Scale Calibration of Deep Ranking Models

Le Yan
Google
lyyanle@google.com

Zhen Qin
Google
zhenqin@google.com

Xuanhui Wang
Google
xuanhui@google.com

Michael Bendersky
Google
bemike@google.com

Marc Najork
Google
najork@google.com

ABSTRACT

Learning-to-Rank (LTR) systems are ubiquitous in web applications nowadays. The existing literature mainly focuses on improving ranking performance by trying to generate the optimal order of candidate items. However, virtually all advanced ranking functions are *not scale calibrated*. For example, rankers have the freedom to add a constant to all item scores without changing their relative order. This property has resulted in several limitations in deploying advanced ranking methods in practice. On the one hand, it limits the use of effective ranking functions in important applications. For example, in ads ranking, predicted Click-Through Rate (pCTR) is used for ranking and is required to be calibrated for the downstream ads auction. This is a major reason that existing ads ranking methods use scale calibrated pointwise loss functions that may sacrifice ranking performance. On the other hand, popular ranking losses are *translation-invariant*. We rigorously show that, both theoretically and empirically, this property leads to training instability that may cause severe practical issues.

In this paper, we study how to perform scale calibration of deep ranking models to address the above concerns. We design three different formulations to calibrate ranking models through calibrated ranking losses. Unlike existing post-processing methods, our calibration is performed during training, which can resolve the training instability issue without any additional processing. We conduct experiments on the standard LTR benchmark datasets and one of the largest sponsored search ads dataset from Google. Our results show that our proposed calibrated ranking losses can achieve nearly optimal results in terms of both ranking quality and score scale calibration.

CCS CONCEPTS

• **Information systems** → **Information retrieval**.

KEYWORDS

Ranking scale calibration; Learning-to-rank; Sponsored search

ACM Reference Format:
Le Yan, Zhen Qin, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2022. Scale Calibration of Deep Ranking Models. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3534678.3539072

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539072

1 INTRODUCTION

Learning-to-Rank (LTR), the long-established research area at the intersection of machine learning and information retrieval, aims to output a ranked list for a set of candidate documents (throughout the paper we use documents to represent web pages, ads, products, etc.) and is widely deployed in many user facing systems [19].

The LTR literature mainly focuses on improving ranking metrics, such as *Normalized Discounted Cumulative Gain* (NDCG). The dominant setting for LTR is the so-called *score-and-sort*, where a scoring function is learned to score each document and a ranked list is formed by sorting documents according to the scores. Given the fact that the existing LTR literature mainly focuses on the ranking metrics, little attention has been paid to the absolute values of scores, since ranking itself is indifferent to various score transformations (e.g., adding a constant). In fact, as we show in this paper, popular pairwise and listwise ranking losses do not produce scale calibrated ranking scores. This effect, however, can limit the use of advanced ranking models in important practical applications.

For example, in sponsored search, the final presentation to users is a ranked list of ads. The vast majority of work in this domain [10, 11, 20, 33, 35] uses pointwise regression losses that produce calibrated scores. This is because calibrated pCTR is a required input for computing the cost-per-click (CPC) after ad impressions. Though advanced ranking losses may produce significantly better ranking performance, the strict business constraint prohibits their usage in this application, which in turn leads to suboptimal user experience and ads revenue.

Besides prohibiting advanced ranking functions to be used in various important applications, the lack of constraints in score scale distribution can lead to *training instability* issues, as the scores may keep drifting during model training. This will cause numerical failures in real-world large scale learning systems, especially under the continual learning paradigm. The problem is becoming more relevant nowadays as continual training is becoming a norm in many industrial applications [16], in order to keep up with the temporal dynamics of the contents and user behaviors.

How to calibrate the score scales of a ranking model has not been widely studied in the LTR literature, especially in the deep learning context. Existing works on calibration are mainly on classification problems, including Platt scaling [27], isotonic regression [21, 43], histogram binning and Bayesian binning [24, 42], and focal loss [22]. Calibration in classification is quite different from the problem in LTR. In classification, there is a fixed set of candidate classes, and

the output scores should sum up to be probability one, where the methods are usually probabilistic and class-centric, such as addressing the over-confident predictions of certain classes. However, there are no such constraints in LTR, as the number of candidate items is not fixed, but can be arbitrary in general. Furthermore, the ranking scores may go beyond probabilities - they can represent relevance, revenue, or watch time [44]. Few works [7, 36] study the problem of calibrating ranking models. They are all post-processing methods, where a ranker is first trained, followed by a calibration stage. Besides complicating training and serving logic, post-processing methods cannot address training instability. Furthermore, existing work did not study ranking calibration for deep rankers that are ubiquitous nowadays.

We want to highlight that our scale calibration technique aims to align the scores produced by the ranking model with some external scale, for example, the predicted click-through rate of an ads retrieval model. We'll use "calibration" and "scale calibration" interchangeably in the rest of the paper. Readers should not confuse it with a different field called uncertainty calibration [8, 15], where "calibration" is performed to produce confidence intervals.

In this paper, we focus on designing scale calibrated ranking losses to calibrate deep ranking models. In particular, we formalize the *translation-invariant* property and show that most effective ranking losses, including pairwise and listwise ones, are translation-invariant. It is the key reason that leads to uncalibrated ranking scores and training instability. We propose three different methods to calibrate ranking models through designing calibrated ranking losses that are not translation-invariant, including a multi-objective formulation, a multi-task formulation, and a novel reference-based method. All our methods directly train models that output calibrated ranking scores without post-processing, and are thus applicable to various scenarios such as continual training [25].

Our experiments are conducted on standard public LTR datasets and a sponsored search dataset from a one of the world's largest ads ranking systems at Google. Our proposed calibrated ranking losses achieve not only strong ranking quality, but also nearly optimal calibration quality, while no existing methods achieve both. More specifically, we are able to outperform the strong production baseline and fully deploy our methods in an ads ranking system. Our work removes the limitations of using advanced ranking losses in real applications such as sponsored search, as well as addressing the practical issue of training instability.

In summary, our contributions are as follows:

- We formalize the important problem of ranking calibration, showcasing its value in important real-world applications, which is largely ignored in the LTR literature.
- We rigorously study the cause of uncalibration for advanced ranking methods, and propose three methods that can achieve both effective ranking and calibration.
- We perform comprehensive evaluation on both public LTR benchmarks and one of the world's largest ads ranking systems, showing the practical value of our work.

The rest of the paper is organized as follows. In Section 2, we review the related work. We define the problems in Section 3 and present our proposed methods in Section 4. Our experiments are described in Section 5. We conclude this paper in Section 6. All of our proofs can be found in Proofs at the end of this paper.

2 RELATED WORK

Learning-to-Rank (LTR) has been extensively studied in the past [19], where a scoring function is trained to minimize a ranking loss and deployed to score and sort user-facing candidate documents. The focus of LTR research has been to improve ranking metrics, with numerous advances in designing more effective loss function, from pointwise to pairwise to listwise [5, 12], and better model architectures, including support vector machines [13], gradient boosted decision trees [4, 14, 39], and neural networks [2, 3, 17, 26, 40]. Neural ranking models are already commonly deployed in many industrial applications to handle large-scale datasets and different data types [18, 32, 41], and recently achieved state-of-the-art performance on traditional LTR datasets with only numerical features [30, 31]. However, almost none of them have studied the calibration issue we discussed above, limiting their applicability in wider applications.

Calibrating model output scores has been studied more extensively for classification problems. The goal is to have a meaningful probability distribution over classes for an instance so as to inform follow-up actions [21]. Post-processing methods are commonly used, where calibration is performed after the classification model is available. Platt et al. [27] proposed a parametric approach to train a logistic model on the outputs of an SVM model, since outputs of SVM do not have probabilitic interpretations. Isotonic regression [21, 43] and binning method such as histogram binning and Bayesian binning [24, 42] are examples of non-parametric approaches. A recent work by Mukhoti et al. [22] proposed to address the overconfidence issues of deep models.

To the best of our knowledge, there are limited work that studies the ranking calibration problem and existing works mainly use the post-processing methods. Tagami et al. [36] used the pairwise squared hinge loss to train an LTR model for ads ranking. It then used Platt-scaling [27] to convert the ranking scores into probabilities. Recently, Chaudhuri et al. [7] compared different post-processing methods to calibrate the outputs of an ordinal regression model, including Platt scaling and isotonic regression.

Our work has several major differences from the existing literature. First, we focus on ranking instead of classification, whose differences are discussed in the previous section. Second, our methods are integrated into ranking function training and do not need an extra post-processing step. Besides being operationally beneficial for real-world applications (e.g., simplifying the infrastructure), we show that not handling calibration during model training can cause training instability issues that post-processing methods fail to fix. This phenomenon has been largely ignored in the literature. Third, the few ranking calibration papers only studied shallow models. We focus on deep ranking models that are ubiquitous nowadays, and all our methods are end-to-end differentiable.

3 THE PROBLEM

In the *score-and-sort* setting, we define a scoring function, such as a deep neural network (DNN), for a query q with n documents as $s(\mathbf{x}_q; \Theta) : \chi^n \rightarrow \mathbb{R}^n$, which maps a list of n documents $\mathbf{x}_q$

defined in feature space χ^n to n real number scores given the model parameters Θ. The parameters Θ fully describe the scoring function and can be trained to optimize an empirical loss on the training dataset grouped by queries, $\Psi = \{(\mathbf{x}_q, \mathbf{y}_q)|q \in Q\}$, where Q denotes the set of queries, drawn from the distribution on (X, Y):

$$\mathcal{L}(\Theta) = \frac{1}{|Q|} \sum_{q \in Q} \ell^{\text{rank}}(\mathbf{y}_q, s(\mathbf{x}_q; \Theta)), \tag{1}$$

where ℓ^{rank} is the loss for a single query and $\mathcal{L}$ represents the total empirical loss. In the following, we use $s_i = s(\mathbf{x}_q; \Theta)_i$ as the score for document i, $\mathbf{s}_q$ and $\{s_i\}$ as the shortcut and expanded version for $s(\mathbf{x}_q; \Theta)$, $i \in D_q$ as the iterator over all documents of query q, and y_i to denote the label for document i. We use $P = |\Theta|$ and $N = |\Psi|$ to denote the number of parameters and the total number of documents in the training set in the rest of the paper.

3.1 Scale Calibrated Losses

We formally define scale calibrated losses and show that popular pointwise losses are calibrated.

Definition 1. *A ranking scoring function s is scale calibrated if there exists a monotonic function f such that for the marginal conditional distribution on $Y|\mathbf{x}_q$ of a given distribution on (X, Y), the value of $f(s_i(\mathbf{x}_q, \Theta))$ for any candidate i equals to the expectation of Y_i, i.e., $f(s_i(\mathbf{x}_q, \Theta)) = \mathbb{E}_{Y|\mathbf{x}_q}[Y_i]$.*

Definition 2. *A loss function ℓ is a* scale calibrated loss *if the scores that minimize the loss are scale calibrated.*

In the popular pointwise losses, each score is directly anchored to its label. For example, the mean squared loss (MSE), typically used for real-valued labels, is

$$\ell^{\text{MSE}}(\mathbf{y}_q, \mathbf{s}_q) = \frac{1}{2} \sum_{i \in D_q} (y_i - s_i)^2. \tag{2}$$

The score s_i is anchored to $y_i \in \mathbb{R}$. On the other hand, the logistic loss (LogLoss) typically used for binary labels is

$$\ell^{\text{LogLoss}}(\mathbf{y}_q, \mathbf{s}_q) = -\sum_{i \in D_q} [y_i \ln(\sigma(s_i)) + (1 - y_i) \ln(1 - \sigma(s_i))] \tag{3}$$

where $\sigma(s_i)$ is the sigmoid function and $y_i \in \{0, 1\}$ is the binary label. $\sigma(s_i)$ is a monotonic function and anchored to y_i.

It's obvious to show that both MSE and LogLoss are scale calibrated losses. Assume that we draw n_i samples of document i with $y_i^{(k)} \sim Y_i$ as the k-th label sample. It can be seen that MSE loss is minimized when $s_i = \sum_k y_i^{(k)}/n_i$ and LogLoss is minimized when $\sigma(s_i) = \sum_k y_i^{(k)}/n_i$, the empirical expectation of Y_i.

3.2 Translation-Invariant Ranking Losses

We show that more effective (in terms of ranking metrics) ranking losses, including pairwise and listwise ones, are not scale calibrated due to their translation invariance property:

Definition 3. *A ranking loss ℓ is invariant to a transformation $\mathcal{T} : s_i \to s_i'$ where $s_i' \neq s_i$, if*

$$\ell(\mathbf{y}, \mathbf{s}') = \ell(\mathbf{y}, \mathbf{s}). \tag{4}$$

Proposition 4. *A loss ℓ that is invariant to a transformation is not scale calibrated.*

Definition 5. *As a special case, the global translation transforms scores by adding a constant term, i.e., $s_i' = s_i + w$, with $w \neq 0$. A loss function that is invariant to this transformation is called translation-invariant.*

In ranking problems, the ranks of candidate documents, achieved by the sort function, are invariant to any transformation that preserves the order of the scores. Many pairwise and listwise ranking losses that approximate the ranks are thus invariant to some transformations. Especially, the commonly used ranking losses below are *translation-invariant*.

- The pairwise RankNet loss [4]:

$$\ell^{\text{RankNet}}(\mathbf{y}_q, \mathbf{s}_q) = -\sum_{i \neq j \in D_q} \mathbb{I}_{y_i > y_j} \ln \frac{\exp(s_i - s_j)}{1 + \exp(s_i - s_j)}, \tag{5}$$

- The listwise softmax loss [1, 26]:

$$\begin{aligned} \ell^{\text{Softmax}}(\mathbf{y}_q, \mathbf{s}_q) &= -\textstyle\sum_{i \in D_q} y_i \ln \frac{\exp(s_i)}{\sum_{j \in D_q} \exp(s_j)} \\ &= -\textstyle\sum_{i \in D_q} y_i \ln \frac{1}{\sum_{j \in D_q} \exp(s_j - s_i)}, \end{aligned} \tag{6}$$

- The listwise ApproxNDCG loss [29]:

$$\begin{aligned} \ell^{\text{NDCG}}(\mathbf{y}_q, \mathbf{s}_q) &= -\frac{1}{\text{DCG}_q^{\text{ideal}}} \sum_{i \in D_q} \frac{2^{y_i} - 1}{\log_2(1 + r_i(\mathbf{s}_q))}, \\ r_i(\mathbf{s}_q, \beta) &= \sum_{j \in D_q} \frac{\exp[\beta(s_j - s_i)]}{1 + \exp[\beta(s_j - s_i)]} + \frac{1}{2}, \end{aligned} \tag{7}$$

where β is the smoothing parameter. In these popular ranking losses, scores always appear in the *paired* form of $s_i - s_j$ or $s_j - s_i$. So when a constant w is added to each score $s_i' = s_i + w$, the losses stay invariant:

$$\ell(\{s_i'\}) = \ell^{\text{paired}}(\{s_i' - s_j'\}) = \ell^{\text{paired}}(\{s_i - s_j\}) = \ell(\{s_i\}). \tag{8}$$

Many other ranking losses, such as BoltzRank [37], also have the same translation-invariant property.

3.3 Training Instability

As a result, scorers trained with these ranking losses are poorly calibrated. As shown in Fig. 1, such properties can cause training instability issues, where the average score goes to either positive or negative infinity as the training proceeds, which we formalize as:

Proposition 6. *A ranking model trained with a loss ℓ that protects translation invariance diverges indefinitely in continuous training. More specifically, as training steps goes to infinity, $t \to \infty$, the model predictions diverge with $|s| \to \infty$.*

As a result of the proposition, severe numerical failures may arise as training continues when the absolute values of scores become numerically much larger than the typical difference between scores. This potentially causes the ranking model to collapse, especially for large ranking models that need long time to converge or trained in the continual learning framework [25], both of which are common in practice nowadays. Apparently, such an issue cannot be fixed by the post-processing methods such as Platt-scaling [27].

Please note that score divergence is a sufficient but not a necessary condition for score un-calibration. We highlight the issue in this paper as it allows theoretical analysis and is easily reproducible in our production systems. In fact, L2 regularization on

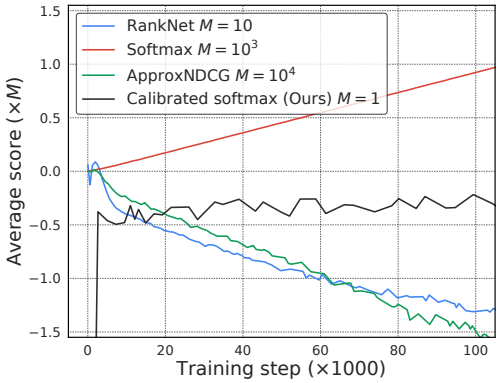

Figure 1: The trend of the average scores of DNN models trained with the RankNet loss Eq. (5) in Blue, softmax loss Eq. (6) in Red, ApproxNDCG loss Eq. (7) in Green, and our calibrated softmax loss Eq. (11) in Black on the Istella dataset. M is the magnitude scale of the y-axis.

model weights as an intuitive way to reduce score value divergence. We tested this, but found the following deficiencies: (1) Adding L2 regularization alone will not achieve score calibration (without post-processing); (2) Blindly constraining score scale by L2 regularization can hurt ranking performance. In the following, we will present our methods to address this issue, which provide both robust ranking and calibration performance.

4 CALIBRATED RANKING LOSSES

In this section, we propose three different methods to calibrate the ranking models by *calibrated ranking losses*. Our goal is to preserve the high ranking quality of ranking losses while achieving scale calibration. The main idea is to explore the one degree of freedom of the scores resulted from their translation-invariance and add proper constraints to make them calibrated.

4.1 The Multi-Objective Method

Our first method is to combine a translation-invariant ranking loss and a calibrated regression loss:

$$\mathcal{L}^{\text{MultiObj}}(\Theta) = \alpha \mathcal{L}^{\text{rank}}(\mathbf{y}^{\text{rank}}, \mathbf{s}) + (1-\alpha)\mathcal{L}^{\text{regr}}(\mathbf{y}^{\text{regr}}, \mathbf{s}), \quad (9)$$

where the ranking loss weight $\alpha \in (0, 1)$. Here we use $\bullet^{\text{regr}}$ to refer to losses and labels in calibrated regression losses, such as Eqs. (2) and (3), and $\bullet^{\text{rank}}$ for uncalibrated ranking losses, such as Eqs. (5), (6) and (7). The multi-objective method results in a direct tradeoff between the ranking loss and the regression loss.

4.2 The Multi-Task Method

Another practical method for deep models is to use the multi-task learning [6, 34]. In this method, we have two output layers after the DNN hidden layers: one as the main output that serves the model predictions and the other as an auxiliary task that outputs ranking scores and is only used during model training. In this way, the main output is always calibrated, while at the same time, the ranking gradients from the ranking loss back-propagate to the shared layers so that the model can push the shared layers towards optimizing for ranking quality.

Formally, a multi-task scorer can be defined as

$$s(\mathbf{x}_q; \Theta, \theta^{\text{regr}}, \theta^{\text{rank}}) : \mathcal{X}^n \to (\mathbb{R} \times \mathbb{R})^n$$

where Θ are model parameters shared by the two outputs, θ^{regr} are parameters of the main output layer and θ^{rank} are of the ranking output layer. The total loss on the multi-task head can be written as $\mathcal{L}^{\text{MultiTask}}(\Theta, \theta^{\text{regr}}, \theta^{\text{rank}}) =$

$$\alpha \mathcal{L}^{\text{rank}}(\mathbf{y}^{\text{rank}}, \mathbf{s}^{\text{rank}}) + (1-\alpha)\mathcal{L}^{\text{regr}}(\mathbf{y}^{\text{regr}}, \mathbf{s}^{\text{regr}}), \quad (10)$$

where again we have the ranking loss weight $\alpha \in (0, 1)$.

4.3 The Reference-based Method

Finally, we introduce a novel reference-based method to derive calibrated ranking losses. Similar to Platt scaling [27], we want to fit the global translation degree of freedom of ranking losses. The difference is that we tune the global shift to optimize the scale calibration in the model training process, but not a post-processing step. We can achieve this goal with the following trick: in addition to the n candidate documents with predicted ranking scores s_i and labels y_i, we add a *virtual candidate* with a fixed ranking score 0 and label y_0. y_0 is a global tunable parameter and serves as a reference for calibration.

Using the softmax loss as an example, by adding the *virtual candidate* as the reference, we get the *calibrated softmax loss* as follows:

$$\begin{aligned} \ell^{\text{CalSoftmax}} &= \ell^{\text{Softmax}}(\{\mathbf{y}_q, y_0\}, \{\mathbf{s}_q, 0\}) \\ &= -\sum_{i \in D_q} y_i s_i + \left(y_0 + \sum_{i \in D_q} y_i\right) \ln\left(1 + \sum_{j \in D_q} \exp(s_j)\right). \end{aligned} \quad (11)$$

Our reference-based method can be extended to other ranking losses easily. Consider the pairwise RankNet loss [4] - if we introduce an anchor candidate with label y_0 to a query, the calibrated RankNet loss is then equivalent to the multi-objective method with an additional LogLoss on each real candidate, and those with label $y_i > y_0$ are treated as positive samples while those with $y_i < y_0$ are taken as negative. The loss boils down to the standard LogLoss when we choose $y_0 \to 0_+$, as

$$\begin{aligned} \ell^{\text{CalRankNet}} = \\ -\sum_{i \neq j \in q} \mathbb{I}_{y_i > y_j} \ln \frac{\exp(s_i - s_j)}{1+\exp(s_i - s_j)} - \sum_{i \in q} \mathbb{I}_{y_i > y_0} \ln \frac{\exp(s_i)}{1+\exp(s_i)} \\ - \sum_{i \in q} \mathbb{I}_{y_i < y_0} \ln \frac{\exp(-s_i)}{1+\exp(-s_i)} = \ell^{\text{RankNet}} + \ell^{\text{LogLoss}}\big|_{y_0} \end{aligned} \quad (12)$$

In this paper, we focus on the listwise softmax loss due to its effectiveness [31] and use the calibrated softmax loss to represent the reference-based method. For the calibrated softmax loss, we have the following theoretical property.

Proposition 7. *Given the scores that minimize the calibrated softmax loss, any global translation of them leads to increased loss. Furthermore, the calibrated softmax loss is a scale calibrated loss.*

5 EXPERIMENTS

We validate our proposed methods on several datasets under various settings, including ranking with logistic prediction on binary

labeled datasets and regression prediction on real-valued datasets. We want to answer the following research questions:

- RQ1: Are the proposed methods able to produce good ranking performance as well as robust scale calibration?
- RQ2: Do the proposed methods generalize to different ranking labels and calibration tasks, including binary and real-valued labels?
- RQ3: Are the proposed methods able to prevent training instability that post-processing methods fail to fix?
- RQ4: How do our proposed methods perform on real-world production systems?
- RQ5: What are the behaviors of the three proposed methods, and which one should practitioners choose?

5.1 Datasets

We use two popular public learning-to-rank datasets, Web30K and Istella, and a real-world ads ranking dataset from Google sponsored search system. For Web30K and Istella, besides studying regression calibration on real-valued labels, we also study logistic calibration by binarizing the labels (positive values as 1 and the rest as 0):

Web30K [28] is a public learning-to-rank 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, 4 are about 51.4%, 32.5%, 13.4%, 1.9%, and 0.8%.

Istella full dataset [9] is a public learning-to-rank dataset 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 judgments also vary from 0 to 4 but with a skewed distribution: 96.3% for 0s, 0.8% for 1s, 1.3% for 2s, 0.9% for 3s, and 0.7% for 4s.

Sponsored Search is a proprietary ads ranking dataset at Google. The labels used in our experiments are binary user clicks. The system employs a continuous training framework, with a huge number of queries per second. Our experiments are conducted on 5 months of data in this paper.

5.2 Experiment Setup

Different applications may rank candidate documents based on different utilities (such as CTR, relevance, or watch time), and need the scores to be calibrated in different manners. In this paper, we study the ranking and score calibration in the following two ways when we have the flexibility to do so (i.e., on Web30K and Istella datasets): (1) In the *regression ranking and calibration* task, the goal is to achieve good ranking performance while correctly predicting the *numeric* relevance label; (2) In the *logistic ranking and calibration* task, the goal is to achieve good ranking performance while correctly predicting the logits to the *binarized* relevance labels (labels 1, 2, 3, 4 as 1 and label 0 as 0). For the sponsored search dataset, we only focus on the logistic ranking and calibration task given user clicks.

Metrics. For ranking performance on public datasets, we adopt the popular NDCG@10 as the evaluation metric. Other ranking metrics such as NDCG@5 are consistent. Higher NDCG values indicate better ranking. Besides ranking metrics, we also care about calibration performance. For regression ranking and calibration, we use the *mean squared error* (MSE) as the evaluation metric, measuring the discrepancy between predicted scores and real-valued labels. For logistic ranking and calibration, we use *logistic loss* (LogLoss) as the metric to measure the discrepancy between predicted click probabilities and binary labels. Lower MSE and LogLoss values indicate better calibration performance. In addition, we consider the Empirical Calibration Error (ECE) [23] as a universal metric of calibration for both regression and logistic tasks. This metric is commonly used in uncertainty calibration. For ranking scale calibration, we divide ranking documents in each query into M bins after we sort them by the model predictions, and compute the ECE by,

$$ECE = \frac{1}{|Q|} \sum_{q \in Q} \sum_{m=1}^{M} \frac{|B_m|}{|D_q|} \left| \frac{1}{|B_m|} \sum_{i \in B_m} y_i - \frac{1}{|B_m|} \sum_{i \in B_m} f(s_i) \right|. \quad (13)$$

In this work, we use $M = 10$ bins with each bin containing approximately the same number of documents with successive predictions. Finally, by taking the last N (e.g., 100) evaluation points on the score-step plot such as in Fig. 1, and making a linear fit, we define *stability* by comparing the delta between two end points of the linear fit with the mean residual of the fit: if the former is greater, we mark the training as not stable. This metric indicates the convergence of the training course, see Fig. 1 for examples of stable vs unstable training curves.

For the sponsored search dataset, we report metrics that are used by the production team. The pCTR is computed as the sigmoid of the output logits and used to compute calibration metrics. Area Under the Curve (AUC) is computed per query and then averaged over all queries to measure ranking quality.

Comparing Methods. The focus of this paper is on the loss function, thus all compared methods on each dataset share the same model architecture. For Web30K and Istella datasets, the neural ranking model contains three layers with 1024, 512, 256 hidden units respectively. In addition, we apply the log1p input transformations, batch normalization, and dropout [31]. Hyperparameters including learning rate, batch normalization momentum, dropout rate, α, and y_0 are tuned for each method when applicable to the validation set.

On the public LTR datasets, for the logistic ranking and calibration task, our baseline includes the pointwise logistic loss ("Pointwise Logistic"), which is the norm in the field. For the regression ranking and calibration task, our baseline includes the pointwise mean square loss ("Pointwise Regression"). For both tasks, we also compare with ranking models that use the listwise softmax loss ("Softmax"), and softmax loss with the post-processing method Platt scaling ("Softmax-Platt"). To the best of our knowledge, Platt scaling is still the most popular and effective calibration method [7, 36] in terms of performance. However, being a post-processing method, it is highly non-trivial to migrate it to large-scale stream learning systems. Our methods can address such constraints without sacrificing ranking performance. We mainly study the listwise softmax loss due to its ranking effectiveness [31].

Table 1: Comparisons on the logistic ranking and calibration task. Bold numbers are the best in each column. Up arrow "↑" and down arrow "↓" indicate statistical significance with p-value=0.01 of better and worse performance than the baseline "Softmax-Platt", respectively. Check mark "✓" indicates stability, and cross mark "✗" for instability.

	Web30K				Istella			
Method	NDCG@10	LogLoss	ECE	Stability	NDCG@10	LogLoss	ECE	Stability
Pointwise Logistic	$0.4566^{\downarrow}$	$\mathbf{0.5864}^{\uparrow}$	$\mathbf{0.1136}^{\uparrow}$	✓	$0.6379^{\downarrow}$	$\mathbf{0.0600}^{\uparrow}$	$\mathbf{0.0175}^{\uparrow}$	✓
Softmax	0.5002	$2.6648^{\downarrow}$	$0.5432^{\downarrow}$	✗	0.6978	$2.5206^{\downarrow}$	$0.5962^{\downarrow}$	✗
Softmax-Platt	0.5002	0.6175	0.1491	✗	0.6978	0.0651	0.0241	✗
MultiObj	0.4971	$0.6013^{\uparrow}$	$0.1264^{\uparrow}$	✓	$0.6857^{\downarrow}$	$0.0618^{\uparrow}$	$0.0207^{\uparrow}$	✓
MultiTask	$0.4952^{\downarrow}$	$0.6019^{\uparrow}$	$0.1272^{\uparrow}$	✓	$0.6615^{\downarrow}$	$0.0638^{\uparrow}$	$0.0228^{\uparrow}$	✓
Calibrated Softmax	**0.5014**	$0.6265^{\downarrow}$	$0.1601^{\downarrow}$	✓	**0.6980**	$0.0645^{\uparrow}$	$0.0229^{\uparrow}$	✓

Table 2: Comparisons on the regression ranking and calibration task. Bold numbers are the best in each column. Up arrow "↑" and down arrow "↓" indicate statistical significance with p-value=0.01 of better and worse performance than the baseline "Softmax-Platt", respectively. Check mark "✓" indicates stability, and cross mark "✗" for instability.

	Web30K				Istella			
Method	NDCG@10	MSE	ECE	Stability	NDCG@10	MSE	ECE	Stability
Pointwise Regression	$0.4956^{\downarrow}$	$0.5456^{\uparrow}$	$0.1883^{\uparrow}$	✓	$0.6786^{\downarrow}$	$\mathbf{0.1243}^{\uparrow}$	$0.0541^{\uparrow}$	✓
Softmax	0.5002	$6.73 \times 10^{4\downarrow}$	$123.3^{\downarrow}$	✗	0.6978	$1.68 \times 10^{7\downarrow}$	$955.4^{\downarrow}$	✗
Softmax-Platt	0.5002	0.5534	0.2084	✗	0.6978	0.1368	0.0621	✗
MultiObj	0.4977	$0.5706^{\downarrow}$	0.1996	✓	$0.6838^{\downarrow}$	$0.2782^{\downarrow}$	$0.1971^{\downarrow}$	✓
MultiTask	0.5005	$\mathbf{0.5411}^{\uparrow}$	$\mathbf{0.1838}^{\uparrow}$	✓	$0.6914^{\downarrow}$	$0.1490^{\downarrow}$	$0.1033^{\downarrow}$	✓
Calibrated Softmax	**0.5010**	$0.5587^{\downarrow}$	$0.2253^{\downarrow}$	✓	**0.6990**	$0.1351^{\uparrow}$	$0.0607^{\uparrow}$	✓

We evaluate all our three proposed methods: (1) the multi-objective method in Eq. (9) ("MultiObj"), where we combine softmax loss with the corresponding pointwise loss. (2) The multi-task method in Eq. (10) ("MultiTask"), where we use an auxiliary softmax head during training to compliment the pointwise loss. (3) The calibrated softmax loss method in Eq. (11) ("Calibrated Softmax").

For the sponsored search dataset for ads ranking, our baseline is the production deep model at the time of our development. The baseline is quite sophisticated in terms of model architecture (e.g, leveraging techniques such as [38]). However, it uses the pointwise logistic loss. The system employs continual training to handle the huge data volume and adapt to the fast-changing environment. For our experiments, we only change the loss function and keep everything else fixed. We implemented the original uncalibrated softmax loss as another baseline. We were able to test the multi-objective method as the treatment and did not test the other methods due to resource constraints.

5.3 Results on the Public LTR Datasets

The main results are shown in Table 1 and 2, for the logistic and regression tasks, respectively. We emphasize that *the goal of this paper is not to get better ranking metrics, but to achieve both strong ranking and calibration metrics*. We can make the following observations:

- The pointwise baselines (Pointwise Logistic and Pointwise Regression) can get very good calibration performance, but their ranking performance is much inferior to other methods that involve listwise ranking losses. Such observations coincide with the vast LTR research literature.
- Our three proposed methods are stable and are able to generate both strong ranking performance and calibration performance, which *answers positively to RQ1*. We highlight that the calibrated softmax method tend to achieve the best ranking performance and as good calibration performance as the pointwise baselines among all methods. Furthermore, the behaviors are pretty consistent across different tasks and datasets, which *answers positively to RQ2*.
- The Softmax baseline can produce strong ranking performance, but is completely unstable and uncalibrated. This verifies our analysis of the uncalibration issue of advanced ranking losses and explains why they are not applicable to some applications that require calibrated scores.
- Softmax-Platt can generate good performance on both fronts. However, it does not resolve the instability issue of the softmax loss. As a result, this baseline may fail in real-world large-scale learning systems with large amount of data and trained continuously, as we will show in Sec. 5.4.
- Importantly, for each configuration, at least one of our models is more competitive than the Softmax-Platt baseline. For example, for Istella dataset, Calibrated Softmax has neutral ranking performance and better calibration performance than Softmax-Platt.

We further analyze the behaviors of the three proposed methods in terms of their trade-offs between the ranking performance (measured by NDCG@10) and the calibration performance (measured by MSE or LogLoss). The results are shown in Figure 2 and Figure 3 for the logistic and regression tasks. Each method has several points

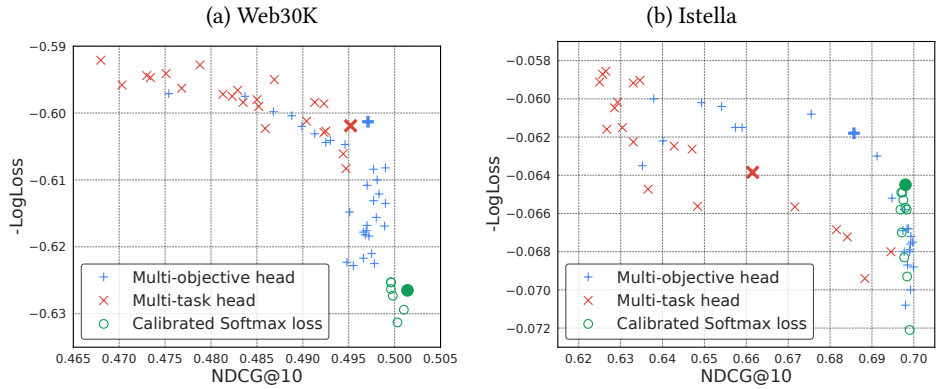

Figure 2: Tradeoffs of methods for logistic ranking and calibration. Negative LogLoss (-LogLoss) is used for illustration.

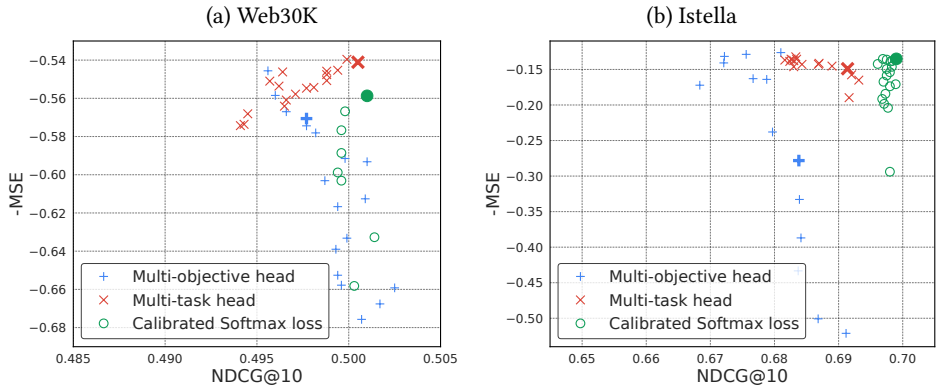

Figure 3: Tradeoffs of methods for regression ranking and calibration. Negative MSE (-MSE) is used for illustration.

in the plot as we vary α or y_0. The settings that were reported in Table 1 and 2 are shown in bold symbols.

We can observe that the three proposed methods have different behaviors: (1) the scattered points of the multi-objective method form a tradeoff frontier and span over a wide range in terms of ranking and calibration performance. (2) The scattered points of the multi-task method stay tightly on the high calibration metrics end. (3) The scattered points of the calibrated softmax loss stay tightly on the high ranking metrics end. These behaviors provide guidance to practitioners and *answers RQ5*: Depending on the business needs, practitioners can bias towards one of them (e.g., prefer calibrated softmax loss if ranking performance is more important), or simply tune the multi-objective method as it provides a more flexible set of solutions to choose from. We provide a more in-depth analysis on the sensitivity of α and y_0 in Supplement Sec. S.1.

5.4 Results on the Sponsored Search Dataset

As mentioned in the experiment setup, we further evaluate the multi-objective method on one of the world's largest ads ranking system at Google. It is a sponsored search application where pCTR prediction is required to calculate costs for the advertisers. The strong production baseline uses the calibrated pointwise logistic loss, and we also compare with the original listwise softmax as an uncalibrated ranking loss. In our multi-objective method, we combine softmax as the ranking loss with pointwise logistic loss. For all these methods, we follow the production setting and train them continuously over the last 5 months of data. We compare the results in Table 3. Note that for proprietary reasons, we only report relative numbers to the production baseline ("Pointwise Logistic") with respect to the average pCTR for scale calibration and AUC for ranking quality. A 0.5% improvement in AUC has very significant effect on core business metrics, and any noticeable difference in pCTR calibration is not acceptable.

Table 3: Relative percentage difference of the average pCTR (for calibration) and AUC (for ranking quality) of different methods relative to the Pointwise Logistic baseline after training on 5 months of data in the Sponsored Search dataset.

Method	pCTR	AUC	Calibration	Stability
Pointwise Logistic	0.0%	0.0%	✓	✓
Softmax	-99.6%	-1.8%	✗	✗
Softmax-Platt	0.0%	-1.8%	✓	✗
MultiObj (Ours)	0.0%	**+0.9%**	✓	✓

From this table, we can see that when the model is trained with softmax loss only, the pCTR values deviated from the Pointwise Logistic baseline noticeably and thus are not applicable. Moreover, we can see a degradation in terms of the ranking quality measured with AUC in the model trained with softmax loss, due to the model instability caused by translation-invariance after training for a long time. Softmax with Platt scaling can not recover from such training failures. Also note that we conducted Platt scaling offline, and it is non-trivial to add an extra post-processing logic into the actual continual learning and serving system. Finally, when our calibrated ranking loss (MultiObj) is applied, we find that both the pCTR is well calibrated and ranking quality is significantly better than the baseline. The proposed method has been successfully deployed into the production system with consistent gains for online metrics. These results show the practical value of our proposed method and *answer positively to RQ3 and RQ4.*

6 CONCLUSION

In this work, we first showed that commonly used ranking losses are translation-invariant. Such a property makes ranking models trained with advanced ranking losses suffer from a global score translation. It leads to uncalibrated model outputs and limits applying advanced ranking models to a wide range of real-world applications where scale calibrated scores are required. Furthermore, this global translation can lead to numeric instability in practice. We then presented three different methods to calibrate deep ranking models directly without requiring post-processing, and show they have different trade-off properties, so practitioners can choose among them depending on business needs. We compare them with baselines on both public LTR datasets and a large-scale online ads ranking application. Our experiments show that our proposed formulations are the first in the literature to be very effective in both ranking and scale calibration, while being stable.

A PROOFS

Proof of Proposition 4. A loss ℓ is invariant to a transformation $\mathcal{T}_w : \mathbb{R} \to \mathbb{R}$, where w is a transformation parameter that specifies one-to-one map $\mathcal{T}_w$. $\{s_i\}$ is the set of ranking scores that will minimize the loss. Assume the loss is calibrated, there exist a one-to-one map f so that $\mathbb{E}[f(s_i)] = \mathbb{E}[y_i]$ for any i. At the same time, $\{\mathcal{T}_w(s_i)\}$ also minimize the loss according to the definition and thus satisfies $\mathbb{E}[f(\mathcal{T}_w(s_i))] = \mathbb{E}[y_i]$. As both f and $\mathcal{T}_w$ are one-to-one maps or bijections, we have $\mathcal{T}_w(\mathbb{E}[y_i]) = \mathbb{E}[y_i]$, or $\mathcal{T}_w$ can only be a bijection onto itself for any w, which contradicts with the condition. So the assumption that the loss is calibrated does not hold.

Proof of Proposition 6. We start with the standard gradient descent. At step t, all queries in Q are used to compute the gradient of each parameter θ_α:

$$\Delta\theta_\alpha^{(t)} = -\lambda \sum_{q\in Q}\sum_{i\in q} \frac{\partial \ell}{\partial s_i^{(t)}} \frac{\partial s_i^{(t)}}{\partial \theta_\alpha} = -\lambda (S^{(t)} \cdot g^{(t)})_\alpha \qquad (14)$$

where λ is the learning rate, $S_{\alpha i} = \frac{\partial s_i}{\partial \theta_\alpha}$ is a $P \times N$ matrix, $g_i = \frac{\partial \ell}{\partial s_i}$ is the loss gradient on the score of the document i, and $g^{(t)}$ is a N-dim vector of gradients, and the dot product is to sum over all documents in Ψ. The corresponding score changes in step t are,

$$\Delta s_i^{(t)} = \sum_\alpha \frac{\partial s_i^{(t)}}{\partial \theta_\alpha} \Delta\theta_\alpha^{(t)} = -\lambda (S^{(t)T} S^{(t)} \cdot g^{(t)})_i,$$

where notation $\bullet^T$ is the transpose of the matrix or vector. The corresponding loss changes by,

$$\Delta\mathcal{L}^{(t)} = \frac{1}{|Q|}\sum_{q\in Q}\sum_{i\in q} \frac{\partial \ell}{\partial s_i} \Delta s_i^{(t)} = \frac{1}{|Q|} g^{(t)T} \cdot \Delta s^{(t)}.$$

In the vicinity of the optimum, $g^{(t)}$ converges to g and $\Delta\mathcal{L}^{(t)} = 0$. We thus have $S^{(t)} \cdot g = 0$ and $\Delta s^{(t)} = 0$. In other words, the model scores are converging to a stationary point in the training.

In practice, we apply stochastic gradient descent with mini-batch of training query set $B^{(t)}$, instead of Q, summed in each step in Eq. (14). Let's say it takes $\tau + 1$ steps to finish an epoch, meaning $Q = \bigcup_{t=t_0}^{t_0+\tau} B^{(t)}$. Then the total change of scores,

$$\Delta s_i^{t_0:t_0+\tau} = \sum_{t=t_0}^{t_0+\tau} \Delta s_i^{(t)} = -\lambda \sum_{t=t_0}^{t_0+\tau} (S^{(t)T} S^{(t)} \cdot g_B^{(t)})_i.$$

In the vicinity of the optimum, we will have

$$\Delta\mathcal{L}^{t_0:t_0+\tau} = \frac{1}{|Q|} g^T \cdot \Delta s^{t_0:t_0+\tau} = 0. \qquad (15)$$

Deep neural network models are in general nonlinear, so that $S^{(t)} \neq S^{(t')}$ if $t \neq t'$. So we can no longer derive the constraint $S^{(t)} \cdot g^{(t)} = 0$ from Eq. (15), nor the condition $\Delta s^{(t)} = 0$. To satisfy constraint Eq. (15), we must have total change Δs perpendicular to g independent of parameter. For the ranking losses discussed above, the global translation $\Delta s_i = w$ invariance offers a non-trivial solution for the constraint. In fact, the global translation speed w is a measure of the curvature near the local optimum of the nonlinear model, which is nontrivial in general. So in most practical situations, one will find that ranking scores do not converge. They may move at a constant rate to positive or negative infinity in the training course, as shown in Fig. 1 for the three ranking losses.

Proof of Proposition 7. We can prove the first part of this proposition. For a list of scores $\mathbf{s}$ that minimizes the calibrated softmax loss, a small global translation w leads to:

$$\ell' = \ell + \frac{1}{2}\left(y_0 + \sum_{i\in q} y_i\right) \frac{\sum_{j\in q}\exp(s_j)}{(1+\sum_{j\in q}\exp(s_j))^2} w^2 + o(w^3).$$

where the notation $o(\cdot)$ denotes that the remaining quantity is negligible when w is small. Therefore, the calibrated softmax loss will be stable to the global translation and the average score converges to a constant.

We can now prove that the calibrated softmax loss is a calibrated loss. It's easy to show that at the optimum of the calibrated softmax loss, the softmax probability for a document i converges to

$$p_i = \frac{\exp(s_i)}{1+\sum_{j\in q}\exp(s_j)} = \frac{y_i}{y_0 + \sum_j y_j}.$$

Normalization of the probability indicates that $y_0 e^{s_i} = y_i$. Therefore, we can define a one-to-one mapping function $f(s) = y_0 e^s$ so that $f(s) = \mathbb{E}[y]$ calibrates the ranking scores s_i with the corresponding labels y_i.

REFERENCES

[1] Sebastian Bruch, Xuanhui Wang, Michael Bendersky, and Marc Najork. An analysis of the softmax cross entropy loss for learning-to-rank with binary relevance. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, page 75–78, 2019.

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

[3] Christopher Burges, Robert Ragno, and Quoc Le. Learning to rank with non-smooth cost functions. In *Advances in Neural Information Processing Systems*. MIT Press, 2007.

[4] Christopher J.C. Burges. From RankNet to LambdaRank to LambdaMART: An overview. Technical Report MSR-TR-2010-82, Microsoft Research, 2010.

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

[6] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[7] Sougata Chaudhuri, Abraham Bagherjeiran, and James Liu. Ranking and calibrating click-attributed purchases in performance display advertising. In *2017 AdKDD & TargetAd*, pages 7:1–7:6, 2017.

[8] Daniel Cohen, Bhaskar Mitra, Oleg Lesota, Navid Rekabsaz, and Carsten Eickhoff. Not all relevance scores are equal: Efficient uncertainty and calibration modeling for deep retrieval models. *arXiv preprint arXiv:2105.04651*, 2021.

[9] Domenico Dato, Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, Raffaele Perego, Nicola Tonellotto, and Rossano Venturini. Fast ranking with additive ensembles of oblivious and non-oblivious regression trees. *ACM Transactions on Information Systems*, 35(2), 2016.

[10] Thore Graepel, Joaquin Quiñonero Candela, Thomas Borchert, and Ralf Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *Proceedings of the 27th International Conference on Machine Learning*, page 13–20, 2010.

[11] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the 8th International Workshop on Data Mining for Online Advertising*, pages 5:1–5:9, 2014.

[12] Rolf Jagerman, Zhen Qin, Xuanhui Wang, Mike Bendersky, and Marc Najork. 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*, 2022.

[13] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142, 2002.

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

[15] Ranganath Krishnan and Omesh Tickoo. Improving model calibration with accuracy versus uncertainty optimization. *arXiv preprint arXiv:2012.07923*, 2020.

[16] Sofia Ira Ktena, Alykhan Tejani, Lucas Theis, Pranay Kumar Myana, Deepak Dilipkumar, Ferenc Huszár, Steven Yoo, and Wenzhe Shi. Addressing delayed feedback for continuous training with neural networks in ctr prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems*, page 187–195, 2019.

[17] Pan Li, Zhen Qin, Xuanhui Wang, and Donald Metzler. 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*, page 2032–2040, 2019.

[18] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. Pretrained transformers for text ranking: Bert and beyond, 2021.

[19] Tie-Yan Liu. *Learning to Rank for Information Retrieval*. Now Publishers Inc, 2009.

[20] H. Brendan McMahan, Gary Holt, D. Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hrafnkelsson, Tom Boulos, and Jeremy Kubica. Ad click prediction: A view from the trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1222–1230, 2013.

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

[22] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. Calibrating deep neural networks using focal loss. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15288–15299, 2020.

[23] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[24] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, page 2901–2907, 2015.

[25] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, May 2019.

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

[27] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In Alexander J. Smola, Peter Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*, page 61–74. MIT Press, 2000.

[28] Tao Qin and Tie-Yan Liu. Introducing letor 4.0 datasets. *arXiv preprint arXiv:1306.2597*, 2013.

[29] Tao Qin, Tie-Yan Liu, and Hang Li. A general approximation framework for direct optimization of information retrieval measures. *Information Retrieval*, 13:375–397, August 2010.

[30] Zhen Qin, Le Yan, Yi Tay, Honglei Zhuang, Xuanhui Wang, Michael Bendersky, and Marc Najork. Born again neural rankers. *arXiv preprint arXiv:2109.15285*, 2021.

[31] Zhen Qin, Le Yan, Honglei Zhuang, Yi Tay, Rama Kumar Pasumarthi, Xuanhui Wang, Michael Bendersky, and Marc Najork. Are neural rankers still outperformed by gradient boosted decision trees? In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[32] Zhen Qin, Honglei Zhuang, Rolf Jagerman, Xinyu Qian, Po Hu, Chary Chen, Xuanhui Wang, Mike Bendersky, and Marc Najork. Bootstrapping recommendations at chrome web store. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2021.

[33] Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *Proceedings of the 16th International Conference on World Wide Web*, page 521–530, 2007.

[34] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

[35] David Stern, Ralf Herbrich, and Thore Graepel. Matchbox: Large scale bayesian recommendations. In *Proceedings of the 18th International World Wide Web Conference*, pages 111–120, 2009.

[36] Yukihiro Tagami, Shingo Ono, Koji Yamamoto, Koji Tsukamoto, and Akira Tajima. Ctr prediction for contextual advertising: Learning-to-rank approach. In *Proceedings of the 7th International Workshop on Data Mining for Online Advertising*, 2013.

[37] Maksims N Volkovs and Richard S Zemel. Boltzrank: learning to maximize expected ranking gain. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1089–1096, 2009.

[38] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*, pages 1–7. 2017.

[39] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. The LambdaLoss framework for ranking metric optimization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1313–1322, 2018.

[40] Le Yan, Zhen Qin, Rama Kumar Pasumarthi, Xuanhui Wang, and Michael Bendersky. Diversification-aware learning to rank using distributed representation. In *Proceedings of the Web Conference 2021*, pages 127–136, 2021.

[41] Le Yan, Zhen Qin, Honglei Zhuang, Xuanhui Wang, Mike Bendersky, and Marc Najork. 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*, 2022.

[42] Bianca Zadrozny and Charles Elkan. 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*, page 204–213, 2001.

[43] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 694–699, 2002.

[44] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. Recommending what video to watch next: A multitask ranking system. In *Proceedings of the 13th ACM Conference on Recommender Systems*, page 43–51, 2019.

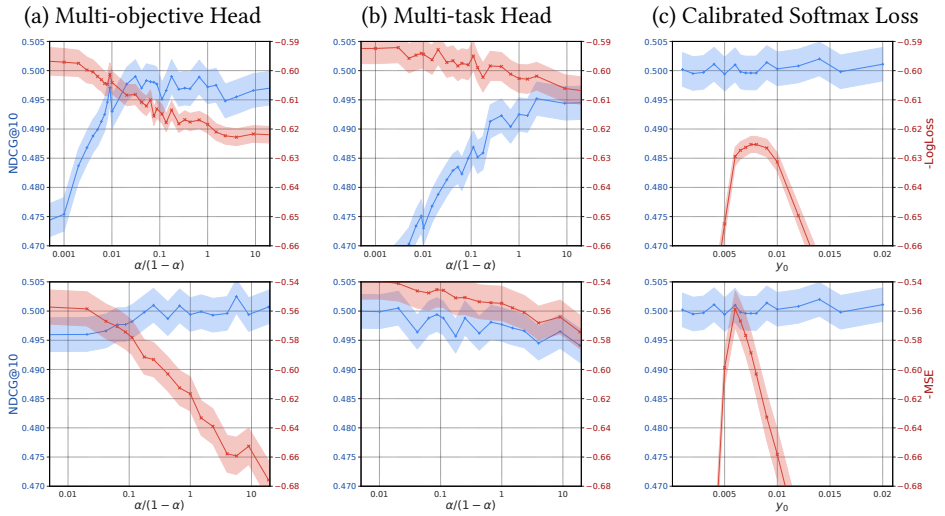

Figure S1: NDCG@10, -LogLoss, and -MSE vs relative ranking weight $\alpha/(1-\alpha)$ for multi-objective and multi-task methods and vs y_0 for calibrated softmax loss on Web30K. All three metrics are the higher, the better.

S SUPPLEMENT

S.1 Impacts of α and y_0

In this supplement section, we give an in-depth analysis on the effects of the parameters α and y_0 to get a deeper understanding of the three proposed methods. The results are illustrated in Figure S1 for the Web30K dataset. The main observations are summarized as follows:

- When we increase the ranking loss weight α in the multi-objective method, the calibration metrics decrease monotonically and ranking metric increases monotonically till an optimal value α^*. It saturates and fluctuates thereafter.
- We observe a similar decrease in the calibration metrics when we increase the ranking loss head weight α in the multi-task method, but the range decreased is much narrower compared to the multi-objective method. As the main head is always directly trained with the regression loss, the calibration performance is much more preserved when more weight is added to the ranking head.
- One interesting observation in the multi-task method is that the calibration performance may not always compete with the ranking performance. This may attribute to the additional freedom in the multi-task method.
- In the models trained with the calibrated softmax loss, we find that the calibration metrics are non-monotonic in terms of the anchor label y_0. Also, in a wide proximity of y_0 that optimizes the calibration metrics, the ranking performance is nearly insensitive to the change of y_0 and is approximately the best ranking performance achieved.