CLC ___________ Number___________

UDC___________ Available for reference □Yes □No

SUSTech Southern University of Science and Technology

Undergraduate Thesis

Thesis Title: **DART: Resolving Training Biases for Recommender Systems via Data Discarding and Relabeling**

Student Name: **Ningzhi Tang**

Student ID: **11912521**

Department: **Computer Science and Engineering**

Program: **Intelligent Science and Technology**

Thesis Advisor: **Chair Professor Yuhui Shi**

Date: June 2, 2023

Letter of Commitment for Integrity

1. I solemnly promise that the paper presented comes from my independent research work under my supervisor’s supervision. All statistics and images are real and reliable.

2. Except for the annotated reference, the paper contents no other published work or achievement by person or group. All people making important contributions to the study of the paper have been indicated clearly in the paper.

3. I promise that I did not plagiarize other people’s research achievement or forge related data in the process of designing topic and research content.

4. If there is violation of any intellectual property right, I will take legal responsibility myself.

Signature: 唐道

Date: 2023年6月2日

DART: Resolving Training Biases for Recommender Systems via Data Discarding and Relabeling

Ningzhi Tang

(Computer Science and Engineering　Department Tutor: Yuhui Shi)

[**ABSTRACT**]: Implicit feedback, such as clicks and purchases, is commonly used to train recommender systems to provide personalized recommendation services. However, the collected training data may not always accurately reflect users' true preferences. For example, a user may be attracted to click on news articles with flashy headlines or play music continuously without much thought. These biases in the training data often manifest as false-positive interactions and can lead to reduced recommendation performance. To alleviate this challenge, this paper proposes DART, a model-agnostic strategy that can be seamlessly integrated into existing recommendation models for resolving such training biases. We identify interactions with a high training loss as biased samples and address them by utilizing a combination of discarding and relabeling techniques. To this end, we conducted extensive offline click-through rate prediction experiments on four public datasets in various recommendation scenarios. Our results demonstrate that our method can accurately identify false-positive interactions and achieve better recommendations, as well as provide evidence for the validity and robustness of our strategy.

[**Key words**]: Recommender systems; Implicit feedback; Click-through rate prediction; Training biases

[摘要]：隐式反馈，如点击和购买，常用于训练推荐系统以提供个性化的推荐服务。然而，它们可能并不总是准确反映用户的真实偏好。例如，用户可能会被花哨的标题吸引而点击新闻文章，或者无意识地连续播放音乐。这些在训练数据中的偏差通常表现为假阳性的交互，并可能导致推荐性能降低。为了解决这个问题，我们的论文提出 DART，一种基于神经网络记忆过程的模型独立的策略，可以无缝地集成到现有的推荐模型中以解决训练偏差问题。我们通过识别具有高训练损失值的交互作为有偏样本，并利用丢弃和重标签技术的组合来处理它们。为此，我们在四个不同的涉及各种推荐场景的公共数据集上，进行了大量的点击率预测的离线实验。我们的结果表明，DART 可以准确识别假阳性交互并实现更好的推荐。实验结果同时为我们的策略的合理性和鲁棒性提供了证据。

[关键词]：推荐系统；隐式反馈；点击率预测；训练偏差

Content

1. Introduction

Recommender systems play a crucial role in providing personalized recommendations to users in navigating vast amounts of data, which are widely deployed in online services such as E-commerce[1], social networks[2], and video-sharing platforms[3]. However, training these systems presents a significant challenge due to data sparsity[1]. With millions of users and items, there exist limited interactions between them[2]. Furthermore, explicit feedback such as ratings and comments is scarce and difficult to collect[4]. To address this problem, implicit feedback (e.g., click and purchase) has become the default choice for training real-world recommender systems due to the significant volume of data available, which could effectively alleviate the data sparsity issues[4].

Nevertheless, this approach introduces additional training biases due to the presence of label noise in implicit feedback[5], which makes it challenging to accurately reflect users' real preferences. This noise arises particularly in the form of false-positive interactions. For instance, in e-commerce, a purchase may result in an unhappy usage experience instead of a positive one. Similarly, in the news or video recommendation systems, a click with short reading or watching time may not indicate actual user interest[5]. Users may be influenced by other factors, such as the first impression of attractive captions[6], to make unintentional clicks. The issue is highly critical in music recommendation scenarios, as music is usually played in the background and users tend to provide limited explicit feedback such as likes and dislikes[7]. The biases can lead to a distribution shift between the train distribution $P(x, y)$ and target distribution $Q(x, y)$, where x is the feature vector and y is the label[8]. This shift can ultimately reduce the accuracy of the recommender system, making it challenging to provide personalized recommendations to users.

Moreover, if these biases are not mitigated, they can reinforce the "closed feedback loop problem"[9], which could further reduce online prediction performance due to label noise. In this scenario, the recommender system generates items that users might be interested in and

[1] https://content-garden.com/click-through-rate-prediction

trains further recommendation models with data from users' feedback on those items. Despite the challenges, there has been limited research on addressing this issue. Many existing approaches require additional user data such as dwell time[10-11] and gaze[12] to identify noisy samples, which can be difficult to obtain in practice. In other fields, such as computer vision, researchers have explored various methods to improve model robustness, including analyzing and modifying training loss[13-15], or using the influence function[16-17]. Similar work in recommendation systems either has high computational complexity[9], or simply modifies the loss function without explicitly detecting noise to improve system transparency[7-8].

Therefore, we propose DART[2], a new framework for resolving training biases to denoise false-positive interactions in real-world data that results in more robust and personalized recommendations. The framework, as shown in Figure 1, consists of two stages: biases identification and biases handling. During the **biases identification** stage, we model positive interactions with higher loss as noisy samples by utilizing a dynamically adjusted identification threshold. This approach is inspired by previous research on deep network memorization[18] and curriculum learning[19], which suggests that neural networks learn easier samples more effectively. After identifying these noisy samples, in the **biases handling** stage, we use a combination of discarding and relabeling methods to improve the quality of our training data. Our framework is model-agnostic, making it easy to integrate into most of the existing recommendation tasks.

Our approach draws inspiration from the idea of improving the robustness of models in computer vision. We are pioneers in applying the idea of explicitly identifying and handling biases to click-through rate (CTR) prediction tasks in recommendation systems. In the biases identification stage, we introduce a new, more dynamic identification threshold that differs from previous work, which provides a larger search space for optimization. In the biases handling stage, we organically combine the discarding and relabeling techniques for the first time, to enhance the diversity of our processing methods and fully leverage the advantages

[2] **DART** 🎯 is the acronym for **D**iscarding **A**nd **R**elabeling **T**raining biases, which also refers to a small pointed missile that can be thrown or fired, is indicative of the accuracy of our method in identifying biases samples.

of both, which is simple yet effective.

We conducted extensive experiments on four different recommendation datasets and used two classic recommendation system models to test the performance of our method and various baselines. We investigated multiple research questions, as stated in Section 4, and demonstrated through experimentation that DART can accurately identify noise (RQ4), improve the model's memory process (RQ2), enhance recommendation performance (RQ1), and not harm recommendations for inactive users (RQ5). We also validated the rationality of using model samples with higher training loss as false-positive interactions (RQ3). We also conducted a sensitivity analysis of hyperparameters to test the robustness of DART. Our code has been made open source and is available at https://github.com/TTangNingzhi/CS490-DART.

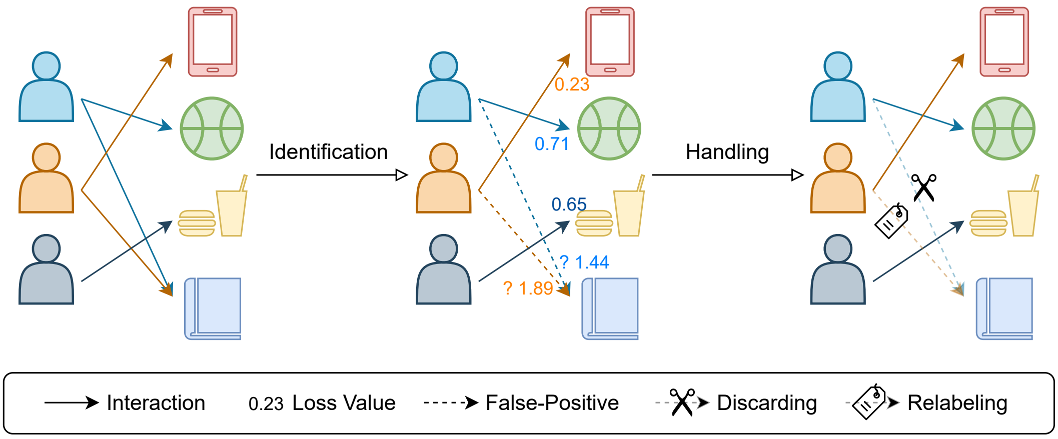


Figure 1 The framework for DART: resolving training biases with biases identification and handling. Positive interactions with higher loss are identified as noisy samples, which are then either discarded or relabeled to improve the training data quality.

In this work, we make the following contributions:

- We conduct a comprehensive survey of existing literature on robust learning in recommender systems. Specifically, we analyze the significance of addressing the training biases problem in the recommendation and review the current denoising methods for this purpose.

- We propose DART, a new framework that aims to resolve training biases from implicit feedback in recommendation systems. It is based on previous research and is model-

and dataset-agnostic, which means that our approach can be effortlessly integrated into existing training frameworks.

- We conduct extensive experiments to demonstrate the effectiveness of the proposed DART on two classic recommendation models on four datasets, specifically under the task of CTR prediction, which outperforms all previous baselines. Furthermore, we conduct an in-depth analysis of the training process to verify its rationality.
- We conducted a comprehensive sensitivity analysis to demonstrate the general robustness of DART. Additionally, we thoroughly discussed the limitations of our method and potential design ideas and optimization directions for future work.

The rest of this paper is organized as follows. Section 2 discusses the relevant literature. Section 3 provides a detailed introduction to the proposed DART framework, while Section 4 describes our experiments. The sensitivity analysis of DART is presented in Section 5, followed by the discussion in Section 6 and the conclusion in Section 7.

2. Related Work

This section provides an overview of the main related works in the field of recommender systems. It covers classical recommender systems, with a particular focus on those trained using deep learning techniques. Additionally, it discusses recent advancements in robust deep learning for recommender systems, including both loss- and influence-based robust learning.

2.1 Recommender Systems

Recommender systems are a popular technique for helping users filter out irrelevant information and discover new items of interest. Collaborative filtering[4] is one of the most widely used approaches, which recommends items to users based on their past interactions with similar users or items. Factorization machines (FM)[20] extend this idea by modeling pairwise interactions between user and item features using low-dimensional embeddings.

In recent years, deep learning-based recommender systems have gained popularity due to their ability to capture complex patterns in user-item interactions. NeuMF[21] makes a combination of matrix factorization with multi-layer perceptrons to learn user and item embeddings. Wide&Deep[22] is a hybrid model that combines linear regression with deep neural networks to capture both shallow and deep feature interactions. DeepFM[23] uses factorization machines to model pairwise feature interactions and a deep neural network to capture higher-order feature interactions.

These models can be adapted to various recommendation scenarios depending on the task requirements. Despite their effectiveness in providing accurate recommendations, they often suffer from training biases that can lead to inaccurate recommendations[5,8]. Therefore, it is crucial to develop methods to mitigate these challenges and ensure that recommender systems provide unbiased recommendations that reflect the user's real preferences.

2.2 Robust Recommendation

Traditionally, there are two main approaches to mitigating training biases, particularly label noise in implicit feedback[8]. One approach is to develop a separate model that predicts false-positive interactions using additional user behaviors such as dwell time[10] and gaze patterns[12]. Another approach is to directly incorporate these behaviors into the training process to reduce the impact of noise[11,24]. For example, Wen *et al.*[25] leverage post-click feedback such as skips and completions to improve the training and evaluation of content recommendations. However, both methods require additional user data, which can be difficult to collect and may exacerbate data sparsity issues. For instance, many users do not provide feedback such as likes or comments after clicking.[8] Therefore, it is important to develop methods for resolving training biases without relying on extra feedback.

2.2.1 Loss-Based Robust Learning

The machine learning community has developed several strategies for training models robustly with noisy samples, which can provide valuable insights for designing bias-resolving frameworks. For example, Bootstrap[13] revises the loss function to focus on the most confident predictions and pay less attention to inconsistent labels. F-correction[26] corrects predictions by estimating a noise transition matrix using a backward and forward procedure. S-model[27] introduces a noise adaptation layer that estimates the probability of each noisy label given the correct label and optimizes it using an EM algorithm. MentorNet[14] dynamically learns a data-driven curriculum that provides sample weights focusing on the correct samples. Co-teaching[15] trains two networks simultaneously and has them teach each other to determine which data should be used for training.

All of these works take into account the impact of training loss when designing their methods. They are typically implemented by either revising the loss function directly or by identifying specific noisy samples. For example, Dai *et al.*[7] propose a dynamic weighting scheme that utilizes model loss for bootstrap to achieve noise-corrected music recommendations. Wang *et al.*[8] raise a training strategy called Adaptive Denoising Training (ADT), which prunes noisy interactions in recommendation with truncated or reweighted loss. Furthermore, the experiments conducted in prior studies[8,14-15] demonstrate an important implication: training samples with larger losses, which are more difficult to memorize, are more likely to be noisy samples. Our work follows this idea and uses it to design and implement a framework for resolving biases in recommender systems.

2.2.2 Influence-Based Robust Learning

The influence function[16] is another important concept in robust learning that was introduced by Pang Wei Koh and Percy Liang. It is used to assess the impact of training samples on validation loss, which can be measured either element-wise[16] or group-wise[28]. The influence is calculated using the formula[16]:

$$\phi(z_i, z_j) \triangleq \frac{dl(z_j, \hat{\theta}_\epsilon)}{d\epsilon}|_{\epsilon=0} = -\nabla_\theta l(z_j, \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta l(z_i, \hat{\theta}) \tag{1}$$

Here, z_i and z_j are training and validation samples, respectively. The Hessian matrix $H_{\hat{\theta}}$ is assumed to be positive definite. $\hat{\theta}_\epsilon$ represents the optimal parameter after a specific training sample z_i is upweighted by a small perturbation ϵ, which is defined as $\hat{\theta}_\epsilon = \arg\min_{\theta\in\Theta} L(z,\theta) + \epsilon l\,(z_s,\theta)$.

The influence function has been used to reweight training samples in low-rank recommendation models[8], including logistic regression (LR)[29] and FM[20]. However, despite its theoretical guarantee and effectiveness[17], there are several challenges that limit its applicability in real-world recommender systems. Firstly, computing the second-order Hessian matrix is expensive[16-17], which can be a bottleneck for deep neural networks. Secondly, a clean validation set is required for evaluation, which may not be feasible in noisy data scenarios. Finally, estimating influence in deep neural networks may not be accurate due to their complex non-linear structure[16]. Therefore, we are focusing on investigating loss-based learning methods to improve unbiased recommendation performance.

3. Methodology

This section details DART, the training biases-resolving framework that we proposed for the recommendation. As shown in Figure 1, the framework is separated into the biases identification stage and the biases handling stage. Prior to that, the task statement is introduced.

3.1 Task Statement

In this study, we use click-through rate (CTR) prediction[30] as the recommendation task to evaluate the effectiveness of various biases-resolving strategies. The training dataset used is noisy and denoted by $\bar{\mathcal{D}} = \{(\mathbf{x}_i, \bar{y}_i)\}_{i=1}^{N}$, where $\mathbf{x}$ represents a feature vector and $\bar{y}_i$ is a binary variable indicating whether the user clicked (1) or did not click (0) on the item. The total number of training samples is N. The goal is to learn a function $p(\mathbf{x}|\Theta) : \mathcal{X} \rightarrow \mathcal{Y}$ that predicts the CTR value. To achieve this, we aim to obtain optimal parameters Θ^* by minimizing the binary cross-entropy (BCE) loss over $\bar{\mathcal{D}}$.

$$\mathcal{L}_{BCE}(\bar{\mathcal{D}}|\Theta^*) = -\frac{1}{N} \sum_{(\mathbf{x},\bar{y})\in\bar{\mathcal{D}}} (\bar{y}\log p(\mathbf{x}|\Theta^*) + (1-\bar{y})\log(1-p(\mathbf{x}|\Theta^*))) \tag{2}$$

It is worth noting that the training dataset may contain a significant number of false-positive interactions that do not accurately reflect users' true preferences. To address this issue, we assume the existence of a clean dataset $\{(\mathbf{x}_i, y_i^*)\}_{i=1}^N$ that represents users' true preferences. False-positive interactions can be formally represented as $\{(\mathbf{x}_i, \bar{y}_i, y_i^*)|\bar{y}_i = 1 \wedge y_i^* = 0\}$. Our work aims to identify and handle these false-positive interactions in order to improve the performance of the recommendation system.

3.2 Training Biases Resolving

3.2.1 Biases Identification

Numerous previous studies have extensively investigated the memorization process[18] of deep neural networks. They have concluded that "*while deep networks are capable of memorizing noisy data, our results suggest that they tend to prioritize learning simple patterns first.*" Based on the observation, several works on robust learning have leveraged it to design their denoising strategies[8,15]. The experiments conducted in these studies demonstrate the validity of this finding. In addition, research on curriculum learning[14,19] has shown that "*humans and animals learn much better when the examples are not randomly presented but organized in a meaningful order which illustrates gradually more concepts, and gradually more complex ones.*" This finding has inspired us to consider the importance of organizing interactions with different levels of difficulty during the learning process.

Based on this observation, we assume that *noisy data are more challenging to fit into the networks during training*. To identify these noisy interactions, we propose to use the training loss as a proxy and assume that interactions with larger training loss are more likely to be false positives. However, since the loss value is decreasing with training iteration T, using a fixed threshold may not work well. Therefore, we model the threshold τ as a dynamically adjusted handling rate related to T. Based on previous research[8,17] and our experiment experience, the design of $\tau(T)$ should have three key characteristics: (1) it needs to have an upper bound to prevent excessive data loss, (2) it needs to reach the upper bound from zero gradually, and (3) it should have a threshold value for T only after which we start to identify false-positive interactions.

By doing so, we can effectively distinguish between noisy and clean data while minimizing the risk of losing valuable information during the learning process. The identification threshold $\tau(T)$ is proposed as follows:

$$\tau(T) = \max\left\{\min\left\{\frac{T - T_i}{T_m} \cdot \tau_m, \tau_m\right\}, 0\right\}, \tag{3}$$

where τ_m is an upper bound that limits the threshold, T_m controls the pace at which the threshold reaches its maximum value, and T_i controls the iteration at which we begin identifying false-positive interactions. In iteration T with the mini-batch data $\bar{\mathcal{D}}_T$, we begin by extracting all positive interactions, denoted as

$$\bar{\mathcal{D}}_T^+ = \{(\mathbf{x}_i, \bar{y}_i)|(\mathbf{x}_i, \bar{y}_i) \in \bar{\mathcal{D}}_T \wedge \bar{y}_i = 1\}. \tag{4}$$

We then proceed to identify any false-positive interactions using the following approach:

$$\hat{\mathcal{D}}_T = \underset{\hat{\mathcal{D}} \in \bar{\mathcal{D}}_T^+, |\hat{\mathcal{D}}| \leq \tau(T) \cdot |\bar{\mathcal{D}}_T^+|}{\arg\max} \mathcal{L}_{BCE}(\hat{\mathcal{D}}|\Theta) \tag{5}$$

which is the top $\tau(T)$ percent of interactions with the highest loss values.

3.2.2 Biases Handling

After identifying the subset of interactions that may contain biases, we developed a biases handling strategy that combines discarding and relabeling. **(1) Discarding** means removing these biased interactions directly from the mini-batch set[8]; **(2) Relabeling**, which involves flipping the identified false-positive samples to negative samples and returning them to the mini-batch for further training.

Both discarding and relabeling strategies have been demonstrated to be effective in other machine learning domains[15,17]. However, due to the limitations of biases identification accuracy, these strategies may result in misidentification of true-positive interactions. We observed that samples with relatively low loss are more likely to be misidentified than those with high loss. Additionally, relabeling can be used as an augmentation technique to utilize false-positive interactions in training rather than simply removing them. It can also help the model forget any memorization of these false interactions that may have occurred during early stages of training. However, if true-positive samples are relabeled, the resulting harm may be greater than that caused by discarding. Therefore, there should be a trade-off between utilizing false-positive interactions and preserving true-positive interactions.

We set a discard/relabel percentage rate r, where the top-r identified samples are relabeled and the bottom-$(1-r)$ are discarded. Our experiments demonstrate the effectiveness of this approach. For simplicity, we typically set r to 0.5, but it can be optimized based on specific recommender system requirements. We calculate $\hat{\mathcal{D}}_T^{Rel}$ and $\hat{\mathcal{D}}_T^{Disc}$ using the following equations.

$$\hat{\mathcal{D}}_T^{Rel} = \underset{\hat{\mathcal{D}} \in \hat{\mathcal{D}}_T, |\hat{\mathcal{D}}| \leq r \cdot |\hat{\mathcal{D}}_T|}{\arg\max} \mathcal{L}_{BCE}(\hat{\mathcal{D}}|\Theta) \tag{6}$$

$$\hat{\mathcal{D}}_T^{Disc} = \hat{\mathcal{D}}_T - \hat{\mathcal{D}}_T^{Rel} \tag{7}$$

To the best of our knowledge, we are the first to organically combine these two strategies and apply them in recommender systems. The overall process is described in Algorithm 1.

Algorithm 1 The proposed DART algorithm

```
Input: training dataset $\bar{\mathcal{D}}$, initial model parameters $\Theta_0$, iteration number $T_{\max}$, loss function
  $\mathcal{L}_{BCE}$, hyperparameters $T_i, T_m, \tau_m$, discard/relabel threshold $r$.
Output: optimized model parameters $\Theta_{T_{\max}}$
  for $T \leftarrow 1$ to $T_{\max}$ do
    Update threshold $\tau(T)$ using hyperparameters $T_i, T_m, \tau_m$          ▷ Equation (3)
    Sample mini-batch data $\bar{\mathcal{D}}_T$ from $\bar{\mathcal{D}}$
    Fetch positive interactions $\bar{\mathcal{D}}_T^+$                           ▷ Equation (4)
    Identify false-positive interactions $\hat{\mathcal{D}}_T^+$ with threshold rate $\tau(T)$   ▷ Equation (5)
    Fetch interactions that need to be relabeled $\hat{\mathcal{D}}_T^{Rel}$ with rate $r$       ▷ Equation (6)
    Obtain remaining interactions that need to be discarded $\hat{\mathcal{D}}_T^{Disc}$       ▷ Equation (7)
    Define relabeled data $\hat{\mathcal{D}}_T^{Rel*} \leftarrow \{(\mathbf{x}_i, 0) | (\mathbf{x}_i, 1) \in \hat{\mathcal{D}}_T^{Rel}\}$
    Obtain handled data $\bar{\mathcal{D}}_T^* \leftarrow (\bar{\mathcal{D}}_T - \hat{\mathcal{D}}_T^{Disc} - \hat{\mathcal{D}}_T^{Rel}) \cup \hat{\mathcal{D}}_T^{Rel*}$
    Optimize $\Theta_T$ from $\Theta_{T-1}$ using $\bar{\mathcal{D}}_T^*$ and $\mathcal{L}_{BCE}$
  end for
```

One potential concern with handling high-loss samples is that it may limit the recommender system's ability to learn from valuable but challenging interactions. Additionally, some true positive samples may be misidentified and prevented from being used in training. In fact, balancing learning and biases-resolving is a trade-off when handling high-loss samples[8], which is controlled by the threshold rate $\tau(T)$. Our experiment also reveals the importance of achieving high precision in identifying biases during the denoising process. Then, we attempted to mitigate the harm caused by misidentified samples by setting the discard/relabel rate r, which was also proven effective in our experiments.

4. Expriments

This section presents a series of extensive experiments designed to address the following research questions (RQs):

- **RQ1:** How does our proposed DART compare to the base noisy training and other state-of-the-art baselines in terms of performance?
- **RQ2:** How does the memorization process of true-positive and false-positive interactions change when biases are resolved compared to when they are not?
- **RQ3:** How reasonable is it to conduct biases identification based on the magnitude of training loss?
- **RQ4:** How do the precision and recall values for false-positive interactions in the bias identification process?
- **RQ5:** To what extent does the design of DART impact the learning of preferences for inactive users?

4.1 Experimental Settings

4.1.1 Datasets

We conducted offline experiments using three widely-used public datasets: MovieLens-100K[31], Adressa[32], Book-Crossing[33], and Jester[34]. They cover a variety of recommendation scenarios. To adapt it for CTR prediction, we consider the existence of interaction records as positive samples and perform negative sampling during the training process.

- **MovieLens-100K[31]:** It contains 100,000 movie ratings (1-5) from 943 users on 1,682 movies, with each user rating at least 20 movies. The data was collected through the MovieLens website[3] over a seven-month period in 1997-1998. We consider ratings greater than or equal to 3 as true-positive samples that reflect users' true preferences, and ratings less than 3 as false-positive samples.
- **Adressa[32]:** The dataset is a real-world news reading dataset that includes news articles in Norwegian from Adressavisen[4]. It contains both the users' clicking and dwell time information, making it a valuable resource for studying user behavior in news reading. To distinguish between true-positive and false-positive samples, we treat clicks with a dwell time of less than 10 seconds as false-positive ones[8].
- **Book-Crossing[33]:** The dataset is sourced from the Book-Crossing community[5] and contains book rating information. Ratings, denoted as 'Book-Rating', are expressed on a scale from 1-10, with higher values indicating higher appreciation. Implicit ratings are expressed as 0. We consider ratings less than or equal to 6 as false positives and ratings greater than 6 as true positives.
- **Jester[34]:** The dataset is collected from the Jester online joke recommender system[6], where users anonymously rate jokes on a scale ranging from -10.00 to +10.00. To distinguish between true-positive and false-positive samples, we consider ratings less than 0 as false positives that should not be recommended later, and ratings greater than 0 as true positives.

Table 1 provides detailed information about each dataset. It is evident that false positive interactions constitute a significant portion of the training data in recommendation systems. However, they do not accurately reflect user preferences, and we need to resolve them appropriately. Nevertheless, this information cannot be leaked during training. We only use a clean

[3] https://movielens.org/

[4] https://www.adressa.no/

[5] https://www.bookcrossing.com/

[6] http://eigentaste.berkeley.edu/

Table 1 The statistics of datasets. TP and FP are abbreviations for True Positive and False Positive, respectively.

Dataset	#User	#Item	#TP	#FP	#FP/(#TP+#FP)
MovieLens-100K	943	1,682	55,375	44,625	0.446
Adressa	207,801	6,154	182,495	290,257	0.614
Book-Crossing	77,805	185,973	326,344	107,326	0.248
Jester	54,905	150	1,218,920	623,450	0.338

test set to verify the effectiveness of our method while keeping the training and validation sets intact. As in previous work[2], we randomly split each dataset into training, validation, and test sets with an 8:1:1 ratio.

4.1.2 Baselines

We evaluate the effectiveness of the proposed DART and other baseline methods on two classical CTR models: factorization machine (FM)[20] and DeepFM[23]. We implement both models using the `torchfm`[7] repository. We compare our strategy with the following baselines, all implemented using their open-source code (Bootstrap[8], LCD[9], ADT[10]) and the same hyperparameter settings.

- **Base:** This refers to direct training with the biased training set without any bias-resolving strategy.
- **Clean:** This refers to training with a clean training and validation set that filters out false-positive interactions. It utilizes additional explicit feedback and serves only as a reference.
- **Bootstrap[13]:** Bootstrap uses a weighted combination of predicted and original labels as corrected labels. The **Hard** version uses binary predicted labels, while the **Soft** version uses the output value passed through the sigmoid function.

[7] https://github.com/rixwew/pytorch-fm

[8] https://github.com/vfdev-5/BootstrappingLoss

[9] https://gitee.com/mindspore/models/tree/master/official/recommend/lcd

[10] https://github.com/WenjieWWJ/DenoisingRec

- **LCD[7]: LCD** ensembles noisy labels and model outputs using a dynamic weighting scheme based on the model loss to perform effective label correction. **LCD-Re** reverses the weights of outputs and labels that are calculated based on loss.

- **ADT[8]:** ADT dynamically prunes large-loss interactions during training, which is similar to our approach. The **CE-R** version reweights training samples based on prediction scores, while the **CE-T** version truncates high-loss samples, which is similar to our discarding strategy, but we utilized a more dynamic threshold and employed a combination of discarding and relabeling.

4.1.3 Evaluation Metrics

CTR prediction involves predicting whether a user will click on an item or not, which can be viewed as a binary classification problem. To evaluate the performance of the model, we use AUC as the evaluation metric[2]. AUC measures how well the model can distinguish between positive and negative samples regardless of the threshold, with a perfect classifier having an AUC of 1 and a random classifier having an AUC of 0.5.

4.1.4 Hyperparameters

The proposed method uses specific hyperparameters, including an embedding size of 16 for all features initialized with Xavier[35]. DeepFM[23] has two fully-connected hidden layers with 16 units and a dropout[36] of 0.2 and the ReLU activation function. The method uses an Adam[37] optimizer with an initial learning rate of 0.001 and weight decay 1e-6, and a batch size of 2048 for all datasets. The model is trained until convergence using an early stopper with two trials. The negative sampling rate is set to 1, following the common practice[38]. All baseline methods are implemented using the code provided by their respective authors. With respect to our bias-resolving strategy DART, we tune the hyperparameters of the initial iteration T_i in [0, 1000, 2000], the increase iteration T_m in [5000, 10000, 20000], the maximum threshold τ_m in [0.005, 0.01, 0.02], and the discard/relabel ratio r tuned in [0, 0.25, 0.5, 0.75, 1]. All offline experiments are run on a single machine with RTX 3090 GPU with 24GB memory.

4.2 Performance Comparison (RQ1)

Table 2 The performance of the proposed method and baselines on three datasets. The best results are highlighted in bold, while the results worse than the Base are marked with an asterisk (*).

Method	MovieLens-100K		Adressa		Book-Crossing		Jester		Δ‰
	FM	DeepFM	FM	DeepFM	FM	DeepFM	FM	DeepFM	
Base	0.8204	0.8079	0.8464	0.8183	0.8139	0.7781	0.8311	0.8343	-
Clean	0.8263	0.8148	0.8589	0.8277	0.8235	0.794	0.8397	0.8465	12.399
Bootstrap-Hard	0.8208	0.8096	0.8487	0.8176*	0.8164	0.7773*	0.8313	0.8413	1.891
Bootstrap-Soft	0.8204*	0.8078*	0.8468	0.8176*	0.8145	0.7773*	0.8301*	0.8403	0.649
LCD	0.8163*	0.7882*	0.8441*	0.8155*	0.8159	0.7699*	0.8265*	0.8175*	-8.659
LCD-Re	**0.8215**	**0.81**	0.8509	0.8166*	0.8168	0.781	**0.8334**	0.8414	3.218
ADT-CE-R	0.8205	0.8084	0.8476	0.8161*	0.8153	0.7769*	0.8303*	0.8367	0.195
ADT-CE-T	0.8205	0.8087	0.8483	0.8206	0.8162	0.7783	0.8307*	0.8375	1.576
DART-Disc	0.8205	0.8087	0.8484	**0.822**	0.8178	0.7873	0.8308*	0.8369	3.421
DART-Rel	0.8205	0.8089	0.8493	0.8207	0.8178	**0.7875**	0.8307*	0.8375	3.493
DART	0.8205	0.8089	**0.8513**	**0.822**	**0.8181**	**0.7875**	0.8315	**0.8417**	**4.783**

We validate the effectiveness of the proposed biases-resolving strategy DART, which involves discarding and relabeling identified noise, on the CTR prediction task. We also computed versions of DART that only use discard ($r = 0$) and relabel ($r = 1$), which we refer to as DART-Disc and DART-Rel, respectively. The experimental results are presented in Table 2, where Δ‰ represents the ratio of relative improvement averaged across all datasets.

- When using a training set that filters out false positive samples, the model's recommendation performance improved significantly compared to Base. This improvement reflects the negative impact of false positive samples on personalized recommendations, confirming the importance of resolving training biases. However, in reality, when using implicit feedback, there is no such clean dataset available.

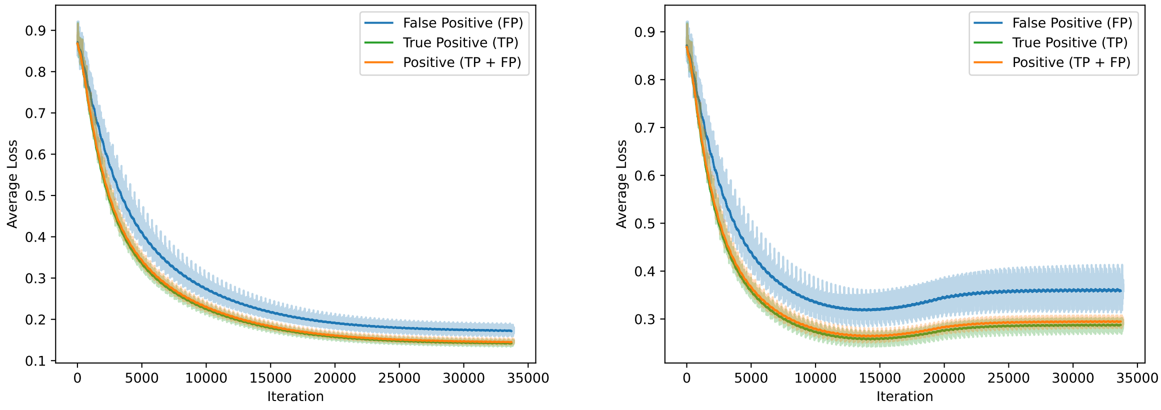
- Most of the baseline denoising methods, including Bootstrap[13], LCD-Re[7], and ADT[8], have shown improved recommendation performance. This demonstrates the effectiveness of robust learning based on training loss. However, LCD[7] significantly reduced performance, indicating that its original dynamic weighting method may not be suitable for our experimental setup.

- Our proposed DART achieves the best results with an average increase of 4.783‰, surpassing all baselines. This indicates that explicitly handling biases has the potential to achieve better performance than implicitly weighting the loss. The significance of this improvement lies in the fact that even a 1‰ increase in real-world recommendation AUC can result in substantial profits for companies[39].

- However, our proposed DART did not achieve the best performance across all datasets and models. LCD-Re[7] outperformed our method in both models on MovieLens-100K[31] and FM[20] on Jester[34] datasets. This also indicates that there is room for improvement in our approach. In the future, it may be necessary to explore the theoretical boundaries of its effectiveness.

- Both DART-Disc and DART-Rel outperform the baseline methods and effectively resolve biases. Furthermore, our approach of combining discarding and relabeling via parameter r provides a more flexible search space to leverage the advantages of both techniques, resulting in superior performance across all settings.

4.3 Memorization Process (RQ2)



a) Base

b) DART

Figure 2 The memorization process of base training (left), and DART (right).

Previous research has demonstrated that neural networks find it more challenging to memorize noisy samples than clean ones[8,14,18]. To investigate this phenomenon and validate the effectiveness of our proposed DART, we analyze the training loss of false-positive (blue) and true-positive interactions (green) during the training process. We use FM[20] trained on Book-Crossing[33] as an example.

As depicted in Figure 2a, the loss of false positives is higher than that of true positives, which confirms previous findings regarding the regular pattern of network memory. However, both losses eventually converge and the gap between them decreases. This demonstrates that the network's powerful memory can ultimately retain difficult samples, but at the cost of impairing its recommendation performance.

Figure 2b illustrates that DART leads to a significant increase in the loss of false-positive interactions as the number of training iterations increases. This finding suggests that the model discards the memorization of harmful false-positive samples, thereby validating the effectiveness of our method. Despite this, the loss of true positives also has a notable increase, which is due to the impact of misidentified samples. However, it can be observed that its improvement rate is lower than that of false-positive samples. Ultimately, our method was effective in closing the gap between the loss of true positives and false-positive samples. In the future, it is crucial to investigate how to minimize the memory loss of misidentified true-positive interactions.

4.4 Strategy Reasonability (RQ3)

Firstly, we analyzed the difference in loss between false-positive and true-positive interactions on the validation set of Base after 2000 training iterations. The results are presented in Figure 3. We observed that the overall loss of false-positive interactions was higher than that of true-positive interactions ($0.251 > 0.204$, Student's t-test with p-value < 0.001). This finding supports the hypothesis proposed in Section 4.3 that false-positive interactions are more difficult to memorize.

Next, we compared the mean loss of different ratings (1-10) on the validation set of

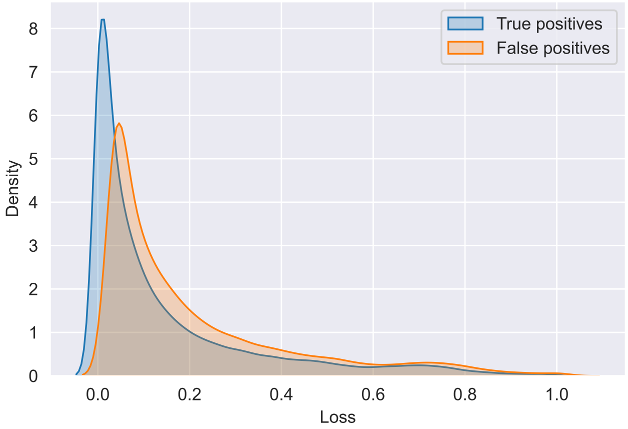


Figure 3 The distribution of losses on the validation set of Base after 2000 iterations of FM trained with Base on Book-Crossing.

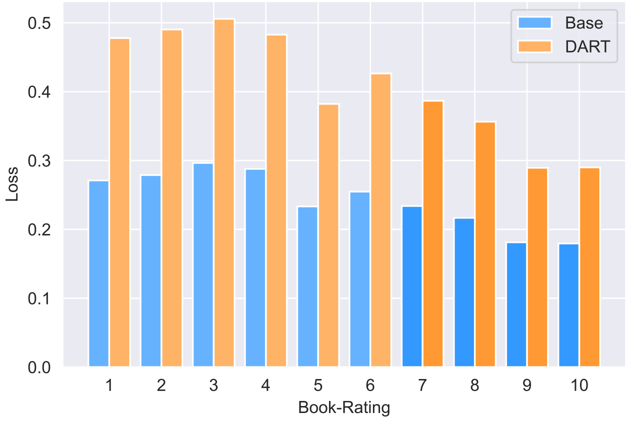


Figure 4 The average loss of different ratings on the validation set after training FM on Book-Crossing.

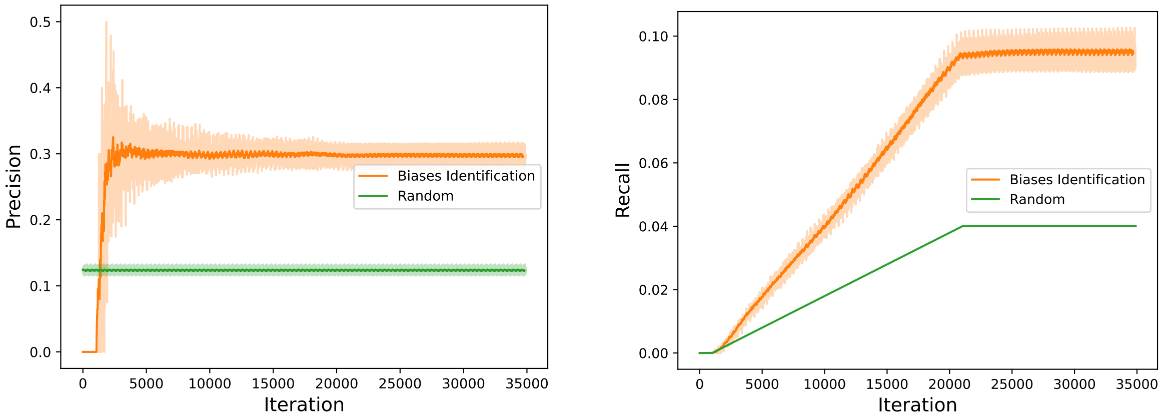
Book-Crossing[33] between Base and DART after training. The results are presented in Figure 4, which reveal the following findings:

- Under Base training, we observed a negative correlation between the average loss value and ratings. The Pearson correlation coefficient was -0.896 (p-value < 0.001), and the Spearman's rank correlation coefficient was -0.855 (p-value = 0.0016 < 0.005). This indicates that lower ratings are more difficult for the model to memorize.

- Under DART training, we observed an increase in the loss of all ratings (average increase of 0.165). However, false-positive samples (ratings 1-6) showed a greater increase compared to true-positives (ratings 7-10) (0.190 > 0.128). This demonstrates that misidentified samples by DART are harmful to both types of samples, but overall effective in achieving the goal of memorizing fewer noisy samples.

The above analysis of loss demonstrates that our strategy's theoretical foundation, "*noisy data are more challenging to fit into the networks during training*," is also reasonable in the context of recommendation systems. However, in the future, more methods will be needed to minimize the impact of misidentified samples.

4.5 Identification Effectiveness (RQ4)



a) Precision **b) Recall**

Figure 5 The precision (left) and recall (right) values for false positive interactions of FM trained with DART on Book-Crossing.

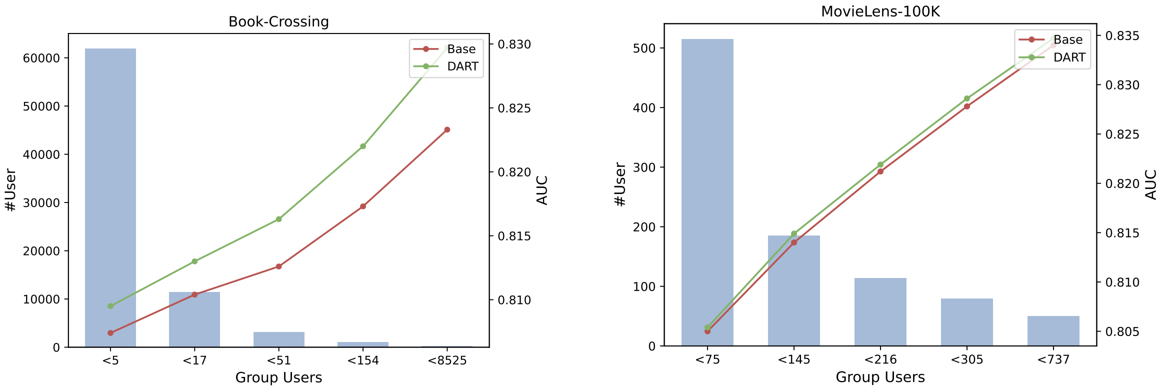
In order to evaluate the effectiveness of our biases identification method, we record how accurately it identifies and handles false-positive interactions[8]. We define *precision* as the proportion of false-positive samples in all handled samples, and *recall* as the proportion of false-positive samples that are identified in the training set. The change in precision and recall values over the process of training is visualized in Figure 5. As a reference, we use random handling, where the recall is equal to the handling threshold $\tau(T)$ during training, and the precision is equal to the proportion of noisy interactions in each training mini-batch at every iteration.

The results in Figure 5 demonstrate that our method can effectively identify approximately 10% of false positive interactions when the identification threshold stabilizes gradually. This significantly outperforms random selection (upper bound 4%) and highlights the effectiveness of using training loss to identify noisy samples. Our method also achieves a precision of approximately 30%, which is again higher than random (about 11%). While these results have led to improvements in recommendation performance, a considerable number of true-positive interactions are still misidentified, and many false-positive samples remain unrecalled. Figure 3 also illustrates that although true-positive and false-positive samples have different distributions, they are still closely intertwined. Therefore, improving the precision and recall of biases identification remains a key focus of this approach. Future research will need to explore new methods for identifying and mitigating the impact of noisy samples on recommendation systems.

4.6 Inactive User Recommendations (RQ5)

Training data in recommendation systems often exhibit a long-tail distribution, where a small portion of users contributes to the vast majority of interaction data, while most users only have a few interactions with items[40]. This can lead to a bias toward recommending items to highly active users, raising fairness concerns that have been explored in previous research. In this section, we explore whether the DART method exacerbates unfair recommendations for inactive users.

Following the former studies[41-42], we divided testing users into five groups based on



a) Book-Crossing **b) MovieLens-100K**

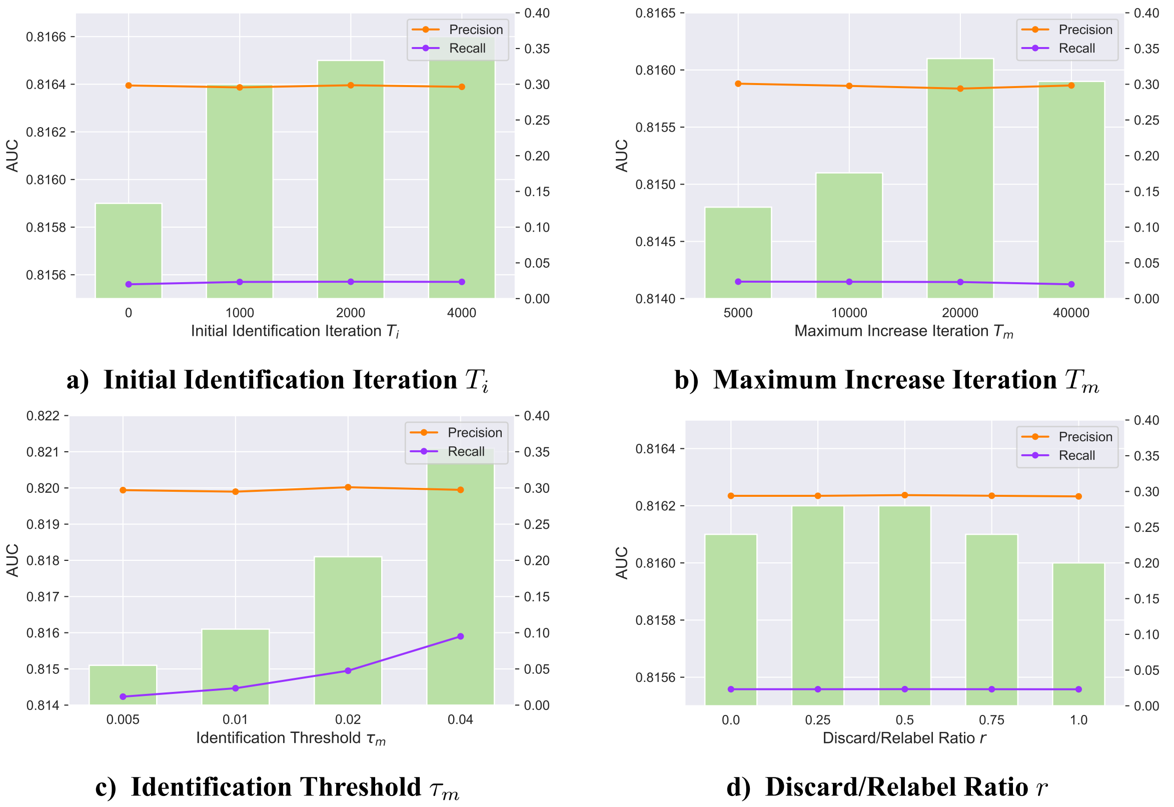
Figure 6 The recommendation performance (AUC) for users with different levels of activity of FM trained with DART, the x-axis represents the range of user interaction numbers.

their range of interaction numbers. To ensure an equal number of positive interactions in each group, we calculated the AUC for Base and DART training on Book-Crossing[33] and MovieLens-100K[31] datasets. As expected, the recommendation quality was significantly better for active users than for inactive ones, confirming the long-tail effect in recommendations. Moreover, the trend of recommendation performance improvement for different user groups varied between the two datasets, possibly due to differences in their distribution. Specifically, while the MovieLens dataset had already undergone a filtering process for inactive users (#interaction < 20), Book-Crossing did not have such preprocessing.

Fortunately, DART improved the recommendation performance for all user groups, albeit to a greater extent for highly active users. Future research directions may explore how to further improve fair recommendations through enhancements to DART.

5. Sensitivity Analysis

In this section, we conduct a systematic sensitivity analysis of the hyperparameters introduced by the DART method to observe its robustness under different settings. We use the results of training FM with DART on Book-Crossing as an example. The hyperparameters we tune include initial iteration T_i in [0, 1000, 2000, 4000], maximum increase iteration T_m in [5000, 10000, 20000, 40000], maximum identification threshold τ_m in [0.005, 0.01, 0.02, 0.04], and discard/relabel ratio r tuned in [0, 0.25, 0.5, 0.75, 1].



a) Initial Identification Iteration T_i

b) Maximum Increase Iteration T_m

c) Identification Threshold τ_m

d) Discard/Relabel Ratio r

Figure 7 The sensitivity analysis of FM trained on Book-Crossing with respect to different settings of hyper-parameters.

Figure 7 shows the sensitivity of the AUC of FM[20] trained on Book-Crossing[33] to the choice of T_i, T_m, τ_m and r. We also visualize its average precision and recall for false positive identification during the last epoch of training. By observing, we can make the following conclusions:

- Figure 7a demonstrates that the design of the initial identification iteration T_i was effective in improving bias resolution performance, as evidenced by a positive correlation within our hyperparameter range. Specifically, allowing the model to memorize noise samples for a period at the beginning of training proved beneficial for later noise identification.

- Figure 7b illustrates the importance of selecting an appropriate value for the maximum increase iteration T_m. Specifically, a moderate increase rate in identification rate is optimal, as both too rapid and too slow increases can negatively impact bias resolution performance.

- Figure 7c demonstrates a positive correlation between the identification threshold τ_m and recommendation performance within our parameter range. Specifically, increasing the threshold appropriately while maintaining a high level of precision in identification can aid in identifying noisy samples and improving recommendation performance. However, experiments on other datasets have shown that setting τ_m too high may lead to more misidentified true-positive interactions.

- As shown in Figure 7d, the setting of the discard/relabel ratio r has a generally weak impact on recommendation performance (within 0.2 AUC). However, selecting an intermediate value for r (e.g., 0.25 or 0.5) to balance the trade-off between discarding and relabeling can achieve the best results, which is consistent with our other experiments.

Overall, our proposed DART is generally robust across various parameter settings and consistently improves recommendation performance (Base AUC 0.8139). A larger parameter search space provides more options for resolving biases in different recommendation scenarios and achieving better performance. Furthermore, the relationship between precision and recall values and recommendation performance was not demonstrated in this experiment. Overall, they maintained a stable value (except for τ_m, which has a natural positive correlation with recall).

6. Discussion and Future Work

Firstly, our proposed DART framework has four hyperparameters for search: T_i, T_m, τ_i, and r. Although their settings have some robustness, as shown in Section 5, they offer greater potential to outperform baseline methods. However, the vast search space for hyperparameters can lead to significant tuning efforts during real-world training and deployment. Therefore, exploring techniques used in *hyperparameter optimization (HPO)* could help alleviate these issues in future work. This could potentially be combined with evolutionary computing[43] during the actual training. On the other hand, some of these parameters may be *adaptively set* by exploring the distribution patterns of data. For example, in future work, we can investigate whether the distribution of training loss is mapped to the optimal discard/relabel ratio r to reduce the workload of hyperparameter search.

Secondly, during the early stages of training, the model must memorize some biased interactions before it can distinguish them. This has inspired us to consider designing a scheme to first identify biases, then reinitialize the model and train it on an unbiased training set. This approach may have the potential to further improve recommendation performance. One possible direction is to use two models for co-training, with one model dedicated to identifying biases and teaching the second model, which is solely responsible for learning. This approach is referred to as "*1st-order differential teaching*". Additionally, we may introduce more dedicated biases identification and teaching models that can work either *serially* or *parallelly* (via voting) to achieve "*kth-order differential teaching*". We leave these ideas for future work.

Finally, the definition of false-positive interactions in different recommendation scenarios is an important aspect of evaluating the performance of bias resolution. In most cases, it can be *intuitively defined*, such as considering clicks with higher ratings as true-positive samples. However, sometimes the noise in the dataset can be complex. For example, in music recommendation scenarios[7], a song played on repeat may be unintentional despite having a high playtime. Defining false-positive interactions requires careful consideration of the distribution patterns in the dataset and ultimately needs to be evaluated based on user feedback or business interests. This is a promising research direction for industry applications.

7. Conclusion

In conclusion, our study highlights the importance of addressing training biases in recommender systems, which can significantly impact the quality of recommendations. Through our investigation of current robust learning methods and our proposed model-agnostic framework, we have shown that it is possible to effectively resolve training biases without relying on additional user data. We propose DART, a model-agnostic two-stage algorithm that first identifies biases based on training loss and then uses a combination of discarding and relabeling to handle them. Our extensive experiments demonstrate the effectiveness of DART in denoising false-positive interactions and improving recommendation performance. Additionally, several research questions and sensitivity analyses were conducted to demonstrate the reasonability and robustness of DART. For future work, we aim to explore methods for mitigating the negative impact of model memorization noise and reducing the search space for hyperparameters. Overall, our study provides valuable insights into addressing training biases in recommender systems and lays a foundation for further research in this area.

References

[1] SMITH B, LINDEN G. Two decades of recommender systems at Amazon. com[J]. Ieee internet computing, 2017, 21(3): 12-18.

[2] QU L, YE Y, TANG N, et al. Single-shot Embedding Dimension Search in Recommender System[C]//Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2022: 513-522.

[3] COVINGTON P, ADAMS J, SARGIN E. Deep neural networks for youtube recommendations[C]//Proceedings of the 10th ACM conference on recommender systems. 2016: 191-198.

[4] SCHAFER J B, FRANKOWSKI D, HERLOCKER J, et al. Collaborative filtering recommender systems[J]. The adaptive web: methods and strategies of web personalization, 2007: 291-324.

[5] CHEN J, DONG H, WANG X, et al. Bias and debias in recommender system: A survey and future directions[J]. ACM Transactions on Information Systems, 2023, 41(3): 1-39.

[6] HOFMANN K, BEHR F, RADLINSKI F. On caption bias in interleaving experiments[C]//Proceedings of the 21st ACM international conference on Information and knowledge management. 2012: 115-124.

[7] DAI Q, LV Y, ZHU J, et al. LCD: Adaptive Label Correction for Denoising Music Recommendation[C]//Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 2022: 3903-3907.

[8] WANG W, FENG F, HE X, et al. Denoising implicit feedback for recommendation [C]//Proceedings of the 14th ACM international conference on web search and data mining. 2021: 373-381.

[9] YU J, ZHU H, CHANG C Y, et al. Influence function for unbiased recommendation [C]//Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2020: 1929-1932.

[10] KIM Y, HASSAN A, WHITE R W, et al. Modeling dwell time to predict click-level satisfaction[C]//Proceedings of the 7th ACM international conference on Web search and data mining. 2014: 193-202.

[11] YI X, HONG L, ZHONG E, et al. Beyond clicks: dwell time for personalization[C]// Proceedings of the 8th ACM Conference on Recommender systems. 2014: 113-120.

[12] ZHAO Q, CHANG S, HARPER F M, et al. Gaze prediction for recommender systems [C]//Proceedings of the 10th ACM Conference on Recommender Systems. 2016: 131-138.

[13] REED S, LEE H, ANGUELOV D, et al. Training deep neural networks on noisy labels with bootstrapping[J]. arXiv preprint arXiv:1412.6596, 2014.

[14] JIANG L, ZHOU Z, LEUNG T, et al. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels[C]//International conference on machine learning. 2018: 2304-2313.

[15] HAN B, YAO Q, YU X, et al. Co-teaching: Robust training of deep neural networks with extremely noisy labels[J]. Advances in neural information processing systems, 2018, 31.

[16] KOH P W, LIANG P. Understanding black-box predictions via influence functions [C]//International conference on machine learning. 2017: 1885-1894.

[17] KONG S, SHEN Y, HUANG L. Resolving training biases via influence-based data relabeling[C]//International Conference on Learning Representations. 2021.

[18] ARPIT D, JASTRZĘBSKI S, BALLAS N, et al. A closer look at memorization in deep networks[C]//International conference on machine learning. 2017: 233-242.

[19] BENGIO Y, LOURADOUR J, COLLOBERT R, et al. Curriculum learning[C]// Proceedings of the 26th annual international conference on machine learning. 2009: 41-48.

[20] RENDLE S. Factorization machines[C]//2010 IEEE International conference on data mining. 2010: 995-1000.

[21] HE X, LIAO L, ZHANG H, et al. Neural collaborative filtering[C]//Proceedings of the 26th international conference on world wide web. 2017: 173-182.

[22] CHENG H T, KOC L, HARMSEN J, et al. Wide & deep learning for recommender systems[C]//Proceedings of the 1st workshop on deep learning for recommender systems. 2016: 7-10.

[23] GUO H, TANG R, YE Y, et al. DeepFM: a factorization-machine based neural network for CTR prediction[J]. arXiv preprint arXiv:1703.04247, 2017.

[24] YANG B, LEE S, PARK S, et al. Exploiting various implicit feedback for collaborative filtering[C]//Proceedings of the 21st International Conference on World Wide Web. 2012: 639-640.

[25] WEN H, YANG L, ESTRIN D. Leveraging post-click feedback for content recommendations[C]//Proceedings of the 13th ACM Conference on Recommender Systems. 2019: 278-286.

[26] PATRINI G, ROZZA A, KRISHNA MENON A, et al. Making deep neural networks robust to label noise: A loss correction approach[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 1944-1952.

[27] GOLDBERGER J, BEN-REUVEN E. Training deep neural-networks using a noise adaptation layer[C]//International conference on learning representations. 2017.

[28] KOH P W W, ANG K S, TEO H, et al. On the accuracy of influence functions for measuring group effects[J]. Advances in neural information processing systems, 2019, 32.

[29] KUMAR R, NAIK S M, NAIK V D, et al. Predicting clicks: CTR estimation of advertisements using logistic regression classifier[C]//2015 IEEE international advance computing conference (IACC). 2015: 1134-1138.

[30] HUA X S, MEI T, HANJALIC A. Online Multimedia Advertising: Techniques and Technologies: Techniques and Technologies[M]. Igi Global, 2010.

[31] HARPER F M, KONSTAN J A. The movielens datasets: History and context[J]. Acm transactions on interactive intelligent systems (tiis), 2015, 5(4): 1-19.

[32] GULLA J A, ZHANG L, LIU P, et al. The adressa dataset for news recommendation [C]//Proceedings of the international conference on web intelligence. 2017: 1042-1048.

[33] ZIEGLER C N, MCNEE S M, KONSTAN J A, et al. Improving recommendation lists through topic diversification[C]//Proceedings of the 14th international conference on World Wide Web. 2005: 22-32.

[34] GOLDBERG K, ROEDER T, GUPTA D, et al. Eigentaste: A constant time collaborative filtering algorithm[J]. information retrieval, 2001, 4: 133-151.

[35] GLOROT X, BENGIO Y. Understanding the difficulty of training deep feedforward neural networks[C]//Proceedings of the thirteenth international conference on artificial intelligence and statistics. 2010: 249-256.

[36] SRIVASTAVA N, HINTON G, KRIZHEVSKY A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. The journal of machine learning research, 2014, 15(1): 1929-1958.

[37] KINGMA D P, BA J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.

[38] QU L, TANG N, ZHENG R, et al. Semi-decentralized Federated Ego Graph Learning for Recommendation[J]. arXiv preprint arXiv:2302.10900, 2023.

[39] ZHU J, LIU J, YANG S, et al. Fuxictr: An open benchmark for click-through rate prediction[J]. arXiv preprint arXiv:2009.05794, 2020.

[40] PARK Y J, TUZHILIN A. The long tail of recommender systems and how to leverage it[C]//Proceedings of the 2008 ACM conference on Recommender systems. 2008: 11-18.

[41] YAO J, WANG F, JIA K, et al. Device-cloud collaborative learning for recommendation[C]//Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 2021: 3865-3874.

[42] WANG X, HE X, WANG M, et al. Neural graph collaborative filtering[C]// Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. 2019: 165-174.

[43] EIBEN A E, SMITH J E. Introduction to evolutionary computing[M]. Springer, 2015.

Acknowledgement

Upon reflection on my four-year time at the Southern University of Science and Technology (SUSTech), I firstly express gratitude to myself for persevering in my personal growth, despite the accompanying confusion and self-doubt. However, the attainment of such success and happiness would have been impossible without the invaluable guidance and unwavering support of my mentors, collaborators, friends, and family.

I am immensely grateful to my advisor, Professor Yuhui Shi, for his tremendous help throughout the past two years. His erudition, foresight, and dedication to his students are truly admirable and will continue to have a lasting impact on me. I consider myself extremely fortunate to have had the opportunity to receive guidance from such a knowledgeable and supportive mentor, who accompanied me at every significant juncture of my undergraduate studies. I look forward to continuing to make him proud in my future research career.

I owe a great deal to my senior labmate, Liang Qu. We had countless group meetings and private discussions and he provided me with invaluable assistance. Without his support, I would not have achieved the research results or received the Ph.D. offer that I did. His guidance laid a solid foundation for my knowledge of recommender systems and showed me the challenging yet fascinating aspects of research details. Liang's creativity and rigor in research, as well as his exemplary research skills, will be endless wealth for me in the future.

I would also like to thank Zhuozhao Li, the reviewer of my paper, as well as the other committee members Yinqian Zhang, Jianqiao Yu, and Shan Chen, for their invaluable feedback and continued support. I am grateful to all the professors I encountered at SUSTech, including Chengzhi Hu, Jingde Cheng, Bingsheng He, Steven Pellerine, Qi Hao, Xuan Song, and many more. Their knowledge and character demonstrated in their courses have had a lifelong impact on me.

I am fortunate to have had the opportunity to work with fabulous collaborators on this research, Shuyu Pan and Tengfei Liu. I am also grateful for all the friends I made during my

time in college, especially Chi Xu, Shuwen Dong, Haocheng Qin, Runyi Liu, Chongyang Shi, Yunxiang Yan, and my roommates Muyao Chen, Jiawei Qiu, and Yuanzhe Wei. Their companionship made my college experience meaningful.

I am especially grateful to my girlfriend, Gelei Xu. Meeting her was the greatest stroke of luck during my time in college. Her companionship and embrace have been invaluable to me during times of uncertainty and despair about the future. I look forward to spending the rest of my life with her, from SUSTech to the University of Notre Dame, as better individuals, researchers, and lovers.

Finally, I would like to express my gratitude to my parents and all other family members. Their unconditional love and support have been the driving force behind my life. During countless late-night runs, I shared with my parents every small achievement and joy that I had gained, which has become my happiest memory of the four years in college.