Lower-Left Partial AUC: An Effective and Efficient Optimization Metric for Recommendation

Wentao Shi*
University of Science and Technology of China
Hefei, China
shiwentao123@mail.ustc.edu.cn

Chenxu Wang*
University of Science and Technology of China
Hefei, China
wcx123@mail.ustc.edu.cn

Fuli Feng
University of Science and Technology of China
Hefei, China
fulifeng93@gmail.com

Yang Zhang
University of Science and Technology of China
Hefei, China
zy2015@mail.ustc.edu.cn

Wenjie Wang†
National University of Singapore
Singapore, Singapore
wenjiewang96@gmail.com

Junkang Wu
University of Science and Technology of China
Hefei, China
jkwu0909@gmail.com

Xiangnan He†
University of Science and Technology of China
Hefei, China
xiangnanhe@gmail.com

ABSTRACT

Optimization metrics are crucial for building recommendation systems at scale. However, an effective and efficient metric for practical use remains elusive. While Top-K ranking metrics are the gold standard for optimization, they suffer from significant computational overhead. Alternatively, the more efficient accuracy and AUC metrics often fall short of capturing the true targets of recommendation tasks, leading to suboptimal performance. To overcome this dilemma, we propose a new optimization metric, Lower-Left Partial AUC (LLPAUC), which is computationally efficient like AUC but strongly correlates with Top-K ranking metrics. Compared to AUC, LLPAUC considers only the partial area under the ROC curve in the Lower-Left corner to push the optimization focus on Top-K. We provide theoretical validation of the correlation between LLPAUC and Top-K ranking metrics and demonstrate its robustness to noisy user feedback. We further design an efficient point-wise recommendation loss to maximize LLPAUC and evaluate it on three datasets, validating its effectiveness and robustness. The code is available at https://github.com/swt-user/LLPAUC.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Partial AUC; Recommendation System; Optimization Metrics

*Both authors contributed equally to this research.
†Corresponding author

WWW '24, May 13–17, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0171-9/24/05
https://doi.org/10.1145/3589334.3645371

ACM Reference Format:
Wentao Shi, Chenxu Wang, Fuli Feng, Yang Zhang, Wenjie Wang, Junkang Wu, and Xiangnan He. 2024. Lower-Left Partial AUC: An Effective and Efficient Optimization Metric for Recommendation. In *Proceedings of the ACM Web Conference 2024 (WWW '24), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3589334.3645371

1 INTRODUCTION

Recommender systems, core engines for Web applications, aim to alleviate Web information overload by recommending the Top-K most relevant items for each user [26, 36]. They are widely adopted in large-scale Web applications such as Amazon and TikTok [5], and typically learned from historical user feedback using optimization metrics related to item ranking [28]. While Top-K ranking metrics such as NDCG@K and Recall@K align well with the goals of recommendation tasks, they are not suitable for practical use at scale due to their substantial computational cost [28]. There thus remains a need to explore effective and efficient optimization metrics for recommender models.

Prior research pursues the target through the trade-off between efficiency and alignment with the Top-K ranking. One approach is to frame the recommendation task as a classification problem and optimize the accuracy metric [6], which inherently deviates from the Top-K ranking. Another approach optimizes the Area Under the Receiver Operating Characteristic (ROC) curve (AUC) metric [29] as shown in Figure 1(a), which quantifies the probability of ranking a random positive item higher than a negative one. AUC accounts for item ranking but treats all items equally, which may not improve the ranking quality for Top-K items when optimized, leading to suboptimal recommendation performance.

In this work, we propose a new optimization metric, Lower-Left Partial AUC, designed to be more correlated with Top-K ranking than the traditional AUC metric. LLPAUC introduces constraints on the upper bound of False Positive Rate (FPR) and True Positive Rate (TPR), *i.e.,* focusing on the partial area under the ROC curve in the Lower-Left corner as depicted in Figure 1(b). These constraints can narrow the ranking to only include the top-ranked items as shown in Figure 1(c), strengthening the correlation with Top-K metrics. Our theoretical analysis shows that LLPAUC can tighter bound Top-K ranking metrics. Notably, the constraint on TPR can also

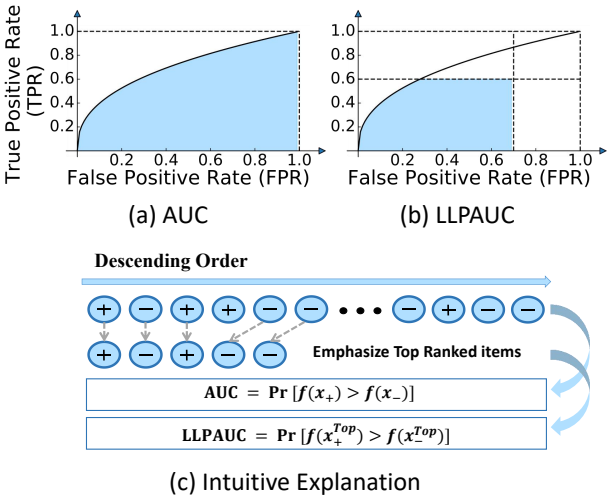

Figure 1: (a) AUC measures the entire area under the ROC curve; (b) LLPAUC considers the lower-left corner; (c) Compared to AUC, LLPAUC only considers the ranking for top-ranked items.

prevent the optimization from overfitting noisy user feedback [33], making LLPAUC more robust than AUC.

Nevertheless, the optimization of LLPAUC is non-trivial due to the non-differentiable and computationally expensive TPR and FPR constraint operations. To address these challenges, following [30], we reformulate the constraint operations using the average Top-K loss [8] to make it differentiable and amenable to mini-batch optimization. On top of these efforts, we propose a minimax point-wise loss function, which efficiently maximizes the LLPAUC metric. Moreover, both time complexity analysis and empirical results on real-world datasets verify its efficiency.

The main contributions of the paper are summarized as follows:

- We propose a new optimization metric LLPAUC for recommendation, and provide both theoretical and empirical evidence on its stronger correlation with Top-K ranking metrics.
- We derive an efficient point-wise loss function for maximizing the LLPAUC metric, which has comparable complexity as conventional point-wise recommendation losses.
- We conduct extensive experiments on three datasets under both clean and noisy settings, demonstrating the effectiveness and robustness of optimizing LLPAUC for recommendation.

2 RELATED WORK

In this section, we briefly introduce the optimization metrics and loss functions for the recommendation task and review recent studies in partial AUC and its optimization.

2.1 Optimization Metrics In Recommendation

In general, there are two common types of loss functions in recommender systems. Point-wise loss functions such as Binary Cross Entropy (BCE) loss [18] cast the recommendation task into a classification problem and optimize the accuracy metric. Pair-wise loss functions such as Bayesian Personalized Ranking (BPR) loss [29] are optimized to maximize the AUC metric. In addition, softmax cross-entropy loss [6] is also widely used to maximize the likelihood estimation of classification. Despite their optimization efficiency, these loss functions have a significant gap with the ideal Top-K ranking metrics.

Beyond these employed loss functions, some approaches aim to directly optimize Top-K ranking metrics, such as NDCG@K [28] and Recall@K [27, 32]. However, these methods are computationally expensive and are not suitable for large-scale applications. To tackle this issue, recent studies have proposed the pAp@K metric [3, 19], which combines partial AUC metric and Precision@K metric. The pAp@K metric represents a specific instance of LLPAUC and offers better alignment with Top-K metrics, which lacks theoretical support. On the contrary, our study introduces the more generalized LLPAUC metric and conducts theoretical analyses and simulated experiments to establish the strong relationship between the LLPAUC metric and Top-K metrics.

2.2 Partial AUC And Its Optimization

The concept of partial AUC was initially introduced by [22]. In various applications, such as drug discovery and graph anomaly detection [10–12], only the partial AUC up to a low false positive rate is of interest [24], which motivates the research on One-way Partial AUC (OPAUC). [31] first discusses the correlation between OPAUC and Top-K metrics for recommendation. Later, [37] argues that a practical classifier must simultaneously have a high TPR and a low FPR. Hence, they propose a new metric named Two-way Partial AUC (TPAUC), which pays attention to the upper-left head region under the ROC curve. Then, [39] first proposes an end-to-end TPAUC optimization framework, which has a profound impact on subsequent work [40]. Nevertheless, TPAUC does not align with the Top-K ranking metrics in the recommendation. The proposed LLPAUC metric exhibits a stronger correlation with Top-K ranking metrics. Beyond that, LLPAUC can additionally alleviate the issue of label noise in recommender systems.

Regarding the optimization of partial AUC, previous works [7, 20, 23, 25] rely on full-batch optimization and the approximation of the Top (Bottom)-K ranking, leading to immeasurable biases and inefficiency. Recently, novel end-to-end mini-batch optimization frameworks have been proposed [39, 41, 43]. These methods can be extended to optimize our proposed LLPAUC metric. In this work, we utilize an unbiased mini-batch optimization scheme [30] due to its superiority in the previous investigation.

3 PRELIMINARY

In this section, we present our task formulation and partial AUC formulation for recommendation.

3.1 Task Formulation

The primary objective of a recommender is to learn a score function $f(u, i|\theta)$ which is parameterized by θ and predicts the preference of a user $u \in \mathcal{U}$ on an item $i \in \mathcal{I}$. In this work, we only focus on f: $\mathcal{U} \times \mathcal{I} \rightarrow [0, 1]$. For convenience, we use $f_{u,i}$ to denote $f(u, i|\theta)$. This work focuses on the implicit feedback setting [38], where positive interactions contain all items interacted with by u (denoted by $\mathcal{I}_u^+ \subseteq \mathcal{I}$), and negative interactions correspond to all

non-interacted items (denoted by $\mathcal{I}_u^- \subseteq \mathcal{I}$). Typically, the learning process is formulated as:

$$\min_{\theta} \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}_u^+} \sum_{j \in \mathcal{I}_u^-} \frac{1}{|\mathcal{I}_u^+| \cdot |\mathcal{I}_u^-|} L(\theta, u, i, j), \tag{1}$$

where $L(\theta, u, i, j)$ denotes the fitting loss for the the positive item i and negative item j of user u. The choice of $L(\cdot)$ determines the optimization metrics. For example, the BPR loss [29] can be selected to optimize AUC, while binary cross-entropy loss [6] can be used to optimize accuracy metrics. During serving, the recommender generates a Top-K recommendation list for each user based on the prediction scores. This work aims to develop optimization metrics that are better aligned with the Top-K ranking metrics and can be optimized efficiently.

3.2 AUC And Partial AUC

AUC is a widely considered optimization metric in the recommendation, which is defined as the region enclosed by the ROC curve [2], as Figure 1(a) shows. Given a threshold t and a score function f, we can define true positive rates (TPR) and false positive rates (FPR) as $\text{TPR}_u(t) = \mathbf{Pr}(f_{u,i} > t | i \in \mathcal{I}_u^+)$ and $\text{FPR}_u(t) = \mathbf{Pr}(f_{u,j} > t | j \in \mathcal{I}_u^-)$, respectively. For a given value $\xi \in [0, 1]$, let $\text{TPR}_u^{-1}(\xi) = \inf\{t \in \mathbb{R}, \text{TPR}_u(t) < \xi\}$ and $\text{FPR}_u^{-1}(\xi) = \inf\{t \in \mathbb{R}, \text{FPR}_u(t) < \xi\}$. Then, according to Figure 1(a), AUC can be formulated as:

$$\text{AUC} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \int_0^1 \text{TPR}_u\left[\text{FPR}_u^{-1}(\xi)\right] \mathrm{d}\xi. \tag{2}$$

In the recommendation, AUC quantifies the overall ranking quality with consideration of all items in $\mathcal{I}$, and we can reformulate it to a pair-wise ranking form [15] as follows:

$$\text{AUC} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbf{Pr}_{i \sim \mathcal{I}_u^+, j \sim \mathcal{I}_u^-}[f_{u,i} > f_{u,j}], \tag{3}$$

where $\mathbf{Pr}_{i \sim \mathcal{I}_u^+, j \sim \mathcal{I}_u^-}[f_{u,i} > f_{u,j}]$ represents the probability that a positive item i is ranked higher than a negative item j for user u.

Recently, One-way Partial AUC (OPAUC) [7] is proposed to better measure Top-K recommendation quality. Different from AUC, OPAUC just focuses on the area with FPR $\leq \beta$, which is equivalent to just focusing on pair-wise ranking between positive items and highly scored negative items (with prediction scores in $[\eta_\beta, 1]$, where η_β satisfies $\mathbf{Pr}_{j \sim \mathcal{I}_u^-}[f_{u,j} \geq \eta_\beta] = \beta$). Formally,

$$\text{OPAUC}(\beta) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbf{Pr}_{i \sim \mathcal{I}_u^+, j \sim \mathcal{I}_u^-}[f_{u,i} > f_{u,j}, f_{u,j} \geq \eta_\beta]. \tag{4}$$

Based on the definition, we could write a non-parametric estimator for OPAUC(β) as follows:

$$\widehat{\text{OPAUC}(\beta)} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}_u^+} \sum_{j \in \mathcal{I}_u^-} \frac{\mathbb{I}[f_{u,i} > f_{u,j}] \cdot \mathbb{I}[f_{u,j} \geq \eta_\beta]}{n_u^+ \cdot n_u^-}, \tag{5}$$

where $\mathbb{I}(\cdot)$ denotes the indicator function, n_u^+ denotes the size of $\mathcal{I}_u^+$, and n_u^- denotes the size of $\mathcal{I}_u^-$.

4 WHEN LLPAUC MEETS WITH RECOMMENDER SYSTEM

In this paper, we introduce a novel metric called Lower-Left Partial AUC, which differs from OPAUC by imposing constraints on both FPR and TPR (*i.e.*, TPR$\leq \alpha$, FPR$\leq \beta$) as shown in Figure 1(b). By placing additional constraints on TPR, LLPAUC can more closely approach Top-K metrics and effectively address noisy user feedback issues. We next present the formal definition of LLPAUC and subsequently provide theoretical and empirical analyses to demonstrate its effectiveness in aligning with Top-K metrics.

• **LLPAUC Definition.** LLPAUC(α,β), as illustrated in Figure 1(b), is defined as *the area of the ROC space that lies below the ROC curve with TPR $\leq \alpha$ and FPR $\leq \beta$*. Similarly to OPAUC, for each user u, the constraint TPR$\leq \alpha$ implies only considering positive items with prediction scores in $[\eta_\alpha, 1]$, where η_α satisfies that $\mathbf{Pr}_{i \sim \mathcal{I}_u^+}[f_{u,i} \geq \eta_\alpha] = \alpha$. The constraint FPR$\leq \beta$ means considering only negative items with prediction scores in $[\eta_\beta, 1]$, where η_β satisfies that $\mathbf{Pr}_{j \sim \mathcal{I}_u^-}[f_{u,j} \geq \eta_\beta] = \beta$. These constraints will make LLPAUC focus on measuring the ranking quality between such highly scored positive items and negative items, and we can accordingly formulate LLPAUC(α,β) for model f as:

$$\text{LLPAUC}(\alpha, \beta) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbf{Pr}_{i \sim \mathcal{I}_u^+, j \sim \mathcal{I}_u^-}[f_{u,i} > f_{u,j}, f_{u,i} \geq \eta_\alpha, f_{u,j} \geq \eta_\beta]. \tag{6}$$

We can also formulate it in an empirical form as follows:

$$\widehat{\text{LLPAUC}(\alpha, \beta)} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}_u^+} \sum_{j \in \mathcal{I}_u^-} \frac{\mathbb{I}\left[f_{u,i} > f_{u,j}\right] \cdot \mathbb{I}\left[f_{u,i} \geq \eta_\alpha\right] \cdot \mathbb{I}\left[f_{u,j} \geq \eta_\beta\right]}{n_u^+ \cdot n_u^-}. \tag{7}$$

It is apparent that both AUC and OPAUC are special instances of our proposed LLPAUC metric. Specifically, we have AUC=LLPAUC(1,1) and OPAUC(β) = LLPAUC(1,β).

4.1 Theoretical Analysis

In this subsection, we present theoretical evidence that LLPAUC(α,β) is highly correlated with Top-K metrics such as Recall@K and Precision@K when α and β are appropriately set.

Theorem 1. *Suppose there are n^+ positive items and n^- negative items, where $n^+ > K$ and $n^- > K$. Ranking all items in descending order according to the prediction scores obtained from any model f, we have*

$$\frac{1}{n^+} \lfloor \mathcal{G}_{lower}(LLPAUC(\alpha, \beta)) \rfloor \leq Recall@K \leq \frac{1}{n^+} \lceil \mathcal{G}_{higher}(LLPAUC(\alpha, \beta)) \rceil, \tag{8}$$

$$\frac{1}{K} \lfloor \mathcal{G}_{lower}(LLPAUC(\alpha, \beta)) \rfloor \leq Precision@K \leq \frac{1}{K} \lceil \mathcal{G}_{higher}(LLPAUC(\alpha, \beta)) \rceil, \tag{9}$$

where $\alpha = \frac{K}{n^+}$, $\beta = \frac{K}{n^-}$, and

$$\begin{aligned} &\mathcal{G}_{lower}(LLPAUC(\alpha, \beta)) = K - \sqrt{K^2 - n^+ n^- \times LLPAUC(\alpha, \beta)}, \\ &\mathcal{G}_{higher}(LLPAUC(\alpha, \beta)) = \sqrt{n^+ n^- \times LLPAUC(\alpha, \beta)}. \end{aligned} \tag{10}$$

Theorem 2. *The bounds for Top-K metrics in Eq.* (8) *and Eq.* (9) *are tighter than the bounds obtained with OPAUC in Theorem 3 of [31].*

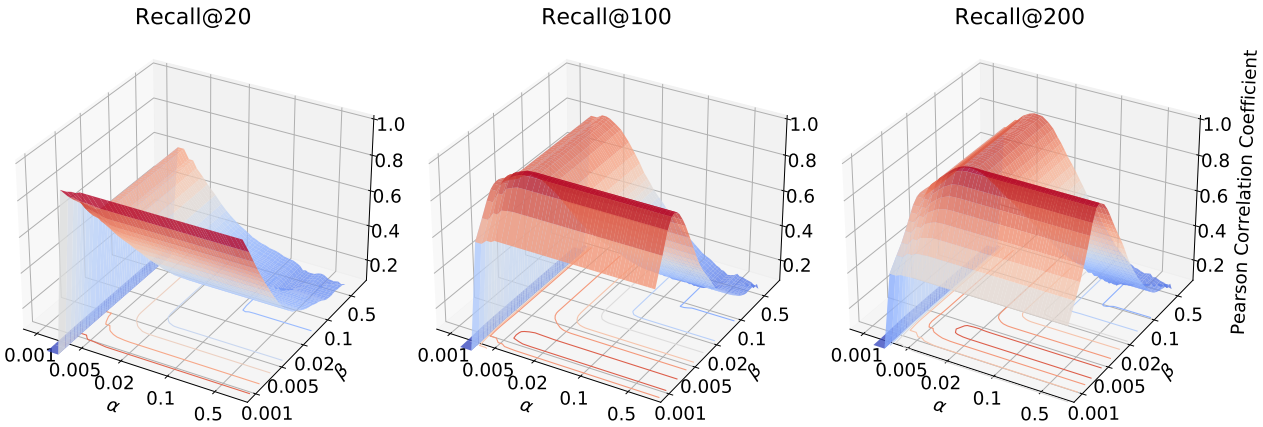

Figure 2: Pearson correlation coefficient between Recall@K and LLPAUC(α, β).

The proof of Theorem 1 and 2 can be found in Appendix A and B, respectively. Based on the two theorems, we conclude that:

- LLPAUC(α,β) exhibits a stronger correlation with Top-K metrics like Precision@K and Recall@K, when compared to OPAUC(β) and AUC. Therefore, optimizing LLPAUC is expected to yield superior performance in the Top-K metrics.
- In the derived bounds, both $\alpha = \frac{K}{n^+}$ and $\beta = \frac{K}{n^-}$ decrease as K decreases. This implies that while manipulating the value of K, adjustments to α and β should be made in order to maintain a robust correlation between LLPAUC and the corresponding Top-K metrics.

4.2 Empirical Analysis

We now provide empirical evidence to further substantiate the strong correlation between LLPAUC and Top-K metrics. We perform Monte Carlo sampling experiments via simulation. Specifically, we assume that there are n^+ positive items and n^- negative items, and take each possible permutation of all items to represent a possible ranking list. We randomly sample 10,000 permutations and calculate the Pearson correlation coefficient between LLPAUC(α, β) and Recall@K with different α, β, and K. It should be noted that the trend is consistent across simulations with different numbers of positive and negative samples (n^+ and n^-). Therefore, without loss of generality, we set $n^+ = 1000$ and $n^- = 50000$, where α and β are logarithmically scaled. It is worth noting that the correlations between Recall@K and OPAUC(β) (or AUC) can be observed by examining LLPAUC(1,β) (or LLPAUC(1,1)). From the Figure 2, we observe that:

(1) The maximum correlation coefficient is obtained when $\alpha < 1$ and $\beta < 1$, with a value exceeding 0.8. This observation provides empirical evidence supporting the proposition that LLPAUC(α, β) exhibits a stronger correlation with Top-K metrics compared to OPAUC and AUC metrics, thus validating Theorem 2.
(2) As K decreases, the point that corresponds to the maximum correlation coefficient shifts towards smaller values of α and β. This aligns with the conclusion drawn from the conditions $\alpha = \frac{K}{n^+}$ and $\beta = \frac{K}{n^-}$ in the bounds of Eq. (8), further reinforcing the validity of our Theorem 1.

Furthermore, we observe that using both α and β to regulate TPR or FPR could enhance the alignment of LLPAUC with the Top-K ranking. Additionally, utilizing α to regulate TPR can also increase the robustness against noise, which we next discuss.

• **LLPAUC Enhancing Robustness Against Noise.** As stated in [33], noise-positive interactions are harder to fit in the early training stage for the recommendation, which results in relatively larger losses (lower predicted score) of noise interactions. As aforementioned, the constraint TPR$\leq \alpha$ implies LLPAUC only considers positive items with prediction scores $f_{u,i} \geq \eta_\alpha$. In this way, lots of noise-positive interactions are filtered out, which makes LLPAUC enhance model robustness against noise.

5 METHOD

In this section, we first introduce the loss function that enables efficient optimization of LLPAUC. We then describe the learning algorithm and discuss its time complexity.

5.1 Loss Function

To optimize LLPAUC during model learning, it is necessary to further convert the LLPAUC(α,β) in Eq. (7) to a loss function that can be efficiently optimized. This involves transforming the non-differentiable and computationally expensive terms in Eq. (7), including the pair-wise ranking term ($\mathbb{I}\left[f_{u,i} > f_{u,j}\right]$) and TPR and FPR constraint terms ($\mathbb{I}\left[f_{u,i} \geq \eta_\alpha\right]$ and $\mathbb{I}\left[f_{u,j} \geq \eta_\beta\right]$), into low-complexity point-wise loss functions. To this end, we replace the pair-wise ranking term with a decouplable surrogate loss and design an *Average Top-K Trick* inspired by [30] to transform the constraint terms. Specifically, we follow the four steps to derive our loss:

• ***Step 1: replacing*** $\mathbb{I}\left[f_{u,i} > f_{u,j}\right]$ ***with surrogate loss function.*** The non-continuous and non-differentiable $\mathbb{I}[f_{u,i} > f_{u,j}]$ in Eq. (7) is also appeared in AUC and OPAUC formulation. To convert it, we adopt an approach similar to that used for AUC and OPAUC, which involves replacing it with a continuous surrogate loss $\ell(f_{u,i} - f_{u,j})$. Under the assumptions below, the surrogate $\ell(\cdot)$ is consistent for LLPAUC maximization [9].

Assumption 1. *We assume $\ell(\cdot)$ is a convex, differentiable and monotonically decreasing function when $\ell(\cdot) > 0$, and $\ell'(0) < 0$.*

Then, maximizing LLPAUC(α, β) in Eq. (7) is equivalent to minimizing the following loss:

$$\min_{\theta} \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}_u^+} \sum_{j\in\mathcal{I}_u^-} \frac{\ell(f_{u,i} - f_{u,j}) \cdot \mathbb{I}\left[f_{u,i} \geq \eta_\alpha\right] \cdot \mathbb{I}\left[f_{u,j} \geq \eta_\beta\right]}{n_u^+ \cdot n_u^-}. \tag{11}$$

• ***Step 2: decoupling pair-wise loss into point-wise loss.*** By setting $\ell(x) = (1-x)^2$, a square loss satisfying Assumption 1, we could decouple the total loss into positive and negative item components, resulting in a point-wise loss.

Lemma 1. *(Proof in Appendix C) With $\ell(x) = (1-x)^2$, the LLPAUC(α, β) optimization problem in Eq.* (11) *is equal to*

$$\min_{\theta,(a,b)\in[0,1]^2} \max_{\gamma\in[-1,1]} \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{I}_u^+} \frac{\ell_+(f_{u,i})\mathbb{I}\left[f_{u,i} \geq \eta_\alpha\right]}{n_u^+} + \sum_{j\in\mathcal{I}_u^-} \frac{\ell_-(f_{u,j})\mathbb{I}\left[f_{u,j} \geq \eta_\beta\right]}{n_u^-} - \gamma^2, \tag{12}$$

where a, b and γ are learnable parameters, $\ell_+(f_{u,i}) = (f_{u,i} - a)^2 - 2(1+\gamma)f_{u,i}$, and $\ell_-(f_{u,j}) = (f_{u,j} - b)^2 + 2(1+\gamma)f_{u,j}$.

• ***Step 3: reformulating TPR and FPR constraint terms using an average top-K trick.*** The constraint terms $\mathbb{I}\left[f_{u,i} \geq \eta_\alpha\right]$ and $\mathbb{I}\left[f_{u,j} \geq \eta_\beta\right]$ require selecting highly scored positive and negative items, which renders the loss in Eq. (12) still non-differentiable and difficult to optimize. Fortunately, under certain conditions, $\ell_+(f_{u,i})$ is a monotonic decreasing function *w.r.t* $f_{u,i}$ and $\ell_-(f_{u,j})$ is a monotonic increasing function *w.r.t* $f_{u,j}$, as proven in Appendix D. Then, we could make the item selection process differentiable using the average Top-K reformulation trick introduced below.

Lemma 2. *(Proof in Appendix E) Suppose $\ell_+(f_{u,i})$ is monotonic decreasing w.r.t. $f_{u,i}$ and $\ell_-(f_{u,j})$ is monotonic increasing w.r.t. $f_{u,j}$, then we have*

$$\sum_{i\in\mathcal{I}_u^+} \left[\ell_+(f_{u,i}) \cdot \mathbb{I}[f_{u,i} \geq \eta_\alpha]\right] = \max_{s^+\in\mathbb{R}} \sum_{i\in\mathcal{I}_u^+} \left[-\alpha s^+ - [-\ell_+(f_{u,i}) - s^+]_+\right],$$

$$\sum_{j\in\mathcal{I}_u^-} \left[\ell_-(f_{u,j}) \cdot \mathbb{I}[f_{u,j} \geq \eta_\beta]\right] = \min_{s^-\in\mathbb{R}} \sum_{j\in\mathcal{I}_u^-} \left[\beta s^- + [\ell_-(f_{u,j}) - s^-]_+\right],$$

where s^+ and s^- are learnable parameters, and $[x]_+ = max(0, x)$.

By leveraging the average Top-K reformulation trick presented in the lemma, we can reformulate the LLPAUC optimization problem in Eq. (12) as follows:

$$\min_{\theta,(a,b)\in[0,1]^2} \max_{\gamma\in\Omega_\gamma} \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} \{\max_{s^+\in\mathbb{R}} \sum_{i\in\mathcal{I}_u^+} \frac{-\alpha s^+ - [-\ell_+(f_{u,i}) - s^+]_+}{n_u^+} + \min_{s^-\in\mathbb{R}} \sum_{j\in\mathcal{I}_u^-} \frac{\beta s^- + [\ell_-(f_{u,j}) - s^-]_+}{n_u^-} - \gamma^2\}, \tag{13}$$

where $\Omega_\gamma = [\max(-a, b-1), 1]$.

• ***Step 4: swapping min-max operations.*** Solving Eq. (13) directly is challenging since it involves a complicated min-max-min sub-problem (it also contains a manageable min-max-max sub-problem). However, as done in [30], we could swap the order of the latter $\max_\gamma$ and $\min_{s^-}$ operations for the min-max-min sub-problem after applying two preprocessing steps: 1) replacing the non-smooth function $[\cdot]_+$ with the softplus function [13] and 2) adding an L_2 regularizer to make Eq. (13) strongly-concave *w.r.t.* γ. Finally, according to the min-max theorem [1], we could merge the consecutive min (or max) operations, converting the overall optimization problem into a min-max form. Formally, Eq. (13) could be reformulated as (see Appendix F for the proof):

$$\min_{\{\theta,(a,b)\in[0,1]^2, s^-\in\mathbb{R}\}} \max_{\{\gamma\in\Omega_\gamma, s^+\in\mathbb{R}\}} \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} \{ \sum_{i\in\mathcal{I}_u^+} \frac{-\alpha s^+ - r_\kappa(-\ell_+(f_{u,i}) - s^+)}{n_u^+} + \sum_{j\in\mathcal{I}_u^-} \frac{\beta s^- + r_\kappa(\ell_-(f_{u,j}) - s^-)}{n_u^-} - (w+1)\gamma^2\}, \tag{14}$$

where $\Omega_\gamma = [\max(-a, b-1), 1]$, and r_κ denotes the softplus function. Formally, $r_\kappa(x) = \frac{1}{\kappa}\log(1 + \exp(\kappa \cdot x))$, where κ is a hyper-parameter. It is easy to show that $r_\kappa(x) \overset{\kappa\to\infty}{\longrightarrow} [x]_+$, which leads to asymptotically unbiased optimization.

Remark. *Our final loss function in Eq.* (14) *is similar to the one proposed in [30]. However, it is important to emphasize that the primary contribution of our work is not the introduction of a completely new optimization scheme. Rather, our main contribution lies in extending existing optimization methods to align with our novel LLPAUC metric while addressing challenges associated with the coexistence of minima and maxima optimizations.*

• **Learning Algorithm and Time Complexity Analysis.** To solve the above minimax optimization in Eq. (14), we employ a stochastic gradient descent ascent (SGDA) method. The detailed algorithm can be found in Appendix G. Based on it, we derive that the total per-iteration complexity of our method is the same as classical loss functions such as BPR [29] and BCE [6]. The detailed derivation process can be found in Appendix G.

6 EXPERIMENTS

In this section, we conduct a series of experiments on three datasets to evaluate the effectiveness and robustness of our proposed optimization metric LLPAUC along with the loss function. Due to space limitations, additional experimental results, including some supplemental results during the rebuttal stage, can be found in the ArXiv version of the paper.

6.1 Experiments Setting

Dateset. We conduct experiments on three real-world datasets: Adressa, Yelp, and Amazon-book. Our dataset selection was made intentionally to cover a broad range of recommendation scenarios and accommodate different dataset sizes. **Adressa** is a news reading dataset from Adressavisen [14], where the clicks with dwell time < 10s are thought of as noisy interactions [33]. **Yelp**[1] is a restaurant recommendation dataset with user ratings from one to five. **Amazon-book**[2] is from the Amazon-Review [16] datasets, containing user interaction ratings with extensive books. A rating score below 3 on Yelp and Amazon-book is regarded as a noisy interaction.

[1]https://www.yelp.com/dataset/challenge.
[2]https://jmcauley.ucsd.edu/data/amazon/.

Table 1: Performance comparison on three datasets with clean training. The best results are highlighted in bold.

	Method	Adressa		Yelp		Amazon	
		Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
MF	BCE	0.1573±0.0251	0.0793±0.0181	0.0814±0.0004	0.0448±0.0005	0.0663±0.0006	0.0363±0.0002
	BPR	0.1800±0.0204	0.0991±0.0144	0.0647±0.0005	0.0358±0.0002	0.0695±0.0001	0.0384±0.0007
	SCE	0.2001±0.0031	0.1057±0.0015	0.0762±0.0007	0.0425±0.0003	0.0894±0.0012	0.0507±0.0009
	CCL	0.1956±0.0110	0.0911±0.0028	0.0842±0.0002	0.0486±0.0000	0.0944±0.0001	0.0551±0.0008
	DNS(M, N)	0.1877±0.0025	0.0965±0.0010	0.0856±0.0005	0.0489±0.0002	0.1012±0.0006	0.0580±0.0003
	Softmax_v(ρ, N)	0.1849±0.0105	0.0949±0.0088	0.0824±0.0008	0.0470±0.0004	0.1024±0.0001	0.0592±0.0001
	PAUCI(OPAUC)	0.2021±0.0014	0.1086±0.0007	0.0821±0.0004	0.0479±0.0003	0.0991±0.0001	0.0549±0.0002
	LLPAUC	**0.2166±0.0022**	**0.1214±0.0009**	**0.0884±0.0005**	**0.0505±0.0003**	**0.1076±0.0007**	**0.0612±0.0004**
	Method	Adressa		Yelp		Amazon	
		Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
LightGCN	BCE	0.1897±0.0004	0.0935±0.0002	0.0905±0.0003	0.0517±0.0004	0.1149±0.0003	0.0660±0.0003
	BPR	0.1737 ±0.0006	0.0923 ±0.0004	0.0802 ±0.0005	0.0453 ±0.0003	0.0922±0.0002	0.0520±0.0001
	SCE	0.1729 ±0.0008	0.0960 ±0.0007	0.0890 ±0.0005	0.0506 ±0.0004	0.1115±0.0004	0.0640±0.0002
	CCL	0.1926±0.0008	0.1014 ±0.0009	0.0915 ±0.0006	0.0528 ±0.0005	0.1007±0.0000	0.0614±0.0001
	DNS(M, N)	0.1830±0.0035	0.0952 ±0.0006	0.0962 ±0.0003	0.0550 ±0.0002	0.1056±0.0004	0.0597±0.0002
	Softmax_v(ρ, N)	0.1923±0.0107	0.1056±0.0117	0.0975±0.0001	0.0567±0.0000	0.1128±0.0007	**0.0724±0.0006**
	LLPAUC	**0.2311 ±0.0004**	**0.1312 ±0.0002**	**0.1002 ±0.0003**	**0.0573 ±0.0004**	**0.1201±0.0003**	0.0684±0.0003

Training Settings. We employed two training settings, **clean training** and **noise training**, to verify the effectiveness and robustness of our proposed loss. Following [34], clean training filters out noisy user interactions and divides the remaining data into separate training, validation, and testing sets. In contrast, noise training retains the same testing set as clean training yet adds noisy interactions to the training and validation sets. Note that we keep the numbers of noisy training and validation interactions on a similar scale as clean training for a fair comparison.

Evaluation Protocols. Following existing studies [18, 29], we adopt the full-ranking evaluation setting, where we calculate the metrics using all negative samples. Meanwhile, we utilize two popular metrics to evaluate models, Recall@K and NDCG@K with $K = 20$, where higher scores indicate better performance.

Baselines. We compare our LLPAUC surrogate loss function with the following representative recommender losses. 1) **Bayesian Personalized Ranking (BPR)** [29] loss is a pair-wise loss function, which optimizes the AUC metric. 2) **Binary Cross-Entropy (BCE)** [18] loss optimizes accuracy metric. 3) **Softmax Cross-Entropy (SCE)** [6] loss is widely used for classification problems and maximizes likelihood estimation of classification. 4) **DNS(M, N) and Softmax_v(ρ, N)** are advanced OPAUC-based loss functions for recommendation system. 5) **PAUCI(OPAUC)** [30] is also an advanced OPAUC-based loss function. For clean training, recent 5) **Cosine Contrastive Loss (CCL)** [21] is included in the comparison. For noise training, we add strong denoising baselines 6) **RCE** and **TCE** [33] for comparison.

In Appendix H, we also compare LLPAUC loss function with advanced learning-to-rank (LTR) methods.

Parameter Settings. For a fair comparison, we choose two representative recommender models, Matrix Factorization (MF) and graph neural network model LightGCN [17], as the backbones for all loss functions. All the models are optimized by the Adam optimizer with a learning rate of 0.001 and a batch size of 128. In the training process, we adopt widely used negative sampling trick [21] to improve the training efficiency. The number of negative items for each positive item is set to 100. For the proposed LLPAUC surrogate loss function, we tune α and β within the ranges of {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9} and { 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 0.7, 0.9}. All hyperparameter searches are done relying on the validation set. We reported results (mean and standard deviation) based on three repeats with distinct random seeds.

6.2 Main Results

Clean Training. Table 1 shows the performance comparison between the LLPAUC surrogate loss function with various baselines under the clean training setting with MF and LightGCN backbones. Several key observations can be made from the results: 1) LLPAUC consistently achieves the best performance in most cases across all three datasets with different backbones, outperforming the other loss functions significantly. This demonstrates that LLPAUC strongly correlates with Top-K metrics compared to other optimization metrics, which is consistent with our previous theoretical analysis and independent of the dataset and the backbones. 2) The performance of BPR is noticeably inferior to that of DNS(M, N) and Softmax_v(ρ, N) on all datasets with different backbones. Drawing upon the prior knowledge that OPAUC has a stronger correlation with Top-K compared to AUC, we can infer that optimization metrics closely tied to Top-K yield superior performance.

Table 2: Performance comparison on three datasets with noise training. The best results are highlighted in bold.

	Method	Adressa		Yelp		Amazon	
		Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
MF	BCE	0.1551±0.0025	0.0762±0.0007	0.0799±0.0014	0.0438±0.0009	0.0911±0.0009	0.0515 ±0.0009
	BPR	0.1666±0.0215	0.0880±0.0139	0.0626±0.0014	0.0341±0.0009	0.0663±0.0008	0.0363±0.0006
	SCE	0.1938±0.0010	0.1062±0.0007	0.0738±0.0003	0.0406±0.0009	0.0840±0.0010	0.0470±0.0011
	TCE	0.1465±0.0022	0.0862±0.0007	0.0826±0.0008	0.0456±0.0005	0.0906±0.0018	0.0514±0.0011
	RCE	0.1617±0.0329	0.0819±0.0221	0.0818±0.0009	0.0452±0.0005	0.0965±0.0017	0.0549±0.0015
	DNS(M, N)	0.1802±0.0125	0.0847±0.0097	0.0844±0.0016	0.0477±0.0008	0.0966±0.0003	0.0543±0.0003
	Softmax_v(ρ, N)	0.1801±0.0086	0.0922±0.0054	0.0816±0.00014	0.0452±0.0005	0.0954±0.0002	0.0536±0.0001
	LLPAUC	**0.2127±0.0014**	**0.1189±0.0009**	**0.0847±0.0007**	**0.0481±0.0001**	**0.0998±0.0008**	**0.0566±0.0006**
	Method	Adressa		Yelp		Amazon	
		Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
LightGCN	BCE	0.1844 ±0.0005	0.0874 ±0.0002	0.0888 ±0.0003	0.0497 ±0.0001	0.1095±0.0003	0.0620±0.0001
	BPR	0.1661 ±0.0007	0.0914 ±0.0006	0.0800 ±0.0005	0.0448 ±0.0002	0.0884±0.0005	0.0492±0.0002
	SCE	0.1732 ±0.0008	0.0936 ±0.0005	0.0916 ±0.0003	0.0514 ±0.0003	0.1068±0.0003	0.0604±0.0002
	TCE	0.2184±0.0005	0.1187±0.0005	0.0923 ±0.0004	0.0522 ±0.0003	0.1085 ±0.0004	0.0611 ±0.0002
	RCE	0.2204 ±0.0007	0.1219 ±0.0007	0.0941 ±0.0006	0.0536 ±0.0008	0.1126 ±0.0004	0.0639 ±0.0005
	DNS(M, N)	0.1701±0.0017	0.0889 ±0.0011	0.0948 ±0.0002	0.0536 ±0.0001	0.1012±0.0002	0.0570±0.0001
	Softmax_v(ρ, N)	0.1815±0.0047	0.0939±0.0084	0.0957±0.0002	0.0549±0.0002	0.1076±0.0003	**0.0682±0.0004**
	LLPAUC	**0.2228±0.0006**	**0.1231 ±0.0005**	**0.0981 ±0.0007**	**0.0558 ±0.0004**	**0.1165±0.0007**	0.0655±0.0005

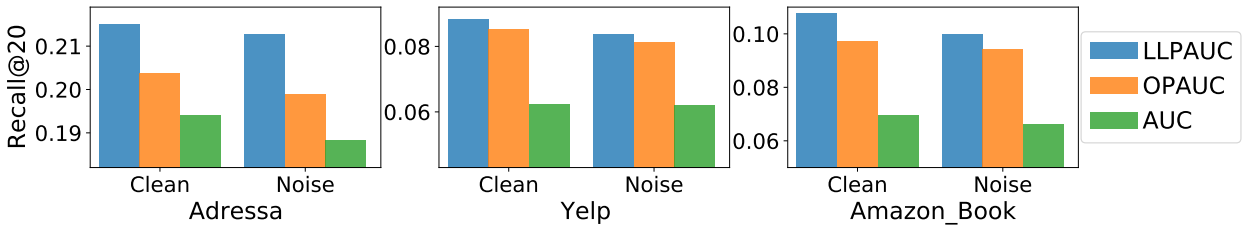

Figure 3: Ablation studies among different AUC metrics with clean training and noise training.

This finding validates our motivation for proposing LLPAUC. 3) In contrast to BPR and BCE, other losses can implicitly pay more attention to hard negative items, resulting in their superior performance. In LLPAUC, we can similarly adjust the attention to hard negative items by varying the β parameter. 4) LightGCN outperforms MF in most cases, highlighting its superior strength as a representative graph neural network backbone.

Noise Training. In real-world recommender systems, the user interactions collected through implicit feedback often contain natural false-positive interactions. To evaluate the robustness of LLPAUC, we compare LLPAUC with other loss functions under the noise training setting in Table 2. Notably, we have the following observation: 1) Across all three datasets, the model performance under the noise training setting drops for all loss functions, when compared to the clean training setting. This observation makes sense because it is more challenging to predict user preference from noisy interactions. 2) Denoising baselines like RCE and TCE achieve better performance than other baselines across all datasets, highlighting the importance of noise removal. 3) LLPAUC surpasses all baselines in most cases on all datasets, verifying the strong robustness against natural noises. The robustness of LLPAUC stems from its emphasis on higher-ranked positive items, which can be adjusted by hyperparameter α.

6.3 In-depth Analysis

Ablation Study. We next conduct ablation studies to assess the significance of the TPR and FPR constraints in LLPAUC(α, β). Note that restriction on the upper bound of TPR and FPR represents the emphasis on high-ranked positive and negative items in LLPAUC, respectively. As shown in Eq. (6), OPAUC(β) = LLPAUC(1, β) and AUC = LLPAUC(1, 1). Based on it, we obtain ablation loss functions of AUC and OPAUC(β) by setting α and β in Eq. (14). The results of ablation studies are summarized in Figure 3, where we can observe that: 1) Under clean training, LLPAUC outperforms OPAUC, and OPAUC perform better than AUC. This verifies both

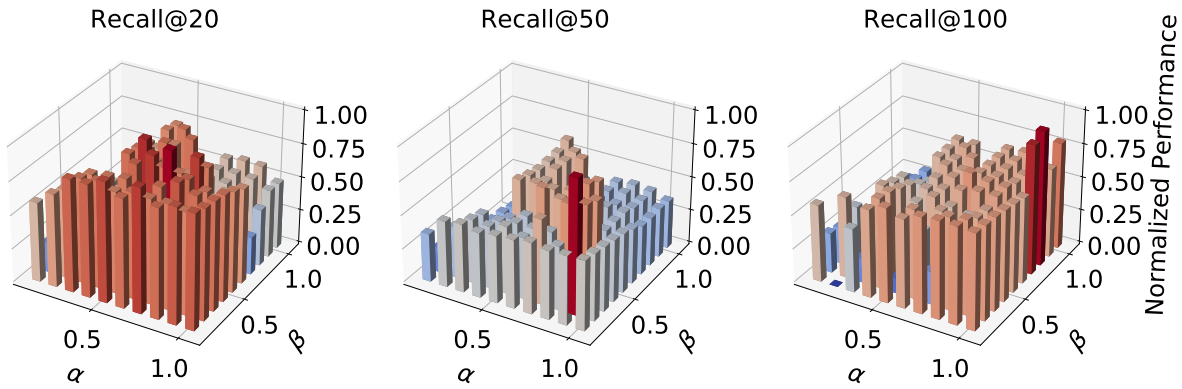

Figure 4: Normalized Recall@K on Adressa dataset under clean training for K=20, 50 and 100.

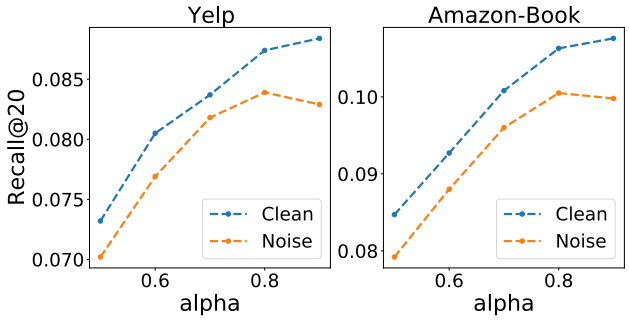

Figure 5: Given a fix β, the hyperparameter analysis of α in LLPAUC(α, β) on different datasets under clean training setting and noise training setting.

emphases on high-ranked positive items and high-ranked negative items strengthen the correlation between LLPAUC and Top-K metrics. 2) When exposed to noisy interactions, LLPAUC demonstrates relatively minor performance degradation compared to OPAUC and AUC, showcasing its robustness against noise. This is attributed to the emphasis on high-ranked positive items and avoidance of noise samples with low ranks in LLPAUC.

Hyperparameter Analysis. To verify the impact of the constraints of LLPAUC, we conduct the grid search experiments on hyperparameters α and β and present the corresponding Recall@K performance in Figure 4. To facilitate a better comparison, we report the normalized Recall@K metrics, where Normalized_Recall = $\frac{\text{Recall}-\text{Min\_Recall}}{\text{Max\_Recall}-\text{Min\_Recall}}$. From the figure, we observe that: 1) The maximum performance is obtained with $\alpha < 1$ and $\beta < 1$. Recall that AUC=LLPAUC(1,1) and OPAUC=LLPAUC(1,β). Hence, this demonstrates both restrictions of α and β of LLPAUC enhance its correlation with the Top-K metric, which is consistent with our Theorem 2 and empirical analysis in Section 4.2. 2) As K in Recall@K decreases, we should shift towards a smaller value of α and β to achieve the best performance, empirically corroborating the bound conditions in our Theorem 1. This means we could emphasize different Top-K performances for different K by adjusting α and β in LLPAUC.

Analysis of Robustness. In this subsection, we conduct experiments to analyze the impact of hyperparameter α on the robustness of the model. Given a fix β, Figure 5 shows how the LLPAUC model's performance changes *w.r.t* α under clean training and noise training setting. Since the natural noise in the Adressa dataset is relatively weak, we do not include it in our comparison. From the figure, we observe that: 1) Since the noisy interactions impede the model's ability to learn the true interests of users, the performance in the noise training setting consistently falls below that of the clean training setting. This is consistent with our observation in Table 1. 2) Given a fix β, the maximum Recall@20 performance of LLPAUC is achieved with $\alpha = 0.9$ under clean training settings, and $\alpha = 0.8$ under noisy training settings. This means under the noise training setting, we should choose smaller α to enhance the robustness. Since α constrains TPR in LLPAUC as stated in Eq. (6), we conclude that the emphasis on high-ranked positive items could enhance the model robustness.

7 CONCLUSION AND FUTURE WORK

In this work, we presented a novel optimization metric for recommender systems, LLPAUC, to alleviate the dilemma of balancing effectiveness and computational efficiency in previous optimization metrics. In particular, LLPAUC is efficient like AUC while strongly correlating with Top-K ranking metrics, leading to superior Top-K recommendation performance. To optimize LLPAUC, we developed a point-wise loss function and conducted experiments on three datasets, demonstrating its efficiency, effectiveness, and robustness under clean and noise settings.

Future work could shed light on the following limitations of our work: 1) Only focusing on high-ranked positive samples like LLPAUC is not sufficient to fully mitigate the impact of natural noise. 2) The TPR and FPR constraint terms in LLPAUC could be more efficiently reformulated. 3) The relationship between LLPAUC and other partial AUC metrics needs to be analyzed more comprehensively.

ACKNOWLEDGMENTS

We would like to express our gratitude to Professor Jiaxin Mao for his guidance during the rebuttal phase of this paper. This work is supported by the National Natural Science Foundation of China (62272437 and 62121002) and the CCCD Key Lab of Ministry of Culture and Tourism.

REFERENCES

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

[2] Andrew P Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern recognition* 30, 7 (1997), 1145–1159.

[3] Amar Budhiraja, Gaurush Hiranandani, Navya Yarrabelly, Ayush Choure, Oluwasanmi Koyejo, and Prateek Jain. 2020. Rich-Item Recommendations for Rich-Users via GCNN: Exploiting Dynamic and Static Side Information. *CoRR* abs/2001.10495 (2020).

[4] Christopher J. C. Burges, Robert Ragno, and Quoc Viet Le. 2006. Learning to Rank with Nonsmooth Cost Functions. In *NIPS*. MIT Press, 193–200.

[5] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and Debias in Recommender System: A Survey and Future Directions. *CoRR* abs/2010.03240 (2020).

[6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *RecSys*. ACM, 191–198.

[7] Lori E. Dodd and Margaret S. Pepe. 2003. Partial AUC Estimation and Regression. *Biometrics* 59, 3 (2003), 614–623.

[8] Yanbo Fan, Siwei Lyu, Yiming Ying, and Bao-Gang Hu. 2017. Learning with Average Top-k Loss. In *NIPS*. 497–505.

[9] Wei Gao and Zhi-Hua Zhou. 2015. On the Consistency of AUC Pairwise Optimization. In *IJCAI*. AAAI Press, 939–945.

[10] Yuan Gao, Xiang Wang, Xiangnan He, Huamin Feng, and Yong-Dong Zhang. 2023. Rumor detection with self-supervised learning on texts and social graph. *Frontiers Comput. Sci.* 17, 4 (2023), 174611.

[11] Yuan Gao, Xiang Wang, Xiangnan He, Zhenguang Liu, Huamin Feng, and Yongdong Zhang. 2023. Addressing Heterophily in Graph Anomaly Detection: A Perspective of Graph Spectrum. In *WWW*. ACM, 1528–1538.

[12] Yuan Gao, Xiang Wang, Xiangnan He, Zhenguang Liu, Huamin Feng, and Yongdong Zhang. 2023. Alleviating Structural Distribution Shift in Graph Anomaly Detection. In *WSDM*. ACM, 357–365.

[13] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep Sparse Rectifier Neural Networks. In *AISTATS (JMLR Proceedings, Vol. 15)*. JMLR.org, 315–323.

[14] Jon Atle Gulla, Lemei Zhang, Peng Liu, Özlem Özgöbek, and Xiaomeng Su. 2017. The Adressa dataset for news recommendation. In *WI*. ACM, 1042–1048.

[15] J.A. Hanley and Barbara Mcneil. 1982. The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve. *Radiology* 143 (05 1982), 29–36. https://doi.org/10.1148/radiology.143.1.7063747

[16] Ruining He and Julian J. McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*. ACM, 507–517.

[17] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*. ACM, 639–648.

[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. ACM, 173–182.

[19] Gaurush Hiranandani, Warut Vijitbenjaronk, Sanmi Koyejo, and Prateek Jain. 2020. Optimization and Analysis of the pAp@k Metric for Recommender Systems. In *ICML (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 4260–4270.

[20] Abhishek Kumar, Harikrishna Narasimhan, and Andrew Cotter. 2021. Implicit rate-constrained optimization of non-decomposable objectives. In *ICML (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 5861–5871.

[21] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. 2021. SimpleX: A Simple and Strong Baseline for Collaborative Filtering. In *CIKM*. ACM, 1243–1252.

[22] Donna Katzman McClish. 1989. Analyzing a Portion of the ROC Curve. *Medical Decision Making* 9, 3 (1989), 190–195. https://doi.org/10.1177/0272989X8900900307 PMID: 2668680.

[23] Harikrishna Narasimhan and Shivani Agarwal. 2013. A Structural SVM Based Approach for Optimizing Partial AUC. In *ICML (1) (JMLR Workshop and Conference Proceedings, Vol. 28)*. JMLR.org, 516–524.

[24] Harikrishna Narasimhan and Shivani Agarwal. 2013. $\text{SVM}_{\text{pAUC}}^{\text{tight}}$: a new support vector method for optimizing partial AUC based on a tight convex upper bound. In *KDD*. ACM, 167–175.

[25] Harikrishna Narasimhan and Shivani Agarwal. 2017. Support Vector Algorithms for Optimizing the Partial Area under the ROC Curve. *Neural Computation* 29, 7 (07 2017), 1919–1963.

[26] Hang Pan, Jiawei Chen, Fuli Feng, Wentao Shi, Junkang Wu, and Xiangnan He. 2023. Discriminative-Invariant Representation Learning for Unbiased Recommendation. In *IJCAI*. ijcai.org, 2270–2278.

[27] Yash Patel, Giorgos Tolias, and Jirí Matas. 2022. Recall@k Surrogate Loss with Large Batches and Similarity Mixup. In *CVPR*. IEEE, 7492–7501.

[28] Zi-Hao Qiu, Quanqi Hu, Yongjian Zhong, Lijun Zhang, and Tianbao Yang. 2022. Large-scale Stochastic Optimization of NDCG Surrogates for Deep Learning with Provable Convergence. In *ICML (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 18122–18152.

[29] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. AUAI Press, 452–461.

[30] Huiyang Shao, Qianqian Xu, Zhiyong Yang, Shilong Bao, and Qingming Huang. 2022. Asymptotically Unbiased Instance-wise Regularized Partial AUC Optimization: Theory and Algorithm. In *NeurIPS*.

[31] Wentao Shi, Jiawei Chen, Fuli Feng, Jizhi Zhang, Junkang Wu, Chongming Gao, and Xiangnan He. 2023. On the Theories Behind Hard Negative Sampling for Recommendation. In *WWW*. ACM, 812–822.

[32] Yongxiang Tang, Wentao Bai, Guilin Li, Xialong Liu, and Yu Zhang. 2022. CROLoss: Towards a Customizable Loss for Retrieval Models in Recommender Systems. In *CIKM*. ACM, 1916–1924.

[33] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising Implicit Feedback for Recommendation. In *WSDM*. ACM, 373–381.

[34] Wenjie Wang, Yiyan Xu, Fuli Feng, Xinyu Lin, Xiangnan He, and Tat-Seng Chua. 2023. Diffusion Recommender Model. *CoRR* abs/2304.04971 (2023).

[35] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The LambdaLoss Framework for Ranking Metric Optimization. In *CIKM*. ACM, 1313–1322.

[36] Le Wu, Xiangnan He, Xiang Wang, Kun Zhang, and Meng Wang. 2023. A Survey on Accuracy-Oriented Neural Recommendation: From Collaborative Filtering to Information-Rich Recommendation. *IEEE Trans. Knowl. Data Eng.* 35, 5 (2023), 4425–4445.

[37] Hanfang Yang, Kun Lu, Xiang Lyu, and Feifang Hu. 2019. Two-way partial AUC and its properties. *Statistical Methods in Medical Research* 28, 1 (2019), 184–195.

[38] Jheng-Hong Yang, Chih-Ming Chen, Chuan-Ju Wang, and Ming-Feng Tsai. 2018. HOP-rec: high-order proximity for implicit recommendation. In *Proceedings of the 12th ACM conference on recommender systems*. 140–144.

[39] Zhiyong Yang, Qianqian Xu, Shilong Bao, Yuan He, Xiaochun Cao, and Qingming Huang. 2021. When All We Need is a Piece of the Pie: A Generic Framework for Optimizing Two-way Partial AUC. In *ICML (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 11820–11829.

[40] Zhiyong Yang, Qianqian Xu, Shilong Bao, Yuan He, Xiaochun Cao, and Qingming Huang. 2023. Optimizing Two-Way Partial AUC With an End-to-End Framework. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 8 (2023), 10228–10246.

[41] Yao Yao, Qihang Lin, and Tianbao Yang. 2022. Large-scale Optimization of Partial AUC in a Range of False Positive Rates. In *NeurIPS*.

[42] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *SIGIR*. ACM, 785–788.

[43] Dixian Zhu, Gang Li, Bokun Wang, Xiaodong Wu, and Tianbao Yang. 2022. When AUC meets DRO: Optimizing Partial AUC for Deep Learning with Non-Convex Convergence Guarantee. In *ICML (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 27548–27573.

A PROOF OF THEOREM 1

Proof. For any given model f, we suppose there are $i (i < K)$ positive items among the Top-K items ranked according to f. Then we have Recall@$K = i/n^+$. Under this condition, easily, we can find out the case which has the maximum value of LLPAUC(α, β), where $\alpha = \frac{K}{n^+}$ and $\beta = \frac{K}{n^-}$:

$$\underbrace{+\cdots+}_{i}\ \underbrace{-\cdots-}_{K-i}\Big|\underbrace{+\cdots+}_{K-i}\ \underbrace{-\cdots-}_{i}.$$

Hence, as stated in Eq. (7), the maximum value of LLPAUC(α, β) is $\frac{-i^2+2Ki}{n^+n^-}$. Given this, we can deduce the maximum value of Recall@K when LLPAUC(α, β) takes a certain value. Note that i can only be integers, we derive that:

$$\frac{1}{n^+}\left\lfloor K - \sqrt{K^2 - n^+n^- \times \text{LLPAUC}(\alpha, \beta)} \right\rfloor \leq \text{Recall@}K.$$

Similarly, the case that has the minimum value of LLPAUC(α, β) is :

$$\underbrace{-\cdots-}_{K-i}\ \underbrace{+\cdots+}_{i}\Big|\underbrace{-\cdots-}_{i}\ \underbrace{+\cdots+}_{K-i}.$$

Based on Eq. (7), the minimum value of LLAUC(α, β) is $\frac{i^2}{n^+n^-}$. Similarly, we derive the minimum value of Recall@K when LLPAUC(α, β)

takes a certain value:

$$\text{Recall@}K \le \frac{1}{n^+}\left\lceil \sqrt{n^+n^- \times \text{LLPAUC}(\alpha,\beta)} \right\rceil.$$

These complete the proof of Eq. (8). Noticing that for a given permutation, $\text{Precision@}K = \frac{n^+}{K} \cdot \text{Recall@}K$, where $\frac{n^+}{K}$ is a constant. Hence, we can easily derive the Eq. (9). □

B PROOF OF THEOREM 2

Reminder of Theorem 2 The bounds for Top-K metrics in Eq. (8) and Eq. (9) are tighter than the bounds obtained with OPAUC in Theorem 3 of [31].

Proof. Note that the bounds obtained with OPAUC(β) in [31] is:

$$\frac{1}{n^+}\lfloor \mathcal{H}_{lower}(\text{OPAUC}(\beta))\rfloor \le \text{Recall@}K \le \frac{1}{n^+}\lceil \mathcal{H}_{higher}(\text{OPAUC}(\beta))\rceil,, \quad (15)$$

$$\frac{1}{K}\lfloor \mathcal{H}_{lower}(\text{OPAUC}(\beta))\rfloor \le \text{Precision@}K \le \frac{1}{K}\lceil \mathcal{H}_{higher}(\text{OPAUC}(\beta))\rceil, \quad (16)$$

where $\beta = \frac{K}{n^-}$ and

$$\begin{aligned}\mathcal{H}_{lower}(\text{OPAUC}(\beta)) &= \frac{n^+ + K - \sqrt{(n^+ + K)^2 - 4n^+n^- \times \text{OPAUC}(\beta)}}{2},\\ \mathcal{H}_{higher}(\text{OPAUC}(\beta)) &= \sqrt{n^+n^- \times \text{OPAUC}(\beta)}.\end{aligned} \quad (17)$$

Without loss of generality, we consider the bounds of Recall@K first. To prove that Eq. (8) is a tighter bound than Eq. (15), we need prove that $\mathcal{H}_{lower}(\text{OPAUC}(\beta)) \le \mathcal{G}_{lower}(\text{LLPAUC}(\alpha,\beta))$ and $\mathcal{H}_{higher}(\text{OPAUC}(\beta)) \ge \mathcal{G}_{higher}(\text{LLPAUC}(\alpha,\beta))$.

Step 1: Proof of $\mathcal{H}_{higher}(\textbf{OPAUC}(\beta)) \ge \mathcal{G}_{higher}(\textbf{LLPAUC}(\alpha,\beta))$ For any ranking list ranked by model f, we calculate LLPAUC(α,β) and OPAUC(β) as following:

$$\text{LLPAUC}(\alpha,\beta) = \sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}_u^-}\frac{\mathbb{I}\left[f_{u,i} > f_{u,j}\right]\cdot\mathbb{I}\left[f_{u,i}\ge\eta_\alpha\right]\cdot\mathbb{I}\left[f_{u,j}\ge\eta_\beta\right]}{n^+\cdot n^-},$$

$$\text{OPAUC}(\beta) = \sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}_u^-}\frac{\mathbb{I}\left[f_{u,i} > f_{u,j}\right]\cdot\mathbb{I}\left[f_{u,j}\ge\eta_\beta\right]}{n^+\cdot n^-}.$$

where

$$\eta_\alpha = \text{argmin}_{\eta\in\mathbb{R}}\left[\mathbb{E}_{i\sim\mathcal{I}_u^+}[\mathbb{I}(f_{u,i}\ge\eta)] = \alpha\right],$$

and

$$\eta_\beta = \text{argmin}_{\eta\in\mathbb{R}}\left[\mathbb{E}_{j\sim\mathcal{I}_u^-}[\mathbb{I}(f_{u,j}\ge\eta)] = \beta\right].$$

When $\alpha = \frac{K}{n^+}$ and $\beta = \frac{K}{n^-}$, LLPAUC(α,β) and OPAUC(β) can be reformulated as:

$$\text{LLPAUC}(\alpha,\beta) = \sum_{i=1}^{K}\sum_{j=1}^{K}\frac{\mathbb{I}\left[f_{u,[i]} > f_{u,[j]}\right]}{n^+n^-},$$

$$\text{OPAUC}(\beta) = \sum_{i=1}^{n_u^+}\sum_{j=1}^{K}\frac{\mathbb{I}\left[f_{u,i} > f_{u,[j]}\right]}{n^+n^-},$$

where $f_{u,[i]}$ denotes the i-th largest score among positive items and $f_{u,[j]}$ denotes the j-th largest score among negative items. This means LLPAUC(α,β) only considers K positive items with the largest prediction scores and K negative items with the largest prediction scores. And OPAUC(β) considers all positive items and K negative items with the largest prediction scores.

We categorize and discuss the possible scenarios of the ranking list. In the first scenario, the number of positive samples appearing in descending order reaches K first:

$$\underbrace{\cdots+}_{K\text{ positive},S\text{ negative}} \quad | \quad \underbrace{\cdots}_{(n^+-K)\text{ positive},(n^--S)\text{ negative}},$$

where $S < K$. And we could observe that LLPAUC(α,β) $\le$ OPAUC(β). When we keep LLPAUC(α,β) fixed, the maximum value of OPAUC(β) can be achieved as following:

$$\underbrace{\cdots+}_{K\text{ positive},S\text{ negative}} \quad | \quad \underbrace{+\cdots+}_{(n^+-K)\text{ positive}} \quad \underbrace{-\cdots-}_{(n^--S)\text{ negative}},$$

Hence, in the first scenario, we can conclude that

$$\text{LLPAUC}(\alpha,\beta) \le \text{OPAUC}(\beta) \le \text{LLPAUC}(\alpha,\beta) + \frac{(n^+-K)\cdot(n^--S)}{n^+n^-}. \quad (18)$$

Easily, we could further obtain that

$$\text{LLPAUC}(\alpha,\beta) \ge \frac{K(n^--S)}{n^+n^-}. \quad (19)$$

In the second scenario, the number of negative samples appearing in descending order reaches K first:

$$\underbrace{\cdots-}_{S'\text{ positive},K\text{ negative}} \quad | \quad \underbrace{\cdots}_{(n^+-S')\text{ positive},(n^--K)\text{ negative}}.$$

And we could find that:

$$\text{OPAUC}(\beta) = \text{LLPAUC}(\alpha,\beta). \quad (20)$$

Taking into account the two scenarios discussed above, we can easily conclude that OPAUC(β) $\ge$ LLPAUC(α,β), which results in $\mathcal{H}_{higher}(\text{OPAUC}(\beta)) \ge \mathcal{G}_{higher}(\text{LLPAUC}(\alpha,\beta))$.

Step 2: Proof of $\mathcal{H}_{lower}(\textbf{OPAUC}(\beta)) \le \mathcal{G}_{lower}(\textbf{LLPAUC}(\alpha,\beta))$ First, we have

$$\begin{aligned}&\mathcal{H}_{lower}(\text{OPAUC}(\beta)) - \mathcal{G}_{lower}(\text{LLPAUC}(\alpha,\beta))\\ =&\frac{n^+ + K - \sqrt{(n^+ + K)^2 - 4n^+n^-\text{OPAUC}(\beta)}}{2} -\\ &\qquad (K - \sqrt{K^2 - n^+n^-\text{LLPAUC}(\alpha,\beta)})\\ =&\frac{1}{2}[(n^+ - K) - (\sqrt{(n^+ + K)^2 - 4n^+n^-\text{OPAUC}(\beta)} -\\ &\qquad 2\sqrt{K^2 - n^+n^-\text{LLPAUC}(\alpha,\beta)})]\end{aligned} \quad (21)$$

Similar to Step 1, we consider two scenarios. In the first scenario, using Eq. (18), we have:

$$\begin{aligned}&\mathcal{H}_{lower}(\text{OPAUC}(\beta)) - \mathcal{G}_{lower}(\text{LLPAUC}(\alpha,\beta))\\ \le&\frac{1}{2}\{(n^+ - K) + [4K^2 - 4n^+n^-\text{LLPAUC}(\alpha,\beta)]^{\frac{1}{2}} - [(n^+ - K)^2 +\\ &4K(n^+ - K) + 4K^2 - 4n^+n^-\text{LLPAUC}(\alpha,\beta) - 4(n^+ - K)\cdot(n^- - S)]^{\frac{1}{2}}\}\end{aligned} \quad (22)$$

It's notable that when $\sqrt{A^2}+\sqrt{C^2}-\sqrt{A^2+B^2+C^2}\le 0$, we have $2\sqrt{A^2C^2}-B^2\le 0$. Hence, when $A^2=(N^+-K)^2$, $B^2=4K(n^+-K)-4(n^+-K)\cdot(n^--S)$, $C^2=4K^2-4n^+n^-\text{LLPAUC}(\alpha,\beta)$, to prove

$$\mathcal{H}_{lower}(\text{OPAUC}(\beta))-\mathcal{G}_{lower}(\text{LLPAUC})\le 0, \tag{23}$$

we need to prove that

$$2\sqrt{(n^+-K)^2\cdot(4K^2-4n^+n^-\text{LLPAUC}(\alpha,\beta))} - 4(n^+-K)(n^--S-K)\le 0. \tag{24}$$

It's equal to prove that:

$$K^2-n^+n^-\text{LLPAUC}(\alpha,\beta)-(n^--S-K)^2\le 0. \tag{25}$$

$$\iff n^+n^-\text{LLPAUC}(\alpha,\beta)\ge -(n^--S)^2+2(n^--S)K \tag{26}$$

Since we already have $\text{LLPAUC}(\alpha,\beta)\ge\frac{K(n_u^--S)}{n^+n^-}$ in Eq. (19), we can easily complete the proof in the first scenario.

For the second scenario, Eq. (21) can be reformulated as

$$\frac{1}{2}\{(n^+-K)+\sqrt{4K^2-4n^+n^-\text{LLPAUC}(\alpha,\beta)} - \sqrt{(n^+-K)^2+4K(n^+-K)+4K^2-4n^+n^-\text{LLPAUC}(\alpha,\beta)} \tag{27}$$

Similarly, to prove

$$\mathcal{H}_{lower}(\text{OPAUC}(\beta))-\mathcal{G}_{lower}(\text{LLPAUC})\le 0, \tag{28}$$

we need to prove that

$$2\sqrt{(n^+-K)^2\cdot(4K^2-4n^+n^-\text{LLPAUC}(\alpha,\beta))}+4(n^+-K)K\le 0. \tag{29}$$

It's equal to prove that:

$$K^2-n^+n^-\text{LLPAUC}(\alpha,\beta)+K^2\le 0. \tag{30}$$

$$\iff \text{LLPAUC}(\alpha,\beta)\ge 0 \tag{31}$$

This is trivially true, thus we have completed the proof for the second scenario. □

C PROOF OF LEMMA 1

Proof. According to the proof of Theorem 7 in [30], we can easily derive the proof of Lemma 1, the detailed proof can be seen in the Arxiv version of the paper. □

D PROOF OF FUNCTION

In this subsection, we utilize the following lemmas to substantiate our argument.

Lemma 3. *If $\gamma\in[b-1,1]$, $\ell_-(f_{u,j})=(f_{u,j}-b)^2+2(1+\gamma)f_{u,j}$ is an increasing function w.r.t $f_{u,j}$, when $j\in\mathcal{I}_u^-$ and $f_{u,j}\in[0,1]$.*

The proof can be found in Appendix F.2.2 in [30].

Lemma 4. *If $\gamma\in[\max\{b-1,-a\},1]$, $\ell_+(f_{u,i})=(f_{u,i}-a)^2-2(1+\gamma)f_{u,i}$ is an increasing function w.r.t $f_{u,i}$, when $i\in\mathcal{I}_u^+$ and $f_{u,i}\in[0,1]$.*

The proof can be found in Appendix F.3.2 in [30].

E PROOF OF LEMMA 2

Reminder of Lemma 2 Suppose $\ell_+(f_{u,i})$ is monotonic decreasing *w.r.t.* $f_{u,i}$ and $\ell_-(f_{u,j})$ is monotonic increasing *w.r.t.* $f_{u,j}$, then we have

$$\sum_{i\in\mathcal{I}_u^+}\left[\ell_+(f_{u,i})\cdot\mathbb{I}[f_{u,i}\ge\eta_\alpha]\right]=\max_{s^+\in\mathbb{R}}\sum_{i\in\mathcal{I}_u^+}\left[-\alpha s^+-[-\ell_+(f_{u,i})-s^+]_+\right], \tag{32}$$

$$\sum_{j\in\mathcal{I}_u^-}\left[\ell_-(f_{u,j})\cdot\mathbb{I}[f_{u,j}\ge\eta_\beta]\right]=\min_{s^-\in\mathbb{R}}\sum_{j\in\mathcal{I}_u^-}\left[\beta s^-+[\ell_-(f_{u,j})-s^-]_+\right], \tag{33}$$

where s^+ and s^- are learnable parameters, and $[x]_+=max(0,x)$ for any x.

Proof. For Eq. (33), the proof can be found in Lemma 1 in [8]. To prove Eq. (32), we first denote that $(-\ell_+(f_{u,i}))$ is monotonic increasing *w.r.t* $f_{u,i}$ and then obtain:

$$\begin{aligned}&\sum_{i\in\mathcal{I}_u^+}\left[\ell_+(f_{u,i})\cdot\mathbb{I}[f_{u,i}\ge\eta_\alpha]\right]\\ =&-\sum_{i\in\mathcal{I}_u^+}\left[(-\ell_+(f_{u,i}))\cdot\mathbb{I}[f_{u,i}\ge\eta_\alpha]\right]\\ =&-\min_{s^+\in\mathbb{R}}\sum_{i\in\mathcal{I}_u^+}\left[\alpha s^++[-\ell_+(f_{u,i})-s^+]_+\right]\\ =&\max_{s^+\in\mathbb{R}}\sum_{i\in\mathcal{I}_u^+}\left[-\alpha s^+-[-\ell_+(f_{u,i})-s^+]_+\right].\end{aligned} \tag{34}$$

This completes our proof of Eq. (32). Notably in the final line of derivation, we employ $-\min f(x)=\max -f(x)$. □

F PROOF OF MIN-MAX SWAP

Proof. To swap $\max_\gamma$ and $\min_{s^-}$, according to the min-max theorem [1], we need to check the second part of Eq. (13) strongly-concave *w.r.t.* γ. Concretely, the function is:

$$\mathcal{F}_2=\sum_{j\in\mathcal{I}_u^-}\frac{\beta s^-+\frac{1}{\kappa}(\log(1+\exp(\kappa\cdot(\ell_-(f_{u,j})-s^-))))}{n_u^-}-(w+1)\gamma^2, \tag{35}$$

where $\ell_-(f_{u,j})=(f_{u,j}-b)^2+2(1+\gamma)f_{u,j}$. Hence,

$$\frac{\partial\mathcal{F}_2}{\partial\gamma}=\frac{1}{n_u^-}\sum_{j\in\mathcal{I}_u^-}\frac{\exp(\kappa\cdot(\ell_-(f_{u,j})-s^-))}{1+\exp(\kappa\cdot(\ell_-(f_{u,j})-s^-))}\cdot 2f_{u,j}-2w\gamma, \tag{36}$$

$$\frac{\partial^2\mathcal{F}_2}{\partial\gamma^2}=\frac{1}{n_u^-}\sum_{j\in\mathcal{I}_u^-}\frac{\exp(\kappa\cdot(\ell_-(f_{u,j})-s^-))}{\left[1+\exp(\kappa\cdot(\ell_-(f_{u,j})-s^-))\right]^2}\cdot 4\kappa f_{u,j}^2-2w. \tag{37}$$

Since $f_{u,j}\in[0,1]$ and $\frac{\exp(\kappa\cdot(\ell_-(f_{u,j})-s^-))}{\left[1+\exp(\kappa\cdot(\ell_-(f_{u,j})-s^-))\right]^2}\in(0,1)$, with sufficiently large $w>4\kappa$, we have $\frac{\partial^2\mathcal{F}_2}{\partial\gamma^2}<0$. Therefore, with sufficiently large w, Eq. (13) is strongly-concave *w.r.t.* γ. □

G TIME COMPLEXITY OF METHOD

In this section, we show the detailed algorithm in Algorithm 1. Concretely, after each update of the gradient, we clip the parameters to ensure that they are within the constraints of the domain.

Table 3: Performance comparison with Learning-To-Rank methods with clean and noise training on Adressa and Yelp. The best results are highlighted in bold.

	Method	Clean		Noise	
		Recall@20	NDCG@20	Recall@20	NDCG@20
Adressa	Lambdarank [4]	0.1997±0.001	0.1093±0.0032	0.1731±0.0032	0.0936±0.0018
	NDCG-Loss1 [35]	0.1866±0.0049	0.1009±0.0037	0.1774±0.0006	0.0948±0.002
	NDCG-Loss2 [35]	0.2002±0.0031	0.1053±0.0018	0.1854±0.0069	0.0955±0.0037
	ARP-Loss2 [35]	0.1957±0.0009	0.1089±0.0008	0.1759±0.0008	0.0946±0.0016
	LLPAUC	**0.2166**±0.0022	**0.1214**±0.0009	**0.2127**±0.0014	**0.1189**±0.0009
	Method	Clean		Noise	
		Recall@20	NDCG@20	Recall@20	NDCG@20
Yelp	Lambdarank [4]	0.0716±0.0015	0.0396±0.0009	0.0695±0.0001	0.0379±0.0001
	NDCG-Loss1 [35]	0.081±0.0011	0.0450±0.0006	0.0806±0.0011	0.0447±0.0004
	NDCG-Loss2 [35]	0.0784±0.0012	0.0442±0.0009	0.0798±0.0006	0.0449±0.0004
	ARP-Loss2 [35]	0.065±0.00016	0.0356±0.0009	0.0622±0.0023	0.0340±0.0011
	LLPAUC	**0.0884**±0.0005	**0.0505**±0.0003	**0.0847**±0.0007	**0.0481**±0.0001

Algorithm 1 Stochastic Gradient Descent Ascent Algorithm

Input: User set $\mathcal{U}$, Item set $\mathcal{I}$, learning parameters $\{\theta, a, b, s^+, s^-, \gamma\}$
Initialize: Randomly select $\{\theta, a, b, s^+, s^-, \gamma\}$. Let $\tau = \{\theta, a, b, s^-\}$, $\tau' = \{\gamma, s^+\}$
for $t = 0, 1, \cdots, T$ **do**
  Sample a mini-batch positive interaction $\mathcal{B}^+$
  Uniformly sample a mini-batch $\mathcal{B}_u^- \in \mathcal{I}_u^-$ for each $(u, i) \in \mathcal{B}^+$.
  Compute $\mathcal{F}(\tau, \tau')$ defined in Eq.(14).
  Update $\tau_{t+1} = \tau_t - \eta \cdot \nabla_\tau \mathcal{F}(\tau, \tau')$;
  Update $\tau'_{t+1} = \tau'_t + \eta \cdot \nabla_{\tau'} \mathcal{F}(\tau, \tau')$;
  Update $\tau_{t+1} = \text{Clip}(\tau_{t+1})$;
  Update $\tau'_{t+1} = \text{Clip}(\tau'_{t+1})$;
end for
Return θ_{T+1}

For time complexity analysis, we need to consider both forward and backward computational complexity. As stated in Eq. (14), the function is:

$$\mathcal{F} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \{ \sum_{i \in \mathcal{I}_u^+} \frac{-\alpha s^+ - r_\kappa(-\ell_+(f_{u,i}) - s^+)}{n_u^+} + \sum_{j \in \mathcal{I}_u^-} \frac{\beta s^- + r_\kappa(\ell_-(f_{u,j}) - s^-)}{n_u^-} - (w+1)\gamma^2 \}. \quad (38)$$

Hence, the complexity of forward propagation is $O(|\mathcal{B}^+||\mathcal{B}^-|d^2)$, where d is the embedding size of user and item, $\mathcal{B}^+$ and $\mathcal{B}^+$ is the mini batch size. For backward propagation, we first derive the gradient of the function $\mathcal{F}$:

$$\frac{\partial \mathcal{F}}{\partial \theta} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \{ \frac{1}{n_u^+} \sum_{i \in \mathcal{I}_u^+} \frac{\exp(\kappa \cdot (-\ell_+(f_{u,i}) - s^+))}{1 + \exp(\kappa \cdot (-\ell_+(f_{u,i}) - s^+))} \cdot \frac{\partial \ell_+(f_{u,i})}{\partial \theta} + \frac{1}{n_u^-} \sum_{j \in \mathcal{I}_u^-} \frac{\exp(\kappa \cdot (\ell_-(f_{u,j}) - s^-))}{1 + \exp(\kappa \cdot (\ell_-(f_{u,j}) - s^-))} \cdot \frac{\partial \ell_-(f_{u,j})}{\partial \theta} \}. \quad (39)$$

Easily, we obtain the complexity of Eq. (39) is $O(|\mathcal{B}^+||\mathcal{B}^-|d^2)$. The partial derivatives of the function with respect to other parameters have a similar form and the same computational complexity. Hence, the total complexity per iteration is $O(|\mathcal{B}^+||\mathcal{B}^-|d^2)$, which is the same with other baseline models such as BPR loss and BCE loss.

H COMPARISON WITH LEARNING-TO-RANK METHODS

To further prove the efficiency of the proposed LLPAUC loss, we compare it with advanced learning-to-rank (LTR) methods. However, directly applying LTR methods, such as the Lambda Framework [35], to collaborative filtering (CF) tasks is very challenging and time-consuming. This stems from the requirement to rank all items in order to calculate $\triangle \text{NDCG}_{ij}$ for different pairs of items. In IR tasks, the retrieval model limits the candidate documents to a small number (*e.g.*, 1,000), while all unobserved items (up to 3,113,576 items) in CF tasks are potential candidates due to lacking explicit query (please refer to [42] for more detailed discussion).

In order to implement these LTR methods, we design a simulated retrieval model for the CF task. For each positive interaction, we randomly sample 100 negative items (keeping the same with other baselines) to form the candidate item set. Then we report the results in Table 3. From it, we observe that the mainstream LTR methods achieve better results than some baselines such as BPR and BCE methods. However, these methods are still inferior to our proposed methods LLPAUC, indicating that LLPAUC can be more strongly correlated with the Top-K metrics.