) \log(1 + |x|)$ transformation on input features for Web30K and Istella datasets due to its effectiveness [28]. We run the experiments on GPUs and use 128 as the training batch size. We use Adam [15] as the optimizer, and perform an extensive grid search of learning rates (LRs) and α over $[0.01, 0.001, 0.0001] \times [0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 0.9, 0.95, 0.99, 0.995, 0.999, 1]$. Note that, for MSE(softplus) and MSE(softplus) + ListCE(softplus), the MSE metric is computed after softplus transformations. To study the behavior of the combined approaches in the context of multi-objective optimization, we evaluate all experiments on the test data, and plot each experiment as a regression-ranking metrics data point and draw the Pareto frontiers.

5.2 Experimental Results

5.2.1 Main Comparisons. In real-world applications, we assume both regression and ranking metrics are critical, however, there might be an application-dependent priority on either ranking or regression metric. Therefore, we report results on test set after model selection on validation set by both regression and ranking metrics. We define a configuration being a dataset and a model selection method. It is easy to see that each configuration associated with a specified loss function produces exactly one ranker/model which

Table 2: Comparisons on logistic ranking tasks. Model selection is done on the validation set by either NDCG@10 or LogLoss metric, with test set results reported. ▲ and ▼ indicate statistical significance with p-value=0.05 of better and worse results than SigmoidCE + SoftmaxCE**.**

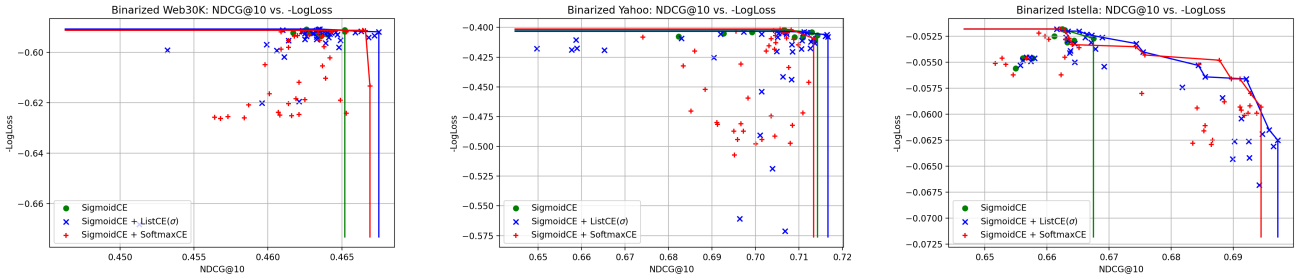
Datasets	Web30K				Yahoo				Istella			
Model Selection	By NDCG@10		By LogLoss		By NDCG@10		By LogLoss		By NDCG@10		By LogLoss	
Metrics	NDCG@10	LogLoss	NDCG@10	LogLoss	NDCG@10	LogLoss	NDCG@10	LogLoss	NDCG@10	LogLoss	NDCG@10	LogLoss
SigmoidCE	0.4626▼	**0.5908**▲	0.4626	**0.5908**▲	0.7130	**0.4041**▲	**0.7130**▲	0.4041▼	0.6675▼	**0.0527**▲	0.6612	0.0525▼
SoftmaxCE	0.4606▼	12.1008▼	0.4590▼	2.0298▼	0.7021▼	1.5221▼	0.7021▼	1.5221▼	0.6906▼	55.3837▼	0.6874▲	1.5417▼
ListCE(σ)	0.4569▼	0.6609▼	0.4569▼	0.6609▼	0.7022▼	0.6288▼	0.6878▼	0.6117▼	0.6956▲	0.0622▼	**0.6956**▲	0.0622▼
SigmoidCE + SoftmaxCE	**0.4669**	0.6134	**0.4664**	0.5912	0.7123	0.4462	0.7057	**0.4014**	0.6938	0.0599	0.6622	**0.0518**
SigmoidCE + ListCE(σ)	0.4649	0.5951▲	0.4634	**0.5908**▲	**0.7165**▲	0.4062▲	0.7112▲	0.4039▼	**0.6972**▲	0.0625▼	0.6616	**0.0518**

Table 3: Comparisons on regression ranking tasks. Model selection is done on the validation set by either NDCG@10 or MSE metric, with test set results reported. ▲ and ▼ indicate statistical significance with p-value=0.05 of better and worse results than MSE + SoftmaxCE**. We use** MSE(s.p.) + ListCE(s.p.) **to represent** MSE(softplus) + ListCE(softplus) **for space limitation.**

Datasets	Web30K				Yahoo				Istella			
Model Selection	By NDCG@10		By MSE		By NDCG@10		By MSE		By NDCG@10		By MSE	
Metrics	NDCG@10	MSE	NDCG@10	MSE	NDCG@10	MSE	NDCG@10	MSE	NDCG@10	MSE	NDCG@10	MSE
MSE	0.5040	0.5400▲	**0.5040**▲	0.5400▼	0.7712	**0.5768**▲	**0.7723**	0.5762▼	0.7177▼	0.1128▲	0.7158	0.1119▼
MSE(softplus)	0.5026▼	**0.5398**▲	0.5037	**0.5382**▲	0.7711	0.5790▲	0.7709	0.5731▲	0.7158▼	**0.1096**▲	0.7158	0.1096▲
SoftmaxCE	0.5010▼	1690.7▼	0.5012	16.674▼	0.7689▼	2214.0▼	0.7653▼	24.654▼	0.7248	3549.3▼	0.7163	38.3080▼
ListCE(softplus)	0.5013▼	10.014▼	0.5006	1.2247▼	0.7702▼	1.6955▼	0.7693	0.6888▼	0.7230	2.2196▲	0.7168▲	0.5215▼
MSE + SoftmaxCE	0.5048	0.5546	0.5023	0.5387	**0.7724**	0.5945	0.7716	0.5750	0.7245	2.3167	0.7143	0.1099
MSE(s.p.) + ListCE(s.p.)	**0.5054**	0.5399▲	0.5035	0.5383▲	0.7718	0.5769▲	0.7722	**0.5719**▲	**0.7257**	0.1153▲	**0.7183**▲	**0.1086**▲

Table 4: Summary of experimental results on the public LTR datasets in terms of winning on both regression and ranking metrics, where #N, #W and %W stand for the number of comparisons, the number of winnings on both metrics, and the winning rate respectively. "All Others" include all comparing approaches except the approach in subject itself.

	v.s. Pointwise			v.s. Listwise			v.s. Multi-Objective			v.s. RCR			v.s. All Others		
	#N	#W	%W	#N	#W	%W	#N	#W	%W	#N	#W	%W	#N	#W	%W
Pointwise				36	24	66.7%	18	3	16.7%	18	1	5.6%	72	28	38.9%
Listwise	36	0	0.0%				24	0	0.0%	24	0	0.0%	84	0	0.0%
Multi-Objective	18	1	5.6%	24	17	70.8%				12	1	8.3%	54	19	35.18%
RCR (proposed)	18	7	**38.9%**	24	21	**87.5%**	12	6	**50.0%**				54	34	**63.0%**
All Others	72	8	11.1%	84	62	73.8%	54	9	16.7%	54	2	**3.7%**			

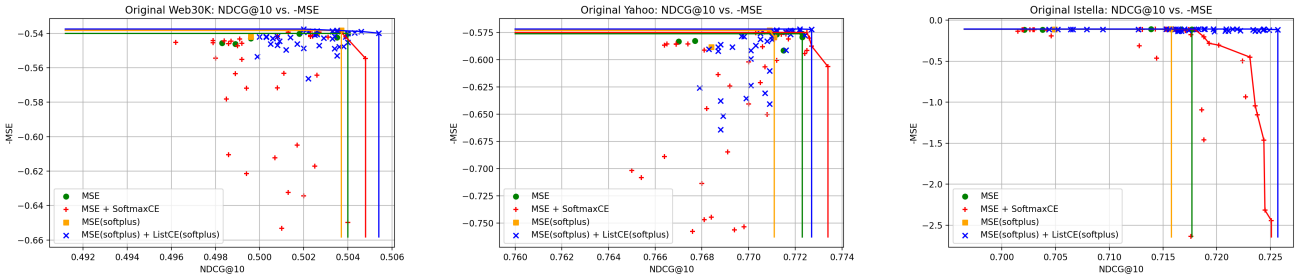


(a) Binarized Web30K. **(b) Binarized Yahoo.** **(c) Binarized Istella.**

Figure 1: Pareto frontiers on the binarized Web30K, Yahoo and Istella datasets.

deterministically gives a pair of regression and ranking metrics. The main results are shown in Tables 2 and 3 for logistic and regression ranking tasks respectively.

Given a configuration, we treat the comparison between two rankers produced by two loss functions as a game, where one ranker strictly wins the other if and only if it achieves the best performance (inclusive) on both regression and ranking metrics. For example, given binarized Web30K as the dataset and NDCG@10 as the model selection method, let Ranker 1 be the model produced by SigmoidCE + ListCE(σ), and Ranker 2 be the model produced by SigmoidCE + SoftmaxCE. Suppose the NDCG@10 and LogLoss metrics of Ranker $i \in \{1, 2\}$ are m_i and l_i respectively. Then Ranker 1 strictly wins Ranker 2 if and only if $m_1 \geq m_2$ and $l_1 \leq l_2$. It is worth noting that,



(a) Original Web30K. **(b) Original Yahoo.** **(c) Original Istella.**

Figure 2: Pareto frontiers on the original Web30K, Yahoo and Istella datasets.

in the context of multi-objective optimization, the strict winning relationship is also known as domination.

We group a set of loss functions originated from the same category as an approach: 1) the Pointwise approach includes SigmoidCE, MSE and MSE(softplus); 2) the Listwise approach includes SoftmaxCE, ListCE(σ) and ListCE(softplus); 3) the Multi-Objective approach includes SigmoidCE + SoftmaxCE and MSE + SoftmaxCE; and, 4) the RCR approach includes SigmoidCE + ListCE(σ) and MSE(softplus) + ListCE(softplus).

When comparing two approaches, we compare every pair of produced rankers from different approaches and count the number of strict winnings of one vs. the other. Table 4 summarizes the results of Tables 2 and 3 in terms of strict winning relationships among Pointwise, Listwise, Multi-Objective and RCR approaches. From the results, we can make the following observations:

- The Pointwise approach is very competitive. It consistently dominates the Listwise approach, achieving a domination rate of 66.7%. This is mainly because it is optimal in terms of regression metrics by design, and high regression performance usually implies high ranking performance (although not necessarily being optimal).
- The Listwise approach is not competitive at all. This aligns with the literature that although it has high performance on ranking metrics, its performance on regression metrics is very poor.
- The simple Multi-Objective approach has similar performance comparing with the Pointwise approach. This aligns with our hypothesis that the simple Multi-Objective approach fails to find a solution that is ideal for both regression and ranking metrics.
- The proposed RCR approach consistently dominates other approaches, achieving the highest domination rate of 63.0% globally. It also achieves the lowest rate of being dominated by other approaches, which is only 3.7%.

These observations indicate the proposed RCR approach is stable and performs well in terms of both regression and ranking metrics on a variety of configurations.

5.2.2 Pareto Frontier Comparisons. In the context of multi-objective optimization, the Pareto frontier is the set of Pareto optimal solutions where there is no scope for further Pareto improvement which is defined as a new solution where at least one objective gains, and no objectives lose. For each method, we evaluate all models over the hyper parameter space on the test data, plot each result as a regression-ranking metrics data point, and draw the Pareto frontier. The results are shown in Figures 1 and 2 for logistic and regression ranking tasks respectively. Note that we use -LogLoss and -MSE in the figures so the Pareto frontier corresponds to the maxima of a point set. In this section, we mainly compare RCR with pointwise baselines and the standard multi-objective baselines, since the listwise baselines are consistently inferior to other methods per previous section. From the figures, we can see that RCR dominates pointwise baselines in all tasks consistently. Furthermore, it dominates the multi-objective baseline in all tasks except binarized Istella and original Yahoo, where it achieves competitive Pareto frontiers. This suggests RCR could improve the Pareto frontiers and give new Pareto optimal solutions in a wide range of tasks.

6 EXPERIMENTS ON YOUTUBE SEARCH

We further evaluate our approach on a real-world dataset gathered from YouTube Search.

6.1 Background

In YouTube Search, real-time user interaction data, represented as item-click pairs, is streaming to the training infrastructure in a continuous way. Our baseline is a pCTR model that is equipped with the traditional SigmoidCE loss. Recently, new data with search page information, represented as page-item-click tuples, is made available to training, which gives the opportunity to directly improve its ranking quality within a search page. However, as stated previously, a direct switch from pointwise pCTR model to pairwise/listwise ranking model will not work in practice due to score calibration issues. It is required to improve the ranking quality of the model without affecting the values of its click predictions in any noticeable way.

6.2 Experiments

In this context, we compared the standard multi-objective approach – SigmoidCE + SoftmaxCE – and our RCR approach – SigmoidCE + ListCE(σ) – following the same setting in our baseline and train them continuously over the past ~1 week of data over the same number of training steps. The weight α is set to be 0.001 for both methods. We use AUC_PR and LogLoss for regression accuracy and NDCG for ranking quality, where AUC_PR is defined as the area under the Precision-Recall curve. Higher AUC_PR or lower LogLoss indicate better regression accuracy. We report the results in Table 5. Note that for proprietary reasons, we only report relative numbers

Table 5: Comparisons with respect to relative differences in YouTube Search with SigmoidCE as the baseline.

	AUC_PR	LogLoss	NDCG@1	NDCG@5	NDCG@10	NDCG
Multi-Objective: SigmoidCE + SoftmaxCE	-0.37%	+0.13%	**+0.30%**	**+0.16%**	**+0.15%**	**+0.15%**
RCR (proposed): SigmoidCE + ListCE(σ)	**+0.22%**	**+0.03%**	+0.27%	+0.13%	+0.13%	+0.13%

to our baseline (SigmoidCE). From the results, we can see that the standard multi-objective approach improved the pCTR model on the NDCG ranking metric, but it caused significant degradation in both AUC_PR and LogLoss metrics. Such a degradation can significantly affect the downstream stages negatively, making the models not suitable for the system. The proposed RCR approach not only improved the ranking quality, but also brought gains to pCTR predictions in terms of AUC_PR. This is because in our approach the ranking component optimizes for the ranking capability directly in a way that is compatible with the regression component and is acting as a valid and aligned in-list constraint for regression – which eventually helps the learning on regression. We also noticed that the proposed approach had a slight increase on LogLoss. This might be because the additional weight added on the ranking loss caused the learning on the regression loss to be less efficient than our baseline which is regression-only, thus the proposed approach may need more training steps for convergence. The result from this experiment suggests that the proposed approach generalize well to real-world production systems. The added ranking constraint not only improves ranking, but also benefits regression.

7 CONCLUSION

In this paper, we propose the novel regression compatible ranking (RCR) approach for logistic and regression ranking tasks in general. Theoretically, we show that the regression and ranking components within the approach share the same minimizer at global minima which indicates that RCR optimizes for regression and ranking simultaneously without conflict. Empirically, we show that RCR performs well on both regression and ranking metrics on several public LTR datasets, and significantly improves the Pareto frontiers in the context of multi-objective optimization. Furthermore, we show that RCR successfully improves both regression and ranking performances of a production pCTR model in YouTube Search. We expect RCR to bring new opportunities for harmonious regression and ranking and to be applicable in a wide range of real-world applications where there is a list structure. In future work, we are interested in exploring more formulations for regression compatible ranking and beyond.

A APPENDIX

A.1 Proof of Proposition 3

Proof. In training, given query q and a document x, its label can be considered as a sample from the underlying distribution. Let $\overline{y} = \mathbb{E}[y|q,x]$ be the expected label of query-document pair (q,x). Applying the ListCE loss on $(x,y) \in D$ is then equivalent to applying it on $(x,\overline{y})$ in expectation. Given transformation T, and predicted scores $s_{1:N}$, we define $p_i = T(s_i)/\sum_{j=1}^N T(s_j)$. The proposed ListCE loss then becomes:

$$\text{ListCE}(T, s_{1:N}, \overline{y}_{1:N}) = -\frac{1}{\sum_{j=1}^N \overline{y}_j} \sum_{i=1}^N \overline{y}_i \log p_i, \tag{30}$$

subject to

$$\sum_{i=1}^N p_i = 1. \tag{31}$$

Let's construct the following Lagrangian formalization:

$$\mathcal{L}(p_{1:N}, \lambda) = -\frac{1}{\sum_{j=1}^N \overline{y}_j} \sum_{i=1}^N \overline{y}_i \log p_i + \lambda(\sum_{i=1}^N p_i - 1). \tag{32}$$

Finding the extremum value of Equation 30 is then equivalent to finding the stationary points of Equation 32, which requires:

$$\frac{\partial \mathcal{L}(p_{1:N}, \lambda)}{\partial p_i} = -\frac{\overline{y}_i}{p_i \sum_{j=1}^N \overline{y}_j} + \lambda = 0, \tag{33}$$

and

$$\frac{\partial \mathcal{L}(p_{1:N}, \lambda)}{\partial \lambda} = \sum_{i=1}^N p_i - 1 = 0. \tag{34}$$

Note that Equations 33 and 34 give us a system of $N+1$ equations on $N+1$ unknowns. It is easy to see that the unique solution is

$$p_i = \frac{\overline{y}_i}{\sum_{j=1}^N \overline{y}_j}, \tag{35}$$

and

$$\lambda = 1. \tag{36}$$

This indicates the unique global extremum at

$$\frac{T(s_i)}{\sum_{j=1}^N T(s_j)} \rightarrow \frac{\mathbb{E}[y_i|q,x_i]}{\sum_{j=1}^N \mathbb{E}[y_j|q,x_j]}. \tag{37}$$

It is easy to verify that this unique global extremum attributes to the global minima which concludes the proof. ■

A.2 Proof of Proposition 5

Proof. The SigmoidCE component achieves global minima when $\sigma(s_i) \rightarrow \mathbb{E}[y_i|q,x_i]$ which implies

$$\frac{\sigma(s_i)}{\sum_{j=1}^N \sigma(s_j)} \rightarrow \frac{\mathbb{E}[y_i|q,x_i]}{\sum_{j=1}^N \mathbb{E}[y_j|q,x_j]}, \tag{38}$$

which minimizes ListCE(σ) at its global minima. ■

A.3 Proof of Proposition 7

Proof. The MSE(softplus) component achieves global minima when $\text{softplus}(s_i) \rightarrow \mathbb{E}[y_i|q,x_i]$ which implies

$$\frac{\text{softplus}(s_i)}{\sum_{j=1}^N \text{softplus}(s_j)} \rightarrow \frac{\mathbb{E}[y_i|q,x_i]}{\sum_{j=1}^N \mathbb{E}[y_j|q,x_j]}, \tag{39}$$

which minimizes ListCE(softplus) at its global minima. ■

REFERENCES

[1] Christopher Burges, Robert Ragno, and Quoc Le. 2007. Learning to Rank with Nonsmooth Cost Functions. In *Advances in Neural Information Processing Systems*. MIT Press.

[2] 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.

[3] Christopher JC Burges. 2010. From RankNet to LambdaRank to LambdaMART: An Overview. *Learning* 11, 23-581 (2010), 81.

[4] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*. 129–136.

[5] Olivier Chapelle and Yi Chang. 2011. Yahoo! learning to rank challenge overview. *Proceedings of Machine Learning Research* 14 (2011), 1–24.

[6] Sougata Chaudhuri, Abraham Bagherjeiran, and James Liu. 2017. Ranking and Calibrating Click-Attributed Purchases in Performance Display Advertising. In *2017 AdKDD & TargetAd*. 7:1–7:6.

[7] Daniel Cohen, Bhaskar Mitra, Oleg Lesota, Navid Rekabsaz, and Carsten Eickhoff. 2021. Not all relevance scores are equal: Efficient uncertainty and calibration modeling for deep retrieval models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 654–664.

[8] 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)* 35, 2 (2016), 1–31.

[9] Rolf Jagerman, Harrie Oosterhuis, and Maarten de Rijke. 2019. To model or to intervene: A comparison of counterfactual and online learning to rank from user interactions. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 15–24.

[10] Rolf Jagerman, Zhen Qin, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2022. On Optimizing Top-K Metrics for Neural Ranking Models. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[11] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[12] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 133–142.

[13] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*. 781–789.

[14] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*. 3146–3154.

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Ranganath Krishnan and Omesh Tickoo. 2020. Improving model calibration with accuracy versus uncertainty optimization. *Advances in Neural Information Processing Systems* 33 (2020), 18237–18248.

[17] Cheng Li, Yue Lu, Qiaozhu Mei, Dong Wang, and Sandeep Pandey. 2015. Click-through Prediction for Advertising in Twitter Timeline. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1959–1968.

[18] Pan Li, Zhen Qin, Xuanhui Wang, and Donald Metzler. 2019. Combining Decision Trees and Neural Networks for Learning-to-Rank in Personal Search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2032–2040.

[19] Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.

[20] Aditya Krishna Menon, Xiaoqian Jiang, Shankar Vembu, Charles Elkan, and Lucila Ohno-Machado. 2012. Predicting accurate probabilities with a ranking loss. In *Proceedings of the 29th International Conference on Machine Learning*. 703–710.

[21] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. 2015. Obtaining Well Calibrated Probabilities Using Bayesian Binning. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. 2901–2907.

[22] Liang Pang, Jun Xu, Qingyao Ai, Yanyan Lan, Xueqi Cheng, and Jirong Wen. 2020. Setrank: Learning a permutation-invariant ranking model for information retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 499–508.

[23] Rama Kumar Pasumarthi, Sebastian Bruch, Xuanhui Wang, Cheng Li, Michael Bendersky, Marc Najork, Jan Pfeifer, Nadav Golbandi, Rohan Anil, and Stephan Wolf. 2019. TF-Ranking: Scalable tensorflow library for learning-to-rank. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2970–2978.

[24] Gustavo Penha and Claudia Hauff. 2021. On the calibration and uncertainty of neural learning to rank models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

[25] John Platt. 2000. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In *Advances in Large Margin Classifiers*, Alexander J. Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans (Eds.). MIT Press, 61–74.

[26] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 datasets. *arXiv preprint arXiv:1306.2597* (2013).

[27] Zhen Qin, Zhongliang Li, Michael Bendersky, and Donald Metzler. 2020. Matching cross network for learning to rank in personal search. In *Proceedings of The Web Conference 2020*. 2835–2841.

[28] Zhen Qin, Le Yan, Honglei Zhuang, Yi Tay, Rama Kumar Pasumarthi, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2021. Are Neural Rankers still Outperformed by Gradient Boosted Decision Trees?. In *Proceedings of the 9th International Conference on Learning Representations*.

[29] David Sculley. 2010. Combined regression and ranking. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 979–988.

[30] Yukihiro Tagami, Shingo Ono, Koji Yamamoto, Koji Tsukamoto, and Akira Tajima. 2013. CTR Prediction for Contextual Advertising: Learning-to-Rank Approach. In *Proceedings of the 7th International Workshop on Data Mining for Online Advertising*. Article 4, 8 pages.

[31] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 115–124.

[32] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The LambdaLoss Framework for Ranking Metric Optimization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1313–1322.

[33] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning*. 1192–1199.

[34] Le Yan, Zhen Qin, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2022. Scale Calibration of Deep Ranking Models. In *Proceedings of the 28th ACM SIGKDD international conference on Knowledge discovery and data mining*.

[35] Le Yan, Zhen Qin, Honglei Zhuang, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2022. Revisiting Two-Tower Models for Unbiased Learning to Rank. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2410–2414.

[36] Bianca Zadrozny and Charles Elkan. 2001. Learning and Making Decisions When Costs and Probabilities Are Both Unknown. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 204–213.

[37] Bianca Zadrozny and Charles Elkan. 2002. Transforming Classifier Scores into Accurate Multiclass Probability Estimates. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 694–699.

[38] Xiaofeng Zhu and Diego Klabjan. 2020. Listwise learning to rank by exploring unique ratings. In *Proceedings of the 13th international conference on web search and data mining*. 798–806.