Learning Robust Sequential Recommenders through Confident Soft Labels

SHIGUANG WU, XIN XIN, PENGJIE REN, ZHUMIN CHEN, and JUN MA, Shandong University, Qingdao, China
MAARTEN DE RIJKE, University of Amsterdam, Amsterdam, The Netherlands
ZHAOCHUN REN, Leiden University, Leiden, The Netherlands

Sequential recommenders that are trained on implicit feedback are usually learned as a multi-class classification task through softmax-based loss functions on one-hot class labels. However, one-hot training labels are sparse and may lead to biased training and sub-optimal performance. Dense, soft labels have been shown to help improve recommendation performance. However, how to generate high-quality and confident soft labels from noisy sequential interactions between users and items is still an open question.

We propose a new learning framework for sequential recommenders, CSRec, which introduces confident soft labels to provide robust guidance when learning from user–item interactions. CSRec contains a teacher module that generates high-quality and confident soft labels and a student module that acts as the target recommender and is trained on the combination of dense, soft labels and sparse, one-hot labels. We propose and compare three approaches to constructing the teacher module: (i) model-level, (ii) data-level, and (iii) training-level. To evaluate the effectiveness and generalization ability of CSRec, we conduct experiments using various state-of-the-art sequential recommendation models as the target student module on four benchmark datasets. Our experimental results demonstrate that CSRec is effective in training better-performing sequential recommenders.

CCS Concepts: • **Information systems** → **Recommender systems**; **Personalization**; **Retrieval models and ranking**; *Novelty in information retrieval*;

Shiguang Wu and Xin Xin contributed equally to this research.

This research was funded by the Natural Science Foundation of China (62272274, 61972234, 62072279, 62102234, 62202271), Meituan, the Natural Science Foundation of Shandong Province (ZR2022QF004), the Key Scientific and Technological Innovation Program of Shandong Province (2019JZZY010129), Shandong University multidisciplinary research and innovation team of young scholars (2020QNQT017), the Tencent WeChat Rhino-Bird Focused Research Program (JR-WXG2021411), the Fundamental Research Funds of Shandong University, the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organization for Scientific Research, https://hybrid-intelligence-centre.nl, project LESSEN with project number NWA.1389.20.183 of the research program NWA ORC 2020/21, which is (partly) financed by the Dutch Research Council (NWO), project ROBUST with project number KICH3.LTP.20.006, which is (partly) financed by the Dutch Research Council (NWO), DPG Media, RTL, and the Dutch Ministry of Economic Affairs and Climate Policy (EZK) under the program LTP KIC 2020-2023, and the FINDHR (Fairness and Intersectional Non-Discrimination in Human Recommendation) project that received funding from the European Union's Horizon Europe research and innovation program under grant agreement No. 101070212.

Authors' Contact Information: Shiguang Wu, Shandong University, Qingdao, China; e-mail: shiguang.wu@mail.sdu.edu.cn; Xin Xin, Shandong University, Qingdao, China; e-mail: xinxin@sdu.edu.cn; Pengjie Ren, Shandong University, Qingdao, China; e-mail: jay.ren@outlook.com; Zhumin Chen, Shandong University, Qingdao, China; e-mail: chenzhumin@sdu.edu.cn; Jun Ma, Shandong University, Qingdao, China; e-mail: majun@sdu.edu.cn; Maarten de Rijke, University of Amsterdam, Amsterdam, The Netherlands; e-mail: m.derijke@uva.nl; Zhaochun Ren (corresponding author), Leiden University, Leiden, The Netherlands; e-mail: z.ren@liacs.leidenuniv.nl.

ACM 1558-2868/2024/12-ART21
https://doi.org/10.1145/3700876

Additional Key Words and Phrases: Sequential recommendation, Recommender systems, Soft labels, Robustness, Implicit feedback

ACM Reference format:
Shiguang Wu, Xin Xin, Pengjie Ren, Zhumin Chen, Jun Ma, Maarten de Rijke, and Zhaochun Ren. 2024. Learning Robust Sequential Recommenders through Confident Soft Labels. *ACM Trans. Inf. Syst.* 43, 1, Article 21 (December 2024), 27 pages.
https://doi.org/10.1145/3700876

1 Introduction

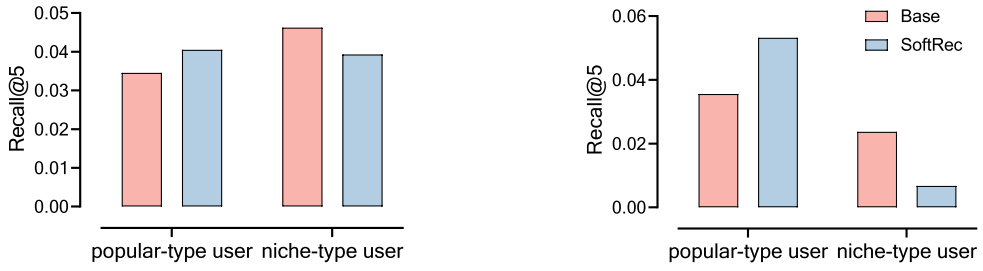
Generating next-item recommendations from sequential implicit user feedback is a widely adopted approach to training recommender systems. This method is common in scenarios like e-commerce [70], video platforms [66], and streaming music services [40]. The problem of learning sequential recommenders based on implicit feedback can be formulated as a multi-class classification task, where each candidate item corresponds to a class. A list of recommendations is then generated by selecting items with the highest classification logits.

Deep neural networks have been widely used to address such classification tasks through softmax-based loss functions over one-hot class labels. The items with which a user has interacted can be viewed as being labeled as 1 s, while all other items in the item set are labeled as 0 s [23, 25, 41, 61]. Thus, the items a user has interacted with are interpreted as the user's positive preferences and are pushed toward higher classification logits during the training process, while all other candidate items are assumed to represent negative user preferences. However, one-hot training labels are sparse and can easily be corrupted [37, 69]. For example, interactions with an item may only be due to presentation bias [6, 7], and a lack of interaction may be attributed to user unawareness, as the item may not have been exposed to the user [43]. Hence, simply promoting high values for items labeled with a 1 and demoting other candidate items can lead to a misunderstanding of user preferences.

Soft Labels. Recent work [8, 28] has shown that compared with sparse one-hot training labels, *soft labels* can help improve recommendation performance. *Soft* labels use class probabilities produced by models instead of the hard zero-one representation, i.e., one-hot labels. They can be seen as a dense distribution over candidate items given the current sequence of items. Liu et al. [28] propose a debiasing recommendation framework based on soft labels derived from knowledge distillation of uniformly exposed data. Cheng et al. [8] use popularity-based or user-based soft labels to train recommenders.

Motivated by [8, 28], we hypothesize that *high-quality and confident soft labels can help train more robust sequential recommenders.* A soft label is *confident* if the supervision signals it provides reflect actual user preferences and have low variance. A sequential recommender is *robust* if it can generate recommendations that lead to a positive user experience, even if the training data are corrupted or contains noise. Unfortunately, the uniformly exposed data used in [28] will always be limited and expensive to collect, as doing so may negatively affect user experience by exposing irrelevant items. The soft labels used in [8] can easily become biased or corrupted by the training data and the teacher model itself.

As shown in Figure 1, niche-type users, who select niche items as their next choices, receive worse recommendation results after using the soft labels proposed in [8]. The bias of the soft labels used in [8] results in reduced exposure to niche items, which is detrimental to both users and merchants in the long run. Hence, the challenge of creating confident, soft labels from noisy sequential interactions between users and items remains an open question.



(a) Yelp (b) Electronics

Fig. 1. Performance of Base recommendation model and SoftRec method [8] on different groups of users. The items are divided into two bins, 20% and 80%, based on the frequency. We then split users based on which kind of items they choose as the next interactions, popular or niche items. The two user groups are referred to as *popular-type* and *niche-type* users. We evaluate the recommendation performance of each group separately. We choose GRU4Rec [41] as the Base recommendation model and SoftRec [8] as the comparison method on the (a) Yelp and (b) Electronics datasets. The niche-type users receive worse recommendation results after using the soft labels proposed in [8].

Recommendations with Confident Soft Labels. We propose **Recommendations with Confident Soft Labels (CSRec)**, a learning framework for training robust sequential recommenders from implicit user feedback. The core idea is to introduce confident, soft labels that complement one-hot labels during the learning process. CSRec consists of a teacher module that generates confident, soft labels from noisy sequential interactions between users and items, and a student module that serves as the target recommender. We propose three methods for constructing the teacher module: (i) model-level, (ii) data-level, and (iii) training-level. These alternatives are motivated as follows:

—Recent research [11, 62] has shown that different models, or even a set of instances of the same model initialized with different random seeds, introduce different kinds of bias into model outputs. Our *model-level* method constructs confident, soft labels from a multi-model ensemble to reduce the bias and variance from the model itself.
—Our *data-level* method reduces the bias or noise signals originating from the data by employing sub-sampling procedures to feed the teacher models with different subsets of the data. The confident, soft labels are, once again, generated from a multi-data ensemble.
—The model-level and data-level methods use ensemble approaches to obtain confident soft labels. Instead, the *training-level* method focuses on directly training a teacher module in an end-to-end fashion. The key insight is to minimize the **Kullback–Leibler (KL)** divergence between the predictions of two teacher models, as confident, soft labels from different models should be consistent to provide more robust guidance.

Given confident, soft labels from the teacher module, the target student recommender is trained using a combination of dense, soft labels and sparse, one-hot labels. Although knowledge distillation also utilizes soft labels, the motivation is quite different. Knowledge distillation aims to compress a large neural network model into a relatively smaller model for more efficient inference, while we aim to utilize robust soft labels to generate robust recommendations from biased data, rather than to speed up inference. Additionally, existing ensemble methods perform ensembling in both training and inference stages, whereas our method involves multiple models to generate confident soft labels during the training stage, with only a single robust target recommender used during inference, thus achieving higher recommendation efficiency.

Technical Comparison. A commonly adopted strategy for enhancing the one-hot representation with soft labels is label smoothing [32, 45], where another rectification distribution is mixed as the soft label. Cheng et al. [8] use the popularity distribution as the soft label, which is a fixed empirical prior distribution over the entire item set. We, however, use the distribution generated from the collaboration of multiple models, which is more accurate, adaptive, and less biased, e.g., with less popularity bias. Moreover, we use the recommendation models to build the supervision signal for themselves without the need for additional labels, as used by Liu et al. [28], which are usually unavailable in practice. Hu et al. [21] recently proposed a decoupled progressive distillation framework, DePoD, for the sequential recommendation. They distill peer models into each other and also use these models during inference. In contrast, we propose a general learning framework to enhance the current target model and directly utilize the collaboration from multiple models for additional robust supervision signals.

Experimental Comparison. To assess the effectiveness and generalization capability of recommender systems trained using the labels produced by CSRec, we conduct experiments on four benchmark datasets using different state-of-the-art sequential recommendation models as the target recommendation module: (i) the **recurrent neural network (RNN)**-based GRU4Rec [41], (ii) the **convolutional neural network (CNN)**-based Nextitnet [61], (iii) the attention-based NARM [25], and (iv) the self-attentive SASRec [23]. Experimental results demonstrate the effectiveness and generalization capability of the CSRec learning framework.

Contributions. In summary, the contributions of our work are as follows:

- We propose CSRec, a learning framework to enhance implicit feedback-based sequential recommenders through a teacher module that provides confident, soft labels and a target student recommender trained on both sparse, one-hot labels and dense, soft labels.
- We propose three methods for constructing the teacher module: (i) a model-level method, (ii) a data-level method, and (iii) a training-level method, all aimed at reducing the effect of bias and noisy interaction signals.
- We evaluate CSRec with eight kinds of state-of-the-art deep learning-based sequential recommendation models and conduct experiments on four benchmark datasets. The results demonstrate the effectiveness of the CSRec learning framework.

2 Related Work

We review related work on sequential recommendation and learning from soft labels.

2.1 Sequential Recommendation

Early work on sequential recommendation mainly relied on factorization methods [46, 50] or Markov chains [15, 58]. Recently, deep learning-based approaches for sequential recommendation have gained significant attention due to their superior learning capacity. These models can be categorized into: (i) RNN-based models [17, 31, 41], (ii) CNN-based models [42], (iii) attention-based models [23, 30, 39], (iv) **graph neural network (GNN)**-based models [36, 38, 49, 67], (v) contrastive learning-based models [33, 55, 68], (vi) generative model-based methods [12, 27, 59], and (vii) large language model-based methods [10, 19, 20, 26].

GRU4Rec [41] is a representative RNN-based model that utilizes gated recurrent units to capture sequential signals. Caser [42] and Nextitnet [61] are CNN-based models that effectively capture skip signals within interaction sequences. NARM [25] introduces an attention mechanism that assigns varying levels of importance to items a user interacts with within a sequence. SASRec [23] employs self-attention and a transformer decoder [44] for sequential recommendation. Unlike the causal decoding used by SASRec, BERT4Rec [39] applies the masked language model objective for

training the recommender. SRGNN [51] represents session sequences as graph-structured data, leveraging GNNs [36] to capture complex item transitions. CL4SRec [55] uses the contrastive learning framework to extract self-supervision signals from original user behavior sequences. DreamRec [59] reshapes sequential recommendation into a learning-to-generate paradigm using a guided diffusion model. Recformer [26] is a large language model-based approach that represents an item sequence as a sentence by flattening item key-value attributes described in the text.

Traditionally, sequential recommenders trained with implicit feedback use either point-wise binary cross-entropy loss or pairwise ranking loss, e.g., Bayesian Personalized Ranking [35]. Both approaches require a negative sampling strategy to sample negative instances from unobserved interactions. The size and distribution of these samples can significantly impact recommendation performance [2]. Although non-sampling-based training methods have been proposed, they often suffer from high computational costs [4] or are limited to shallow linear models [56].

Deep learning-based approaches to sequential recommendation are often formulated as multi-class classifiers, where each candidate item corresponds to a class. These deep models are typically trained using softmax classification loss over sparse one-hot class labels, where items that users interact with are labeled as 1 s, and all other candidate items are labeled as 0 s [3, 23, 25, 39, 41, 61]. This approach assumes that items with which users interact indicate positive preferences, while all other candidate items reflect negative preferences. However, this assumption rarely holds in real-world scenarios [43]. User interactions may be influenced by various types of presentation biases [6], and in some cases, a lack of interaction may simply be due to the user's unawareness. Thus, sparse one-hot training labels can easily become corrupted and fail to capture unobserved user–item interactions. Relying on these labels for sequential recommendation leads to biased training and sub-optimal performance [6, 28, 69].

In this article, we propose to enhance sequential recommenders by incorporating confident, soft labels in addition to the conventional sparse one-hot labels when learning from implicit feedback data.

2.2 Learning from Soft Labels

Soft labels have been successfully utilized in fields such as **computer vision (CV)** and **natural language processing (NLP)**. Knowledge distillation [18] is a widely adopted framework that compresses large complex models into simpler, smaller models to enable faster inference. In this process, knowledge distillation extracts hidden knowledge from large teacher networks in the form of soft labels to guide the training of smaller student networks. Multi-model distillation has also been explored in CV and NLP. A common approach involves averaging the responses from multiple teacher models as part of the supervision signal for the student [14, 60, 63]. For example, You et al. [60] average the outputs of different teachers and reduce the dissimilarity between the student and teachers; Du et al. [13] use multi-objective optimization to determine the best direction that accommodates different teachers; and Liu et al. [29] average the outputs of teachers with weights generated by the teachers themselves.

Recent work has also explored the use of soft labels to improve recommendation systems. Although the predictions from well-trained teacher models contain useful knowledge that could provide additional supervision for student models, naive distillation might introduce unwanted bias and noisy signals. This issue arises because recommendation data, compared to data in CV and NLP, often contains more noise and biases. Chen et al. [5] identified the biased student phenomenon during model compression using knowledge distillation. They proposed a grouping method specifically to address popularity bias in general recommendation systems, though their approach heavily depends on prior knowledge of the bias. Their method aims to enhance the distillation process in the presence of biased soft labels, whereas our research focuses on inferring

more robust and confident soft labels, which addresses a different stage of the process. Liu et al. [28] introduced a debiasing framework for recommendations using soft labels learned from uniform interaction data. However, collecting uniform interaction data is often challenging and costly as it can negatively impact user experience. Cheng et al. [8] used popularity-based and user-based soft labels to enhance sequential recommenders, but their generated soft labels are prone to corruption from noisy implicit feedback. There has also been considerable work on using distillation to enhance GNN-based recommendation models. Ren et al. [34] proposed extracting knowledge from user-bundle graphs and distilling it into a graph-masked autoencoder designed to capture significant local and global user–item relationships. Xia et al. [52] directly applied distillation and contrastive learning to tackle the over-smoothing issue and scalability problems. Xia et al. [54] employed a self-supervised knowledge distillation framework to reduce the model size for on-device recommendation systems. Xia et al. [53] utilized distilled global collaborative effects among users and items, maintained by a hyper-graph transformer network, to enhance user representations in GNN-based models. These methods primarily use distillation to address data sparsity, over-smoothing, and scalability challenges in GNN-based models.

In conclusion, while knowledge distillation in CV and NLP primarily focuses on using soft labels for model compression or transfer learning, the challenge of generating confident soft labels from noisy implicit feedback for robust sequential recommendation remains an open **research question (RQ)** in recommender systems. We address this challenge directly by proposing a new learning framework that leverages soft labels to achieve robust sequential recommendations from implicit feedback data.

3 Method

In this section, we first present our notation and task formulation. We then describe three approaches for constructing a teacher module that provides confident, dense, and soft labels. Finally, we detail the training procedure for the target student recommender.

3.1 Notation and Task Formulation

We focus on the task of learning sequential recommenders from implicit user feedback. We denote the user set as $\mathcal{U}$ and the item set as $\mathcal{I}$. Each user $u \in \mathcal{U}$ interacts with a sequence of items $\boldsymbol{s}_u = (i_1, i_2, \ldots, i_{|s_u|})$, sorted by time. Ideally, the next interacted item is the one with the highest probability in the user's true preference distribution $P(i \mid \boldsymbol{s}_u)$.

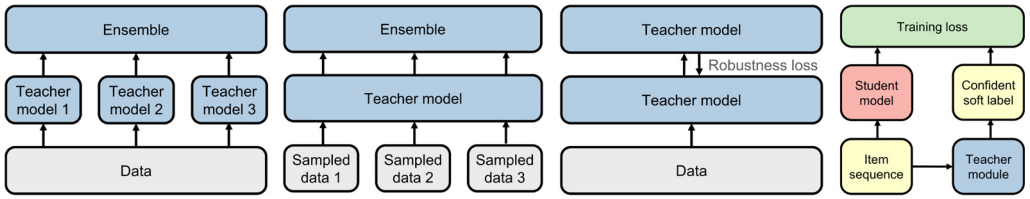
However, the observed next interacted item $i_{|s_u|+1}$ may be corrupted by various types of noise or bias. For example, the user–item interaction might be influenced by popularity bias [6], which can lead to a negative user experience as indicated by Wang et al. [48]. Additionally, in E-commerce, many clicks do not result in purchases, and some purchases receive negative reviews [47]. In our study, we model the noisy signal as an additive error [9] resulting from unknown noise or bias, which can be formulated as follows:

$$i_{|s_u|+1} = \arg\max_{i \in \mathcal{I}} P(i \mid \boldsymbol{s}_u) + \epsilon, \tag{1}$$

where ϵ denotes the noisy signal. The task is to train a target recommender f that can approximate the user's true preference, such that $P_f(i \mid \boldsymbol{s}_u) \approx P(i \mid \boldsymbol{s}_u)$. During model inference, recommendations can be generated as follows:

$$\tilde{i} = \arg\max_{i \in \mathcal{I}} f(\boldsymbol{s}_u), \tag{2}$$

where $f(\boldsymbol{s}_u)$ denotes the output logits of the recommender f. This task can be formulated as a multi-class classification problem, with $\mathcal{I}$ being the candidate class set. Conventional methods



(a) Model-level teacher module (b) Data-level teacher module (c) Teacher-level teacher module (d) Learning student model

Fig. 2. Framework of CSRec, with (a) a model-level teacher module, (b) a data-level teacher module, (c) a training-level teacher module, and (d) a training student model. (a) The model-level teacher module obtains the soft labels by averaging the outputs of multiple teacher models. (b) The data-level teacher module trains teacher models on different data subsets. (c) The training-level teacher module directly learns a robust teacher module without relying on a post-training ensemble and uses the robustness loss function Equation (12) to fuse different models. (d) We use the confident soft labels generated by the teacher module together with the sparse one-hot label to train student recommender through the training loss Equation (20).

Table 1. Main Notation Used in This Work

Symbol	Description
$\mathcal{U}, \mathcal{I}$	User set, and item set
$u \in \mathcal{U}, i \in \mathcal{I}$	User, and item
$s_u = (i_1, \ldots, i_{\lvert s_u \rvert})$	A sequence of items interacted by user u, sorted by time
$i_{\lvert s_u \rvert+1}$	Observed next interacted item, which contains noise and bias
j	Ideal next user-preferred item that cannot be directly observed
f	Student recommender
$\{g_1, \ldots, g_m\}$	Teacher modules
$P(i \mid s_u), P_f(i \mid s_u)$	Real preference distribution, and approximated predicted distribution by model f

that ignore the noisy signal ϵ use a softmax-based cross-entropy loss to train the recommender f by minimizing the difference between $\tilde{i}$ and $i_{\lvert s_u \rvert+1}$. However, due to the presence of ϵ, $i_{\lvert s_u \rvert+1}$ is not truly sampled from $P(i \mid s_u)$, resulting in a discrepancy between the potentially biased learned distribution $P_f(i \mid s_u)$ and the ideal unbiased distribution $P(i \mid s_u)$.

To address this issue, the proposed CSRec framework treats the target recommender f as a student module and introduces a teacher module that consists of a set of models $\{g_1, g_2, \ldots, g_m\}$. The teacher module is designed to generate confident, soft labels to (i) provide dense supervision signals for the many missing interactions; and (ii) alleviate the effect of the noisy signal ϵ. The student model f is then trained on a combination of dense, soft labels and sparse data observations of $i_{\lvert s_u \rvert+1}$. Figure 2 illustrates three alternative teacher modules for CSRec, which we detail in the following sections. We have summarized the main notations used in our study in Table 1.

3.2 Confident Teacher Module

We describe three alternative teacher modules for CSRec.

3.2.1 Model-Level Teachers. Recent research [11, 62] shows that different models, or even the same models trained with different random seeds, can introduce various types of bias into model

Algorithm 1: Model-Level Teachers

Require: $\{g_1, g_2, \ldots, g_m\}$: teacher models; η: learning rate; $\ell_{ce}(\cdot)$: loss for teacher models; $\theta(\cdot)$: parameters; D: training set

```
repeat
    for all s_u ∈ D do
        θ(g_k) ← θ(g_k) − η · ∇ℓ_ce(s_u, g_k, i_{|s_u|+1})        ▷ k = 1, 2, ..., m
    end for
until converged
for all s_u ∈ D do
    e_u ← Σ g_k(s_u)/m                                          ▷ ensemble of g_k
end for
return e_u
```

outputs. This suggests that different models capture different aspects of ϵ, referred to as model-specific noisy signals ϵ_m. We propose constructing a confident teacher module from multiple models to leverage this insight. The key idea is to average the outputs of multiple teacher models. By averaging the noisy signals ϵ_m from each model, we obtain a more uniform error distribution and more robust predictions, which are used to generate soft labels.

We use a set of models $\{g_1, g_2, \ldots, g_m\}$ that share the same architecture as the target student recommender f, but with different random seeds. We train $\{g_1, \ldots, g_m\}$ individually on the entire dataset using different training seeds. The softmax-based cross-entropy loss, denoted as ℓ_{ce}, is used as the training loss function for the teacher models. We then average the outputs to generate soft labels that guide the student model.

The training process for model-level teachers is described in Algorithm 1. Model-level teachers are expected to mitigate bias or noise introduced by the model itself, i.e., ϵ_m.

3.2.2 Data-Level Teachers. In addition to the noisy signal ϵ_m, which is related to the model-level perspective, there may also be data-level bias, ϵ_d, as part of the noisy signal ϵ. We argue that ϵ_d is data-specific and has different distributions across various data subsets. Therefore, a multi-data ensemble of models trained on different data subsets could help alleviate the effect of ϵ_d and provide more robust guidance for the target student recommender.

To achieve this, we propose constructing a confident teacher module from a data-level perspective. We perform sub-sampling with a uniform probability p on the entire dataset to generate different subsets. Teacher models $\{g_1, g_2, \ldots, g_m\}$, which share the same model structure, are then trained on different data subsets. We fuse the outputs of $\{g_1, g_2, \ldots, g_m\}$ to generate more robust teacher outputs.

The data-level teacher's algorithm is described in Algorithm 2.

3.2.3 Training-Level Teachers. Model- and data-level teachers use ensemble-based methods to fuse multi-source outputs, which can be referred to as *post-training* strategies. Training-level methods aim to directly learn a robust teacher module without relying on a post-training ensemble. The key idea is based on two principles:

(1) Confident, soft labels should be consistent with the latent true user preference.
(2) Confident, soft labels from different models should be consistent with each other.

The first principle is directly derived from the definition of confident labels in Section 1, which is the expected property of soft labels in many recent works [8, 34]. The second principle is based on the observation by Wang et al. [48] that different models tend to make similar predictions for clean

Algorithm 2: Data-Level Teachers

Require: $\{g_1, g_2, \ldots, g_m\}$: teacher models; η: learning rate; $\ell_{ce}(\cdot)$: loss for teacher models; $\theta(\cdot)$: parameters; D: training dataset

randomly sample p percent of D for m times as $D_1, \ldots, D_m$
repeat
  for all $\boldsymbol{s}_u \in D_k$ **do** ▷ $k = 1, 2, \ldots, m$
    $\theta(g_k) \leftarrow \theta(g_k) - \eta \cdot \nabla \ell_{ce}(\boldsymbol{s}_u, g_k, i_{|\boldsymbol{s}_u|+1})$
  end for
until converged
for all $\boldsymbol{s}_u \in D$ **do**
  $\mathbf{e}_u \leftarrow \sum g_k(\mathbf{s}_u)/m$ ▷ ensemble of g_k
end for
return $\mathbf{e}_u$

examples compared to noisy ones. Therefore, if the confident, soft labels reflect the latent true user preference, different models should also be consistent with that preference.

Here, we introduce the main teacher model g_1 and a side teacher model g_2. We then describe how to directly train a robust g_1 with the help of g_2.

As discussed earlier, the outputs of g_1 and g_2 should be consistent with each other if they are expected to generate confident, soft labels. Therefore, given an item sequence $\boldsymbol{s}_u$, we aim to minimize their KL divergence as follows:

$$\mathrm{KL}(P_{g_2}(\cdot \mid \boldsymbol{s}_u) \| P_{g_1}(\cdot \mid \boldsymbol{s}_u)) = \mathbb{E}_{j \sim P_{g_2}}[\log(P_{g_2}(j \mid \boldsymbol{s}_u)) - \log(P_{g_1}(j \mid \boldsymbol{s}_u))], \tag{3}$$

where j is a random variable representing the ideal user-preferred item, which cannot be directly observed. What we can observe from the data is $i_{|\boldsymbol{s}_u|+1}$, which contains noise and bias. For simplicity, in the following description, we will use i to denote $i_{|\boldsymbol{s}_u|+1}$. Moreover, $\boldsymbol{s}_u$ will occasionally be omitted, and $P(\cdot \mid \boldsymbol{s}_u)$ will be abbreviated as P if necessary.

According to Bayes' Theorem and the discussion, we have

$$P_{g_1}(j \mid \boldsymbol{s}_u) \sim P(j \mid \boldsymbol{s}_u) = \frac{P(i \mid \boldsymbol{s}_u) P(j \mid i, \boldsymbol{s}_u)}{P(i \mid j, \boldsymbol{s}_u)}. \tag{4}$$

Substituting Equation (4) into Equation (3), we obtain

$$\begin{aligned} \mathrm{KL}(P_{g_2} \| P_{g_1}) &\approx \mathbb{E}_{j \sim P_{g_2}} \left[\log(P_{g_2}(j \mid \boldsymbol{s}_u)) - \log \frac{P(i \mid \boldsymbol{s}_u) P(j \mid i, \boldsymbol{s}_u)}{P(i \mid j, \boldsymbol{s}_u)} \right] \\ &= \mathrm{KL}(P_{g_2} \| P) - \log(P(i \mid \boldsymbol{s}_u)) + \mathbb{E}_{j \sim P_{g_2}}[\log P(i \mid j, \boldsymbol{s}_u)]. \end{aligned} \tag{5}$$

Rearranging Equation (5), we have

$$\mathrm{KL}(P_{g_2} \| P) - \log(P(i \mid \boldsymbol{s}_u)) \approx \mathrm{KL}(P_{g_2} \| P_{g_1}) - \mathbb{E}_{j \sim P_{g_2}}[\log P(i \mid j, \boldsymbol{s}_u)]. \tag{6}$$

Since $\mathrm{KL}(P_{g_2} \| P) \geq 0$, the right-hand side of Equation (6) is an approximate upper bound of the negative logarithm likelihood. We can regard $\mathbb{E}_{j \sim P_{g_2}}[\log P(i \mid j, \boldsymbol{s}_u)]$ as a regularization term to adjust the agreement across P_{g_1}, P_{g_2}, and P.

Through a similar derivation, we also have

$$\mathrm{KL}(P_{g_1} \| P) - \log(P(i \mid \boldsymbol{s}_u)) \approx \mathrm{KL}(P_{g_1} \| P_{g_2}) - \mathbb{E}_{j \sim P_{g_1}}[\log P(i \mid j, \boldsymbol{s}_u)]. \tag{7}$$

Combining the right-hand sides of both Equations (6) and (7), we obtain the following regularization term:

$$\begin{aligned}\ell_{reg} = \alpha \left\{ \mathrm{KL}(P_{g_2} \| P_{g_1}) - \mathbb{E}_{j \sim P_{g_2}} [\log P(i \mid j, \mathbf{s}_u)] \right\} \\ + (1-\alpha) \left\{ \mathrm{KL}(P_{g_1} \| P_{g_2}) - \mathbb{E}_{j \sim P_{g_1}} [\log P(i \mid j, \mathbf{s}_u)] \right\}. \end{aligned} \tag{8}$$

Since our goal is to train the main teacher g_1 with the help of the side teacher g_2, the term $\mathbb{E}_{j \sim P_{g_2}} [P(i \mid j, \mathbf{s}_u)]$ does not contribute to the training of g_1, so we omit it and revise our regularization function to

$$\ell_1 = \alpha \mathrm{KL}(P_{g_2} \| P_{g_1}) + (1-\alpha) \mathrm{KL}(P_{g_1} \| P_{g_2}) - \mathbb{E}_{j \sim P_{g_1}} [\log P(i \mid j, \mathbf{s}_u)]. \tag{9}$$

In practice, for simplicity, we assume that the observation of i given the real user-preferred item j is conditionally independent of the item sequence $\mathbf{s}_u$. This is a reasonable assumption since the correlation between the observed item i and the ideal item j mainly depends on noisy signals. As a result, $P(i \mid j, \mathbf{s}_u)$ can be approximated through an auxiliary model h as

$$h(i, j) \approx P(i \mid j) \approx P(i \mid j, \mathbf{s}_u). \tag{10}$$

A concise solution to learn h is to use matrix factorization and factorize $h(i \mid j)$ as

$$[h]_{|\mathcal{I}| \cdot |\mathcal{I}|} = \mathcal{M}_{|\mathcal{I}| \cdot d} \cdot \mathcal{N}_{d \cdot |\mathcal{I}|}, \tag{11}$$

where $d \ll \mathcal{I}$; $[h]$ is the matrix form of the model h and can be viewed as a global noise matrix. $\mathcal{M}$ and $\mathcal{N}$ are its low-rank factorization matrices.

To summarize, given an item sequence $\mathbf{s}_u$ and the observed next item i as the label, we obtain the training-level robust regularization:

$$\ell_r = \alpha \mathrm{KL}(P_{g_2} \| P_{g_1}) + (1-\alpha) \mathrm{KL}(P_{g_1} \| P_{g_2}) - \sum_{j \in \mathcal{I}} \log(h_{i,j}) P_{g_1}(j \mid \mathbf{s}_u). \tag{12}$$

Since in practice, h outputs logits, which we denote as $\tilde{h}$, instead of probabilities, we need to normalize it through softmax. However, the softmax operation requires summing over $\exp \tilde{h}_{i,j}$, which is computationally expensive. To avoid normalization and enable more efficient computation, we relax the term $-\sum_{j \in \mathcal{I}} \log(h_{i,j}) P_{g_1}(j \mid \mathbf{s}_u)$ as follows:

$$-\sum_{j \in \mathcal{I}} \log(h_{i,j}) P_{g_1}(j \mid \mathbf{s}_u) \tag{13}$$

$$= -\sum_{j \in \mathcal{I}} \log \left(\frac{\exp \tilde{h}_{i,j}}{\sum_k \exp \tilde{h}_{k,j}} \right) P_{g_1}(j \mid \mathbf{s}_u) \tag{14}$$

$$\leq -\sum_{j \in \mathcal{I}} \left[\log \left(\exp \tilde{h}_{i,j} \right) - \log \left(|\mathcal{I}| \max_k \exp \tilde{h}_{k,j} \right) \right] P_{g_1}(j \mid \mathbf{s}_u) \tag{15}$$

$$\leq \sum_{j \in \mathcal{I}} \left[\log \left(1 + \exp \left(-\tilde{h}_{i,j} \right) \right) + \log |\mathcal{I}| + \max_k \tilde{h}_{k,j} \right] P_{g_1}(j \mid \mathbf{s}_u). \tag{16}$$

Then, the robustness loss function used in practice becomes:

$$\begin{aligned}\ell_r = \alpha \mathrm{KL}(P_{g_2} \| P_{g_1}) + (1-\alpha) \mathrm{KL}(P_{g_1} \| P_{g_2}) \\ + \sum_{j \in \mathcal{I}} \left[\log \left(1 + \exp \left(-\tilde{h}_{i,j} \right) \right) + \log |\mathcal{I}| + \max_k \tilde{h}_{k,j} \right] P_{g_1}(j \mid \mathbf{s}_u). \end{aligned} \tag{17}$$

Algorithm 3: Training-Level Teachers

Require: g_1, g_2: two teacher models; η: learning rate; $\ell_{ce}(\cdot)$: loss function for the side teacher g_2; $\ell_t(\cdot)$: loss function for the main teacher g_1; $\theta(\cdot)$: parameters; D: training dataset

randomly sample p percent data D' from D.
repeat
  for all $s_u \in D'$ **do**
    $\theta(g_2) \leftarrow \theta(g_2) - \eta \cdot \nabla \ell_{ce}(s_u, g_2, i_{|s_u|+1})$ ▷ pretrain g_2
  end for
until converged
repeat
  for all $s_u \in D$ **do**
    $\theta(g_1) \leftarrow \theta(g_1) - \eta \cdot \nabla \ell_t(s_u, g_1, i_{|s_u|+1})$ ▷ train g_1
  end for
until converged
for all $s_u \in D$ **do**
  $\mathbf{e}_u \leftarrow g_1(\mathbf{s}_u)$
end for
return $\mathbf{e}_u$

Finally, we combine the robust loss ℓ_r with the softmax cross-entropy loss (i.e., ℓ_{ce}) to form the final loss function, which constructs the training-level confident teacher module:

$$\ell_t = \ell_r + \ell_{ce}. \tag{18}$$

We pretrain the side teacher model g_2 on a sub-sampled data split to enhance learning stability and introduce data-level robustness. The algorithmic process for training-level teachers is described in Algorithm 3. First, we use the robustness loss function to train one of the teacher models. Then, this model is used to generate confident soft labels directly for the student model. The robustness loss function ensures that this teacher model is aware of the soft labels generated by other teacher models, allowing them to be fused through training.

3.3 Learning of Student Recommenders

Given the input sequence $\mathbf{s}_u$ and dense logits $\mathbf{e}_u$ (as produced by Algorithms 1, 2, and 3) from the confident teacher module, we define the soft labels for the student recommender f as:

$$\mathbf{r}_u = \frac{1}{2}(\text{softmax}(\mathbf{e}_u/\mathcal{T}) + \text{onehot}(i_{|s_u|+1})), \tag{19}$$

where $\mathcal{T}$ is a temperature parameter that smooths the soft label distribution. A larger $\mathcal{T}$ results in smoother soft labels. As $\mathcal{T} \to \infty$, the soft labels effectively perform naive label smoothing.

Finally, the training loss for the target student recommender f is defined as:

$$\ell_s = (1-\beta)\ell_{\text{ce}}(\mathbf{s}_u, f, i_{|s_u|+1}) + \beta \text{KL}(P_f(\cdot \mid \mathbf{s}_u) \| \mathbf{r}_u). \tag{20}$$

During inference, the student model f generates a list of recommended items by selecting the top-n items with the highest classification logits.

3.4 Summary and Remarks

We have proposed three strategies for learning a confident teacher module that generates soft labels. Each method leverages the collaboration of multiple teachers, but from different perspectives and at different stages of training. The model-level and data-level strategies involve collaboration after

the teachers' training, where each teacher model is trained independently. Averaging the outputs is an effective way to reduce errors and produce more robust soft labels. In contrast, the training-level strategy involves collaboration between the main teacher model and a side teacher model during the training process. The main teacher model is directly trained to generate confident soft labels without the need for post-training fusion. In the training-level strategy, we use a pre-trained g_2 instead of training both models together, as the pre-trained model significantly accelerates the collaboration process. Additionally, we employ a sub-sampled data split during the pre-training of the side teacher model g_2. This approach helps the main teacher model g_1 benefit from data-level randomness, leading to more robust outputs.

4 Experiments

In this section, we describe our experimental setup for evaluating the effectiveness of our proposed training method. We focus on the following RQs:

(RQ1) What is the overall recommendation performance of the proposed CSRec learning framework?
(RQ2) Does CSRec improve the robustness of sequential recommendations from implicit user feedback?
(RQ3) How does the design of the teacher module impact the performance of the student model?

4.1 Experimental Settings

4.1.1 Datasets. We conduct experiments on four datasets: (i) Last.FM,[1] (ii) Yelp,[2] (iii) Amazon Electronics, and (iv) Amazon Movies and TV.[3] Table 2 shows the dataset statistics.

We exclude users and items with fewer than five interactions from all datasets. Since we focus on sequential recommendation tasks, we split each user's interactions into sequences of fixed lengths. For sessions that are too short, we add padding tokens. For longer sessions, we divide them into several sub-sessions.

4.1.2 Evaluation Protocols. To evaluate recommendation performance, we use the *leave-one-out* evaluation procedure. The last item in each sequence is treated as the test sample, while the second-to-last item is used for validation. The remaining interactions serve as the training data. We use the full item set as the candidate set for ranking.

For evaluation, we use two ranking-based metrics: (i) Recall and (ii) **normalized discounted cumulative gain (NDCG)**. Recall@n measures whether the ground-truth item appears in the top-n positions of the recommendation list. NDCG is a weighted metric that gives higher importance to items at the top positions.

To assess the robustness of the recommenders (i.e., their ability to generate higher user ratings from noisy implicit feedback), we introduce filtered versions of these metrics for the two Amazon datasets, which contain rating information.

The original Recall is defined as

$$\text{Recall@}n = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbb{1}(\text{rank}_u \leq n),$$

[1] https://grouplens.org/datasets/hetrec-2011/
[2] https://www.yelp.com/dataset
[3] http://jmcauley.ucsd.edu/data/amazon/

Table 2. Statistics of Datasets

Dataset	#Users	#Items	#Interactions	Length
Last.FM	3,173	3,646	57,826	20
Yelp	43,947	20,033	361,290	10
Electronics	102,187	29,351	763,813	10
Movies and TV	82,156	21,035	697,924	10

where rank_u denotes the rank of the ground-truth item and $\mathbb{1}(\cdot)$ is the indicator function. For the *filtered version*, denoted as Recall$^+$, we have

$$\text{Recall}^+@n = \frac{1}{\sum_{u \in \mathcal{U}} \mathbb{1}(r_u \geq \delta)} \sum_{u \in \mathcal{U}} \mathbb{1}(r_u \geq \delta \wedge \text{rank}_u \leq n),$$

where r_u is the rating of the ground-truth item and δ is a threshold value set to 4 (ratings range from 0 to 5). This filtered metric measures whether the recommended items can lead to positive user preferences.

Similarly, we have a filtered version of the NDCG metric, denoted as NDCG$^+$. Note that ratings are only used during the evaluation phase to verify robustness. For training, we use binary implicit user feedback (i.e., items that a user interacted with are labeled as positive, while other interactions are considered missing).

4.1.3 Baselines. To assess the effectiveness of training recommenders with CSRec, we use the following models as student models:

—GRU4Rec [41] uses a GRU to model sequential user behavior, and we use an improved version by Tan et al. [41].
—NextItNet [61] is a simple yet effective CNN-based model capable of learning both short- and long-range dependencies.
—SASRec [23] is based on the transformer decoder [44], which employs multi-head self-attention to capture user preferences.
—BERT4Rec [39] uses deep bidirectional self-attention to model user behavior sequences and adopts the Cloze objective for sequential recommendation by predicting randomly masked items in the sequence using both left and right context.
—NARM [25] utilizes an attention mechanism to capture user intention and computes the recommendation score using bi-linear matching based on latent representations.
—SRGNN [51] models session sequences as graph-structured data and uses a GNN [36] to capture complex transitions between items.
—CL4SRec [55] employs a contrastive learning framework to derive self-supervision signals from original user behavior sequences.
—DreamRec [59] approaches sequential recommendation as a learning-to-generate paradigm achieved through a guided diffusion model.

Each student recommendation model is trained using the following methods:

—*Base* refers to the standard training framework that optimizes the cross-entropy between the output and the sparse one-hot label.
—*SoftRec* denotes the item-based method proposed by Cheng et al. [8], which uses a popularity-based teacher model to generate soft labels.

— *CSRec-M* is our proposed framework that utilizes the model-level teacher module to produce confident, soft labels.
— *CSRec-D* is our proposed framework that employs data-level teachers to provide confident, soft labels.
— *CSRec-T* is our proposed framework that uses training-level teachers to generate confident, soft labels.

4.1.4 Implementation Details. We use implementations from Recbole [64][4] and Yang et al. [59][5] for the eight baseline models listed above.

The embedding size for all models, including d in the training-level method, is fixed at 64 for a fair comparison. Both α and β in Equations (12) and (20) are chosen from $\{0.25, 0.5, 0.75\}$, and the temperature $\mathcal{T}$ in Equation (20) is chosen from $\{1, 3, 6, 9\}$. For CSRec-M and CSRec-D, the number of teachers (m) is set to 2. For CSRec-D, the sub-sampling ratio p is set to 0.8.

For GRU4Rec, the number of layers is set to 2 for all datasets, with a dropout rate of 0.5. For NextItNet, the kernel size is set to 3, the number of blocks is 5 for all datasets, and the dilations are set to 1 after tuning. For SASRec, the number of layers is 2, the number of heads is 2, and the dropout rate is 0.3. For NARM, the hidden size is 128, and the number of layers is 1. For BERT4Rec, the hidden size is 64, the number of layers is 2, and the dropout rate is 0.3. For SRGNN, the hidden size is 64, and the number of layers is 2. For CL4SRec, the hidden size is 64, the number of layers is 2, and the dropout rate is 0.5. For DreamRec, we extensively tuned the original model code on our chosen datasets but did not achieve reasonable results. Consequently, we replaced the MSE loss in the diffusion module with cross-entropy loss. The hidden size is set to 64, the dropout rate to 0.1, the initializer range to 0.02, the timestep to 500, and the beta schedule is exponential, starting at 0.0001 and ending at 0.02.

We use Adam [24] as the training optimizer, setting the learning rate to 0.001. For CSRec, we use the same model architecture for the teacher models g as for the student models f to ensure a fair comparison.[6]

5 Results and Analysis

5.1 Overall Performance Comparison (RQ1)

Table 3 compares the recommendation performance of all models under different training regimes. Training with the three proposed CSRec methods (i.e., CSRec-M, CSRec-D, and CSRec-T) consistently leads to improved recommendation performance across four datasets and four student models. Notably, the model-level and training-level methods achieve significant improvements over the SoftRec method in most cases.

When breaking down the results by dataset, we observe that training with CSRec-D sometimes yields higher recommendation performance compared to the other two proposed methods on the Last.FM dataset. In contrast, on the other three datasets, training with either CSRec-M or CSRec-T results in the highest recommendation performance in most cases. This difference could be due to the denser interactions in Last.FM compared to the other datasets. As the data-level teachers omit some samples, more variations and randomness are introduced, allowing CSRec-D to capture a more robust signal from the data. Training with CSRec-T consistently results in good recommendation performance on the Yelp and Electronics datasets, which are both relatively sparse. Thus, CSRec-D appears suitable for dense datasets, while CSRec-T is more effective for sparse datasets. Additionally, CSRec-M is applicable in more general cases.

[4] https://github.com/RUCAIBox/RecBole
[5] https://github.com/YangZhengyi98/DreamRec
[6] The code and data used are available at https://github.com/Furyton/CSRec/

Table 3. Comparison of the Top-n Recommendation Performance of Different Models (n = 10) on Four Datasets

		GRU4Rec		NextItNet		NARM		SASRec	
Dataset	**Method**	RC (%)	NG (%)	RC (%)	NG (%)	RC (%)	NG (%)	RC (%)	NG (%)
Last.FM	Base	16.68	10.08	15.45	11.56	20.26	**13.20**	20.35	11.43
	SoftRec	17.42	10.62	15.47	11.82	20.48	12.82	20.74	11.95
	CSRec-M	18.16[a]	10.67	**16.46[a]**	**12.47[a]**	**20.81[a]**	13.05	20.87	11.97
	CSRec-D	**18.60[a]**	**10.99[a]**	16.20[a]	12.05	20.73[a]	12.96	**21.13[a]**	**12.25[a]**
	CSRec-T	18.18[a]	10.76[a]	15.68	12.00	20.69	12.87	21.04	12.00
Yelp	Base	5.81	3.51	8.07	5.74	8.58	5.64	8.52	5.32
	SoftRec	6.10	3.58	8.30	5.92	8.10	5.02	9.05	5.58
	CSRec-M	6.40[a]	3.82[a]	8.58[a]	6.17[a]	**8.90[a]**	**5.77[a]**	**9.48[a]**	**5.90[a]**
	CSRec-D	6.23	3.64	8.50	6.10	8.72[a]	5.48[a]	9.48	5.89[a]
	CSRec-T	**6.52[a]**	**3.89[a]**	**8.58[a]**	**6.17[a]**	8.70[a]	5.53[a]	9.42	5.88
Electronics	Base	5.17	2.85	4.90	3.21	5.21	3.05	6.33	4.00
	SoftRec	5.43	3.04	5.20	3.42	5.49	3.19	6.36	4.08
	CSRec-M	5.54[a]	3.13[a]	**5.66[a]**	**3.78[a]**	5.78[a]	3.43[a]	6.73	3.94
	CSRec-D	5.51[a]	3.10	5.30[a]	3.50[a]	5.84[a]	3.48[a]	6.72	3.92
	CSRec-T	**5.59[a]**	**3.16[a]**	5.35[a]	3.58[a]	**5.90[a]**	**3.52[a]**	**7.00[a]**	**4.10**
Movies and TV	Base	8.41	4.77	7.37	4.44	9.38	5.51	11.09	6.61
	SoftRec	9.54	5.36	8.22	4.97	9.40	5.45	10.90	6.54
	CSRec-M	9.63	5.52[a]	**8.74[a]**	**5.30[a]**	**9.99[a]**	**5.88[a]**	**11.19**	6.70
	CSRec-D	9.45	5.42[a]	8.36	5.02	9.75[a]	5.71[a]	11.19	**6.71**
	CSRec-T	**9.70[a]**	**5.52[a]**	8.46[a]	5.11[a]	9.87[a]	5.77[a]	11.19	6.70

(Continued)

Examining the results by student model, we find that RNN-based and CNN-based models, such as GRU4Rec and NextItNet, experience the most notable improvements with CSRec training. In contrast, SASRec and BERT4Rec models, which utilize self-attention mechanisms, show relatively smaller improvements. Recent research [16, 57] indicates that pre-trained transformers handle noisy examples more effectively than CNN or RNN-based models. Similar findings are reported in computer vision [1]. However, CSRec still leads to significant performance improvements for SASRec in most cases. DreamRec employs a small transformer model to encode the sequence as the condition for the guided diffusion module. We observe a substantial improvement with this architecture, suggesting that our generated confident soft labels can enhance the denoising process. CL4SRec, which already uses many data augmentation techniques, benefits only slightly from our confident soft labels.

In summary, the CSRec framework improves the performance of various recommenders compared to standard training regimes and SoftRec. The choice of teacher modules can be tailored to different dataset characteristics to achieve optimal performance.

5.2 Robustness Performance (RQ2)

5.2.1 Evaluation on Real Positive User Preference. We use filtered evaluation metrics on two Amazon datasets to determine whether CSRec training results in recommenders that align with real positive user preferences (i.e., higher ratings). The results, shown in Table 4, indicate that our

Table 3. Continued

		BERT4Rec		SRGNN		DreamRec		CL4SRec	
Dataset	**Method**	RC (%)	NG (%)	RC (%)	NG (%)	RC (%)	NG (%)	RC (%)	NG (%)
Last.FM	Base	17.27	9.62	14.31	9.11	15.44	8.68	18.34	10.19
	SoftRec	17.39	8.77	14.35	9.30	16.07	8.97	17.89	9.68
	CSRec-M	18.12[a]	10.03[a]	15.19[a]	**9.91[a]**	19.00[a]	**11.29[a]**	**19.00[a]**	**10.72[a]**
	CSRec-D	17.84	9.54	15.29[a]	9.78[a]	18.69[a]	10.86[a]	18.94	10.58
	CSRec-T	**18.47[a]**	**10.35[a]**	**15.51[a]**	9.44	**19.10[a]**	11.18[a]	18.63	10.35
Yelp	Base	7.15	4.34	6.91	4.72	6.99	4.22	8.59	5.48
	SoftRec	7.16	4.05	6.93	4.62	6.45	3.94	8.23	4.56
	CSRec-M	**7.55**	**4.61**	7.51	**5.11[a]**	**8.37[a]**	**4.97**	9.00	5.54
	CSRec-D	7.42	4.48	**7.55[a]**	4.97	8.30[a]	4.85	**9.07**	**5.59**
	CSRec-T	7.40	4.51	7.31	4.77	8.18[a]	4.71	8.97	5.46
Electronics	Base	4.95	2.73	5.15	3.64	3.29	2.01	6.32	3.47
	SoftRec	5.06	2.63	5.44	3.57	4.06	2.50	**6.58**	3.54
	CSRec-M	5.61[a]	3.05	**5.66**	**3.93**	4.43	2.63	6.44	3.66
	CSRec-D	5.58[a]	3.05	5.35	3.68	4.86[a]	**2.83[a]**	6.48	**3.67**
	CSRec-T	**5.64[a]**	**3.10**	5.59	3.74	**4.88[a]**	2.80[a]	6.47	3.63
Movies and TV	Base	8.44	4.69	7.66	4.71	7.00	4.29	10.32	6.27
	SoftRec	8.12	4.15	7.30	4.37	7.26	4.33	9.80	5.52
	CSRec-M	**8.95[a]**	**4.99**	8.42[a]	**5.28[a]**	8.52[a]	5.26[a]	**10.50**	**6.33**
	CSRec-D	8.61	4.75	8.03[a]	4.93	**8.67[a]**	**5.36[a]**	10.48	6.32
	CSRec-T	8.82	4.91	**8.48[a]**	5.23[a]	8.65[a]	**5.36[a]**	10.22	6.16

RC is short for Recall@10. NG is short for NDCG@10. Boldface denotes the highest score.
[a]indicates significant improvements over the corresponding baseline ($p < 0.05$).

proposed training methods outperform the base training and SoftRec on these datasets. Most CSRec results demonstrate significant improvements, suggesting that our methods effectively capture real user preferences even in the presence of noisy training instances (i.e., items a user interacted with but rated low). Furthermore, the high performance achieved with CSRec-M and CSRec-T underscores the generalization capability of our methods.

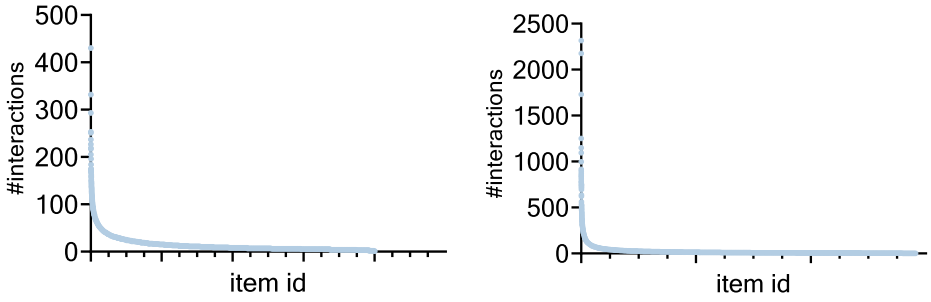
5.2.2 Effect on Reducing Popularity Bias. The long tail phenomenon is prominent in recommendation datasets. As illustrated by the popularity distribution in Figure 3, very few items have interactions that far exceed those of most other items. We assess the popularity bias of the student module in Figure 4. Our methods improve recommendation performance for tail items with only a minor reduction in performance for the most popular items. On the Yelp dataset, we observe significant improvements for both popular and niche users. This indicates that our training methods alleviate popularity bias and better focus on niche items, which benefits the long-term profitability of service providers.

For a deeper analysis, it is important to note that popularity bias is not always detrimental. According to Zhao et al. [65], popular items can reflect general user interests. A single teacher model trained separately (e.g., SoftRec [8]) may exhibit both harmful biases and beneficial general interests, thereby enhancing popularity bias. In contrast, CSRec mitigates harmful bias through collaboration among multiple teacher modules while retaining beneficial general interests. This is

Table 4. Comparison of the Top-n Recommendation Performance of Different Models (n = 10) on Two Rating Datasets Using the Filtered Metrics Described in Section 4.1.2

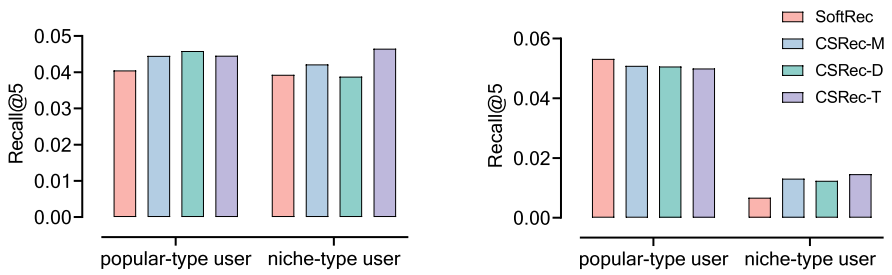
		GRU4Rec		NextItNet		NARM		SASRec	
Dataset	**Method**	RC$^+$ (%)	NG$^+$ (%)	RC$^+$ (%)	NG$^+$ (%)	RC$^+$ (%)	NG$^+$ (%)	RC$^+$ (%)	NG$^+$ (%)
Electronics	Base	5.57	3.07	5.11	3.34	5.57	3.26	7.39	4.27
	SoftRec	5.81	3.26	5.48	3.60	5.90	3.43	7.43	4.35
	CSRec-M	5.93[a]	3.35[a]	**5.94**[a]	**3.96**[a]	6.19[a]	3.67[a]	7.23	4.23
	CSRec-D	5.90[a]	3.31[a]	5.55	3.66[a]	6.25[a]	3.72[a]	7.20	4.20
	CSRec-T	**5.97**[a]	**3.38**[a]	5.63[a]	3.75[a]	**6.32**[a]	**3.76**[a]	**7.45**	**4.38**
Movies and TV	Base	8.60	4.92	7.55	4.58	9.71	5.79	11.60	7.07
	SoftRec	9.81	5.59	8.38	5.12	9.72	5.70	11.29	6.87
	CSRec-M	9.92[a]	5.74[a]	**8.99**[a]	**5.50**[a]	**10.36**[a]	**6.19**[a]	11.64[a]	7.07
	CSRec-D	9.73[a]	5.65[a]	8.55	5.19	10.07[a]	6.01[a]	**11.64**[a]	**7.08**
	CSRec-T	**9.95**[a]	**5.74**[a]	8.64[a]	5.26[a]	10.17[a]	6.05[a]	11.60	7.04
		BERT4Rec		SRGNN		DreamRec		CL4SRec	
Dataset	**Method**	RC$^+$ (%)	NG$^+$ (%)	RC$^+$ (%)	NG$^+$ (%)	RC$^+$ (%)	NG$^+$ (%)	RC$^+$ (%)	NG$^+$ (%)
Electronics	Base	5.34	2.95	5.36	3.78	3.37	2.07	6.79	**3.94**
	SoftRec	5.50	2.88	5.76	3.75	4.25	2.62	6.70	3.43
	CSRec-M	**6.10**[a]	**3.32**[a]	**5.90**[a]	**4.08**	4.63	2.75	6.85	3.89
	CSRec-D	6.04	3.31	5.61	3.84	**5.12**[a]	**2.98**[a]	**6.90**	3.90
	CSRec-T	5.89	3.20	5.46	3.73	4.35	2.90	6.83	3.89
Movies and TV	Base	8.82	4.96	7.83	4.88	7.25	4.51	10.75	6.63
	SoftRec	8.48	4.37	7.48	4.50	6.49	4.23	9.19	5.78
	CSRec-M	**9.38**[a]	**5.32**[a]	**8.69**[a]	**5.49**[a]	8.81	5.54[a]	**10.96**	**6.71**
	CSRec-D	9.00	5.03	8.20[a]	5.08	**8.96**	**5.62**[a]	10.90	6.68
	CSRec-T	9.24[a]	5.21	8.66[a]	5.36[a]	8.91	5.6[a]1	10.62	6.49

RC$^+$ is short for Recall$^+$@10. NG$^+$ is short for NDCG$^+$@10. Boldface denotes the highest score.
[a]denotes significant improvements over the corresponding baseline ($p < 0.05$).



(a) Yelp (b) Electronics

Fig. 3. Item popularity distribution in the dataset (a) Yelp and (b) Electronics. The x-axis is the item id sorted by the number of interactions.



(a) Yelp (b) Electronics

Fig. 4. Performance of SoftRec and the three proposed training methods, i.e., CSRec-M, CSRec-D, and CSRec-T, on different groups of users. Similar settings to Figure 1 are adopted. We choose GRU4Rec [41] as the base model on the (a) Yelp and (b) Electronics datasets.

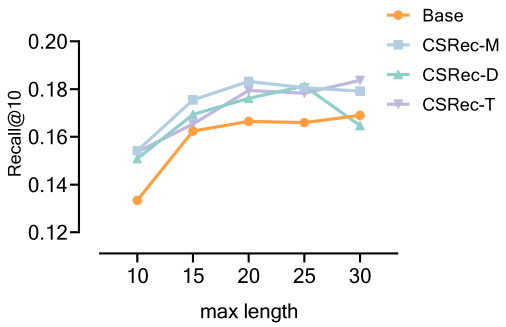


Fig. 5. Performance of the base model and the three proposed training methods, i.e., CSRec-M, CSRec-D, and CSRec-T, on Last.FM dataset with different sequence lengths. We choose GRU4Rec [41] as the base model.

because beneficial general user interests should be consistent across models and thus preserved during model collaboration.

5.2.3 Effect of Sequence Length. We also investigate how the sequence length of user–item interactions affects performance, as shown in Figure 5. Users with longer interaction histories do not necessarily receive better recommendations [22]. However, training with our proposed methods shows increased resilience to variations in interaction history length. This observation suggests that CSRec consistently improves recommendation performance regardless of sequence length.

In conclusion, our proposed training methods not only enhance performance in standard evaluation scenarios but also improve outcomes in evaluations of positive user preferences. The CSRec teacher module effectively guides the student recommender to produce recommendations that align with real positive user preferences despite corrupted training data, thereby enhancing robustness. Additionally, CSRec shows promise in mitigating popularity bias and performing robustly across varying sequence lengths and sparsity levels.

5.3 Performance of Teacher Modules (RQ3)

As the construction of teacher modules is a major contribution, we evaluate their performance to assess the quality of the soft labels they generate. We examine the evaluation results on overall

Table 5. The Performance of the Base Model and the Three *Teachers* from the Proposed Methods, i.e., CSRec-M, CSRec-D, and CSRec-T

		GRU4Rec				NARM			
Dataset	**Method**	RC (%)	RC^+ (%)	NG (%)	NG^+ (%)	RC (%)	RC^+ (%)	NG (%)	NG^+ (%)
Electronics	Base	5.46	5.93	2.99	3.24	5.18	5.54	3.03	3.25
	CSRec-M	**5.73**	**6.15**	**3.24**	**3.47**	5.72	6.14	3.38	3.61
	CSRec-D	5.41	5.85	3.00	3.23	**5.87**	**6.26**	**3.64**	**3.87**
	CSRec-T	5.62	6.03	3.17	3.40	5.53	5.93	3.25	3.49
Movies and TV	Base	8.48	8.61	4.79	4.91	9.59	9.90	5.68	5.96
	CSRec-M	**9.52**	**9.72**	**5.53**	**5.70**	**10.06**	**10.42**	**5.97**	**6.29**
	CSRec-D	8.70	8.93	4.94	5.13	9.11	9.33	5.28	5.48
	CSRec-T	8.49	8.69	4.72	4.89	9.02	9.26	5.17	5.36

Here, we investigate the results of GRU4Rec and NARM. NG is short for NDCG@10. RC is short for Recall@10. NF^+ is the filtered version metrics described in Section 4.1.2. Boldface indicates the highest score among different teachers and base models.

data and clean data, using the original metrics and the filtered metrics defined in Section 4.1.2. We also consider adverse metrics Recall^- and NDCG^-, which focus on disliked interactions, to determine if the model performs better by fitting noisy examples. The detailed results are presented in Table 5. Some teacher modules, such as the model-level teacher, achieve superior performance, indicating high-quality soft labels. While some teachers, such as the training-level teachers, show less promising results, they still guide the student models to better performance. This suggests that teachers do not need to excel in predicting ground truth items to provide helpful soft labels.

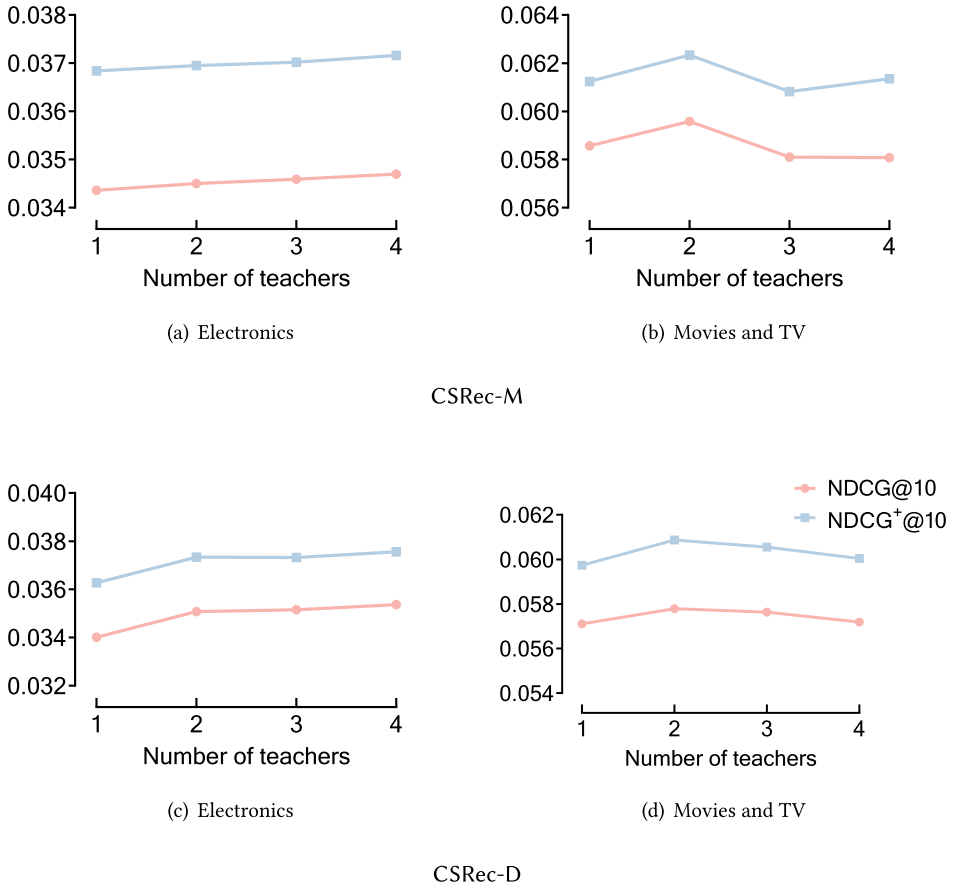
In short, the proposed teacher modules can generate high-quality confident soft labels when training on a clean sample, and give much more conservative guidance on noisy samples, which helps the students focus on the clean ones and achieve better performance.

5.4 Ablation Study

We investigate the impact of the teacher module, tradeoff coefficient, sub-sampling ratio, and robust training loss on performance.

Teacher Module. We analyze how the design of the teacher module affects student performance. Since CSRec-M and CSRec-D use multiple teachers, we first explore the effect of the number of teachers (i.e., m). The results, using NARM as the student model on two Amazon datasets, are shown in Figure 6. Results for other student models follow similar trends. Performance improves as the number of teachers increases from 1 to 2, suggesting that multiple teachers benefit the student. However, further increasing the number of teachers yields little improvement, indicating that more teachers do not necessarily provide better soft labels and recommendation results.

Tradeoff Coefficient. We examine how the tradeoff coefficient α in Equation (12), β in Equation (20), and the smoothing parameter $\mathcal{T}$ affect performance. Results for the GRU4Rec model are displayed in Figure 7, and for the SASRec model in Figure 8. For GRU4Rec, higher tradeoff parameter β significantly improves performance, highlighting the importance of soft label guidance during training. In contrast, for SASRec, lower β surpasses higher values at a small temperature but performs worse at higher temperatures. For both CSRec-M and CSRec-D, performance converges but slightly decreases as temperature increases. As noted in Section 3.3, when $\mathcal{T} \to \infty$, the soft label approximates label smoothing and provides less useful information. For the training-level



(a) Electronics (b) Movies and TV

(c) Electronics (d) Movies and TV

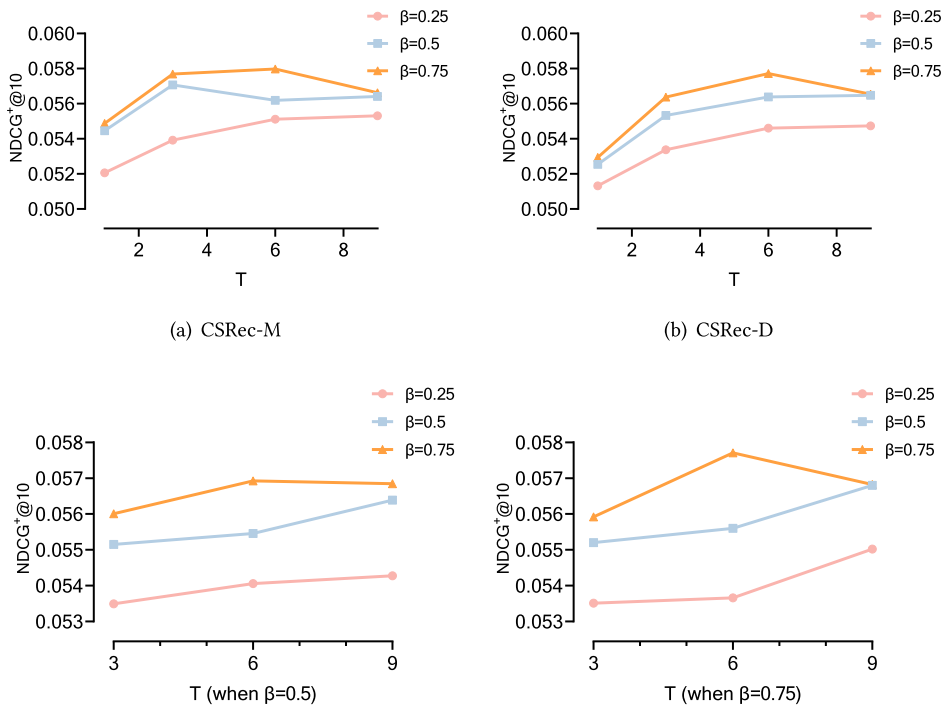
Fig. 6. Effect of the number of teacher models using NARM as the student model.

method, increasing α improves accuracy with $\beta = 0.75$ at some temperatures, while $\beta = 0.5$ shows a slight decrease.

Sub-Sampling Ratio. For CSRec-D, we also explore the effect of the sub-sampling ratio. Results using NARM as the student model are shown in Figure 9. A large sampling ratio (i.e., $p > 0.8$) does not always improve performance. Larger sub-datasets may have more overlapping samples, making the teacher modules more similar and sharing the same data-level bias and noise. Conversely, a very small sampling ratio introduces excessive randomness and degrades performance. A ratio of $p = 0.8$ is a robust and reliable choice.

Robust Training Loss. Finally, for CSRec-T, we investigate the effect of the robust training loss ℓ_r (Equation (12)). Since ℓ_r resembles the regular distillation loss except for the term $-\sum_{j\in\mathcal{I}} \log(h_{i,j}) P_{g_1}(j \mid \mathbf{s}_u)$, we assess the effectiveness of this expectation term and report the results in Figure 10, using NARM as the student model. We find that the expectation term significantly enhances the student's performance, demonstrating the effectiveness of our proposed methods.

In conclusion, there is a clear tradeoff in choosing hyper-parameters. Our hyper-parameter study highlights key factors that improve recommendation performance and demonstrates the effectiveness of the soft labels and proposed methods.



(a) CSRec-M

(b) CSRec-D

(c) CSRec-T

Fig. 7. The figures for model-level and data-level teachers, i.e., (a) and (b), show the effect of the temperature $\mathcal{T}$ and the tradeoff coefficient β in Equation (20). The figures for the training-level teacher, i.e., (c), show the effect of $\mathcal{T}$, β, and the tradeoff coefficient α in Equation (12), which is set to 0.50 and 0.75. The base model architecture is *GRU4Rec* and we conduct the experiment on dataset Movies and TV.

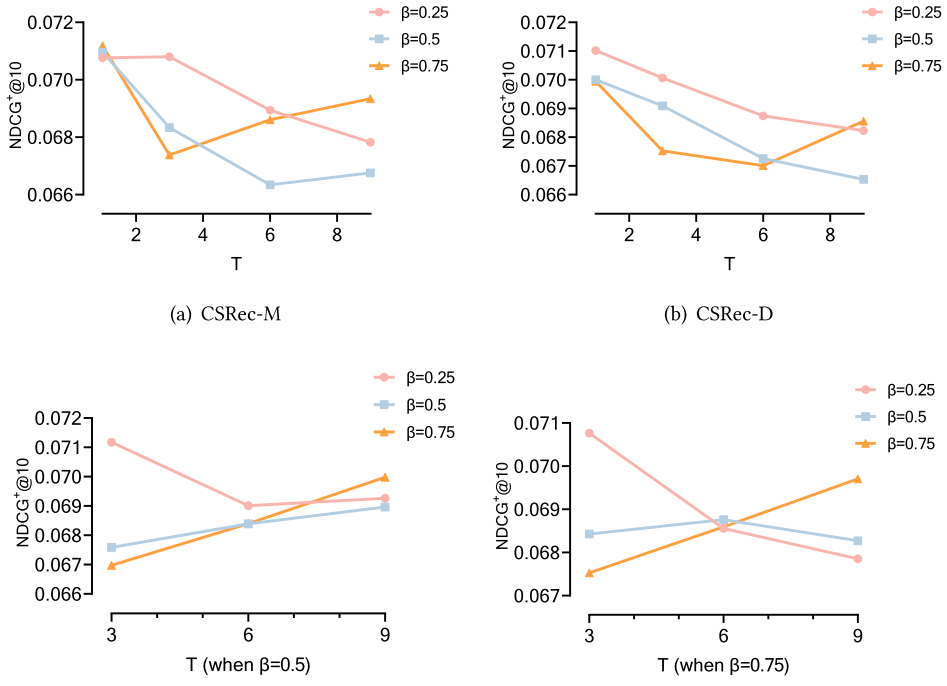
6 Conclusions

We have proposed a new learning framework, CSRec, to train a robust sequential recommender from noisy, implicit feedback. The key idea is to introduce confident, soft labels to provide robust guidance in the learning process. We have presented three teacher modules, i.e., model-level, data-level, and training-level, for generating high-quality and confident, soft labels from noisy user–item interactions. We have conducted extensive experiments to assess the effectiveness of the proposed learning framework.

Experimental results on four datasets and diverse student models demonstrate that training with the proposed learning framework CSRec helps to improve the recommendation performance. It can be applied to various deep-learning-based sequential recommendation models.

The broader implications of our work are that we have demonstrated the potential of soft labels for training sequential recommender systems, which we believe should be further investigated and developed as a generic framework for producing better soft label guidance for a broader set of recommendation tasks.

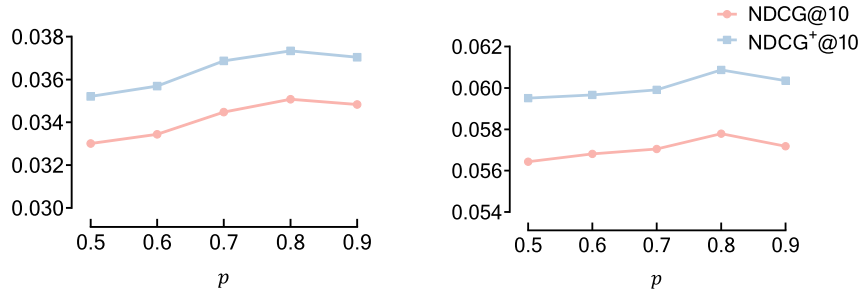
Limitations of our work concern the inefficiency of the whole training framework pipeline due to the required multiple well-trained teacher models. The lack of exploration of different ensemble methods for further analysis and comparison is also one of the limitations.



(a) CSRec-M (b) CSRec-D

(c) CSRec-T

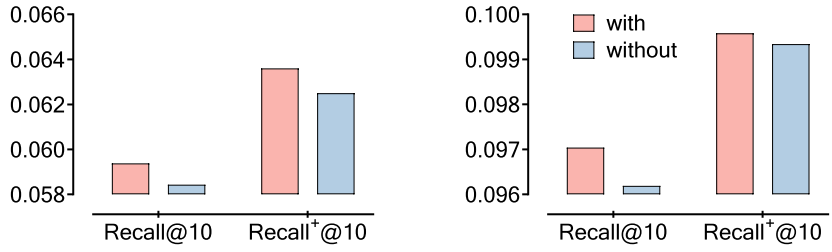
Fig. 8. The figures for model-level and data-level teachers, i.e., (a) and (b), show the effect of the temperature $\mathcal{T}$ and the tradeoff coefficient β in Equation (20). The figures for the training-level teacher, i.e., (c), show the effect of $\mathcal{T}$, β, and the tradeoff coefficient α in Equation (12), which is set to 0.50 and 0.75. The base model architecture is *SASRec* and we conduct the experiment on dataset Movies and TV.



(a) Electronics (b) Movies and TV

Fig. 9. Effect of the sub-sampling ratio p in CSRec-D, using NARM as the student model.

As to future work, we see many promising directions. First, more intuitive and general teacher modules could be designed beyond the ones we considered in this article, perhaps targeting specific types of noise or bias present in logged interaction data. Second, we aim to develop a more in-depth analysis to see how soft labels intrinsically affect the student model learning.



(a) Electronics (b) Movies and TV

Fig. 10. Effect of the term $-\sum_{j\in\mathcal{I}} \log(h_{i,j}) P_{g_1}(j \mid \mathbf{s}_u)$ in the robust loss ℓ_r of CSRec-T. Here, "with" and "without" denote whether the term is used or not. The student model used is NARM.

Acknowledgments

We thank our anonymous reviewers for their constructive feedback that helped us to improve our article. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

References

[1] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. 2021. Understanding Robustness of Transformers for Image Classification. In *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV '21)*, 10211–10221.

[2] Hugo Caselles-Dupré, Florian Lesaint, and Jimena Royo-Letelier. 2018. Word2vec Applied to Recommendation: Hyperparameters Matter. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*. Sole Pera, Michael D. Ekstrand, Xavier Amatriain, and John O'Donovan (Eds.), ACM, 352–356. DOI: https://doi.org/10.1145/3240323.3240377

[3] Jianxin Chang, Chen Gao, Y. Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential Recommendation with Graph Neural Networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 378–387.

[4] Chong Chen, Weizhi Ma, Min Zhang, Chenyang Wang, Yiqun Liu, and Shaoping Ma. 2022. Revisiting Negative Sampling vs. Non-Sampling in Implicit Recommendation. *ACM Transactions on Information Systems* (2022), Article No. 12. DOI: https://doi.org/10.1145/3522672

[5] Gang Chen, Jiawei Chen, Fuli Feng, Sheng Zhou, and Xiangnan He. 2023. Unbiased Knowledge Distillation for Recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23)*. DOI: https://doi.org/10.1145/3539597.3570477

[6] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2023. Bias and Debias in Recommender System: A Survey and Future Directions. *ACM Trans. Inf. Syst.* 41, 3 (2023), Article 67, 39 pages. DOI: https://doi.org/10.1145/3564284

[7] Jiawei Chen, Xiang Wang, Fuli Feng, and Xiangnan He. 2021. Bias Issues and Solutions in Recommender System: Tutorial on the RecSys 2021. In *Proceedings of the 15th ACM Conference on Recommender Systems*, 825–827.

[8] Mingyue Cheng, Fajie Yuan, Qi Liu, Shenyang Ge, Zhi Li, Runlong Yu, Defu Lian, Senchao Yuan, and Enhong Chen. 2021. Learning Recommender Systems with Implicit Feedback via Soft Target Enhancement. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 575–584.

[9] Ilias Diakonikolas and Daniel M. Kane. 2023. *Algorithmic High-Dimensional Robust Statistics*. Cambridge University Press.

[10] Hao Ding, Anoop Deoras, Yuyang (Bernie) Wang, and Hao Wang. 2022. Zero Shot Recommender Systems. In *Proceedings of the Workshop on Deep Generative Models for Highly Structured Data (ICLR '22)*. Retrieved from https://www.amazon.science/publications/zero-shot-recommender-systems

[11] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping. arXiv:2002.06305. Retrieved from https://doi.org/10.48550/arXiv.2002.06305

[12] Hanwen Du, Huanhuan Yuan, Zhen Huang, Pengpeng Zhao, and Xiaofang Zhou. 2023. Sequential Recommendation with Diffusion Models. arXiv:2304.04541. Retrieved from https://arxiv.org/abs/2304.04541

[13] Shangchen Du, Shan You, Xiaojie Li, Jianlong Wu, Fei Wang, Chen Qian, and Changshui Zhang. 2020. Agree to Disagree: Adaptive Ensemble Knowledge Distillation in Gradient Space. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, (NeurIPS '20)*. Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.), Retrieved from https://proceedings.neurips.cc/paper/2020/hash/91c77393975889bd08f301c9e13a44b7-Abstract.html

[14] Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge Distillation: A Survey. *International Journal of Computer Vision* 129, 6 (2021), 1789–1819. DOI: https://doi.org/10.1007/s11263-021-01453-z

[15] Ruining He and Julian McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *Proceedings of the 2016 IEEE 16th International Conference on Data Mining (ICDM '16)*, 191–200.

[16] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained Transformers Improve Out-of-Distribution Robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2744–2751. DOI: https://doi.org/10.18653/v1/2020.acl-main.244

[17] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-Based Recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 843–852.

[18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. arXiv:1503.02531. Retrieved from https://doi.org/10.48550/arXiv.1503.02531

[19] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large Language Models Are Zero-Shot Rankers for Recommender Systems. In *Advances in Information Retrieval: 46th European Conference on Information Retrieval (ECIR 2024)*. Springer-Verlag, Berlin, 364–381. DOI: https://doi.org/10.1007/978-3-031-56060-6_24

[20] Jun Hu, Wenwen Xia, Xiaolu Zhang, Chilin Fu, Weichang Wu, Zhaoxin Huan, Ang Li, Zuoli Tang, and Jun Zhou. 2024. Enhancing Sequential Recommendation via LLM-Based Semantic Embedding Learning. In *Companion Proceedings of the ACM on Web Conference 2024 (WWW '24)*. ACM, New York, NY, 103–111. DOI: https://doi.org/10.1145/3589335.3648307

[21] Kaixi Hu, Lin Li, Qing Xie, Jianquan Liu, Xiaohui Tao, and Guandong Xu. 2023. Decoupled Progressive Distillation for Sequential Prediction with Interaction Dynamics. *ACM Transactions on Information Systems* 42, 3, Article 72 (Dec. 2023), 35 pages. DOI: https://doi.org/10.1145/3632403

[22] Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. 2022. Do Loyal Users Enjoy Better Recommendations? Understanding Recommender Accuracy from a Time Perspective. In *Proceedings of the 2022 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '22)*. ACM, New York, NY, 92–97.

[23] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. In *Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM '18)*, 197–206.

[24] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conferenceon Learning Representations (ICLR '15)*. Yoshua Bengio and Yann LeCun (Eds.). arXiv:1412.6980. Retrieved from http://arxiv.org/abs/1412.6980

[25] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-Based Recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. Ee-Peng Lim, Marianne Winslett, Mark Sanderson, Ada Wai-Chee Fu, Jimeng Sun, J. Shane Culpepper, Eric Lo, Joyce C. Ho, Debora Donato, Rakesh Agrawal, Yu Zheng, Carlos Castillo, Aixin Sun, Vincent S. Tseng, and Chenliang Li (Eds.), ACM, 1419–1428. DOI: https://doi.org/10.1145/3132847.3132926

[26] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023. Text Is All You Need: Learning Language Representations for Sequential Recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)*. ACM, New York, NY, 1258–1267. DOI: https://doi.org/10.1145/3580305.3599519

[27] Zihao Li, Aixin Sun, and Chenliang Li. 2023. DiffuRec: A Diffusion Model for Sequential Recommendation. *ACM Transactions on Information Systems* 42, 3, Article 66 (Dec. 2023), 28 pages. DOI: https://doi.org/10.1145/3631116

[28] Dugang Liu, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2020. A General Knowledge Distillation Framework for Counterfactual Recommendation via Uniform Data. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. ACM, 831–840.

[29] Yuang Liu, Wei Zhang, and Jun Wang. 2020. Adaptive Multi-Teacher Multi-Level Knowledge Distillation. *Neurocomputing* 415 (2020), 106–113. DOI: https://doi.org/10.1016/j.neucom.2020.07.048

[30] Anjing Luo, Pengpeng Zhao, Yanchi Liu, Fuzhen Zhuang, Deqing Wang, Jiajie Xu, Junhua Fang, and Victor S. Sheng. 2020. Collaborative Self-Attention Network for Session-Based Recommendation. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI '20)*. Christian Bessiere (Ed.), ijcai.org, 2591–2597. DOI: https://doi.org/10.24963/ijcai.2020/359

[31] Naresh Nelaturi and Golagani Devi. 2019. A Product Recommendation Model Based on Recurrent Neural Network. *Journal Européen des Systèmes Automatisés* 52, 5 (2019), 501–507. DOI: https://doi.org/10.18280/jesa.520509

[32] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing Neural Networks by Penalizing Confident Output Distributions. arXiv:1701.06548. Retrieved from https://arxiv.org/abs/1701.06548

[33] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive Learning for Representation Degeneration Problem in Sequential Recommendation. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining (WSDM '22)*. ACM, New York, NY, 813–823. DOI: https://doi.org/10.1145/3488560.3498433

[34] Yuyang Ren, Zhang Haonan, Luoyi Fu, Xinbing Wang, and Chenghu Zhou. 2023. Distillation-Enhanced Graph Masked Autoencoders for Bundle Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23)*. ACM, New York, NY, 1660–1669. DOI: https://doi.org/10.1145/3539618.3591666

[35] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09)*. AUAI Press, Arlington, VA, 452–461. DOI: https://dl.acm.org/doi/10.5555/1795114.1795167

[36] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The Graph Neural Network Model. *IEEE Transactions on Neural Networks* 20, 1 (2009), 61–80. DOI: https://doi.org/10.1109/TNN.2008.2005605

[37] Hwanjun Song, Minseok Kim, Dongmin Park, and Jae-Gil Lee. 2022. Learning from Noisy Labels with Deep Neural Networks: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* 34, 11 (2022), 8135–8153.

[38] Jiajie Su, Chaochao Chen, Weiming Liu, Fei Wu, Xiaolin Zheng, and Haoming Lyu. 2023. Enhancing Hierarchy-Aware Graph Networks with Deep Dual Clustering for Session-Based Recommendation. In *Proceedings of the ACM Web Conference 2023 (WWW '23)*. ACM, New York, NY, 165–176. DOI: https://doi.org/10.1145/3543507.3583247

[39] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*. Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.), ACM, 1441–1450. DOI: https://doi.org/10.1145/3357384.3357895

[40] Yasufumi Takama, Jing-cheng Zhang, and Hiroki Shibata. 2021. Context-Aware Music Recommender System Based on Implicit Feedback. *Transactions of the Japanese Society for Artificial Intelligence* 36, 1 (2021), WI2–D_ 1–10. DOI: https://doi.org/10.1527/tjsai.36-1_wi2-d

[41] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. (DLRS '16)*. ACM, New York, NY, 17–22.

[42] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining (WSDM '18)*. Yi Chang, Chengxiang Zhai, Yan Liu, and Yoelle Maarek (Eds.), ACM, 565–573. DOI: https://doi.org/10.1145/3159652.3159656

[43] Ali Vardasbi, Harrie Oosterhuis, and Maarten de Rijke. 2020. When Inverse Propensity Scoring Does Not Work: Affine Corrections for Unbiased Learning to Rank. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20)*. ACM, 1475–1484.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*. Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.), 5998–6008. Retrieved from https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[45] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. Knowledge-Aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. ACM, New York, NY, 968–977. DOI: https://doi.org/10.1145/3292500.3330836

[46] Pengfei Wang, Jiansheng Chen, and Shaozhang Niu. 2018. CFSH: Factorizing Sequential and Historical Purchase Data for Basket Recommendation. *PLoS One* 13, 10 (2018), e0203191. DOI: https://doi.org/10.1371/journal.pone.0203191

[47] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising Implicit Feedback for Recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM '21)*. ACM, New York, NY, 373–381. DOI: https://doi.org/10.1145/3437963.3441800

[48] Yu Wang, Xin Xin, Zaiqiao Meng, Joemon M. Jose, Fuli Feng, and Xiangnan He. 2022. Learning Robust Recommenders through Cross-Model Agreement. In *Proceedings of the ACM Web Conference 2022 (WWW '22)*. ACM, New York, NY, 2015–2025. DOI: https://doi.org/10.1145/3485447.3512202

[49] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. Global Context Enhanced Graph Neural Networks for Session-Based Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. ACM, New York, NY, 169–178. DOI: https://doi.org/10.1145/3397271.3401142

[50] Nuan Wen and Fang Zhang. 2020. Extended Factorization Machines for Sequential Recommendation. *IEEE Access* 8 (2020), 41342–41350.

[51] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence and 31st Innovative Applications of Artificial Intelligence Conference and 9th AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI '19/IAAI '19/EAAI '19)*. AAAI Press, Article 43, 8 pages. DOI: https://doi.org/10.1609/aaai.v33i01.3301346

[52] Lianghao Xia, Chao Huang, Jiao Shi, and Yong Xu. 2023. Graph-Less Collaborative Filtering. In *Proceedings of the ACM Web Conference 2023 (WWW '23)*. ACM, New York, NY, 17–27. DOI: https://doi.org/10.1145/3543507.3583196

[53] Lianghao Xia, Chao Huang, and Chuxu Zhang. 2022. Self-Supervised Hypergraph Transformer for Recommender Systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*. ACM, New York, NY, 2100–2109. DOI: https://doi.org/10.1145/3534678.3539473

[54] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Guandong Xu, and Quoc Viet Hung Nguyen. 2022. On-Device Next-Item Recommendation with Self-Supervised Knowledge Distillation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. ACM, New York, NY, 546–555. DOI: https://doi.org/10.1145/3477495.3531775

[55] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Bolin Ding, and Bin Cui. 2021. Contrastive Learning for Sequential Recommendation. arXiv:2010.14395. Retrieved from https://arxiv.org/abs/2010.14395

[56] Xin Xin, Fajie Yuan, Xiangnan He, and Joemon M. Jose. 2018. Batch IS NOT Heavy: Learning Word Representations from All Samples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Long Papers, Vol. 1, Association for Computational Linguistics, Melbourne, Australia, 1853–1862. DOI: https://doi.org/10.18653/v1/P18-1172

[57] Keyang Xu, Tongzheng Ren, Shikun Zhang, Yihao Feng, and Caiming Xiong. 2021. Unsupervised Out-of-Domain Detection via Pre-Trained Transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, Long Papers, Vol. 1, 1052–1061.

[58] Yeongwook Yang, Hong Jun Jang, and Byoungwook Kim. 2020. A Hybrid Recommender System for Sequential Recommendation: Combining Similarity Models with Markov Chains. *IEEE Access* 8 (2020), 190136–190146.

[59] Zhengyi Yang, Jiancan Wu, Zhicai Wang, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. Generate What You Prefer: Reshaping Sequential Recommendation via Guided Diffusion. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23)*. Curran Associates Inc., Red Hook, NY, Article 1054, 15 pages.

[60] Shan You, Chang Xu, Chao Xu, and Dacheng Tao. 2017. Learning from Multiple Teacher Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1285–1294.

[61] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M. Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining (WSDM '19)*. J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman (Eds.), ACM, 582–590. DOI: https://doi.org/10.1145/3289600.3290975

[62] Stone Yun and Alexander Wong. 2020. Where Should We Begin? A Low-Level Exploration of Weight Initialization Impact on Quantized Behaviour of Deep Neural Networks. arXiv:2011.14578. Retrieved from https://doi.org/10.48550/arXiv.2011.14578

[63] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018. Deep Mutual Learning. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4320–4328.

[64] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al. 2020. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. DOI: https://doi.org/10.48550/ARXIV.2011.01731

[65] Zihao Zhao, Jiawei Chen, Sheng Zhou, Xiangnan He, Xuezhi Cao, Fuzheng Zhang, and Wei Wu. 2021. Popularity Bias Is Not Always Evil: Disentangling Benign and Harmful Bias for Recommendation. *IEEE Trans. on Knowl. and Data Eng.* 35, 10 (Oct. 2023), 9920–9931. DOI: https://doi.org/10.1109/TKDE.2022.3218994

[66] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed H. Chi. 2019. Recommending What Video to Watch Next: A Multitask Ranking System. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys '19)*. Toine Bogers, Alan Said, Peter Brusilovsky, and Domonkos Tikk (Eds.), ACM, 43–51. DOI: https://doi.org/10.1145/3298689.3346997

[67] Xiaolin Zheng, Jiajie Su, Weiming Liu, and Chaochao Chen. 2022. DDGHM: Dual Dynamic Graph with Hybrid Metric Training for Cross-Domain Sequential Recommendation. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22)*. ACM, New York, NY, 471–481. DOI: https://doi.org/10.1145/3503161.3548072

[68] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM '20)*. ACM, New York, NY, 1893–1902. DOI: https://doi.org/10.1145/3340531.3411954

[69] Xiong Zhou, Xianming Liu, Chenyang Wang, Deming Zhai, Junjun Jiang, and Xiangyang Ji. 2021. Learning with Noisy Labels via Sparse Regularization. In *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV '21)*, 72–81.

[70] Jinghua Zhu, Xu Guo, and Shengchao Ma. 2018. Attentive Neural Network Recommendation for E-Commerce Based on Various Implicit Feedback. In *Proceedings of the 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS '18)*, 889–892.

Received 25 January 2024; revised 21 August 2024; accepted 29 September 2024