Received November 23, 2021, accepted January 7, 2022, date of publication January 11, 2022, date of current version January 20, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3142096

PeerRank: Robust Learning to Rank With Peer Loss Over Noisy Labels

XIN WU[1], QING LIU[2], JIARUI QIN[1], AND YONG YU[1]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[2]Huawei Noah's Ark Laboratory, Shenzhen 518129, China

Corresponding author: Yong Yu (yyu@apex.sjtu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62177033, and in part by the Huawei Innovation Research Program. This work was completed while Qing Liu was with Huawei Noah's Ark Lab.

ABSTRACT User-generated data are extensively utilized in learning to rank as they are easy to collect and up-to-date. However, the data inevitably contain noisy labels attributed to users' annotation mistakes, lack of domain knowledge, system failure, etc., making building a robust model challenging. On account of the remarkable nature of deep neural networks in fitting datasets, the noisy labels significantly degrade the performance of learning-to-rank algorithms. To cope with this problem, previous studies have put forward several methods for label de-noising. However, they are either susceptible to the noise distribution on datasets, raising the demand for clean data or incurring more computational costs. Moreover, most of them are tough to extend to different scenarios. This paper proposes a simple yet effective framework named PeerRank that can be applied in broad applications such as click-through rate prediction and commercial web search in learning-to-rank tasks. PeerRank is a robust, effective, and adaptable framework that can couple with numerous models with theoretical guarantees. Extensive experiments on three public real-world datasets with thirteen point-wise base models and four semi-synthetic generation datasets with four pair-wise base models show the consistent improvement of PeerRank. The results comparing PeerRank with seven classic and state-of-the-art de-noising methods validate the advantages of PeerRank framework for learning to rank over noisy labels.

INDEX TERMS Learning to rank, noisy labels, robustness.

I. INTRODUCTION

Learning to rank (LTR) approaches heavily rely on the large-scale labeled data to build ranking models. Editorial labeled training data that requires experts to annotate is time-consuming and costly to obtain. Thus, more and more research works use user-generated data, which are easily accessible and up-to-date, to train ranking models [1]. However, user-generated data inevitably contain noise for many reasons [2]. For example, the vague definition of relevance levels or the lack of domain knowledge makes it difficult for the users to give a reliable label to each data point. Xu *et al.* [3] proved that label noise in training data, no matter being randomly generated or existing in real-world data, can significantly degrade the performance of LTR algorithms. Fig. 1 shows an example to elaborate how noisy labels affect the ranking results. Suppose there are four items in candidate pool where x_1, x_2, x_3 are relevant items while x_4 is not. However, x_2 is observed as irrelevant from user interactions, i.e., noisy data point. Thus, the ranking model learned from the noisy data probably ranks the items as $\tilde{\boldsymbol{\pi}}:\langle x_1, x_4, x_3, x_2\rangle$, which brings about a decrease of metric values compared with the ranking result $\boldsymbol{\pi}:\langle x_1, x_2, x_4, x_3\rangle$ from the clean data. If we consider a longer ranking list, a noisy interaction on a single item will lead to a large number of mislabeled pairs, which deteriorates overall ranking performance [4]. Making it worse, the state-of-the-art (SOTA) performance of LTR is achieved by deep models [5]–[7], where deep neural networks are more likely to fit noisy labels than traditional lower-rank ranking algorithms [8]. A robust model, defined as one that can tolerate perturbations in the data, is urgently needed to address the issues.

In the literature, several solutions have been proposed for designing a robust model and label de-noising. Recently, some works train models that are invulnerable to outliers by developing robust loss functions [9]–[13], applying

The associate editor coordinating the review of this manuscript and approving it for publication was Hengyong Yu.

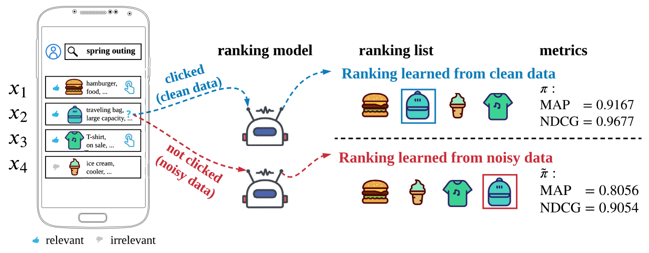


FIGURE 1. An example illustrating how noisy labels affect ranking results. A model learned from noisy data will perform inferior.

regularization techniques [14]–[17] or selecting reliable samples [3], [18]–[22]. However, methods of these types either are significantly affected by the changes in noise distribution [23], improve marginally [24] or carry the risk of eliminating clean data [2], [23].

This paper proposes a novel framework, named PeerRank, that can be easily applied to a broad class of applications with noisy labels in LTR. The PeerRank framework is built on the Peer Loss [25], which deals with noisy labels in binary classification tasks. LTR can be mapped to classification tasks if we consider whether an item is relevant to a user or not, corresponding to point-wise LTR approaches, or whether an item is more relevant than another one, corresponding to pair-wise LTR approaches. PeerRank constructs a peer sample for each training instance. It trains a ranking model based on Peer Loss with both observed data samples and peer samples. The principle is that a model trained based on Peer Loss with observed samples and peer samples is equivalent to a model trained based on empirical risk minimization (ERM) with clean data. As the properties of Peer Loss [25], which can be used as the point-wise LTR loss function, have been discussed in the original paper, we extend its properties to pair-wise approaches. Notice that the pair-wise unified framework is slightly different from that in point-wise cases. Considering the characteristics of the real-world data, we give the range of the hyper-parameter, which is missing from the original paper. More details can be found in Section IV. To the extent of our knowledge, this is the first time Peer Loss has been applied in LTR applications with solid theoretical guarantees.

PeerRank inherits a bunch of merits. Firstly, it can positively adapt to different data distributions and varying degrees of data noise without prior knowledge of the noise rate. Secondly, it can easily fit different LTR algorithms, referring to either point-wise or pair-wise and some list-wise approaches. The framework is relatively simple and general, requiring no dedicated architecture design for each algorithm or dataset. Thirdly, it does not restrict any specific ERM methods, so a wide range of loss functions is applicable. Finally, we theoretically prove that PeerRank has the properties of robustness and effectiveness, which are essentially desired by all de-noising approaches.

We conduct extensive and comprehensive quantified experiments to testify that PeerRank is easy to couple with thirteen SOTA point-wise and four pair-wise approaches and achieves better performance than those LTR approaches without PeerRank. We empirically prove the advantages of PeerRank over seven classic and SOTA de-noising methods. We also find the boundaries for the only hyper-parameter of PeerRank. Besides, we explore the effect of noise rate on the performance of PeerRank. And we do further experiments comparing PeerRank with some SOTA de-biasing methods.

II. RELATED WORK

A. LEARNING-TO-RANK APPROACHES

LTR algorithms include *point-wise*, *pair-wise* and *list-wise* approaches. In point-wise approaches, many SOTA models focus on feature interactions design, such as DeepFM [26], PNN [27], DCN [28], xDeepFM [29], DCN-M [6] and AutoInt [5]. Other models try to capture users' sequential interest patterns, including GRU4Rec [30], Caser [31], SASRec [32], MIMN [33], HPMN [34], DIN [35] and DIEN [36]. In pair-wise approaches, SVMRank [37] is the pioneer to transform ranking tasks to classification tasks. RankNet [38] and LambdaRank [39] act as the foundation of other algorithms, e.g., DirectRanker [7] generalizes the RankNet architecture. List-wise approaches [40]–[42] optimize evaluation measures of the entire list. Though these algorithms achieve remarkable results, model robustness is not taken into account.

B. DE-NOISING APPROACHES

Several previous robust training methods have been proposed for label de-noising [24]. Whereas most of them are applied in computer vision, and few are involved in information retrieval (IR).

Most of the works try to design a robust loss function. GCE [10], TCE [12] and SCE [11] are proposed based on mean absolute error and categorical cross-entropy to exploit the advantages of their robustness, fast convergence, and generalization [9]. Some adjustments to the loss functions improve robustness. For example, BootStrap (BS) [13] uses label refurbishment to update the training labels. Regularization [14]–[17] is utilized to prevent a model from over-fitting noisy labels. e.g., Label Smoothing (LS) [17] estimates the marginalized effect of label noise during training to prevent the neural network from fully calculating the loss of noisy training samples.

Sample selection is another widely used method trying to distinguish and remove noisy data samples to pursue robust learning. For example, Co-teaching (CT) [20] selects samples with low losses and feeds them to another network for further training. The noise rate (denoted as τ in CT [20]) is required for hyper-parameter setting. Reweight [2] contains two dedicated steps to calculate the probability of a label being noisy in the first step and reweight the loss in the second step. Similar ideas can be found in [3], [18], [19], [21]–[23].

However, several drawbacks exist in the methods mentioned above. The loss function design methods are sensitive to the changes in noise distribution [23], which greatly reduces their applicability. Since the definition of a clean sample is vague [2], [23], sample selection methods may eliminate numerous clean and sound samples while excluding noisy and unreliable samples. Other works applying meta-learning [23], [43], [44] or semi-supervised learning [45] either require a certain amount of clean data that may be unavailable in real-world scenarios or bring about an inevitable increase in computational cost [24].

In view of all these limitations, we develop a more effective and more applicable method, named PeerRank, to deal with noisy training data for LTR based on Peer Loss [25], a new family of loss functions that copes with noisy labels in binary classification tasks without prior knowledge of the noise rate. Wide ranges of algorithms can be further improved with our proposed PeerRank framework. In this paper, we examine how PeerRank takes effect on three branches of LTR approaches.[1] We prove theoretically and empirically that PeerRank is more robust and effective for ranking with noisy labels than the existing approaches.

III. PRELIMINARY

A. PROBLEM SETUP

We denote the *clean* dataset as $D = (X, Y)$ where X represents the features of instances, and Y represents the clean labels of the instances. Each sample in D is independently and identically drawn from an implicit data distribution $\mathcal{D}$, i.e., $D \sim \mathcal{D}$. The objective of LTR is to give a permutation of n items where the items that the user is interested in are ranked ahead. Denote the permutation that optimally coincides with the user's interests as π^*. Let $\boldsymbol{\pi}(\cdot)$ be the permutation produced by a ranking model with n input items. The ranking objective is to minimize the ranking risk [47] measuring the gap between $\boldsymbol{\pi}(\cdot)$ and π^*:

$$\mathbb{E}_{\mathcal{D}}[\ell(\boldsymbol{\pi}(X_1, X_2, \ldots, X_n), \pi^*)], \tag{1}$$

where ℓ is a loss function measuring the distance between $\boldsymbol{\pi}(\cdot)$ and π^*. The objective can be instantiated in three different LTR approaches. We mainly focus on the point-wise and pair-wise approaches widely used in commercial search engines to illustrate our proposed PeerRank framework. PeerRank is also applicable to specific list-wise algorithms that optimize the entire list but are trained in a pair-wise mode, such as LambdaRank [39].

In real world, however, only the *observed* data are available which contain both clean and noisy labels, i.e., $\tilde{D} = (X, \tilde{Y}) \sim \tilde{\mathcal{D}}$ where $\tilde{Y}$ can be clean or noisy. Following [25], we define error transition probabilities as

$$\begin{aligned} e^+ &= Pr(\tilde{Y} = -1 | Y = +1), \\ e^- &= Pr(\tilde{Y} = +1 | Y = -1), \end{aligned} \tag{2}$$

[1]Algorithms like LambdaRank [39] and LambdaMART [46] can be viewed as list-wise approaches as they consider the whole rank, but they learn in a pair-wise mode. We advocate that list-wise algorithms with such property can be optimized with our PeerRank framework.

where $e^+(e^-)$ represents the probability of samples that should be positive (negative) are observed as negative (positive).

In the field of IR, for each user, the observed data is $\tilde{D} = \{(\boldsymbol{x}_i, c_i)_{i=1}^n\}$ where a sample $\boldsymbol{x}_i \in X$ contains the features of the user, an item, and context, and $c_i = \{0, 1\}$ indicates whether the item is clicked (1) or not (0).

1) POINT-WISE APPROACH

The point-wise approach learns a scoring function that takes the feature vector of an instance as input

$$f(\boldsymbol{x}_i) \equiv f_\Theta(\boldsymbol{x}_i), \tag{3}$$

where Θ is the model parameters. f predicts the relevance of the current item to the user. All items are then ranked according to the inferred scores from the learned scoring function. Without loss of generality, we adopt the widely used logistic regression to calculate the probability $h(\boldsymbol{x}_i)$ that the target item i is relevant,

$$h(\boldsymbol{x}_i) = \frac{1}{1 + e^{-\sigma f(\boldsymbol{x}_i)}}, \tag{4}$$

where σ is the shape parameter. A higher $h(\boldsymbol{x}_i)$ yields a higher ranking of the item. The objective of the model is to minimize the ranking risk

$$\mathbb{E}_{(X,Y)\sim\mathcal{D}}[\ell(h(X), Y)] \tag{5}$$

point-wisely, where ℓ denotes the loss function. As click c_i serves as the label in point-wise approach, i.e., $\tilde{y}_i = c_i$, noise in clicks directly influences the performance of the algorithm.

2) PAIR-WISE APPROACH

The goal of the pair-wise LTR approach is to minimize the number of misclassified item pairs [46], [48]. Any two candidate items, i_1 and i_2, are paired to form a training instance $(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \tilde{y}_i)$ where the pair label $\tilde{y}_i = c_{i_1} > c_{i_2}$ indicates the pair-wise preference and we use Y_i to denote the pair labels. The classifier $h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2})$ outputs the probability of item i_1 being more relevant than item i_2 to the user as

$$h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}) = \frac{1}{1 + e^{-\sigma(f(\boldsymbol{x}_{i_1}) - f(\boldsymbol{x}_{i_2}))}}, \tag{6}$$

where f is a linear [37] or non-linear [7], [39], [46] scoring function as in (3), excepting some algorithms like Greedy-Order [49]. The ranking model tries to minimize the ranking risk

$$\mathbb{E}_{(X_{i_1}, X_{i_2}, Y_i)\sim\mathcal{D}}[\ell(h(X_{i_1}, X_{i_2}), Y_i)] \tag{7}$$

pair-wisely. The influence of noisy clicks is more severe in the pair-wise setting than in the point-wise setting since one noisy click incurs $O(n)$ noisy pair labels.

B. PEER LOSS FUNCTION

Peer Loss is proposed in [25], which can be served as a robust loss function to deal with noisy training data without the prior knowledge of the noise rate.

Definition 1: Given classifier h and instance $(x_i, \tilde{y}_i)$, we randomly sample two additional instances $(x_j, \tilde{y}_j)$, $(x_k, \tilde{y}_k)$ from $\tilde{D}$ to form a peer sample $(x_j, \tilde{y}_k)$. The peer loss function is defined as

$$\ell_{peer}(h(x_i), \tilde{y}_i) = \ell(h(x_i), \tilde{y}_i) - \ell(h(x_j), \tilde{y}_k). \qquad (8)$$

Peer Loss draws inspiration from peer prediction, which is a method to truthfully elicit information from different sources with no ground truth verification. The noisy labels and classifier outputs are treated as two sources of information, and clean labels are treated as the information to elicit. The Peer Loss is served as a scoring function from peer prediction literature [50], [51] to evaluate the quality of information source, i.e., the model outputs. Intuitively speaking, the second term represents how well the model predicts artificially created noise labels and consequently "punishes" models that predict noise well.

IV. PeerRank METHOD AND ANALYSIS

This section presents our framework, named *PeerRank*, which is robust to unexpected noise in the training data. We firstly explain the general idea of PeerRank. Whereafter, the robustness, effectiveness, and adaptability of PeerRank are proven theoretically.

The structure of our framework is displayed in Fig. 2. A traditional LTR algorithm feeds the feature vector $\boldsymbol{x}_i$ into a model, which might include hidden layers and a prediction layer. Then, the model outputs the ranking score for each item. In traditional LTR, the model is learned by optimizing the loss function $\ell(h(\boldsymbol{x}_i), \tilde{y}_i)$. In PeerRank, we construct a peer instance $(\boldsymbol{x}_{\text{peer}}, \tilde{y}_{\text{peer}})$ for each input in the batch training data. Both the features of batch training data $\boldsymbol{x}_i$ and the features of generated peer instances $\boldsymbol{x}_{\text{peer}}$ will be fed into the network, the predicted values of which are $h(\boldsymbol{x}_i)$ and $h(\boldsymbol{x}_{\text{peer}})$, respectively. The model is learned by optimizing the peer loss function as shown in Fig. 2 where the peer instances are used to compute the second term in the loss function.

The generation of peer instances is illustrated in Fig. 3. For each sample $(\boldsymbol{x}_i, \tilde{y}_i)$ used in a point-wise approach, we randomly and uniformly sample another two instances $(\boldsymbol{x}_j, \tilde{y}_j)$, $(\boldsymbol{x}_k, \tilde{y}_k)$ from the batch data where the feature vector of the first instance $\boldsymbol{x}_j$ and the label of the second instance $\tilde{y}_k$ are assembled as the peer instance $(\boldsymbol{x}_j, \tilde{y}_k)$. In the pair-wise approach, for each input $(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \tilde{y}_i)$, the peer instance $(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \tilde{y}_k)$ is assembled from the two instances, i.e., $(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \tilde{y}_j)$ and $(\boldsymbol{x}_{k_1}, \boldsymbol{x}_{k_2}, \tilde{y}_k)$, randomly and pair-wisely sampled from the batch data.

A. PeerRank LOSS FUNCTIONS

1) POINT-WISE PeerRank

Referring to the definition of peer loss function in Section III-B, we perform ERM on (5) as

$$\min \quad \mathcal{J}_{\Theta,\text{peer}}(\tilde{D}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{\text{peer}}(\boldsymbol{x}_i, \tilde{y}_i), \qquad (9)$$

$$\mathcal{L}_{\text{peer}}(\boldsymbol{x}_i, \tilde{y}_i) = \ell(h(\boldsymbol{x}_i), \tilde{y}_i) - \ell(h(\boldsymbol{x}_j), \tilde{y}_k). \qquad (10)$$

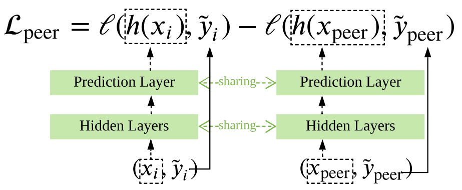

$\mathcal{L}_{\text{peer}} = \ell(h(x_i), \tilde{y}_i) - \ell(h(x_{\text{peer}}), \tilde{y}_{\text{peer}})$
Prediction Layer
sharing
Prediction Layer
Hidden Layers
sharing
Hidden Layers
$(x_i, \tilde{y}_i)$
$(x_{\text{peer}}, \tilde{y}_{\text{peer}})$

FIGURE 2. The structure of PeerRank. The features of both training samples and peer samples are fed into the networks with hidden layers and a prediction layer. In our work, the network trained with the training samples and the network trained with the peer samples share the layer parameters.

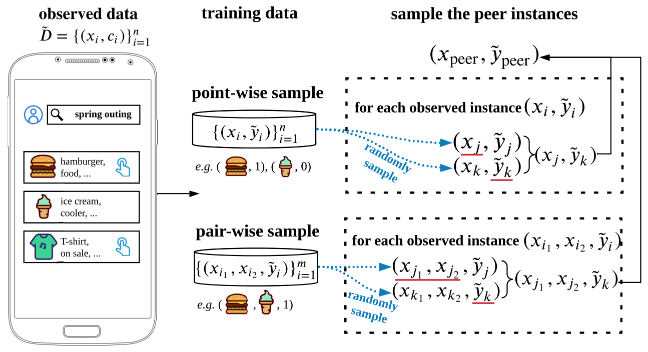

observed data
$\tilde{D} = \{(x_i, c_i)\}_{i=1}^n$
training data
sample the peer instances
$(x_{\text{peer}}, \tilde{y}_{\text{peer}})$
spring outing
hamburger, food, ...
ice cream, cooler, ...
T-shirt, on sale, ...
point-wise sample
$\{(x_i, \tilde{y}_i)\}_{i=1}^n$
e.g. (, 1), (, 0)
for each observed instance $(x_i, \tilde{y}_i)$
randomly sample
$(x_j, \tilde{y}_j)$
$(x_k, \tilde{y}_k)$
$(x_j, \tilde{y}_k)$
pair-wise sample
$\{(x_{i_1}, x_{i_2}, \tilde{y}_i)\}_{i=1}^m$
e.g. (, , 1)
for each observed instance $(x_{i_1}, x_{i_2}, \tilde{y}_i)$
randomly sample
$(x_{j_1}, x_{j_2}, \tilde{y}_j)$
$(x_{k_1}, x_{k_2}, \tilde{y}_k)$
$(x_{j_1}, x_{j_2}, \tilde{y}_k)$

FIGURE 3. The process of sampling peer instance for each current instance in point-wise and pair-wise approaches. For each training instance, a peer instance is generated. The feature and the label of peer instance are randomly and uniformly sampled from the data, respectively.

$(\boldsymbol{x}_j, \tilde{y}_j)$, $(\boldsymbol{x}_k, \tilde{y}_k)$ are randomly sampled from observed data $\tilde{D}$ while only the feature vector $\boldsymbol{x}_j$ and the label $\tilde{y}_k$ are used. ℓ can be 0-1 loss or any surrogate loss functions, e.g., ℓ usually refers to cross-entropy loss in click-through rate (CTR) prediction tasks.

2) PAIR-WISE PeerRank

We unify pair-wise approaches such as RankNet [38] and LambdaRank [39] into a framework

$$\mathcal{L}(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \tilde{y}_i) = \triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), \tilde{y}_i). \qquad (11)$$

The explanation of $\triangle$Goal varies with different algorithms. For RankNet [38], $\triangle\text{Goal}(i_1, i_2)$ equals to 1. For LambdaRank [39], $\triangle\text{Goal}(i_1, i_2)$ refers to the change of the ranking metric after swapping the positions of these two items, such as NDCG. Suppose there are m item pairs from a list of n items. Taking (8) and (11) into consideration, we perform ERM on (7) as

$$\min \quad \mathcal{J}_{\Theta,\text{peer}}(\tilde{D}) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}_{\text{peer}}(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \tilde{y}_i)$$

$$\mathcal{L}_{\text{peer}}(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \tilde{y}_i) = \triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), \tilde{y}_i) \qquad (12)$$
$$- \triangle\text{Goal}(j_1, j_2)\ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), \tilde{y}_k), \qquad (13)$$

where $(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2})$ is randomly sampled from paired feature space and $\tilde{y}_k$ is randomly sampled from the set of pair labels.

B. THEORETICAL ANALYSIS

The point-wise PeerRank, inherited from [25], has the properties of robustness, effectiveness, and adaptability. Here, we prove that pair-wise PeerRank also satisfy those properties of robustness (Theorem 1), effectiveness (Theorem 2 and 3) and adaptability. We emphasize that the extension of Peer Loss to pair-wise LTR approaches acts as our main contribution. We also give the range of the only hyper-parameter of PeerRank taking account of the real-world data distribution, which is lacking in the original paper. With all these properties, PeerRank is able to withstand errors in the input and derives an optimal or near-optimal model regardless of the label noise distributing in the data.

1) ROBUSTNESS

Robustness of PeerRank refers to its ability to learn a model whose performance is stable despite the existence of noise in the data. We demonstrate that pair-wise PeerRank has maintained the nature of resisting noise.

Theorem 1: Optimizing the PeerRank in (13) *over observed data is equivalent to optimizing that over the clean data. That is,* (13) *is invariant to label noise in $\tilde{D}$ in expectation,*

$$\mathbb{E}_{\tilde{\mathcal{D}}}[\mathcal{L}_{peer}(X_{i_1}, X_{i_2}, \tilde{Y}_i)] = (1 - e^- - e^+)\mathbb{E}_{\mathcal{D}}[\mathcal{L}_{peer}(X_{i_1}, X_{i_2}, Y_i)].$$

The proof is given in Appendix A. We have the inequality $0 \le e^+ + e^- < 1$ holds for the reason that massive errors are unlikely to happen in real life based on the rationality of the vast majority of users. Theorem 1 shows that PeerRank on a noisy training data is proportional to that on a clean data. Therefore, the model trained with loss function (13) is robust since it is invariant to noise in the training data.

2) EFFECTIVENESS

Effectiveness of pair-wise PeerRank refers to the optimization guarantee that pair-wise PeerRank can produce an optimal or near-optimal model as if performing ERM on the clean data.

Denote the *true* risk measure of model f on clean data as $R_{\mathcal{D}}(f) = \mathbb{E}_{\mathcal{D}}[\triangle\text{Goal}(i_1, i_2)\mathbb{1}(h(X_{i_1}, X_{i_2}), Y_i)]$ where (X_{i_1}, X_{i_2}, Y_i) is one instance and $\mathbb{1}(\cdot)$ is 0-1 loss. The empirical risk is defined as

$$\hat{R}_D = \frac{1}{m}\sum_{i=1}^{m} \triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), y_i). \tag{14}$$

When trained on clean data that is large enough, the empirical risk converges to the true risk. We now illustrate the connection between PeerRank's loss function and the true risk $R_{\mathcal{D}}(f)$. For the convenience of induction, we take $\ell(\cdot)$ as 0-1 loss, which has the property:

Lemma 1: If $\ell(\cdot) = \mathbb{1}(\cdot)$, $\ell(h(X), 0) + \ell(h(X), 1) = 1$.

Theorem 2 is put forward for balanced datasets where the number of positive and negative instances is almost the same. Theorem 3 is put forward for unbalanced datasets. The proofs are given in Appendix B and C respectively.

Theorem 2: When $p = Pr(Y = 1) = 0.5$, $\ell(\cdot) = \mathbb{1}(\cdot)$, we have

$$\arg\min_f \mathbb{E}_{\tilde{\mathcal{D}}}[\mathcal{L}_{peer}(X_{i_1}, X_{i_2}, \tilde{Y}_i)] = \arg\min_f R_{\mathcal{D}}(f).$$

In a more general case where the dataset is unbalanced, i.e., $p \neq 0.5$, the gap between the risk of a model trained by PeerRank and the true risk is still bounded.

Theorem 3: Denote $R_{peer,\tilde{\mathcal{D}}}(f)$ as the risk measure of model f learned from pair-wise PeerRank on noisy data. Let $\tilde{f}^*_{peer} = \arg\min_f R_{peer,\tilde{\mathcal{D}}}(f)$. Denote $R_{\mathcal{D}}(f^*)$ the optimal true risk where $f^* = \arg\min_f R_{\mathcal{D}}(f)$. Let $C_2 = 4|p - 0.5|\max_{X_i,X_j} \triangle Goal(i,j)$ and $\ell(\cdot) = \mathbb{1}(\cdot)$. The discrepancy between risk $R_{\mathcal{D}}(\tilde{f}^*_{peer})$ and $R_{\mathcal{D}}(f^*)$ is bounded by C_2. That is,

$$|R_{\mathcal{D}}(\tilde{f}^*_{peer}) - R_{\mathcal{D}}(f^*)| \le 4|p - 0.5| \max_{X_i,X_j} \triangle Goal(i,j).$$

Theorem 2 is a special case of Theorem 3 when C_2 equals to zero. In this special case, PeerRank is strongly guaranteed by Theorem 2 to produce an optimal model. For the case that the dataset is unbalanced, Theorem 3 guarantees that the optimized empirical model using PeerRank is near-optimal to minimize the risk on clean data.

3) ADAPTABILITY

Adaptability of pair-wise PeerRank refers to its ability to adapt to different datasets and different degrees of noise. For dataset that are severely unbalanced, i.e., p being distant from 0.5, Theorem 3 provides a loose bound. In this case, α-weighted PeerRank can be adopted to optimizing the ranking risk.

a: α-WEIGHTED PeerRank

When $p = Pr(Y = 1) \neq 0.5$, we adopt α-weighted PeerRank loss function in (13) where a parameter α is added to the second term:

$$\mathcal{L}_{\alpha\text{-peer}}(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \tilde{y}_i) = \triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), \tilde{y}_i) - \alpha\triangle\text{Goal}(j_1, j_2)\ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), \tilde{y}_k). \tag{15}$$

The hyper-parameter α can be regarded as a control parameter modulated with the label distribution on the dataset. Typically, when $\triangle\text{Goal}(i_1, i_2) = 1$, the optimal value of α can be calculated as claimed in [25] as

$$\begin{aligned}\alpha^* &= 1 - (1 - e^- - e^+) \cdot \frac{\delta_p}{\delta_{\tilde{p}}},\\ \delta_p &= Pr(Y = 1) - Pr(Y = 0),\\ \delta_{\tilde{p}} &= Pr(\tilde{Y} = 1) - Pr(\tilde{Y} = 0).\end{aligned} \tag{16}$$

Denote $R_{\mathcal{D}}(\hat{f}^*_{\alpha^*-\text{peer}})$ the true risk taking $\hat{f}^*_{\alpha^*-\text{peer}}$ as scoring function optimized empirically by (15) with α^*. $\hat{f}^*_{\alpha^*-\text{peer}}$ is proven to converge to the optimal scoring function by the lemma below.

Lemma 2: $\ell(\cdot) = \mathbb{1}(\cdot)$, according to Hoeffding's inequality, with a probability of at least $1 - \delta$, we have

$$R_{\mathcal{D}}(\hat{f}^*_{\alpha^*-peer}) - R_{\mathcal{D}}(f^*) \le \frac{1 + \alpha^*}{1 - e^- - e^+}\sqrt{\frac{2\log 2/\delta}{m}}.$$

Lemma 2 ensures the model trained by α-weighted PeerRank converges to the implicit optimal model. The value of α^* depends on the error transition probabilities e^+, e^- as in (16), which might not be available in real-world data.

Following, we would like to discuss the valid range of α, and we find that in most scenarios of IR, $\alpha^* \in (0, 1)$. Firstly, as we work in IR cases, the number of negative samples (non-clicked) is often much larger than positive ones (clicked). Thus, we consider δ_p and $\delta_{\tilde{p}}$ in (16) to have the same sign. Also, $1 - e^+ - e^-$ is positive if we consider the majority of users are rational and give clean labels. Then, we have $\alpha^* < 1$. In the following, we show that $\alpha^* > 0$ holds:

$$\begin{aligned}\alpha^* > 0 \Leftrightarrow 1 > (1 - \frac{\#e^+}{\#Y^+} - \frac{\#e^-}{\#Y^-}) \\ \cdot \frac{\#Y^- - \#Y^+}{(\#Y^- + \#e^+ - \#e^-) - (\#Y^+ + \#e^- - \#e^+)} \\ \Leftrightarrow \#e^+\#Y^-(\#Y^+ + \#Y^-) > \#e^-\#Y^+(\#Y^+ + \#Y^-) \\ \Leftrightarrow \frac{\#e^+}{\#Y^+} > \frac{\#e^-}{\#Y^-} \Leftrightarrow e^+ > e^-\end{aligned}$$

where $\#e^+$ ($\#e^-$) stands for the number of samples that change from positive (negative) to negative (positive), and $\#Y^+$ ($\#Y^-$) stands for the number of positive (negative) samples in the clean data.

The condition indicated by the inequalities in the last line is easy to satisfy in large-scale datasets. As $\#Y^-$ is extremely large, $\#e^-$ should be large enough in the real world to reverse the inequality, which is a rare scene in IR since most of the user behaviors make sense. Thus, mistakes on a large scale could hardly happen. On the other hand, $\#e^+$ might be relatively large in IR since users often overlook relevant items not on the top pages. This can also be validated by our experiment detailed in Section V. In semi-synthetic Yahoo dataset using PBM as click model with 0.05 rate of noise, $e^- = 0.0373$, $e^+ = 0.5037$, we have $e^+ \gg e^-$.

V. EXPERIMENTS

In this section, we conduct experiments based on two applications, CTR prediction and web search ranking, which are common scenarios of point-wise and pair-wise LTR approaches, respectively, to evaluate the performance of PeerRank.[2] We mainly focus on answering the following research questions (RQs).

- **RQ1**: Does PeerRank easily couple with SOTA point-wise and pair-wise LTR approaches and make significant improvement?
- **RQ2**: Does PeerRank achieve better performance than other SOTA de-noising methods?
- **RQ3**: How does the noise rate affect the performance of PeerRank?
- **RQ4**: How does PeerRank perform compared to the de-biasing methods?
- **RQ5**: Does PeerRank works on data involving noise caused by biases (e.g., the position bias)?

[2]Code for reproduction can be found in https://bit.ly/3cQivE4

A. EXPERIMENTAL SETTINGS

1) DATASETS

We start with the introduction of datasets and pre-processing details.

a: CTR PREDICTION

We use three large-scale public available real-world datasets for CTR prediction to evaluate the performance of PeerRank on point-wise approaches. These datasets naturally contain noisy clicks.

- **Tmall**[3] contains 847,568 behavior sequences from 423,784 users' shopping logs on the Tmall e-commerce platform in 6 months.
- **Taobao**[4] contains 1,962,046 behavior sequences from 981,023 users on the Taobao e-commerce platform from Nov. 25th to Dec. 3rd, 2017.
- **Alipay**[5] contains 996,616 online shopping behavior sequences from 498,308 users accumulated from Jul. 1st to Nov. 30th, 2015.

We follow UBR [52] to process the datasets.

b: WEB SEARCH RANKING

Our experimental data for web search ranking is derived from two widely used expert-annotated LTR datasets. Both of the datasets supply the 5-level relevance label (0-4).

- **Yahoo! LTR set 1**[6] contains 29,921 queries and 701k documents, where 700 features are extracted from each query-document pair.
- **Istella-S**[7] is composed of 33,018 queries and 3,408k documents, where each query-document pair has 220 features.

To simulate the real-world scenario where only the implicit click feedback is available, we first generate click behaviors following user click model PBM [53] and CCM [54]. The probability of document i being correlated with the user is calculated by

$$P(r_i = 1) = \frac{2^{s_i} - 1}{2^{s_{\max}} - 1}, \tag{17}$$

where s_i is the expert-annotated relevance score with the maximum value of $s_{\max}$. We then adding noise manually to the generated clicks by randomly flipping the click labels at the noise rate ϵ. In our experiments, we choose $\epsilon = 0.05$.[8]

We also do extra experiments under the setting of [55] where noise is added by transforming (17) to $P(r_i = 1) = \epsilon + (1 - \epsilon)\frac{2^{s_i}-1}{2^{s_{\max}}-1}$, and $\epsilon = 0.1$ according to [55]. It introduces feature-dependent noise into labels, which does not conform with the assumption in Peer Loss [25]. Whereas we

[3]https://tianchi.aliyun.com/dataset/dataDetail?dataId=42

[4]https://tianchi.aliyun.com/dataset/dataDetail?dataId=649

[5]https://tianchi.aliyun.com/dataset/dataDetail?dataId=53

[6]https://webscope.sandbox.yahoo.com

[7]http://quickrank.isti.cnr.it/istella-dataset/

[8]We tried different rates of noise. From statistics, noise larger than this ratio will make the distribution of generated click labels inconsistent with that of most data in IR scenarios.

conduct experiments in this setting and empirically prove that PeerRank still takes effect. More details can be found in Section V-F.

2) EVALUATION METRIC

We evaluate the performance of point-wise PeerRank with the area under the ROC curve (AUC) as it is a universal criterion for judging the merits of CTR prediction. For pair-wise PeerRank, we record Mean Average Precision (MAP, document i is regarded as relevant if $s_i \geq 1$ [55]) and Normalized Discounted Cumulative Gain (NDCG@10, abbreviated as NDCG) calculated by original 5-level relevance labels.

3) BASE MODELS

Multiple algorithms introduced in Section II served as the base models of PeerRank. We name the models **PeerX** for algorithms X coupled with PeerRank. For the CTR prediction task, we couple PeerRank with 13 point-wise LTR algorithms as in Table 1. We adopt the same hyper-parameter setting for the models and their peer versions as in [52]. The exclusive parameters for every model are tuned according to the best performance on the validation set. Specifically, we set an additional 2-layer cross net for DCN [28] and DCN-M [6]. The stacked way is chosen for DCN-M as it performs better than the parallel way. We set 3-layer compressed interaction network with 7 vectors per layer for xDeepFM [29]. We set 3 interacting layers with 2 heads per layer for AutoInt [5], and the dimension of Q, K, V are 6.

For the web search ranking task, we combine PeerRank with 4 pair-wise algorithms as in Table 2. The initial learning rate is searched from {0.0005, 0.001, 0.005, 0.01}. The batch size is set as 256. All multi-layer perceptron models are configured with a 4-layer network of [512, 256, 128, 1].

4) STATE-OF-THE-ART DE-NOISING METHODS

To demonstrate the superiority of PeerRank in de-noising, we compare our framework with several classic and widely recognized de-noising methods introduced in Section II as in Table 3. Two types of BS [13], "soft" and "hard", are implemented.

B. OVERALL PERFORMANCE (RQ1)

Table 1 shows the performance of 13 point-wise base models on 3 real-world datasets, i.e., "base" columns, and the performance of the PeerRank on these base models, i.e., "+peer" columns. Since the experimental settings are the same, some results in Table 1 with notes * are directly referred from [52]. Table 2 shows the performance of 4 pair-wise base models on 4 semi-synthetic datasets and the performance of the PeerRank on these base models.

Three properties of PeerRank can be revealed from Table 1 and Table 2. (i) PeerRank can easily couple base models and significantly achieve better performance, particularly on the Alipay dataset (improve 0.20% to 14.4%). In experiments of web search ranking where noise is only added to the training data, the clean relevance labels are used when evaluating.

TABLE 1. Results (AUC) of CTR prediction on real-world data. The "base" column represents the results of base models. "+peer" refers to the results of PeerRank coupled with base models.

	Tmall		Taobao		Alipay	
	base	+peer	base	+peer	base	+peer
GRU4Rec	0.7620*	0.7631	0.6770*	0.6811	0.6130*	0.7012
Caser	0.7620*	0.7657	0.6730*	0.6855	0.6550*	0.7133
SASRec	0.7550*	0.7658	0.6700*	0.6721	0.6480*	0.6920
HPMN	0.7630*	0.7633	0.6680*	0.6782	0.6150*	0.6958
MIMN	0.7530*	0.7750	0.6620*	0.6681	0.6640*	0.7211
DeepFM	0.7710	0.7769	0.6658	0.6695	0.6590	0.6817
IPNN	0.7384	0.7396	0.6481	0.6507	0.6900	0.6917
DCN	0.7650	0.7671	0.6224	0.6262	0.6985	0.7005
xDeepFM	0.7502	0.7548	0.6782	0.6790	0.6897	0.6916
DCN-M	0.7482	0.7492	0.6281	0.6302	0.7008	0.7022
DIN	0.7660*	0.7677	0.6780*	0.6786	0.7320*	0.7349
DIEN	0.7750*	0.7772	0.6770*	0.6811	0.7300*	0.7755
AutoInt	0.7669	0.7677	0.6356	0.6366	0.7065	0.7092

Values with * are referred from [52]. All the results of PeerRank are significant with p-value<0.05 comparing with the base model.

As can be observed from Table 2, the superior performance of PeerRank demonstrates it is invariant to label noise in the training data and achieves better results concerning both MAP and NDCG over the base models. (ii) In the CTR prediction task experiments, the noise distribution remains the same in the training and test set since they are segmented from the same observed real-world dataset. PeerRank is insensitive to noise in the test data and beats the base models, verifying its robustness. (iii) PeerRank improves all the base models on multiple datasets. The improvement of PeerRank coupled with both linear [37] and non-linear [7], [38], [39] models in web search ranking also proves its adaptability.

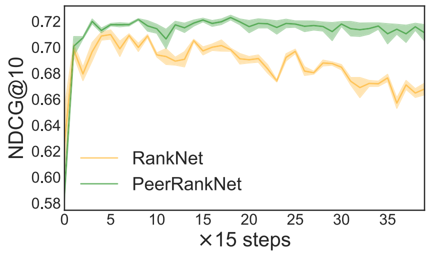


FIGURE 4. NDCG@10 on Yahoo (PBM) validation set. The NDCG of RankNet drops after several rounds, whilst the performance of PeerRank remains high and stable.

We also cast sight into why PeerRank takes effect. We experiment PeerRankNet comparing with RankNet on Yahoo (PBM) and plot NDCG during training in Fig. 4. We find that the performance of RankNet first climbs high but suffers from downdrift later on. On the contrary, the performance of PeerRankNet remains at a high level as training going on. This is probably because when no de-noising method is applied, RankNet is sensitive to noise in the data and fits the outliers. While PeerRank can prevent over-fitting to such noise and behaves well.

C. COMPARISON WITH DE-NOISING MODELS (RQ2)

In this experiment, we fix DIEN, which is regarded as the SOTA model from Alibaba Group, as the base model for the point-wise approach, and RankNet, which has stable

TABLE 2. Results of web search ranking on semi-synthetic datasets.

	Yahoo (PBM)				Yahoo (CCM)				Istella-S (PBM)				Istella-S (CCM)			
	base		+peer		base		+peer		base		+peer		base		+peer	
	MAP	**NDCG**	**MAP**	**NDCG**	**MAP**	**NDCG**	**MAP**	**NDCG**	**MAP**	**NDCG**	**MAP**	**NDCG**	**MAP**	**NDCG**	**MAP**	**NDCG**
SVMRank	0.8507	0.7081	0.8534†	0.7140†	0.8644	0.7239	0.8664†	0.7368†	0.7846	0.6725	0.7896†	0.6862†	0.7858	0.6721	0.7918†	0.6832†
RankNet	0.8526	0.7259	0.8613†	0.7325†	0.8671	0.7384	0.8684†	0.7421†	0.8050	0.6929	0.8060†	0.6951†	0.8037	0.6900	0.8053†	0.6910
LambdaRank	0.8601	0.7323	0.8627†	0.7341	0.8635	0.7380	0.8659†	0.7388†	0.8043	0.6933	0.8048†	0.6949†	0.8076	0.6926	0.8100†	0.6938
DirectRanker	0.8534	0.7290	0.8561†	0.7311†	0.8688	0.7413	0.8694	0.7429†	0.8046	0.6924	0.8056†	0.6937	0.8039	0.6902	0.8063†	0.6928†

† implies the results are significant with p-value<0.05 comparing the PeerRank with the base model.

TABLE 3. Results of the state-of-the-art de-noising methods applied to base model, i.e., DIEN (point-wise) and RankNet (pair-wise).

	Alipay		Yahoo (PBM)		
	Par.	**AUC**	**Par.**	**MAP**	**NDCG**
base		0.7300		0.8526	0.7259
+BS (soft)	β=0.95	0.7687	β=0.97	0.8572	0.7288
+BS (hard)	β=0.8	0.7747	β=0.7	0.8581	0.7295
+LS	ϵ=0.01	0.7742	ϵ=0.05	0.8580	0.7297
+GCE	q=0.5	0.7729	q=0.1	0.8540	0.7261
+CT	-	-	t=15	0.8571	0.7266
+TCE	t=6	0.7727	t=2	0.8588	0.7289
+Reweight	-	-	b=10	0.8506°	0.7167°
+PeerRank	α=0.05	**0.7755**	α=0.1	**0.8613**	**0.7325**

The best values of results are marked as bold. $^\circ$ denotes the results that are inferior to the base.

performance, for the pair-wise approach. We compare PeerRank with other de-noising methods introduced in Section V-A4 on these two models. We test on Alipay dataset for CTR prediction and Yahoo semi-synthetic dataset with PBM for web search ranking, as displayed in Table 3. We specify the exclusive hyper-parameters for each model in the "Par." columns, with which those methods achieve their best performance.

From Table 3, we can observe that (i) In either point-wise or pair-wise, the PeerRank performs the best among all SOTA de-noising approaches, which testifies it is effective when training with noisy labels. (ii) We find that not all de-noising method takes effect. e.g., Reweight [2] performs poorly. This is because it calculates the probability of labels being noisy based on the expert-annotated relevance labels, which might not be available in many IR scenarios where only a binary click label is achievable. (iii) We find it difficult or impossible for some de-noising methods to apply to vast scenarios. e.g., CT [20] algorithm demands prior knowledge of the noise rate, which is not available in many real-world datasets, such as the point-wise cases in our experiment. Reweight [2] assumes the features of an item are independent of each other, which makes it not applicable to methods such as DIEN [36] where the sequential features are correlated. GCE [10] and TCE [12] are restricted to cross-entropy only, hindering them from extending to ranking models like SVMRank. PeerRank is not subject to these limitations. It requires no prior knowledge of the noise rate and can work with click labels. Furthermore, as displayed in Table 1 and Table 2, PeerRank is easy to couple with many loss functions despite the complexity of the base models.

D. PERFORMANCE UNDER DIFFERENT NOISE RATES (RQ3)

We conduct extra experiments on Yahoo (PBM) adding other rates of noise, i.e. $\epsilon = 0.01, 0.03$, to explore how PeerRank works on different noise rates. The results are shown in Table 4. Overall speaking, the performance of both RankNet and PeerRankNet drops as the noise rate increases. Still, PeerRankNet achieves significant better performance than RankNet and drops slower, reflected in the rise of improvement recorded in "Impv." column, referring to the relative NDCG improvement of PeerRankNet over RankNet. This indicates the PeerRankNet adapts well to different noise rates and helps to alleviate the noisy labels issues.

TABLE 4. Results of web search ranking on Yahoo (PBM) under different noise rates.

	RankNet		PeerRankNet		
ϵ	**MAP**	**NDCG**	**MAP**	**NDCG**	**Impv.**
0.01	0.8666	0.7419	0.8719†	0.7452†	0.445%
0.03	0.8687	0.7381	0.8720†	0.7429†	0.650%
0.05	0.8526	0.7259	0.8613†	0.7325†	0.909%

† implies the results are significant with p-value<0.05 comparing the PeerRank with the base model.

E. COMPARISON WITH DE-BIASING MODELS (RQ4)

Since we are working with click data, we conduct extra experiments on Yahoo (PBM) to make a comparison with some de-biasing methods, IPW [56], DLA [57] and PairDebias [58], dealing with the click noise caused by biases, e.g., position bias, as shown in Table 5. To make the comparison fairly and reasonably, we adjust IPW and DLA to pair-wise manner.

We find that the result of model with IPW is inferior to that of RankNet. This is because IPW highly relies on the generated click data to learn user exam propensity weights so as to learn a robust and effective ranking model. As our click noise simulates the real noise not only caused by position bias, the IPW approach might over-fit these noises and perform poorly. The other two SOTA methods improve RankNet as their intelligent way of de-biasing through either dual learning or pair-wisely updating, but both of them consider only the bias correlated to position while neglecting other noise factors like user randomness or system failure, so they do not perform as well as PeerRank.

F. PERFORMANCE ON DATA WITH BIASES (RQ5)

We also conduct extra experiments on semi-synthetic datasets following the setting in [55] to prove that PeerRank can

TABLE 5. Comparison of the results of web search ranking on Yahoo (PBM) with the state-of-the-art de-biasing methods.

	MAP	NDCG
RankNet	0.8526	0.7259
+IPW	0.8456°	0.7156°
+DLA	0.8592	0.7308
+PairDebias	0.8595	0.7305
+PeerRank	**0.8613**	**0.7325**

The best values of results are marked as bold. ° denotes the results that are inferior to the base.

also work in the case where data is involved with biases. The biases are caused by the combination of item displaying positions and user browsing habits. The difference between this setting and that of experiments in Section V-B is the way of calculating relevance probabilities. As a result, the noise in labels is merely caused by biases like position bias. The results are presented in Table 6.

The observations from Table 6 are similar to those discovered from Table 2. This is partly attributed to the data distribution of these datasets being similar according to the statistics in Table 7. The data distributions are slightly different because these two processing methods set up different distributions of noise. Whereas the statistics reflect no apparent difference between the data distribution on the Yahoo datasets, which verifies the reasonability of our setting $\epsilon = 0.05$ when flipping the clicks.

As bias can be regarded as one kind of the noise, theoretically, dealing with bias should fall within the scope of dealing with noise. As PeerRank is a general framework to deal with noisy data, and because of its ability to dispose of various data distributions, it is reasonable to see that PeerRank behaves well under noise caused by typical ranking biases.

VI. CONCLUSION

This paper proposes an easy-to-extend framework, PeerRank, for LTR from noisy data. Specifically, PeerRank randomly samples feature vectors and labels to construct peer samples for each training instance. We propose loss functions in the PeerRank framework for both point-wise approaches and pair-wise approaches. We theoretically prove that PeerRank inherits the properties of robustness, effectiveness, and adaptability. Extensive experiments are conducted on two real-world applications in LTR. Results on three real-world datasets for the CTR prediction task and four semi-synthetic datasets for web search ranking show the superiority of our work. We leave the investigation of other user click models to generate click labels in the pair-wise experiments for future studies. More experiments combining de-noising and de-biasing methods remain to be conducted.

APPENDIX A
THE PROOF OF ROBUSTNESS

Proof: Firstly, from the definition we have

$$\mathbb{E}[\mathcal{L}_{\text{peer}}(X_{i_1}, X_{i_2}, \tilde{Y}_i)] = \mathbb{E}[\triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), \tilde{Y}_i)] - \mathbb{E}[\triangle\text{Goal}(j_1, j_2)\ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), \tilde{Y}_k)].$$

Let $p = Pr(Y = 1)$. Consider the first term on the right-hand-side:

$$\begin{aligned}
&\mathbb{E}[\triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), \tilde{Y}_i)] \\
&= \mathbb{E}_{X_i,X_j,Y=0}[\triangle\text{Goal}(i,j)(Pr(\tilde{Y}=0|Y=0)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) \\
&\quad + Pr(\tilde{Y}=1|Y=0)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))] \\
&\quad + \mathbb{E}_{X_i,X_j,Y=1}[\triangle\text{Goal}(i,j)(Pr(\tilde{Y}=0|Y=1)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) \\
&\quad + Pr(\tilde{Y}=1|Y=1)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))] \\
&= \mathbb{E}_{X_i,X_j,Y=0}[\triangle\text{Goal}(i,j)((1-e^-)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) \\
&\quad + e^-\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))] \\
&\quad + \mathbb{E}_{X_i,X_j,Y=1}[\triangle\text{Goal}(i,j)((1-e^+)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1) \\
&\quad + e^+\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0))] \\
&= \mathbb{E}_{X_i,X_j,Y=0}[\triangle\text{Goal}(i,j)((1-e^- - e^+)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) \\
&\quad + e^+\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) + e^-\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))] \\
&\quad + \mathbb{E}_{X_i,X_j,Y=1}[\triangle\text{Goal}(i,j)((1-e^- - e^+)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1) \\
&\quad + e^+\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) + e^-\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))] \\
&= (1-e^- - e^+)\mathbb{E}_{X_i,X_j,Y}[\triangle\text{Goal}(i,j)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),y)] \\
&\quad + \mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)(e^+\triangle\text{Goal}(i,j)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) \\
&\quad + e^-\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))];
\end{aligned}$$

and consider the second term:

$$\begin{aligned}
&\mathbb{E}[\triangle\text{Goal}(j_1, j_2)\ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), \tilde{Y}_k)] \\
&= \mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)(\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0)Pr(\tilde{Y}=0) \\
&\quad + \ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1)Pr(\tilde{Y}=1))] \\
&= \mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)(\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0)(e^+p + (1-e^-)(1-p)) \\
&\quad + \ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1)((1-e^+)p + e^-(1-p)))] \\
&= \mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)((1-e^- - e^+)(1-p)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) \\
&\quad + (1-e^- - e^+)p\cdot\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))] \\
&\quad + \mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)(e^+\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0)+e^-\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))] \\
&= (1-e^- - e^+)\mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),Y)] \\
&\quad + \mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)(e^+\triangle\text{Goal}(i,j)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),0) \\
&\quad + e^-\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),1))].
\end{aligned}$$

Subtracting the first and second term on right-hand-side we get:

$$\begin{aligned}
&\mathbb{E}[\mathcal{L}_{\text{peer}}(X_{i_1}, X_{i_2}, \tilde{Y}_i)] \\
&\quad = \mathbb{E}[\triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), \tilde{Y}_i)] \\
&\quad\quad - \mathbb{E}[\triangle\text{Goal}(j_1, j_2)\ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), \tilde{Y}_k)] \\
&\quad = (1-e^- - e^+)\mathbb{E}_{X_i,X_j,Y}[\triangle\text{Goal}(i,j)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),y)] \\
&\quad\quad - (1-e^- - e^+)\mathbb{E}_{X_i,X_j}[\triangle\text{Goal}(i,j)\ell(h(\boldsymbol{x}_i,\boldsymbol{x}_j),y)] \\
&\quad = (1-e^- - e^+)\mathbb{E}_{\mathcal{D}}[\mathcal{L}_{\text{peer}}(X_{i_1}, X_{i_2}, Y_i)].
\end{aligned}$$

□

TABLE 6. Results of web search ranking on semi-synthetic datasets under noise caused by biases [55].

	Yahoo (PBM)				Yahoo (CCM)				Istella-S (PBM)				Istella-S (CCM)			
	base		+peer		base		+peer		base		+peer		base		+peer	
	MAP	NDCG	MAP	NDCG	MAP	NDCG	MAP	NDCG	MAP	NDCG	MAP	NDCG	MAP	NDCG	MAP	NDCG
SVMRank	0.8568	0.7193	$0.8612^{\dagger}$	$0.7247^{\dagger}$	0.8612	0.7228	$0.8625^{\dagger}$	$0.7254^{\dagger}$	0.7302	0.6217	$0.7401^{\dagger}$	$0.6388^{\dagger}$	0.7855	0.6717	$0.7930^{\dagger}$	$0.6815^{\dagger}$
RankNet	0.8640	0.7343	$0.8683^{\dagger}$	$0.7381^{\dagger}$	0.8636	0.7304	$0.8648^{\dagger}$	$0.7324^{\dagger}$	0.8002	0.6865	$0.8051^{\dagger}$	$0.6888^{\dagger}$	0.8023	0.6831	$0.8047^{\dagger}$	$0.6850^{\dagger}$
LambdaRank	0.8669	0.7364	0.8675	$0.7391^{\dagger}$	0.8654	0.7284	0.8675	$0.7327^{\dagger}$	0.8039	0.6897	0.8045	$0.6918^{\dagger}$	0.8020	0.6823	$0.8052^{\dagger}$	$0.6847^{\dagger}$
DirectRanker	0.8639	0.7324	$0.8684^{\dagger}$	$0.7365^{\dagger}$	0.8649	0.7287	$0.8669^{\dagger}$	$0.7330^{\dagger}$	0.8021	0.6860	$0.8057^{\dagger}$	0.6870	0.7987	0.6835	$0.8057^{\dagger}$	$0.6862^{\dagger}$

† implies the results are significant with p-value<0.05 comparing the PeerRank with the base model.

TABLE 7. Data distribution of semi-synthetic datasets after simulating clicks in different ways. The "1st way" refers to adding noise by randomly flipping. The "2nd way" refers to adding biases [55].

Positive rate	original	1st way	2nd way
Yahoo (PBM)	0.0281	0.0502	0.0410
Yahoo (CCM)	0.0362	0.0418	0.0419
Istella (PBM)	0.0122	0.0609	0.0191
Istella (CCM)	0.0100	0.0588	0.0102

APPENDIX B
THE PROOF OF EFFECTIVENESS (BALANCED DATA)

Proof: Let $C_1 = 0.5 \cdot (1 - e^- - e^+) \cdot \mathbb{E}_{(X_{i_1}, X_{i_2}, Y_i)\sim\mathcal{D}}[\triangle\text{Goal}(i_1, i_2)]$. Apply Theorem 1,

$$\begin{aligned}
&\mathbb{E}_{\tilde{\mathcal{D}}}[\mathcal{L}_{\text{peer}}(X_{i_1}, X_{i_2}, \tilde{Y}_i)] \\
&= (1 - e^- - e^+)\mathbb{E}_{\mathcal{D}}[\mathcal{L}_{\text{peer}}(X_{i_1}, X_{i_2}, Y_i)] \\
&= (1 - e^- - e^+)[\mathbb{E}[\triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), Y_i)] \\
&\quad - \mathbb{E}_{X_{j_1}, X_{j_2}}[\triangle\text{Goal}(j_1, j_2)\ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), Y_k)]] \\
&= (1 - e^- - e^+)[\mathbb{E}[\triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), Y_i)] \\
&\quad - \mathbb{E}_{X_{j_1}, X_{j_2}}[\triangle\text{Goal}(j_1, j_2) \\
&\quad \cdot (0.5 \cdot \ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), 0) + 0.5 \cdot \ell(h(\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}), 1))]] \\
&= (1 - e^- - e^+)[\mathbb{E}[\triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), Y_i)] \\
&\quad - 0.5 \cdot \mathbb{E}_{X_{j_1}, X_{j_2}}[\triangle\text{Goal}(j_1, j_2)]] \\
&= (1 - e^- - e^+)\mathbb{E}[\triangle\text{Goal}(i_1, i_2)\ell(h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}), Y_i)] - C_1.
\end{aligned}$$

Given the implicit clean dataset D collected from $\mathcal{D}$, $C_1 = 0.5 \cdot (1 - e^- - e^+) \cdot \frac{1}{m}\sum_{i=1}^{m} \triangle\text{Goal}(i_1, i_2)$ is a constant with respect to f, so

$$\mathbb{E}_{\tilde{\mathcal{D}}}[\mathcal{L}_{\text{peer}}(X_{i_1}, X_{i_2}, \tilde{Y}_i)] \propto (1 - e^- - e^+)R_{\mathcal{D}}(f) + \text{const}.$$

□

APPENDIX C
THE PROOF OF EFFECTIVENESS (UNBALANCED DATA)

To prove Theorem 3, we first introduce Lemma 3 and Corollary 1.

Lemma 3: From [25], for all h,

$$\begin{aligned}
&|p\mathbb{E}_X[\ell(h(X), 1)] + (1 - p)\mathbb{E}_X[\ell(h(X), 0)] \\
&\quad - 0.5\mathbb{E}_X[\ell(h(X), 1)] - 0.5\mathbb{E}_X[\ell(h(X), 0)]| \leq |p - 0.5|.
\end{aligned}$$

For ease of expression, we abbreviate $h(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2})$ in (6) as h when there is no confusion. Derive the pair-wise format of Lemma 3:

Corollary 1:

$$\begin{aligned}
&|p\mathbb{E}_{X_i, X_j}[\triangle Goal(i, j)\ell(h, 1)] \\
&\quad + (1 - p)\mathbb{E}_{X_i, X_j}[\triangle Goal(i, j)\ell(h, 0)] \\
&\quad - 0.5\mathbb{E}_{X_i, X_j}[\triangle Goal(i, j)\ell(h, 1)] \\
&\quad - 0.5\mathbb{E}_{X_i, X_j}[\triangle Goal(i, j)\ell(h, 0)]| \\
&\leq |p - 0.5| \cdot 2\max_{X_i, X_j} \triangle Goal(i, j).
\end{aligned}$$

Then the proof of Theorem 3 is given below:

Proof: We short-hand $h_{\tilde{f}^*_{\text{peer}}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ as $h_{\tilde{f}^*_{\text{peer}}}$ substituting $\tilde{f}^*_{\text{peer}}$ as scoring function in (6). Correspondingly, h_{f^*} stands for $h_{f^*}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. From the definition of $\tilde{f}^*_{\text{peer}}$ we have

$$\mathbb{E}[\mathcal{L}_{\text{peer}}(h_{\tilde{f}^*_{\text{peer}}}, \tilde{Y})] \leq \mathbb{E}[\mathcal{L}_{\text{peer}}(h_{f^*}, \tilde{Y})].$$

Apply Theorem 1 we know that

$$\mathbb{E}[\mathcal{L}_{\text{peer}}(h_{\tilde{f}^*_{\text{peer}}}, Y)] \leq \mathbb{E}[\mathcal{L}_{\text{peer}}(h_{f^*}, Y)].$$

Then

$$\begin{aligned}
&R_{\mathcal{D}}(\tilde{f}^*_{\text{peer}}) - 0.5 \cdot \mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)] \\
&= R_{\mathcal{D}}(\tilde{f}^*_{\text{peer}}) - 0.5 \cdot \mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{\tilde{f}^*_{\text{peer}}}, 1)] \\
&\quad - 0.5 \cdot \mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{\tilde{f}^*_{\text{peer}}}, 0)] \\
&\leq R_{\mathcal{D}}(\tilde{f}^*_{\text{peer}}) - p\mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{\tilde{f}^*_{\text{peer}}}, 1)] \\
&\quad - (1 - p)\mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{\tilde{f}^*_{\text{peer}}}, 0)] \\
&\quad + 2|p - 0.5| \max_{X_i, X_j} \triangle\text{Goal}(i, j) \\
&\leq R_{\mathcal{D}}(f^*) - p\mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{f^*}, 1)] \\
&\quad - (1 - p)\mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{f^*}, 0)] \\
&\quad + 2|p - 0.5| \max_{X_i, X_j} \triangle\text{Goal}(i, j) \\
&\leq R_{\mathcal{D}}(f^*) - 0.5 \cdot \mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{f^*}, 1)] \\
&\quad - 0.5 \cdot \mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)\ell(h_{f^*}, 0)] + C_2 \\
&= R_{\mathcal{D}}(f^*) - 0.5 \cdot \mathbb{E}_{X_i, X_j}[\triangle\text{Goal}(i, j)] + C_2.
\end{aligned}$$

The second inequality is obtained from Corollary 1. Eliminate the same terms in both sides of the inequality, we obtain

$$R_{\mathcal{D}}(\tilde{f}^*_{\text{peer}}) - R_{\mathcal{D}}(f^*) \leq 4|p - 0.5| \max_{X_i, X_j} \triangle\text{Goal}(i, j).$$

□

ACKNOWLEDGMENT
All support is gratefully acknowledged.

REFERENCES

[1] S. Sidana, M. Trofimov, O. Horodnytskyi, C. Laclau, Y. Maximov, and M.-R. Amini, "User preference and embedding learning with implicit feedback for recommender systems," *Data Mining Knowl. Discovery*, vol. 35, pp. 568–592, Jan. 2021.

[2] W. Ding, X. Geng, and X.-D. Zhang, "Learning to rank from noisy data," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 1, pp. 1–21, Oct. 2015.

[3] J. Xu, C. Chen, G. Xu, H. Li, and E. R. T. Abib, "Improving quality of training data for learning to rank using click-through data," in *Proc. 3rd ACM Int. Conf. Web Search Data Mining (WSDM)*, New York, NY, USA, Feb. 2010, pp. 171–180.

[4] V. R. Carvalho, J. L. Elsas, W. W. Cohen, and J. G. Carbonell, "A meta-learning approach for robust rank learning," in *Proc. SIGIR Workshop Learn. Rank Inf. Retr.*, vol. 1, 2008, pp. 1–9.

[5] W. Song, C. Shi, Z. Xiao, Z. Duan, Y. Xu, M. Zhang, and J. Tang, "AutoInt: Automatic feature interaction learning via self-attentive neural networks," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Beijing, China, Nov. 2019, pp. 1161–1170.

[6] R. Wang, R. Shivanna, D. Z. Cheng, S. Jain, D. Lin, L. Hong, and E. H. Chi, "DCN-M: Improved deep & cross network for feature cross learning in web-scale learning to rank systems," 2020, *ArXiv:2008.13535*.

[7] M. Köppel, A. Segner, M. Wagener, L. Pensel, A. Karwath, and S. Kramer, "Pairwise learning to rank by neural networks revisited: Reconstruction, theoretical analysis and practical performance," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Würzburg, Germany: Springer, 2019, pp. 237–252.

[8] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *Proc. 5th Int. Conf. Learn. Representations (ICLR)*, in Conference Track Proceedings, Toulon, France, Apr. 2017, pp. 1–15.

[9] A. Ghosh, H. Kumar, and P. S. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proc. 31st AAAI Conf. Artif. Intell.*, San Francisco, CA, USA: AAAI Press, Feb. 2017, pp. 1919–1925.

[10] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Proc. 32nd Conf. Neural Inf. Process. Syst. (NeurIPS)*, Montreal, QC, Canada, Dec. 2018, pp. 8792–8802.

[11] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Seoul, South Korea, Oct. 2019, pp. 322–330.

[12] L. Feng, S. Shu, Z. Lin, F. Lv, L. Li, and B. An, "Can cross entropy loss be robust to label noise?" in *Proc. 29th Int. Joint Conf. Artif. Intell. (IJCAI)*, Jul. 2020, pp. 2206–2212.

[13] S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping," in *Proc. 3rd Int. Conf. Learn. Representations (ICLR)*, 2015, pp. 1–11.

[14] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. 3rd Int. Conf. Learn. Representations (ICLR)*, in Conference Track Proceedings, San Diego, CA, USA, May 2015, pp. 1–11.

[15] I. Jindal, M. Nokleby, and X. Chen, "Learning deep networks from noisy labels with dropout regularization," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 967–972.

[16] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "*mixup*: Beyond empirical risk minimization," in *Proc. 6th Int. Conf. Learn. Representations (ICLR)*, in Conference Track Proceedings, Vancouver, BC, Canada, Apr./May 2018, pp. 1–13.

[17] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. E. Hinton, "Regularizing neural networks by penalizing confident output distributions," in *Proc. 5th Int. Conf. Learn. Representations (ICLR)*, 2017, pp. 1–12.

[18] X. Geng, T. Qin, T.-Y. Liu, and X.-Q. Cheng, "A noise-tolerant graphical model for ranking," *Inf. Process. Manage.*, vol. 48, no. 2, pp. 374–383, Mar. 2012.

[19] L. Jiang, Z. Zhou, T. Leung, L. Li, and L. Fei-Fei, "MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *Proc. Int. Conf. Med. Learn. (ICML)*, in Proceedings of Machine Learning Research, vol. 80. Stockholm, Sweden, 2018, pp. 2309–2318.

[20] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. W. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, Montreal, QC, Canada, Dec. 2018, pp. 8536–8546.

[21] D. T. Nguyen, C. K. Mummadi, T. Ngo, T. H. P. Nguyen, L. Beggel, and T. Brox, "SELF: Learning to filter noisy labels with self-ensembling," in *Proc. 8th Int. Conf. Learn. Representations (ICLR)*, Addis Ababa, Ethiopia, Apr. 2020, pp. 1–15.

[22] H. Han, S.-W. Hwang, Y.-I. Song, and S. Kim, "Training data optimization for pairwise learning to rank," in *Proc. ACM SIGIR Int. Conf. Theory Inf. Retr.*, Sep. 2020, pp. 13–20.

[23] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proc. Int. Conf. Mach. Learn.*, in Proceedings of Machine Learning Research, vol. 80. Stockholm, Sweden, 2018, pp. 4331–4340.

[24] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from noisy labels with deep neural networks: A survey," 2020, *arXiv:2007.08199*.

[25] Y. Liu and H. Guo, "Peer loss functions: Learning from noisy labels without knowing noise rates," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, in Proceedings of Machine Learning Research, vol. 119, Jul. 2020, pp. 6226–6236.

[26] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," in *Proc. 26th Int. Joint Conf. Artif. Intell. (IJCAI)*, Melbourne, VIC, Australia, Aug. 2017, pp. 1725–1731.

[27] Y. Qu, H. Cai, K. Ren, W. Zhang, Y. Yu, Y. Wen, and J. Wang, "Product-based neural networks for user response prediction," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 1149–1154.

[28] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proc. ADKDD*, Aug. 2017, pp. 1–7.

[29] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "XDeepFM: Combining explicit and implicit feature interactions for recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, London, U.K., Jul. 2018, pp. 1754–1763.

[30] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," in *Proc. 4th Int. Conf. Learn. Representations (ICLR)*, in Conference Track Proceedings, San Juan, Puerto Rico, May 2016, pp. 1–10.

[31] J. Tang and K. Wang, "Personalized top-*N* sequential recommendation via convolutional sequence embedding," in *Proc. 11th ACM Int. Conf. Web Search Data Mining (WSDM)*, Marina Del Rey, CA, USA, Feb. 2018, pp. 565–573.

[32] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *Proc. IEEE 18th Int. Conf. Data Mining (ICDM)*, Nov. 2018, pp. 197–206.

[33] Q. Pi, W. Bian, G. Zhou, X. Zhu, and K. Gai, "Practice on long sequential user behavior modeling for click-through rate prediction," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Anchorage, AK, USA, Jul. 2019, pp. 2671–2679.

[34] K. Ren, J. Qin, Y. Fang, W. Zhang, L. Zheng, W. Bian, G. Zhou, J. Xu, Y. Yu, X. Zhu, and K. Gai, "Lifelong sequential modeling with personalized memorization for user response prediction," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Paris, France, Jul. 2019, pp. 565–574.

[35] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, London, U.K., Jul. 2018, pp. 1059–1068.

[36] G. Zhou, N. Mou, Y. Fan, Q. Pi, W. Bian, C. Zhou, X. Zhu, and K. Gai, "Deep interest evolution network for click-through rate prediction," in *Proc. 33rd AAAI Conf. Artif. Intell., 31st Innov. Appl. Artif. Intell. Conf. (IAAI), 9th AAAI Symp. Educ. Adv. Artif. Intell. (EAAI)*, Honolulu, HI, USA: AAAI Press, Jan./Feb. 2019, pp. 5941–5948.

[37] T. Joachims, "Training linear SVMs in linear time," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2006, pp. 217–226.

[38] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, in ACM International Conference Proceeding Series, Bonn, Germany, 2005, pp. 89–96.

[39] P. Donmez, K. M. Svore, and C. J. C. Burges, "On the local optimality of LambdaRank," in *Proc. 32nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2009, pp. 460–467.

[40] J. Xu and H. Li, "AdaRank: A boosting algorithm for information retrieval," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2007, pp. 391–398.

[41] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li, "Listwise approach to learning to rank: Theory and algorithm," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, in ACM International Conference Proceeding Series, Helsinki, Finland, 2008, pp. 1192–1199.

[42] M. Taylor, J. Guiver, S. Robertson, and T. Minka, "SoftRank: Optimizing non-smooth rank metrics," in *Proc. Int. Conf. Web Search Web Data Mining (WSDM)*, Palo Alto, CA, USA, Feb. 2008, pp. 77–86.

[43] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li, "Learning from noisy labels with distillation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*. Venice, Italy: IEEE Computer Society, Oct. 2017, pp. 1928–1936.

[44] Z. Zhang, H. Zhang, S. O. Arik, H. Lee, and T. Pfister, "Distilling effective supervision from severe label noise," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, Jun. 2020, pp. 9291–9300.

[45] J. Li, R. Socher, and S. C. H. Hoi, "DivideMix: Learning with noisy labels as semi-supervised learning," in *Proc. 8th Int. Conf. Learn. Representations (ICLR)*, Addis Ababa, Ethiopia, Apr. 2020, pp. 1–14.

[46] C. J. Burges, "From RankNet to LambdaRank to LambdaMART: An overview," *Learning*, vol. 11, nos. 23–581, p. 81, Jun. 2010.

[47] S. Clémençon, G. Lugosi, and N. Vayatis, "Ranking and scoring using empirical risk minimization," in *Proc. Int. Conf. Comput. Learn. Theory* (Lecture Notes in Computer Science), vol. 3559. Bertinoro, Italy: Springer, Jun. 2005, pp. 1–15.

[48] T.-Y. Liu, "Learning to rank for information retrieval," in *Proc. 33rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Geneva, Switzerland, Jul. 2010, p. 904.

[49] W. Schapire and Y. Singer, "Learning to order things," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 10, 1998, p. 451.

[50] A. Dasgupta and A. Ghosh, "Crowdsourced judgement elicitation with endogenous proficiency," in *Proc. 22nd Int. Conf. World Wide Web (WWW)*, in International World Wide Web Conferences Steering Committee/ACM, Rio de Janeiro, Brazil, 2013, pp. 319–330.

[51] V. Shnayder, A. Agarwal, R. Frongillo, and D. C. Parkes, "Informed truthfulness in multi-task peer prediction," in *Proc. ACM Conf. Econ. Comput. (EC)*, Jul. 2016, pp. 179–196.

[52] J. Qin, W. Zhang, X. Wu, J. Jin, Y. Fang, and Y. Yu, "User behavior retrieval for click-through rate prediction," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Xi'an, China, Jul. 2020, pp. 2347–2356.

[53] M. Richardson, E. Dominowska, and R. Ragno, "Predicting clicks: Estimating the click-through rate for new ads," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, Banff, AB, Canada, May 2007, pp. 521–530.

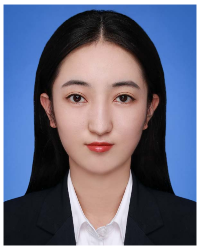
[54] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y.-M. Wang, and C. Faloutsos, "Click chain model in web search," in *Proc. 18th Int. Conf. World wide Web (WWW)*, Madrid, Spain, Apr. 2009, pp. 11–20.

[55] Q. Ai, T. Yang, H. Wang, and J. Mao, "Unbiased learning to rank: Online or offline?" *ACM Trans. Inf. Syst.*, vol. 39, no. 2, pp. 1–29, Apr. 2021.

[56] T. Joachims, A. Swaminathan, and T. Schnabel, "Unbiased learning-to-rank with biased feedback," in *Proc. 10th ACM Int. Conf. Web Search Data Mining (WSDM)*, Cambridge, U.K., Feb. 2017, pp. 781–789.

[57] Q. Ai, K. Bi, C. Luo, J. Guo, and W. B. Croft, "Unbiased learning to rank with unbiased propensity estimation," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Ann Arbor, MI, USA, Jun. 2018, pp. 385–394.

[58] Z. Hu, Y. Wang, Q. Peng, and H. Li, "Unbiased LambdaMART: An unbiased pairwise learning-to-rank algorithm," in *Proc. World Wide Web Conf. (WWW)*, San Francisco, CA, USA, May 2019, pp. 2830–2836.



XIN WU received the B.S. degree in computer science and technology from Nanjing University, Nanjing, China, in 2019. She is currently pursuing the M.S. degree in computer science and technology with Shanghai Jiao Tong University, Shanghai, China. Her research interests include learning to rank, and click through rate prediction in recommender or information systems.

QING LIU received the B.S. degree in computer science and technology from East China Normal University, Shanghai, China, in 2013, and the Ph.D. degree in computer science from the National University of Singapore, Singapore, in 2017.

From 2018 to 2021, she was a Senior Research Engineer with Noah's Ark Laboratory, Huawei, Shenzhen, China. Since 2021, she has been a Senior Algorithmic Engineer with the AI Department, Bilibili, Shanghai. Her research interests include click through rate prediction, unbiased learning to rank, and knowledge transferring in recommender or information systems.

JIARUI QIN received the B.Eng. degree in software engineering from Shanghai Jiao Tong University, in 2019, where he is currently pursuing the Ph.D. degree with the Computer Science Department. He has published several papers on SIGIR, SIGKDD, WSDM, AAAI, IJCAI, and *TOIS*. His research interests include data mining, machine learning, and information retrieval.

YONG YU is currently a Professor with the Department of Computer Science, Shanghai Jiao Tong University. He has published over 200 articles and served as a PC Member of several conferences, including WWW, RecSys, and a dozen of other related conferences, such as NIPS, ICML, SIGIR, and ISWC, in these fields. His research interests include information systems, web search, data mining, and machine learning.