Combating Selection Biases in Recommender Systems with A Few Unbiased Ratings

Xiaojie Wang
Amazon.con, Inc.
xiojie@amazon.com

Rui Zhang*
The University of Melbourne
rui.zhang@unimelb.edu.au

Yu Sun
Twitter Inc.
ysun@twitter.com

Jianzhong Qi
The University of Melbourne
jianzhong.qi@unimelb.edu.au

ABSTRACT

Recommendation datasets are prone to selection biases due to self-selection behavior of users and item selection process of systems. This makes explicitly combating selection biases an essential problem in training recommender systems. Most previous studies assume no unbiased data available for training. We relax this assumption and assume that a small subset of training data is unbiased. Then, we propose a novel objective that utilizes the unbiased data to adaptively assign propensity weights to biased training ratings. This objective, combined with unbiased performance estimators, alleviates the effects of selection biases on the training of recommender systems. To optimize the objective, we propose an efficient algorithm that minimizes the variance of propensity estimates for better generalized recommender systems. Extensive experiments on two real-world datasets confirm the advantages of our approach in significantly reducing both the error of rating prediction and the variance of propensity estimation.

ACM Reference Format:
Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2021. Combating Selection Biases in Recommender Systems with A Few Unbiased Ratings . In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441799

1 INTRODUCTION

Normally, recommendation datasets are collected during users' normal use of recommender systems and are subject to *selection biases* [19, 24]. For example, Fig. 1 shows two types of selection biases in movie recommendation: (1) Systems' item selection process: the right half of Fig. 1 shows that the system aims to recommend movies that Bob may like by filtering out movies with low predicted ratings; (2) Users' self-selection behavior: the left half of Fig. 1 shows that Bob tends to rate recommended movies that he likes and rarely rates recommended movies that he dislikes. Due to such selection biases, high ratings are more likely to be observed than low ratings [13]. Back to the example in Fig. 1, there are a total of two high ratings (the top-right corner) and both ratings are observed. The percentage of high ratings being observed is thus 100%, which is greater than that (17%) of low ratings (one out of six is observed). In fact, studies have shown that high ratings account for the majority of observed ratings and low ratings are under-represented [18, 19]. Usually, observed ratings are used to train a rating prediction model and items are ranked in descending order of predicted ratings when recommended to users [33, 39]. Since observed ratings are biased from the population of all ratings – whether observed or not, it is difficult to accurately estimate the real performance of a rating model. This creates a widely-recognized challenge for training recommender systems on biased datasets [6, 18].

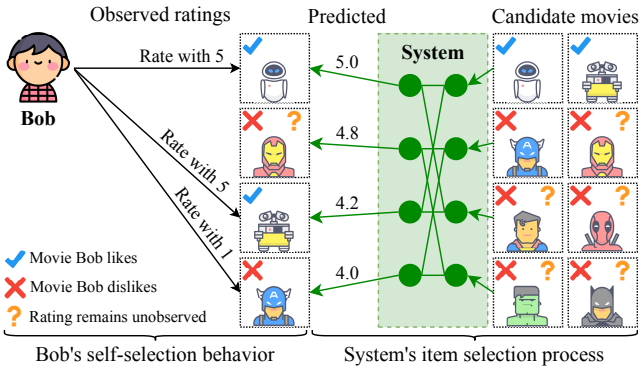


Fig. 1: An example of selection biases in movie recommendation.

To address this challenge, *unbiased estimators* of the real performance of a rating model have been recently introduced [24, 39]. Theoretically, unbiased estimators can yield accurate estimation of a rating model's real performance even on biased datasets. This is achieved mostly by inversely weighting each observed rating with the propensity (i.e., probability) of observing that rating. An intuitive justification is that ratings that are under-represented within observed ratings should be up-weighted. According to the definition of propensity, the ratings that are under-represented are those that have a small propensity of being observed. Those ratings, once weighted by inverse of the small propensity, will be assigned a large weight and thus be correctly up-weighted.

Most previous studies that explicitly handle selection biases assume that no unbiased dataset is available for training [6, 18, 24].

*Corresponding author

WSDM '21, March 8–12, 2021, Virtual Event, Israel

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441799

However, unbiased ratings can be gathered, e.g., by asking users to rate a number of randomly-selected items (thus avoiding the selection biases). Exiting studies have used such unbiased datasets to accurately estimate the real performance of a rating model [24, 39]. Whenever it is possible to gather unbiased ratings, a small unbiased dataset can be curated for training. Hence, we depart from the assumption that all training ratings are biased and assume that a small subset of training ratings is unbiased.

In this paper, we propose an objective that utilizes a small unbiased dataset to explicitly strengthen the generalization ability of a rating model. Specifically, we iteratively validate the performance of the rating model on the unbiased dataset while training the rating model on a biased dataset. The optimal rating model is thus the one that is trained on biased data but performs well on unbiased data. This closely matches the definition of generalization ability because unbiased data can provide accurate estimation of the generalization ability. The key idea of our optimization objective is to introduce a propensity model that uses the unbiased dataset to dynamically determine the propensities of biased ratings. This guides the propensity model to find a weighted combination of biased ratings that can best approximate unbiased performance estimation on unbiased data. This way, training the rating model on biased data will be equivalent to training the rating model on unbiased data. To update the propensity model's parameters, we directly optimize the rating model's performance on the unbiased dataset. The rationale is that the best propensity model should optimize the performance of a rating model on unbiased data that is consistent with the testing procedure. A recent study [39] that also uses unbiased data simply assumes that the propensity of observing a rating only depends on the rating's value. In contrast, our propensity model is free of this assumption and can use all user and item information (e.g., user age) to better model the propensity.

To optimize the objective, we propose an algorithm where the propensity model's parameters are updated simultaneously with the rating model's parameters. This algorithm makes use of automatic differentiation and obviates complex derivation of gradients. Hence, it can be conveniently applied when we use different unbiased estimators and a diverse set of model architectures [27, 40]. This makes the proposed algorithm appealing to practitioners. A well-known challenge in applying unbiased estimators is to obtain stable gradients to update the rating model [31, 32]. This is because inverse propensity weighting may cause large gradient updates to the rating model's parameters, especially when the propensity estimates are small. This motivates us to use the sample variance of propensity estimates to regularize the training of the propensity model, which results in low-variance propensity estimates. Empirical results show that our approach substantially outperforms the state-of-the-art ones that do not employ unbiased data, as well as a recent one that also uses unbiased data [39].

Our main contributions are summarized as follows.

1 We propose a novel objective that utilizes a small set of unbiased data to alleviate selection biases and improve the training of recommender systems on biased data.
2 To optimize the objective, we propose an efficient algorithm that can also effectively reduce the variance of propensity estimation during training model parameters.
3 We conduct extensive experiments on two real-world datasets. The results show that our approach improves the performance of rating prediction by up to 7.9% and reduces the variance of propensity estimation by orders of magnitude.

2 RELATED WORK

2.1 Recommendation Debiasing

Existing studies on recommendation debiasing primarily focus on two tasks: rating prediction and item ranking [29, 30, 38]. The rating prediction task aims to predict the rating that a user may give to an unseen item, while the item ranking task aims to provide a user with an item list that maximizes a ranking metric [28, 37]. Both tasks have been widely studied by academics and industry over the last few years [14, 35]. In this paper, we tackle the rating prediction task. We use existing model architectures [5, 12], which output a rating given user and item features as inputs, to implement underlying rating model. Our approach benefits from advances in designing such model architectures, e.g. the ones based on deep neural networks [5, 26]. Recommendation is closely related to but different from search in that recommendation does not require explicit user queries while search does [7, 15, 16].

A widely-recognized problem in the rating prediction task is that datasets for training are usually biased [19, 22, 23]. To handle biased datasets, early studies optimize a joint likelihood of a propensity model and a rating model, which requires highly complex inferences [6, 18]. To avoid such inference complexity, recent studies adopt two-phase learning, which first learns a propensity model and then applies propensity weighting techniques to train a rating model [24, 39]. The main difference between these studies and our work is that we directly relate the objective for the propensity model to the final goal of the debiasing problem, i.e., optimizing the performance of a rating model on unbiased datasets.

As for the item ranking task, recent studies show that directly using biased datasets in learning to rank approaches usually yields suboptimal results [11]. The suboptimality is observed under various ranking metrics such as Expected Reciprocal Rank (ERR) and Normalized Discounted Cumulative Gain (NDCG) [34, 36]. These studies on item ranking are largely orthogonal to our work on rating prediction.

2.2 Bi-level Optimization

Bi-level optimization, which performs upper-level learning subject to the optimality of lower-level learning, has received increasing attention recently [10, 17]. Among existing approaches of bi-level optimization, the most related one is from Ren et al. [21], which dynamically determines weights of training examples. This approach may generate negative weights, and it resorts to heuristics for adjusting these weights to avoid unstable training behaviors. In contrast, our approach generates non-negative weights based on propensity estimates, and explicitly controls the variance of the weights for a better performance of rating prediction.

Generally, it is challenging to train parameters in bi-level optimization because the lower-level learning cannot be performed in closed form [10, 25]. To tackle this challenge, an early approach applies implicit differentiation, and assumes that the optimality to the lower-level learning uniquely exists [20]. This assumption often

does not hold in practice, and thus this approach may incur large errors in gradient computation. To address this issue, recent studies differentiate a certain parameter update function to better compute gradients [3, 4]. We further develop this approach by regularizing the upper-level learning with the variance of propensity estimates, which helps stabilize training at the lower level and thus results in better generalized rating models.

3 PRELIMINARIES

Let $\{u_m|m = 1, ..., M\}$ be a set of users, $\{i_n|n = 1, ..., N\}$ be a set of items, and $\mathcal{A} = \{(u_m, i_n)|m = 1, ..., M; n = 1, ..., N\}$ be the set of all user-item pairs. Let $\boldsymbol{x}_{u,i} = [x_{u,i,k}|k = 1, ..., K]$ be a feature vector of user u and item i, where $x_{u,i,k} \in \mathbb{R}$ is the k-th feature (e.g., user gender). Let $\mathbf{R} = [r_{u,i}|u, i \in \mathcal{A}]$ be a true rating matrix, where $r_{u,i} \in \mathbb{R}$ is the true rating given by user u to item i. Users may freely select a fraction of items to rate and the ratings to these selected items are observed. The observed ratings, denoted by $\mathbf{R}^{\mathcal{B}}$ ($\mathcal{B} \subseteq \mathcal{A}$), are biased, meaning that the probability of observing a rating depends on that rating's value [19]. Such a probability is often called the propensity $p_{u,i} = p(o_{u,i} = 1)$, where $o_{u,i}$ is a Bernoulli variable indicating whether the true rating $r_{u,i}$ is observed $o_{u,i} = 1$ or missing $o_{u,i} = 0$. Given the biased ratings $\mathbf{R}^{\mathcal{B}}$, a debiasing problem of recommendation aims to learn a rating model $y_\phi(\boldsymbol{x}_{u,i}) \approx r_{u,i}$ (with parameters ϕ) that can accurately predict all true ratings. Formally, the goal is to minimize the *real performance* $\mathcal{E}(\mathcal{A})$ of a rating model, which can be computed if all ratings are observed

$$\mathcal{E}(\mathcal{A}) = \frac{1}{|\mathcal{A}|} \sum_{u,i \in \mathcal{A}} (e_{u,i})^2, \; e_{u,i} = y_\phi(\boldsymbol{x}_{u,i}) - r_{u,i}, \tag{1}$$

where $e_{u,i}$ is a *prediction error*. The real performance can be unbiasedly estimated by a naive estimator that averages prediction errors over a set of unbiased ratings $\mathbf{R}^{\mathcal{U}}$ ($\mathcal{U} \subseteq \mathcal{A}$)

$$\mathcal{E}(\mathcal{A}) \approx \mathcal{E}(\mathcal{U}) = \frac{1}{|\mathcal{U}|} \sum_{u,i \in \mathcal{U}} (e_{u,i})^2. \tag{2}$$

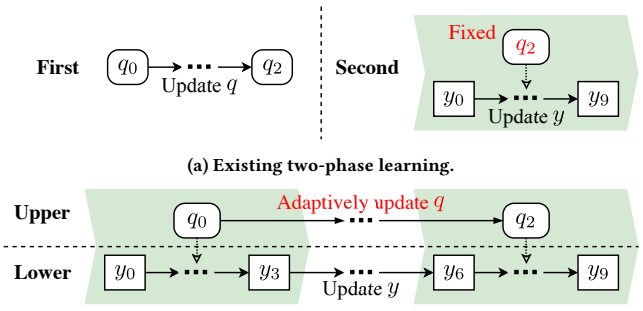
To gather unbiased ratings, we may ask users to rate randomly-selected items. This way, the propensities of observing different ratings are the same and the observed ratings are thus unbiased.

To solve the debiasing problem, recent studies separate learning into two consecutive phases, which we illustrate in Fig. 2a [24, 39]. As shown by the left half of Fig. 2a, the first phase aims to learn a propensity model $q_\theta(\boldsymbol{x}_{u,i}) \approx p_{u,i}$ (with parameters θ) that can accurately estimate the propensity. There are two types of propensity models. The first type is a naive Bayes model [24]. It assumes that the propensity $p_{u,i} = p(o_{u,i} = 1|r_{u,i} = r)$ only depends on the true rating, and estimates the propensity by the Bayes' theorem

$$q_\theta(\boldsymbol{x}_{u,i}) = \frac{p(r_{u,i} = r|o_{u,i} = 1)p(o_{u,i} = 1)}{p(r_{u,i} = r)}. \tag{3}$$

To simplify notation, we define $p_r = p(r_{u,i} = r)$, $p_o = p(o_{u,i} = 1)$, and $p_r^1 = p(r_{u,i} = r|o_{u,i} = 1)$. Given biased ratings $\mathbf{R}^{\mathcal{B}}$ and unbiased ratings $\mathbf{R}^{\mathcal{U}}$, the parameters $\theta = \{p_r, p_o, p_r^1|r = 1, ..., R\}$ are fitted by maximizing a likelihood function as follows

$$\mathcal{F}_{\mathrm{NB}}(\theta) = \prod_{u,i \in \mathcal{U}} p_{r_{u,i}} + \prod_{u,i \in \mathcal{B}} p_o \prod_{u,i \in \mathcal{M}} (1 - p_o) + \prod_{u,i \in \mathcal{B}} p^1_{r_{u,i}}, \tag{4}$$



(a) Existing two-phase learning.

(b) Proposed bi-level optimization.

Fig. 2: During the training process of updating a rating model y_s, two-phase learning (a) uses a fixed propensity model q_2, but bi-level optimization (b) adaptively updates the propensity model q_s.

where $\mathcal{M} = \mathcal{A} \setminus \mathcal{B}$ is the set of missing ratings. The second type is a logistic regression model [24]. It estimates the propensity by applying a logistic function on the user and item features $\boldsymbol{x}_{u,i}$

$$q_\theta(\boldsymbol{x}_{u,i}) = \sigma(\boldsymbol{w}^\top \boldsymbol{x}_{u,i} + b) = \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_{u,i} + b)}{\exp(\boldsymbol{w}^\top \boldsymbol{x}_{u,i} + b) + 1}, \tag{5}$$

where $\theta = \{\boldsymbol{w}, b\}$ are parameters. To fit the parameters, we maximize the likelihood of observing the biased ratings $\mathbf{R}^{\mathcal{B}}$ (i.e., the ratings self-selected by users) as follows

$$\mathcal{F}_{\mathrm{LR}}(\theta) = \prod_{u,i \in \mathcal{B}} q_\theta(\boldsymbol{x}_{u,i}) \prod_{u,i \in \mathcal{M}} (1 - q_\theta(\boldsymbol{x}_{u,i})). \tag{6}$$

Given the propensity model, the second phase aims to train a rating model, as shown in the right half of Fig. 2a. This is achieved by minimizing an unbiased estimator of the real performance, e.g., inverse-propensity-scoring (IPS) estimator [24] and doubly-robust (DR) estimator [39]. The DR estimator needs an error model $g_\xi(\boldsymbol{x}_{u,i}) \approx e_{u,i}$ (with parameters ξ) that can accurately impute the prediction errors.

Note that the naive Bayes model requires an unbiased dataset and has two issues. First, it assumes that the propensity depends on the rating only, which may not hold in practice. Second, if a rating value r is absent on the unbiased dataset, the naive Bayes model $q_\theta(\boldsymbol{x}_{u,i})$ in Eqn. (3) is not well defined because in such case the denominator will be zero (i.e., $p_r = 0$).

4 PROPOSED APPROACH

We study the debiasing problem where a set of unbiased ratings $\mathbf{R}^{\mathcal{U}}$, in addition to a set of biased ratings $\mathbf{R}^{\mathcal{B}}$, is available for training. Unbiased ratings are more costly than biased ratings to gather because asking users to rate randomly-selected items may negatively affect user experience. We thus assume that the number of unbiased ratings is much smaller than that of biased ratings $|\mathcal{U}| \ll |\mathcal{B}|$. Under these problem settings, we propose an objective to effectively utilize the unbiased dataset in Sec. 4.1. Then, we propose an efficient algorithm to optimize the objective in Sec. 4.2.

4.1 Training Objective

Since we assume that the number of unbiased ratings $\mathbf{R}^{\mathcal{U}}$ is small, using only the unbiased dataset to train a rating model may cause severe overfitting. Hence, we use the large biased dataset $\mathbf{R}^{\mathcal{B}}$ to

train the rating model and rely on unbiased estimators to obtain accurate performance estimation on the biased dataset. For the estimators to be unbiased, we need a good propensity model that can produce accurate propensity estimation. In such case, the unbiased estimators will be equivalent to a naive one on the unbiased dataset given by Eqn. (2). This observation motivates us to utilize the unbiased dataset to train a propensity model. In particular, we aim to learn a propensity model such that a rating model trained on the biased dataset performs well on the unbiased dataset. Formally, this goal can be formulated as a bi-level optimization problem

$$\theta^* = \arg\min_{\theta} \mathcal{L}_{\mathrm{U}}(\phi^*(\theta); \mathcal{U}) \tag{7}$$

$$\text{s.t. } \phi^*(\theta) = \arg\min_{\phi} \mathcal{L}_{\mathrm{L}}(\phi, \theta; \mathcal{B}). \tag{8}$$

Following the terminology in bi-level optimization, Eqn. (7) and Eqn. (8) are called *upper level* and *lower level*, respectively. The lower level aims to minimize a lower loss $\mathcal{L}_{\mathrm{L}}$ w.r.t. the rating model's parameters ϕ on the biased dataset $\mathbf{R}^{\mathcal{B}}$ for a given propensity model, whereas the upper level aims to minimize an upper loss $\mathcal{L}_{\mathrm{U}}$ w.r.t. the propensity model's parameters θ on the unbiased dataset $\mathbf{R}^{\mathcal{U}}$.

We define the lower loss based on an unbiased estimator to obtain accurate performance estimation on the biased dataset. To give a unified treatment of the IPS estimator and the DR estimator, we define the lower loss by

$$\mathcal{L}_{\mathrm{L}}(\phi, \theta; \mathcal{B}) = \sum_{u,i \in \mathcal{B}} \ell_{u,i}(\phi, \theta) + \sum_{u,i \in \mathcal{M}} \hat{\ell}_{u,i}(\phi), \tag{9}$$

where $\ell_{u,i}(\phi, \theta)$ and $\hat{\ell}_{u,i}(\phi)$ are losses for the observed ratings $\mathbf{R}^{\mathcal{B}}$ and the missing ratings $\mathbf{R}^{\mathcal{M}}$, respectively. Specifically, we can instantiate the lower loss by IPS as follows. The IPS loss minimizes a *square prediction error* $s_{u,i}(\phi) = (y_\phi(\boldsymbol{x}_{u,i}) - r_{u,i})^2$, inversely weighted by a propensity estimate $q_\theta(\boldsymbol{x}_{u,i})$, for each observed rating and ignores all missing ratings,

$$\ell_{u,i}(\phi, \theta) = \frac{s_{u,i}(\phi)}{q_\theta(\boldsymbol{x}_{u,i})}, \ \hat{\ell}_{u,i}(\phi) = 0. \tag{10}$$

Alternatively, we can instantiate the lower loss by DR as follows. The DR loss treats an imputed error $g_\xi(\boldsymbol{x}_{u,i})$ as the prediction error to train the rating model. It obtains a "true rating" $\hat{r}_{u,i} = y_{\phi_s}(\boldsymbol{x}_{u,i}) + g_{\xi_s}(\boldsymbol{x}_{u,i})$ by adding a predicted rating and an imputed error at current training step s. For each missing rating, it minimizes a square imputed error $\hat{s}_{u,i}(\phi) = (y_\phi(\boldsymbol{x}_{u,i}) - \hat{r}_{u,i})^2$ between the rating model's next prediction and the "true rating"

$$\hat{\ell}_{u,i}(\phi) = \hat{s}_{u,i}(\phi). \tag{11}$$

Once the true rating is observed, it further introduces a correction (i.e., the difference between the square prediction error $s_{u,i}(\phi)$ and the square imputed error $\hat{s}_{u,i}(\phi)$), and inversely weights this correction by the propensity estimate $q_\theta(\boldsymbol{x}_{u,i})$

$$\ell_{u,i}(\phi, \theta) = \hat{s}_{u,i}(\phi) + \frac{s_{u,i}(\phi) - \hat{s}_{u,i}(\phi)}{q_\theta(\boldsymbol{x}_{u,i})}. \tag{12}$$

By the above optimization, for any propensity model q_θ, we can obtain a corresponding rating model $y_{\phi^*(\theta)}$ that minimizes the lower loss. We aim to find the propensity model such that the corresponding rating model performs well on the unbiased dataset. To this end, we define the upper loss as an average of square prediction errors on the unbiased dataset

$$\mathcal{L}_{\mathrm{U}}(\phi^*(\theta); \mathcal{U}) = \sum_{u,i \in \mathcal{U}} s_{u,i}(\phi^*(\theta)) \tag{13}$$

$$= \sum_{u,i \in \mathcal{U}} (y_{\phi^*(\theta)}(\boldsymbol{x}_{u,i}) - r_{u,i})^2. \tag{14}$$

Note that the propensity model often has a small number of parameters, which can be properly fitted on the small unbiased dataset. We illustrate the proposed bi-level optimization in Fig. 2b: instead of using a fixed propensity model, it learns to assign adaptive propensity weights while debiasing the training of a rating model. Hence, we name the proposed approach *learning to debias* (LTD).

4.2 Training Algorithm

Next, we detail how to train the parameters within LTD $\{\phi, \theta, \xi\}$. We encounter two challenges in the training: (1) Since LTD builds on bi-level optimization, it is often computationally expensive to minimize the upper loss to find a good propensity model [4]. This is because the upper loss depends on the best parameter values of the rating model, which has no closed form. (2) LTD applies inverse propensity weighting, and thus it is difficult to obtain stable gradients to update the rating model [32]. The reason for this is that such inverse may cause large gradient updates to the rating model, especially when the propensity estimates are small. Note that the two challenges are not specific to our approach, but are known in approaches based on bi-level optimization [4] and inverse propensity weighting [32]. To tackle these challenges, we propose a variance-regularized training algorithm in this section.

To simplify notation, we define two operators that compute partial gradients w.r.t. parameters $\rho \in \{\phi, \theta, \xi\}$

$$\nabla_{\rho_s}(\cdot) = \frac{\partial}{\partial \rho}(\cdot)\Big|_{\rho=\rho_s}, \quad \nabla^{\mathsf{T}}_{\rho_s}(\cdot) = (\nabla_{\rho_s}(\cdot))^{\mathsf{T}}, \tag{15}$$

where ρ_s are the values of parameters ρ at training step s.

The lower loss is differentiable w.r.t. the rating model's parameters ϕ. Hence, we can apply vanilla stochastic gradient descent (SGD) or other SGD variants to update the rating model. Here, we apply vanilla SGD for illustrative purpose.

It is difficult to minimize the upper loss because the best parameter values $\phi^*(\theta)$ within the upper loss have no closed form. A common practice is to replace the the best parameter values with a sequence of approximate ones that can converge to the best ones [4, 21]. To obtain such approximate parameter values, at each training step s, we define an update function that simulates a single parameter update of the rating model by vanilla SGD

$$\phi_{s+1}(\theta_s) = \phi_s - \eta \nabla_{\phi_s} \mathcal{L}_{\mathrm{L}}(\phi, \theta_s), \tag{16}$$

$$= \phi_s - \eta \sum_{u,i \in \mathcal{B}_s} \nabla_{\phi_s} \ell_{u,i}(\phi, \theta_s) - \eta \sum_{u,i \in \mathcal{M}_s} \nabla_{\phi_s} \hat{\ell}_{u,i}(\phi),$$

where $\eta \in \mathbb{R}^+$ is a learning rate; $\mathcal{B}_s \subset \mathcal{B}$ and $\mathcal{M}_s \subset \mathcal{M}$ are mini-batches of observed and missing ratings. This update function models how the propensity model affects training of the rating model, and does not actually update the rating model's parameters. The update function performs a single parameter update for computational efficiency at training. We find that performing more parameter updates does not further improve the performance but

Algorithm 1: `Vrt`: Variance-Regularized Training

Input: $S, \mathbf{R}^{\mathcal{B}}, \mathbf{R}^{\mathcal{U}}, \phi_0, \theta_0, \xi_0$

1 **for** $s = 0, ..., S-1$ **do**
2 Sample mini-batches $\mathcal{B}_s \subset \mathcal{B}$ ($\mathcal{M}_s \subset \mathcal{M}$) and $\mathcal{U}_s \subset \mathcal{U}$
3 Compute the lower loss $\mathcal{L}_{\mathrm{L}}(\phi_s, \theta_s)$ on $\mathcal{B}_s$ (and $\mathcal{M}_s$)
4 Compute an update function $\phi_{s+1}(\theta_s) \leftarrow \phi_s - \eta \nabla_{\phi_s} \mathcal{L}_{\mathrm{L}}(\phi, \theta_s)$
5 Compute the RU loss $\mathcal{L}_{\mathrm{RU}}(\phi_{s+1}(\theta_s))$ on $\mathcal{U}_s$ and $\mathcal{B}_s$
6 Update the propensity model $\theta_{s+1} \leftarrow \theta_s - \eta \nabla_{\theta_s} \mathcal{L}_{\mathrm{RU}}(\phi_{s+1}(\theta))$
7 Compute the lower loss $\mathcal{L}_{\mathrm{L}}(\phi_s, \theta_{s+1})$ on $\mathcal{B}_s$ (and $\mathcal{M}_s$)
8 Update the rating model $\phi_{s+1} \leftarrow \phi_s - \eta \nabla_{\phi_s} \mathcal{L}_{\mathrm{L}}(\phi, \theta_{s+1})$
9 **end**

Output: ϕ_S, θ_S

Algorithm 2: Doubly-Robust Variance-Regularized Training

Input: $T, S, \mathbf{R}^{\mathcal{B}}, \mathbf{R}^{\mathcal{U}}, \phi_0, \theta_0, \xi_0^0$

1 **for** $t = 0, ..., T-1$ **do**
2 **for** $s = 0, ..., S-1$ **do**
3 Sample a mini-batch $\mathcal{B}_t^s \subset \mathcal{B}$
4 Compute the EI loss $\mathcal{L}_{\mathrm{EI}}(\xi_t^s)$ on $\mathcal{B}_t^s$
5 Update the error model $\xi_t^{s+1} \leftarrow \xi_t^s - \eta \nabla_{\xi_t^s} \mathcal{L}_{\mathrm{EI}}(\xi)$
6 **end**
7 Call Alg. 1 by $\phi_{t+1}, \theta_{t+1} \leftarrow \mathtt{Vrt}(S, \mathbf{R}^{\mathcal{B}}, \mathbf{R}^{\mathcal{U}}, \phi_t, \theta_t, \xi_t^S)$
8 Copy the error model's parameter values $\xi_{t+1}^0 \leftarrow \xi_t^S$
9 **end**

Output: $\phi_T, \theta_T, \xi_T^0$

increases computational cost, which has also been observed previously [10]. Then, we replace the best parameter values $\phi^*(\theta)$ in the upper loss with the approximate ones $\phi_{s+1}(\theta_s)$ computed by the update function, and differentiate this update function to compute gradients w.r.t. the propensity model's parameters

$$\theta_{s+1} = \theta_s - \eta \nabla_{\theta_s} \mathcal{L}_{\mathrm{U}}(\phi_{s+1}(\theta)), \tag{17}$$

where the gradients can be derived by the chain rule

$$\begin{aligned}
\nabla_{\theta_s} \mathcal{L}_{\mathrm{U}}(\phi_{s+1}(\theta)) &= \sum_{v,j \in \mathcal{U}_s} \nabla_{\theta_s} s_{v,j}(\phi_{s+1}(\theta)), \\
&= \sum_{v,j \in \mathcal{U}_s} (\nabla_{\theta_s} \phi_{s+1}(\theta)) \nabla_{\phi_{s+1}(\theta_s)}^{\mathsf{T}} s_{v,j}(\phi), \\
&= \sum_{v,j \in \mathcal{U}_s} \left(\nabla_{\theta_s} \left(-\eta \sum_{u,i \in \mathcal{B}_s} \nabla_{\phi_s} \ell_{u,i}(\phi, \theta) \right) \right) \nabla_{\phi_{s+1}(\theta_s)}^{\mathsf{T}} s_{v,j}(\phi), \\
&= -\eta \sum_{u,i \in \mathcal{B}_s,\ v,j \in \mathcal{U}_s} (\nabla_{\theta_s} \nabla_{\phi_s} \ell_{u,i}(\phi, \theta)) \nabla_{\phi_{s+1}(\theta_s)}^{\mathsf{T}} s_{v,j}(\phi),
\end{aligned}$$

where $\mathcal{U}_s \subset \mathcal{U}$ is a mini-batch of unbiased ratings. The last equation holds because, by definition, the loss $\hat{\ell}_{u,i}(\neg\theta)$ is constant w.r.t. the propensity model's parameters. This equation requires second-order gradients, which can be computed by

$$\nabla_{\theta_s} \nabla_{\phi_s} \ell_{u,i}(\phi, \theta) = -\frac{\nabla_{\theta_s} q_\theta(\boldsymbol{x}_{u,i}) \nabla_{\phi_s}^{\mathsf{T}} (y_\phi(\boldsymbol{x}_{u,i}) - r_{u,i})^2}{q_{\theta_s}(\boldsymbol{x}_{u,i})^2}, \tag{18}$$

$$\nabla_{\theta_s} \nabla_{\phi_s} \ell_{u,i}(\phi, \theta) = -\frac{\nabla_{\theta_s} q_\theta(\boldsymbol{x}_{u,i}) \nabla_{\phi_s}^{\mathsf{T}} (s_{u,i}(\phi) - \hat{s}_{u,i}(\phi))}{q_{\theta_s}(\boldsymbol{x}_{u,i})^2}, \tag{19}$$

when using the IPS estimator and the DR estimator, respectively.

4.2.1 Variance Regularization. The propensity model trained using the above algorithm often has an increasing variance in propensity estimation (see Fig. 5b for empirical evidence). Such variance may cause training of the rating model to diverge because a portion of biased ratings may receive low propensity estimates. This portion of biased ratings will dominate the lower loss and thus lead to large gradients for training the rating model. To reduce the variance of propensity estimation, we propose a regularized-upper (RU) loss that regularizes the upper loss with the sample variance of propensity estimates on the mini-batch $\mathcal{B}_s$ of biased ratings

$$\mathcal{L}_{\mathrm{RU}}(\phi_{s+1}(\theta_s)) = \mathcal{L}_{\mathrm{U}}(\phi_{s+1}(\theta_s)) + \lambda \mathcal{L}_{\mathrm{SV}}(\theta_s). \tag{20}$$

Here, $\lambda \in \mathbb{R}^+$ is a regularization hyper-parameter and

$$\mathcal{L}_{\mathrm{SV}}(\theta_s) = \frac{1}{|\mathcal{B}_s| - 1} \sum_{u,i \in \mathcal{B}_s} \left(q_{\theta_s}(\boldsymbol{x}_{u,i}) - \sum_{v,j \in \mathcal{B}_s} \frac{q_{\theta_s}(\boldsymbol{x}_{v,j})}{|\mathcal{B}_s|} \right)^2 \tag{21}$$

is the sample variance. Since the sample variance is differentiable w.r.t. the propensity model's parameters, we can straightforwardly compute the gradients of the RU loss as

$$\nabla_{\theta_s} \mathcal{L}_{\mathrm{RU}}(\phi_{s+1}(\theta)) = \nabla_{\theta_s} \mathcal{L}_{\mathrm{U}}(\phi_{s+1}(\theta)) + \lambda \nabla_{\theta_s} \mathcal{L}_{\mathrm{SV}}(\theta). \tag{22}$$

where the gradients of the sample variance are given by

$$\begin{aligned}
\nabla_{\theta_s} \mathcal{L}_{\mathrm{SV}}(\theta_s) =& \frac{2}{|\mathcal{B}_s| - 1} \sum_{u,i \in \mathcal{B}_s} q_{\theta_s}(\boldsymbol{x}_{u,i}) \nabla_{\theta_s} q_\theta(\boldsymbol{x}_{u,i}) \\
&- \frac{2}{|\mathcal{B}_s|(|\mathcal{B}_s| - 1)} \sum_{u,i \in \mathcal{B}_s} q_{\theta_s}(\boldsymbol{x}_{u,i}) \sum_{u,i \in \mathcal{B}_s} \nabla_{\theta_s} q_\theta(\boldsymbol{x}_{u,i}).
\end{aligned} \tag{23}$$

We summarize the whole algorithm of training the propensity model and the rating models with variance regularization in Alg. 1. When using the IPS estimator, we do not need to sample mini-batches of missing ratings, which is shown in parentheses in Alg. 1. At each training step s, we update the propensity model (line 3-6) before performing actual update of the rating model (line 7-8). We can directly apply Alg. 1 when using the IPS estimator, but need to train an extra error model when using the DR estimator. Following Wang et al. [39], we train the error model by minimizing an error-imputation (EI) loss, i.e., a weighted average square of the differences between imputed errors $\{g_\xi(\boldsymbol{x}_{u,i})\}$ and prediction errors $\{e_{u,i}\}$ on the biased ratings $\mathbf{R}^{\mathcal{B}}$

$$\min_{\xi} \mathcal{L}_{\mathrm{EI}}(\xi) = \sum_{u,i \in \mathcal{B}} \frac{(g_\xi(\boldsymbol{x}_{u,i}) - e_{u,i})^2}{q_\theta(\boldsymbol{x}_{u,i})}. \tag{24}$$

We show the complete algorithm when using the DR estimator in Alg. 2. We alternate between training the error model and the rating model until a certain stopping criteria is satisfied. Such a joint training algorithm has shown to be effective in many other problems, e.g., dialogue response generation [8, 9].

4.2.2 Training Efficiency. Fig. 3 shows computation performed by LTD to update the propensity model and the rating model at a training step. We can see that LTD performs two forward and backward passes of the rating model on biased ratings and unbiased ratings, respectively, and a forward and backward pass of the propensity model on biased ratings. Then, LTD performs a backward-on-backward pass to obtain gradients for the propensity model, and a final backward pass to obtain gradients for the rating

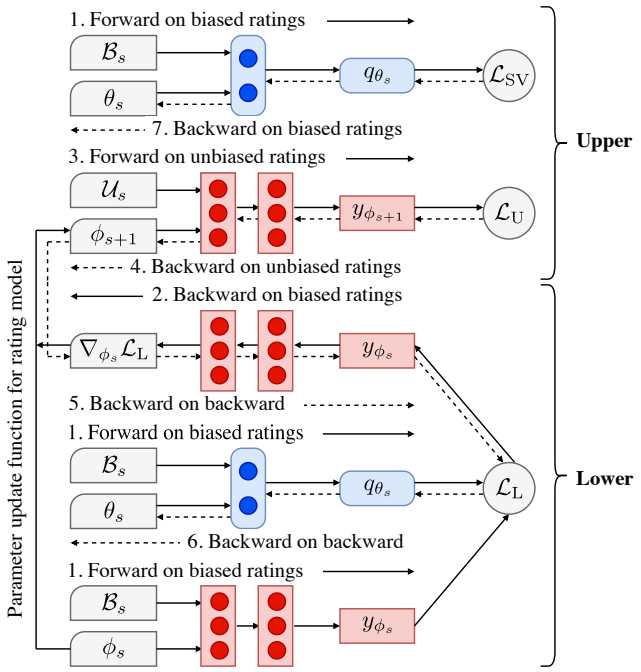


Fig. 3: Computational graph of the proposed variance-regularized training algorithm at training step s, which employs deep neural networks to estimate propensities q_θ and to predict ratings y_ϕ.

model. A backward-on-backward pass often takes about the same time as a forward pass [21]. Suppose both models take the same time for a forward and backward pass and are trained for the same number of steps, LTD only needs about 2× training time compared to two-phase learning, which performs two forward and backward passes for the propensity model and the rating model, respectively.

5 EXPERIMENTS

In this section, we empirically study the proposed LTD with the aim of answering the following research questions:

- **RQ1** What is the performance of LTD in rating prediction compared with existing approaches to the debiasing problem? Does LTD reduce the variance of propensity estimation compared to likelihood functions used by two-phase learning? How does the variance-regularized training algorithm of LTD converge compared to existing two-phase learning?
- **RQ2** How does each component of LTD, the propensity model and the variance regularization, contribute to the performance? How does the size of the unbiased dataset used in the upper-level optimization affect the performance?
- **RQ3** What propensity estimates does LTD automatically learn from an unbiased dataset? How does inverse propensity weighting contribute to improving the performance?

5.1 Experimental Settings

5.1.1 Datasets. Unbiased datasets are usually used to unbiasedly evaluate the performance of rating prediction because it is difficult to obtain unbiased performance estimation on biased datasets [39]. To the best of our knowledge, there are only two public datasets where unbiased ratings are available: (1) Music, which has 311,704 biased ratings and 54,000 unbiased ratings given by 15,400 users to 1,000 songs [19]. (2) Coat, which has 6,960 biased and 4,640 unbiased ratings given by 290 users to 300 coats [24].

5.1.2 Compared Approaches. We refer to approaches that do not explicitly handle biased datasets and those that do as *traditional* and *debiasing* approaches, respectively. We compare with two traditional approaches: (1) Matrix factorization (**MF**) [12]; (2) Neural factorization machine (**NF**) [5]. We compare to debiasing approaches that optimize a joint likelihood function: (3) **CPT-V** [18]; (4) **PMF-NMAR** [6]. We also compare to debiasing approaches that adopt two-phases learning: (5) **(MF/NF)-IPS** where the underlying rating model is MF or NF [24]; (6) **(MF/NF)-DR** where the rating model and the error model are MF or NF [39]. We pretrain a propensity model for (MF/NF)-(IPS/DR) as follows: (1) On Coat, we train a logistic regression model using all pairs of user and item covariates [24]; (2) Due to lack of user and item covariates, we train a naive Bayes model on Music [39].

We use **(MF/NF)-(IPS/DR)-LTD** to denote the proposed approach where the lower level is instantiated with the second phase of two-phase learning approaches (MF/NF)-(IPS/DR), respectively. The propensity model trained by LTD can use all features of observed ratings as inputs. On Music and Coat, we use features: (1) Number of ratings given by a user and received by an item. (2) Average rating given by a user and received by an item. (3) True rating of a user to an item. On Coat, we use additional features: (1) Gender of a user. (2) Age group of a user. (3) Location of a user. (4) How much a user is interested in fashion. (5) Gender of a coat. (6) Jacket type of a coat. (7) Color of a coat. (8) Whether a coat is promoted on front page. We represent discrete features by one-hot encoding and normalize continuous features into unit space.

5.1.3 Evaluation Protocol. To unbiasedly evaluate an approach's capability to handle biased datasets, we split the ratings on Music and Coat as follows. We use 90% of the biased ratings (training subset I) to train a rating model, and use the remaining 10% as a validation set to tune hyper-parameters. We split out 5% of the unbiased ratings (training subset II) to train a propensity model, and use the remaining 95% as a testing set. The 5% of the unbiased ratings is also used to train the rating model of LTD because the resulting rating model performs better. To ensure using the same amount of training data for all approaches, other approaches use both training subsets I and II for training.

We measure the performance by averaging mean square error (MSE) and mean absolute error (MAE) on the testing set over 10 different runs. A straightforward metric of the estimation variance of a propensity model q_θ is the sample variance (SV) of the propensity estimates on an unbiased dataset $\mathbf{R}^{\mathcal{U}}$

$$\mathrm{SV}(q_\theta) = \frac{1}{|\mathcal{U}|-1}\sum_{u,i\in\mathcal{U}}\left(q_\theta(\boldsymbol{x}_{u,i}) - \sum_{u,i\in\mathcal{U}}\frac{q_\theta(\boldsymbol{x}_{u,i})}{|\mathcal{U}|}\right), \tag{25}$$

Another metric is mean inverse square (MIS) of the propensity estimates, which characterizes the variability of a training process using propensity weighting (up to a certain constant) [24]

$$\mathrm{MIS}(q_\theta) = \frac{1}{|\mathcal{U}|}\sum_{u,i\in\mathcal{U}}\frac{1}{q_\theta(\boldsymbol{x}_{u,i})^2}. \tag{26}$$

Table 1: Performance averaged over 10 different runs.

Approach	MSE ± standard deviation		MAE ± standard deviation	
	Music	Coat	Music	Coat
MF	1.951 ± 0.003	1.349 ± 0.007	1.167 ± 0.002	0.948 ± 0.005
NF	1.586 ± 0.007	1.299 ± 0.013	1.034 ± 0.005	0.919 ± 0.009
CPT-V	1.181 ± 0.004	1.512 ± 0.020	0.914 ± 0.003	0.992 ± 0.012
PMF-NMAR	2.243 ± 0.010	1.279 ± 0.009	1.190 ± 0.006	0.910 ± 0.005
MF-IPS	1.069 ± 0.005	1.179 ± 0.012	0.857 ± 0.003	0.891 ± 0.008
NF-IPS	1.047 ± 0.006	1.137 ± 0.011	0.842 ± 0.005	0.863 ± 0.009
MF-DR	1.037 ± 0.003	1.058 ± 0.006	0.793 ± 0.002	0.807 ± 0.004
NF-DR	1.024 ± 0.007	1.033 ± 0.013	0.782 ± 0.003	0.784 ± 0.007
MF-IPS-LTD	1.009 ± 0.003	1.062 ± 0.006	0.836 ± 0.002	0.827 ± 0.004
NF-IPS-LTD	0.992 ± 0.006	1.041 ± 0.009	0.827 ± 0.003	0.812 ± 0.006
MF-DR-LTD	0.982 ± 0.002	0.982 ± 0.003	0.781 ± 0.001	0.770 ± 0.002
NF-DR-LTD	**0.977** ± 0.003	**0.973** ± 0.006	**0.774** ± 0.002	**0.759** ± 0.004

* The bottom four approaches, (MF/NF)-(IPS/DR)-LTD, are proposed.

Since on a biased dataset $\mathbf{R}^{\mathcal{B}}$ the true propensities satisfy a condition $\mathbb{E}_{\mathcal{B}}[\sum_{u,i\in\mathcal{B}} p_{u,i}^{-1}] = MN$, we normalize propensity estimates by $q_\theta(\boldsymbol{x}_{u,i}) \leftarrow (M^{-1}N^{-1}\sum_{u,i\in\mathcal{B}} q_\theta(\boldsymbol{x}_{u,i})^{-1})q_\theta(\boldsymbol{x}_{u,i})$ before comparing the variance of propensity estimation.

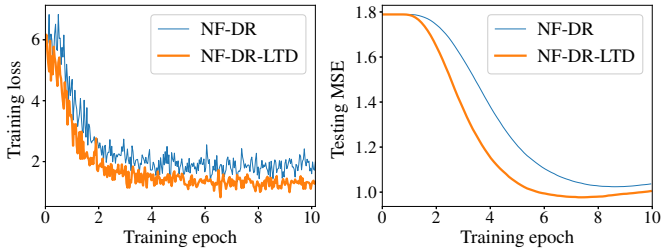
5.1.4 Implementation Details. We implement the propensity model as a multilayer perceptron (MLP) with one hidden layer. For the MLP, we search the activation function from {ReLU, sigmoid, tanh} and the hidden layer size from {4, 16, 64, 256}. We search the regularization hyper-parameter λ from 0.01 to 100. To avoid overfitting, we apply L2 regularization on model parameters and search the L2 regularization from 0.01 to 1. We set the batch size to 128 on Music and 64 on Coat. All models are trained by AdaGrad [2] with a learning rate from {0.001, 0.005, 0.01, 0.05}. We tune all hyper-parameters based on the performance on the validation set.

5.2 RQ1 Comparative Results

5.2.1 Overall Performance. We show the MSE and the MAE of all approaches on Music and Coat in Table 1. We can see that the proposed LTD significantly outperforms the corresponding two-phase learning approach. For example, MF-IPS-LTD (1.062) outperforms MF-IPS (1.179) by 9.9% on Coat. We also find that NF-DR-LTD performs the best on both datasets, e.g., NF-DR-LTD achieves the smallest MSE (0.977) on Music. By comparing (MF/NF)-(IPS/DR)-LTD, we can see that LTD benefits from: (1) Advanced model architecture: NF enhances the expressiveness of MF by modeling non-linear and high-order interactions between features. (2) Improved lower loss: the DR estimator addresses the variance issue of the IPS estimator by introducing an error model. In general, the debiasing approaches outperform the traditional ones by explicitly handling selection biases in the training set. However, the debiasing approaches, CPT-V and PMF-NMAR, perform worst on Coat and Music, respectively. This is probably because these two approaches make strong generative assumptions and require highly complex inferences. Unlike CPT-V and PMF-NMAR, our approaches neither make generative assumptions nor require complex inferences, leading to consistent improvements. The improvements of LTD under MAE are not as significant as those under MSE because the losses of LTD are based on square errors rather than absolute errors.

Table 2: Variance of propensity estimation on Music and Coat

	Approach	SV		MIS	
		Music	Coat	Music	Coat
	Two-phase	$7.079 \cdot 10^{-4}$	$6.827 \cdot 10^{-3}$	$5.996 \cdot 10^{3}$	$2.164 \cdot 10^{3}$
Bi-level	MF-IPS-LTD	$4.876 \cdot 10^{-6}$	$1.635 \cdot 10^{-5}$	$2.682 \cdot 10^{3}$	$1.473 \cdot 10^{2}$
Bi-level	NF-IPS-LTD	$4.136 \cdot 10^{-6}$	$4.962 \cdot 10^{-6}$	$2.661 \cdot 10^{3}$	$1.471 \cdot 10^{2}$
Bi-level	MF-DR-LTD	$9.203 \cdot 10^{-7}$	$9.898 \cdot 10^{-6}$	$2.507 \cdot 10^{3}$	$1.467 \cdot 10^{2}$
Bi-level	NF-DR-LTD	$\mathbf{7.066 \cdot 10^{-7}}$	$\mathbf{4.534 \cdot 10^{-6}}$	$\mathbf{2.494 \cdot 10^{3}}$	$\mathbf{1.466 \cdot 10^{2}}$



(a) DR loss during training. **(b) Testing MSE during training.**

Fig. 4: Comparing convergence of NF-DR and NF-DR-LTD on Music.

Table 3: Performance of NF-DR-LTD on Music and Coat.

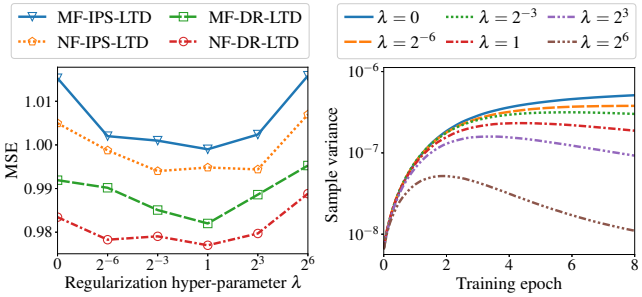
Propensity model	MSE		MAE	
	Music	Coat	Music	Coat
Simple propensity model	1.044	1.028	0.812	0.784
Logistic regression model	1.003	1.006	0.791	0.774
MLP with one hidden layer	**0.977**	**0.973**	**0.774**	**0.759**
MLP with two hidden layers	0.996	0.990	0.783	0.763

5.2.2 Variance of Propensity Estimation. We compare the variance of propensity estimation when completing the training of debiasing approaches based on propensity weighting. We show the results under SV and MIS on Music and Coat in Table 2. The variance of propensity estimation by bi-level optimization is much smaller than that by two-phase learning (up to 3 orders of magnitude).

Since NF-DR-LTD performs the best on all datasets, we will focus on its results in the following discussion. We observe that the results of other LTD approaches are similar to those of NF-DR-LTD.

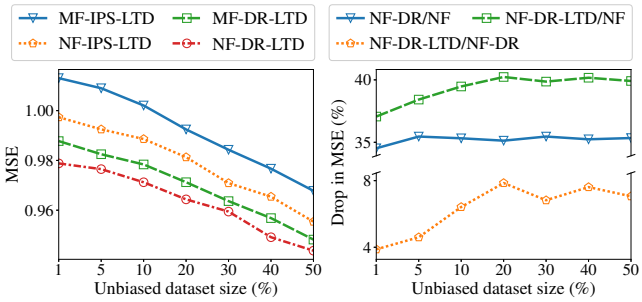
5.2.3 Analysis of Convergence. It is well-known that bi-level optimization is difficult to perform and we further introduce variance regularization into the optimization, so it is meaningful to analyze the convergence of LTD. We plot training loss and testing MSE of NF-DR and NF-DR-LTD against training epochs on Music in Fig. 4a and Fig. 4b, respectively. We can see that after eight epochs, both training loss and testing MSE of NF-DR-LTD converge. The convergence rate of NF-DR-LTD in terms of training epochs is comparable to that of NF-DR.

5.3 RQ2 Ablation Studies



(a) MSE of our approaches. (b) Training NF-DR-LTD.

Fig. 5: Effects of the variance regularization on Music.



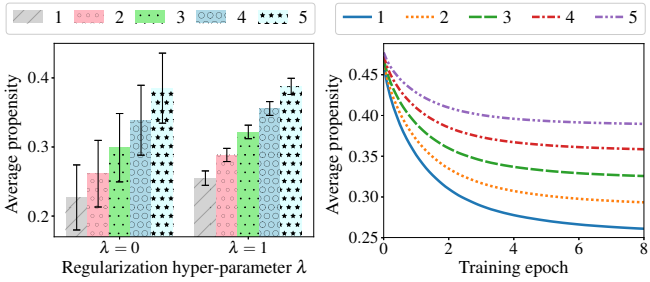
(a) MSE of our approaches. (b) Comparing two approaches.

Fig. 6: Effects of varying unbiased dataset size on Music.

5.3.1 Propensity Model. To study the preferred model architectures for propensity estimation, we use a simple propensity model $q_\theta(\boldsymbol{x}_{u,i}) = \sigma(w_{u,i})$ ($w_{u,i} \in \mathbb{R}$ is a parameter) [21] and a logistic regression model to estimate propensities in NF-DR-LTD. We show the results in Table 3. We can see that using MLP with one hidden layer performs the best. Using more hidden layers does not help largely because the unbiased dataset is too small to train the parameters of additional hidden layers. The simple propensity model performs the worst because it ignores all features (e.g., the true rating), which can be quite predictive in propensity estimation. MLP is preferred over the logistic regression model due to its ability to approximate almost any continuous functions in theory [1].

5.3.2 Variance Regularization. We study how the regularization hyper-parameter λ affects the performance of LTD. We plot the result on Music in Fig. 5a. We can see that the variance regularization is indeed beneficial, e.g., MF-IPS-LTD with $\lambda = 1$ (1.009) outperforms that with $\lambda = 0$ (1.026) by 1.7%. We further compute the sample variance of propensity estimates on the unbiased testing set. We show the result during training NF-DR-LTD on Music in Fig. 5b. We find that the sample variance keeps growing when $\lambda = 0$, but drops after a few epochs when $\lambda \geq 1$. The sample variance in NF-DR-LTD during training is consistently smaller than that in two-phase learning ($7.079 \cdot 10^{-4}$) in Table 2.

5.3.3 Unbiased Dataset Size. We study how the size of the unbiased dataset used for training affects the performance. We resplit unbiased ratings into a fixed testing set (50%) and a varying training set for the propensity model (1% to 50%). We show the results on Music in Fig. 6a. We can see that LTD performs better when the unbiased dataset size increases, as expected. We further explore the unbiased dataset size required to learn a good propensity model within LTD by computing improvements between NF, NF-DR, and NF-DR-LTD. We show the results in terms of percentage decrease in MSE on Music in Fig. 6b. We can see that improvements of NF-DR-LTD over NF and NF-DR keep growing when we vary the unbiased dataset size from 1% to 20%, but do not grow afterwards (improvements over NF-DR are up to a 7.9% drop in MSE). In contrast, improvements of NF-DR over NF keep steady because a small unbiased dataset can fit the naive Bayes model well.



(a) Bars show standard deviation. (b) Hyper-parameter $\lambda = 1$.

Fig. 7: Average propensity of rating 1 to 5 by NF-DR-LTD on Music.

5.4 RQ3 Propensity Weighting

To explore what propensity estimates does LTD learn, we average propensity estimates for each distinct value of the true ratings (i.e., 1 to 5) on the training set. We show the results when we complete training NF-DR-LTD on Music in Fig. 7a. We can see that on average higher ratings do have larger propensity estimates, which is consistent with the findings in previous studies [18, 19]. We study how such results are achieved by computing the average propensity estimate for each distinct value of the true ratings during training. We show the results of training NF-DR-LTD on Music in Fig. 7b. We can see that the higher a rating value is, the slower its average propensity estimate decreases. These results indicate that by learning from an unbiased dataset, LTD correctly assigns larger propensity estimates to higher ratings to make the training sets less biased. We also compare NF-DR-LTD to NF by averaging MSE and MAE for each distinct rating value on Music and Coat in Table 4. Compared to NF, NF-DR-LTD performs better on lower ratings by sacrificing the performance on higher ratings.

6 CONCLUSIONS

In this paper, we showed the impact of having a small set of unbiased ratings on alleviating selection biases when training recommender systems. We proposed learning to debias (LTD), a novel approach that utilizes a few unbiased ratings to improve the generalization ability of a rating model trained on biased datasets. To learn the parameters within LTD, we developed an efficient training algorithm, which can effectively reduce the variance of propensity estimation while training the rating model. We showed how to apply

Table 4: Comparing the performance of NF-DR-LTD to that of NF for each distinct value of the true ratings on Music and Coat.

	MSE dec. (↓) or inc. (↑)		MAE dec. (↓) or inc. (↑)	
Rating value	Music	Coat	Music	Coat
1	↓ 86.77%	↓ 47.88%	↓ 65.54%	↓ 36.26%
2	↓ 60.85%	↓ 21.90%	↓ 25.71%	↓ 12.65%
3	↑ 79.10%	↑ 10.08%	↑ 63.53%	↑ 5.98%
4	↑ 93.52%	↑ 10.72%	↑ 67.60%	↑ 6.24%
5	↑ 82.82%	↑ 10.75%	↑ 58.19%	↑ 7.06%

LTD to improve over two representative types of unbiased performance estimators on two real-world datasets. Compared to the state-of-the-art approaches, LTD achieves consistent performance improvements, and the advantage is up to 7.9% in the error of rating prediction and order of magnitude in the variance of propensity estimation. For future work, we will explore theoretical bounds on the generalization ability of a rating model trained by LTD.

REFERENCES

[1] Balázs Csanád Csáji et al. 2001. Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary* (2001).

[2] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research (JMLR)* (2011).

[3] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. 2017. Forward and reverse gradient-based hyperparameter optimization. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*.

[4] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. 2018. Bilevel programming for hyperparameter optimization and meta-learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*.

[5] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

[6] José Miguel Hernández-Lobato, Neil Houlsby, and Zoubin Ghahramani. 2014. Probabilistic matrix factorization with non-random missing data. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*.

[7] Sha Hu, Zhicheng Dou, Xiaojie Wang, Tetsuya Sakai, and Ji-Rong Wen. 2015. Search result diversification based on hierarchical intents. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM)*.

[8] Xinting Huang, Jianzhong Qi, Yu Sun, and Rui Zhang. 2020. Mala: Cross-domain dialogue generation with action learning. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*.

[9] Xinting Huang, Jianzhong Qi, Yu Sun, and Rui Zhang. 2020. Semi-Supervised Dialogue Policy Learning via Stochastic Reward Estimation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

[10] Simon Jenni and Paolo Favaro. 2018. Deep Bilevel Learning. In *Proceedings of the 15th European Conference on Computer Vision (ECCV)*.

[11] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM)*.

[12] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* (2009).

[13] Guang Ling, Haiqin Yang, Michael R Lyu, and Irwin King. 2012. Response aware model-based collaborative filtering. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*.

[14] Donghua Liu, Jing Li, Bo Du, Jun Chang, and Rong Gao. 2019. DAML: Dual Attention Mutual Learning between Ratings and Reviews for Item Recommendation. In *Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

[15] Jiongnan Liu, Zhicheng Dou, Xiaojie Wang, Shuqi Lu, and Ji-Rong Wen. 2020. DVGAN: A Minimax Game for Search Result Diversification Combining Explicit and Implicit Features. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

[16] Shuqi Lu, Zhicheng Dou, Chenyan Xiong, Xiaojie Wang, and Ji-Rong Wen. 2020. Knowledge Enhanced Personalized Search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

[17] Dougal Maclaurin, David Duvenaud, and Ryan P Adams. 2015. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*.

[18] Benjamin M Marlin and Richard S Zemel. 2009. Collaborative prediction and ranking with non-random missing data. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys)*.

[19] Benjamin M Marlin, Richard S Zemel, Sam Roweis, and Malcolm Slaney. 2007. Collaborative filtering and the missing at random assumption. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI)*.

[20] Fabian Pedregosa. 2016. Hyperparameter optimization with approximate gradient. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*.

[21] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to Reweight Examples for Robust Deep Learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*.

[22] Masahiro Sato, Sho Takemori, Janmajay Singh, and Tomoko Ohkuma. 2020. Unbiased Learning for the Causal Effect of Recommendation. In *Fourteenth ACM Conference on Recommender Systems (RecSys)*.

[23] Tobias Schnabel and Paul N Bennett. 2020. Debiasing Item-to-Item Recommendations With Small Annotated Datasets. In *Fourteenth ACM Conference on Recommender Systems (RecSys)*.

[24] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: debiasing learning and evaluation. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*.

[25] Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. 2019. Truncated back-propagation for bilevel optimization. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[26] Ying Shan, T Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and JC Mao. 2016. Deep crossing: Web-scale modeling without manually crafted combinatorial features. In *Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

[27] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I Nikolenko. 2020. RecVAE: a New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM)*.

[28] Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

[29] Harald Steck. 2013. Evaluation of recommendations: rating-prediction and ranking. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys)*.

[30] Yixin Su, Rui Zhang, Sarah Erfani, and Zhenghua Xu. 2021. Detecting Beneficial Feature Interactions for Recommender Systems. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*.

[31] Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual risk minimization: learning from logged bandit feedback. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*.

[32] Adith Swaminathan and Thorsten Joachims. 2015. The self-normalized estimator for counterfactual learning. In *Proceedings of the 28th Conference on Neural Information Processing Systems (NeurIPS)*.

[33] Menghan Wang, Mingming Gong, Xiaolin Zheng, and Kun Zhang. 2018. Modeling dynamic missingness of implicit feedback for recommendation. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS)*.

[34] Xiaojie Wang, Zhicheng Dou, Tetsuya Sakai, and Ji-Rong Wen. 2016. Evaluating search result diversity using intent hierarchies. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR)*.

[35] Xiaojie Wang, Jianzhong Qi, Kotagiri Ramamohanarao, Yu Sun, Bo Li, and Rui Zhang. 2018. A joint optimization approach for personalized recommendation diversification. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.

[36] Xiaojie Wang, Ji-Rong Wen, Zhicheng Dou, Tetsuya Sakai, and Rui Zhang. 2017. Search result diversity evaluation based on intent hierarchies. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2017).

[37] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2018. Kdgan: Knowledge distillation with generative adversarial networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[38] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2019. Adversarial distillation for learning with privileged provisions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2019).

[39] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2019. Doubly robust joint learning for recommendation on data missing not at random. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*.

[40] Longqi Yang, Eugene Bagdasaryan, Joshua Gruenstein, Cheng-Kang Hsieh, and Deborah Estrin. 2018. Openrec: A modular framework for extensible and adaptable recommendation algorithms. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining (WSDM)*.