Improving Ad Click Prediction by Considering Non-displayed Events

Bowen Yuan
National Taiwan University
f03944049@csie.ntu.edu.tw

Jui-Yang Hsia*
National Taiwan University
hsiajuiyang5174@gmail.com

Meng-Yuan Yang[†]
ETH Zurich
meyang@ethz.ch

Hong Zhu
Huawei Noah's ark lab
zhuhong8@huawei.com

Chih-Yao Chang
Huawei Noah's ark lab
chang.chih.yao@huawei.com

Zhenhua Dong
Huawei Noah's ark lab
dongzhenhua@huawei.com

Chih-Jen Lin
National Taiwan University
cjlin@csie.ntu.edu.tw

ABSTRACT

Click-through rate (CTR) prediction is the core problem of building advertising systems. Most existing state-of-the-art approaches model CTR prediction as binary classification problems, where displayed events with and without click feedbacks are respectively considered as positive and negative instances for training and offline validation. However, due to the selection mechanism applied in most advertising systems, a selection bias exists between distributions of displayed and non-displayed events. Conventional CTR models ignoring the bias may have inaccurate predictions and cause a loss of the revenue. To alleviate the bias, we need to conduct counterfactual learning by considering not only displayed events but also non-displayed events. In this paper, through a review of existing approaches of counterfactual learning, we point out some difficulties for applying these approaches for CTR prediction in a real-world advertising system. To overcome these difficulties, we propose a novel framework for counterfactual CTR prediction. In experiments, we compare our proposed framework against state-of-the-art conventional CTR models and existing counterfactual learning approaches. Experimental results show significant improvements.

CCS CONCEPTS

• **Information systems → Computational advertising**.

KEYWORDS

CTR prediction, Recommender system, Missing not at random, Selection bias, Counterfactual learning

---

1 INTRODUCTION

Online advertising is a major profit model for most Internet companies where advertisers pay publishers for promoting ads on their services. Advertising systems perform in the role of a dispatcher that decides which and where ads are displayed. Though there are many compensation methods in advertising systems, in this work, we focus on cost-per-click (CPC) systems, where advertisers are only charged if their ads are clicked by users. We give a simplified example of CPC systems in Figure 1.

Because the number of positions from publishers is limited, the system must select the most valuable ads to display. Therefore, when the system receives a request, all ads must be ranked according to the following expected revenue

$$\text{ECPM} = P \times R, \tag{1}$$

where

$$P = \text{Pr}(\text{click} \mid \text{event}) \tag{2}$$

is the probability of an event associated with a (request, ad) pair being clicked, and R is the price paid by advertisers if the event is clicked. Then the system selects and displays the top events on the publishers. In Figure 1, the example shows that five events bid for three advertising positions, so the system ranks them by (1). Event '1', '2' and '3' are the top three, and get the chance to be displayed.

Once the event is displayed, the system will receive signals if a customer clicks it, which can be considered as the feedback of this event. Then the system gains revenue from the advertiser. As the example shown in Figure 1, among the three displayed events, event '2' is clicked, while the other two are not clicked.

To maximize the revenue, the estimation of (2), also referred as click-through rate (CTR) prediction, is the core problem of building advertising systems. To estimate CTR, the most widely used

Figure 1: A simplified example to illustrate the procedure of selecting events and collecting feedbacks in CPC systems. The symbols '✓', '✗', and '?' indicate the event is clicked, not clicked, and not displayed, respectively.

approach is to learn from historically displayed events via machine learning techniques. Specifically, assume that we have the following L events

$$(y_1, \boldsymbol{x}^1), \cdots, (y_L, \boldsymbol{x}^L), \tag{3}$$

where y_l and $\boldsymbol{x}^l$ are the label and the feature vector of the lth events, respectively. To model CTR prediction as binary classification problems, we assume that the label y satisfies

$$y = \begin{cases} 1 & \text{the event was displayed and clicked,} \\ -1 & \text{the event was displayed but not clicked.} \end{cases}$$

Then we need to find a decision function $\hat{y}(\boldsymbol{x}) \in [-\infty, \infty]$ so that it can be used to predict the label. For example, in a typical classification setting the predicted label is

$$\begin{cases} 1 & \text{if } \hat{y}(\boldsymbol{x}) \geq 0, \\ -1 & \text{otherwise.} \end{cases}$$

For CTR prediction, a probability output may be needed, so we can transform $\hat{y}(\boldsymbol{x})$ to a value in $[0, 1]$ by a sigmoid function. To obtain a good decision function, many state-of-the-art models for CTR prediction problems have been proposed (e.g., logistic regression [3, 21, 25], factorization machines [24], field-aware factorization machines [13, 14], and deep learning approaches [6, 23, 34]). A review of some of these models will be given in Section 2.1.

Recall the example in Figure 1. Two events are not displayed so their feedbacks are not available. Most works mentioned above ignore non-displayed events. We think the reason supporting this decision is that some pioneering works (e.g., [25, Section 3]) thought causality exists between a display and a click, where the click is understood as a consequence of the display. Then, they give the following basic but far-reaching assumption:

ASSUMPTION 1.1. *The probability of an event being clicked but not displayed is zero.*

With the assumption, no click information is included for non-displayed events. Therefore, the probability of an event being clicked can be measured by CTR defined as

$$\text{CTR} = \frac{\text{number of clicks}}{\text{number of displayed events}} \times 100\%.$$

Table 1: Main notation.

L	numbers of displayed events
$\hat{L}$	numbers of displayed and non-displayed events
$\boldsymbol{x}, y$	feature vector and label
D	numbers of features
S_t	set of events logged by the uniform policy
S_c	set of events logged by non-uniform policies
$\hat{y}(\cdot), \sigma(\cdot)$	CTR model and imputation model
m, n	numbers of requests and ads
Ω	set of displayed (request, ad) pairs
$Y, \hat{Y}, A$	label matrix, output matrix, and the matrix of imputed labels

However, in Section 2.2, by following some past works (e.g., [15, 28]), we argue the importance of handling non-displayed events. That is, through considering non-displayed events as unlabeled instances, CTR prediction should be modeled as a learning problem with labeled and unlabeled instances. Further, through an A/B test conducted in our online system, we show that due to the inconsistency between distributions of labeled and unlabeled samples, the ignorance of non-displayed events may cause a strong bias and inaccurate predictions. To eliminate the bias in CTR prediction, not only displayed events but also non-displayed events should be considered. However, because of lacking labels for non-displayed events, we need to solve a learning problem with partial labels, which is also referred to as "batch learning from logged bandit feedbacks" or "counterfactual learning" [28].

In Section 3, through a review on existing approaches of counterfactual learning, we point out some difficulties of applying them in a real-world CPC system. For example, many existing approaches required that events are sampled to be displayed through the importance sampling technique. However, as illustrated in Figure 1, most real-world CPC systems deterministically select the top events with the highest expected revenues. Therefore, these approaches are inapplicable to practical CTR prediction. Additionally, some existing approaches need to solve a difficult optimization problem, a situation not impractical for industrial-scale CTR prediction. Our main contribution in Section 4 is to propose a novel framework for counterfactual CTR prediction to overcome these difficulties. To address the issue of the training efficiency, we link the proposed framework to recommender systems with implicit feedbacks and develop efficient algorithms. In Section 5, experiments are conducted on a semi-synthetic small data set and a real-world data set collected from our online system. Results show our framework is superior to both conventional and counterfactual learning approaches. Table 1 gives main notation in this paper. Supplementary materials and code for experiments are at https://www.csie.ntu.edu.tw/~cjlin/papers/occtr.

2 HANDLING UNOBSERVED DATA IN CTR PREDICTION

In this section, we first give a brief review on the widely-used settings for CTR prediction. Next we show that under this setting, a bias may exist and cause inconsistency between offline and online evaluations.

2.1 A Review on CTR Prediction by Binary Classification

As mentioned in Section 1, the task of CTR prediction is to find a model, which inputs x, outputs $\hat{y}(x)$ and generates a probability value as the prediction of (2). If we have L displayed events as training samples shown in (3), most CTR works solve the following optimization problem.

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \sum_{l=1}^{L} \ell(y_l, \hat{y}_l), \tag{4}$$

where $\hat{y}_l \equiv \hat{y}(x^l)$, $\mathcal{W}$ is a set of model parameters, $\ell(a, b)$ is a loss function convex in b, and $R(\mathcal{W})$ is the regularization term with the parameter λ decided by users. Logistic regression (LR) is a commonly used model for CTR prediction. Assume D is the number of features in each vector x^l, and a vector $w \in \mathcal{R}^D$ is considered as the set $\mathcal{W}$ of model parameters. LR then sets

$$\hat{y}(x) = w^T x$$

and considers the following logistic loss

$$\ell(a, b) = \log(1 + e^{-ab}). \tag{5}$$

Past studies (e.g., [3, 14]) have shown that learning the effect of feature conjunctions is crucial for CTR prediction. However, due to the high dimensionality and the high sparsity of x, LR may have difficulty to learn from all feature conjunctions of x. To overcome the difficulty, factorization machine (FM) [24] and its variants such as field-aware factorization machine (FFM) [14] are proposed. They consider a low-rank approximation to the coefficient matrix of all feature conjunctions. The output of FM is

$$\hat{y}_{FM}(x) = \sum_{j_1=1}^{D} \sum_{j_2 \ge j_1+1}^{D} (w_{j_1}^T w_{j_2}) x_{j_1} x_{j_2}, \tag{6}$$

where $w_j \in \mathcal{R}^k$, $\forall j$ are the latent vectors and k is a pre-specified value. FFM extends FM to assume that features are grouped to several "fields."

$$\hat{y}_{FFM}(x) = \sum_{j_1=1}^{D} \sum_{j_2 \ge j_1+1}^{D} (w_{j_1, f_{j_2}}^T w_{j_2, f_{j_1}}) x_{j_1} x_{j_2}, \tag{7}$$

where f_{j_1} and f_{j_2} are the corresponding fields of features j_1 and j_2, respectively, and $w_{j_1, f_{j_2}} \in \mathcal{R}^k$ is a latent vector which learns the latent effect between feature j_1 and the interacted features belonging to the field f_{j_2}. It has been shown in some past competitions (e.g., [12, 14]) and industry scenarios [13] that FFM gives state-of-the-art results on CTR prediction.

For evaluation, two metrics are commonly used [3, 7, 14]. Suppose L is the number of events being evaluated. The first criterion is the negative logarithmic loss (NLL):

$$\text{NLL} \equiv -\frac{1}{L} \sum_{l=1}^{L} \log(1 + e^{-y_l \hat{y}_l}). \tag{8}$$

Another one is the area under the ROC curve (AUC):

$$\text{AUC} \equiv \frac{\sum_{l=1}^{L_p} \text{Rank}_l - \binom{L_p}{2}}{(L_p(L - L_p))}, \tag{9}$$
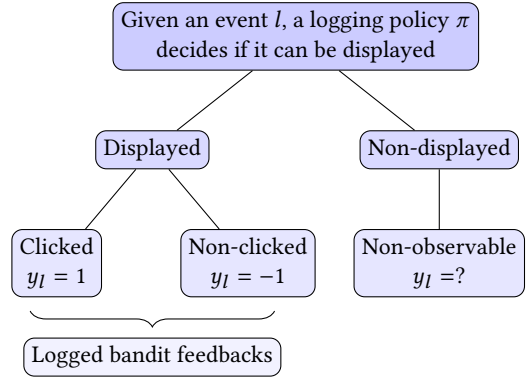
Figure 2: **A diagram of CTR prediction being formulized as a selective labels problem**

where L_p is the number of positive instances, and Rank_l is the rank of a positive event l in all L events, which are ranked in a descending order according to their predicted values.

2.2 The Importance of Handling Non-displayed Events

Most CTR models conducted training and validation on the displayed events. Let

$$P_L = \text{Pr}(\text{click} \mid \text{event}, \text{displayed}) \tag{10}$$

be the probability of displayed events being clicked. Models obtained by solving (4) implicitly assume that

$$P = P_L. \tag{11}$$

However, in the subsequent discussion, we show that this assumption is not precise. We start by following [15] to formalize CTR prediction as a selective labels problem. As illustrated in Figure 2, we can view our system as a labeling platform, where customers help to label $L + L_U$ instances. However, due to the limitation of capacity, an algorithm π (also called logging policy) selects L instances and asks customers to label them. From this view, the displayed events are labeled instances (also called logged bandit feedbacks), while non-displayed events are unlabeled instances. Because of lacking labels, the click distribution of non-displayed events,

$$P_U = \text{Pr}(\text{click} \mid \text{event}, \text{non-displayed}), \tag{12}$$

is unknown rather than zero in Assumption 1.1. It can be known that if the logging policy π completely uniformly selects events by random, distributions of click behavior can be the same on displayed and non-displayed events. Then we can predict P via estimating P_L. Unfortunately, this condition can not be held for CTR prediction due to the non-uniform selection of displayed events as introduced in Section 1.

To better explain this, we deployed a uniform policy on a small part of live traffics in our online system, while the rest was still served by non-uniform policies. According to logging policies, the labeled events are grouped into the following two sets:

$$S_t = \{(y_l, x^l) \mid \text{event } l \text{ was logged by the uniform policy}\},$$

$$S_c = \{(y_l, x^l) \mid \text{event } l \text{ was logged by non-uniform policies}\}.$$
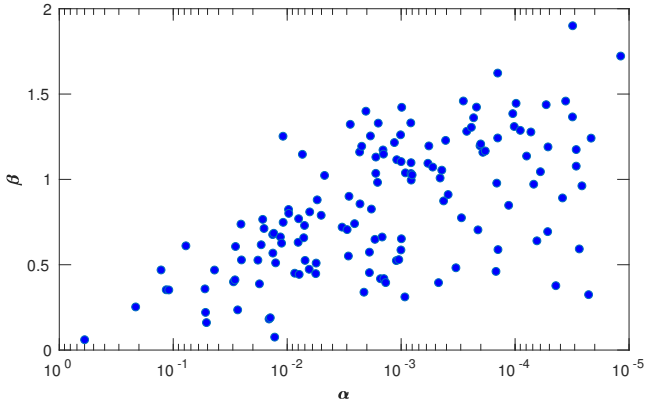
Figure 3: The coordinates (α, β) of each dot indicate an ad's displayed ratio defined in (13) and the log-scaled CTR ratio between events from S_c and S_t; see (14).

In Figure 3, we check how CTR estimated from logs of non-uniform policies differs from the true CTR. The x-axis is the displayed ratio of an ad, where each point is calculated by

$$\alpha_j = \frac{\text{\# displayed events associated with ad } j}{\text{\# (displayed+non-displayed) events associated with ad } j}. \tag{13}$$

The y-axis is the log-scaled ratio of an ad's observed CTR in S_c to that in S_t, where each point is calculated by

$$\beta_j = \log(\frac{\text{the CTR of ad } j \text{ observed in } S_c}{\text{the CTR of ad } j \text{ observed in } S_t}). \tag{14}$$

As we mentioned earlier, for events logged by a uniform policy, $P = P_L$. Thus an ad's CTR value from S_t can be considered as the true P's average over all events associated with this ad. From Figure 3, with $\beta_j > 0$, an ad's CTR from events in S_c is higher than that from events in S_t. This observation confirms a gap between P_L and P when events are logged by non-uniform policies. More importantly, the gap varies on different ads. For example, for those ads with lower α scores, the gap enlarges, while for those ads with higher α scores, the gap narrows. It can be realized that for an ad favoured by the proceeding policies, more associated events can be displayed and fewer events are not labeled. Then the observed CTR is closer to the unbiased one obtained from the fully labeled set S_t. We can conclude that if we fit CTR models on events logged by non-uniform policies, a biased estimator is obtained. It may overestimate the expected revenue of ads with lower α scores and wrongly display too many of them.

The bias caused by using a non-uniform logging policy is often referred to as "selection bias" in the literature of econometrics [8], and "missing not at random" in the field of statistics [18]. Ideally, a simple way to avoid the selection bias is to directly train CTR models on S_t. However, in most scenarios of CTR prediction, the feature set is high dimensional and sparse, so a large training set is required for learning a model with good generalization ability. The higher CTR shown in Figure 3 by displaying events in S_c indicates that if we deploy the uniform policy on more traffics, the overall revenue may decrease. Therefore, forming a large S_t through deploying the uniform policy on a significant amount of traffic is not practically

viable for many internet companies. In this paper, we will develop more practical ways to eliminate the bias carried in the data logged by non-uniform logging policies.

Now we aim to consider both displayed and non-displayed events. Assume we have the following $\hat{L}$ events

$$(y_1, \boldsymbol{x}^1), \cdots, (y_{\hat{L}}, \boldsymbol{x}^{\hat{L}}), \tag{15}$$

where $\hat{L}$ with $\hat{L} \gg L$ is the number of all possible events. To estimate P without bias, we expect to solve the following optimization problem.

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \sum_{l=1}^{\hat{L}} \ell(y_l, \hat{y}_l). \tag{16}$$

However, because feedbacks of non-displayed events can not be observed for most CPC systems, the corresponding y_l for $l = L + 1, \cdots, \hat{L}$ are not available. Therefore, (16) can not be solved by conventional methods based on supervised learning. Solving (16) with selective labels is also referred to as "batch learning from logged bandit feedbacks" and "counterfactual learning" [28]. We give a brief review of existing approaches on this topic in the next section.

3 EXISTING APPROACHES FOR LEARNING FROM SELECTIVE LABELS

Existing approaches for solving (16) with selective labels broadly fall into the following three categories: direct method, the inverse-propensity-scoring method, and the doubly robust method.

3.1 Direct Method

As we mentioned above, the key challenge in solving (16) is that we lack labels of those non-displayed events. Direct methods are based on an intuitive idea that we can find a relationship $\sigma(\cdot)$ between labels and features of displayed events such that $\sigma(\boldsymbol{x}^l) \approx y_l$, for $l = 1, \cdots, L$. We call $\sigma(\cdot)$ as an imputation model, which can impute labels for all events and have the following $\hat{L}$ labeled samples

$$(\sigma_1, \boldsymbol{x}^1), \cdots, (\sigma_{\hat{L}}, \boldsymbol{x}^{\hat{L}}),$$

where $\sigma_l \equiv \sigma(\boldsymbol{x}^l)$. Then we solve the following standard supervised learning problem.

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \sum_{l=1}^{\hat{L}} \ell(\sigma_l, \hat{y}_l). \tag{17}$$

Clearly, the performance of a direct method is determined by the quality of $\sigma(\cdot)$. Many past works found $\sigma(\cdot)$ via conducting machine learning techniques on the labeled data set. For example, ridge linear regression [4], and random forests [16] are considered.

3.2 Inverse-Propensity-Scoring (IPS) Method

By assuming the logging policy is stochastic, the IPS method proposed in [9] weights each labeled event with the inverse of its propensity score, which refers to the tendency or the likelihood of an event being logged. Specifically, given the event l, we can observe an indicator $O_l \in \{0, 1\}$, which indicates if the event was displayed. Following [27, 29], we assume that O_l is sampled from a Bernoulli distribution as $O_l \sim \text{Bern}(z_l)$, where z_l is the probability

of the event being displayed. Here we consider z_l as the propensity scores of event l. The IPS method solves the following problem.

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \sum_{l=1}^{\hat{L}} \frac{\ell(y_l, \hat{y}_l)}{z_l} O_l,$$

which, according to the observations of O_l, $\forall l$, is equivalent to

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \sum_{l=1}^{L} \frac{\ell(y_l, \hat{y}_l)}{z_l}. \tag{18}$$

The following proof has been given in past works, which shows that solving (18) can get an unbiased estimator of P.

$$\mathbb{E}_O[\sum_{l=1}^{\hat{L}} \frac{\ell(y_l, \hat{y}_l)}{z_l} O_l] = \sum_{l=1}^{\hat{L}} \frac{\ell(y_l, \hat{y}_l)}{z_l} \cdot \mathbb{E}_O[O_l] = (16).$$

Before solving (18), we need to know z_l for $l = 1, \cdots, L$. The work [27] differentiates two settings for the assignment mechanism of z_l. The first is an experimental setting, where any event z_l is decided by us and O_l is generated according to z_l. Therefore we can directly solve (18) with known z_l. Many past works and data sets (e.g., [17, 28]) are based on this setting. The second is an observational setting, where O_l is not controlled by us, so z_l is implicit. In this case, we must estimate z_l first by some techniques. For example, naive Bayes [27], survival models [33], and more generally, by assigning

$$O_l = \begin{cases} 1 & \text{the event was displayed,} \\ 0 & \text{the event was not displayed,} \end{cases}$$

we can find a relationship $o(\cdot)$ between O_l and the given features such that $o(\boldsymbol{x}^l) \approx O_l$, for $l = 1, \cdots, \hat{L}$ by any machine learning techniques (e.g., logistic regression [27], gradient boosting [22]). Then $z_l \equiv o(\boldsymbol{x}^l)$ for $l = 1, \cdots, L$ can be used as the propensity scores in (18).

3.3 Doubly Robust Method

Doubly robust method for counterfactual learning was first proposed in [4], which takes advantage of the two aforementioned approaches by combining them together. Therefore, the following optimization problem is solved.

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \underbrace{\sum_{l=1}^{L} \left(\frac{\ell(y_l, \hat{y}_l)}{z_l} - \ell(\sigma_l, \hat{y}_l) \right)}_{\text{IPS method part}} + \underbrace{\sum_{l=1}^{\hat{L}} \ell(\sigma_l, \hat{y}_l)}_{\text{direct method part}}. \tag{19}$$

The analysis in [4] shows that the bias of the doubly robust method is the product of the two parts from the direct method and the IPS method. Therefore, the bias of the doubly robust method can be eliminated if either the direct method part or the IPS method part is unbiased. Empirical studies confirm that the doubly robust method performs better than solely a direct method or an IPS method.

3.4 Discussion on Feasibility of Existing Approaches for CPC Systems

Though the estimators obtained by above-mentioned methods are expected to give unbiased predictions of P, we show that there exist three main obstacles for applying these approaches for CTR prediction in a real-world CPC system.

First, from the theoretic guarantee established by past literatures, IPS works when the logging policy is stochastic such that with the increase of L, sufficient events with various propensity scores can be observed. That is, though the policy tends to select the events with high propensity scores, those with low propensity scores still can have a chance to be displayed and included in the labeled set. For events with low propensity scores, the formulation in (18) leads to larger corresponding instance weights, and then IPS can correct the selection bias. However, commercial products such as CPC systems usually avoid a stochastic policy because of the risk of revenue loss. Instead, as we introduced in Section 1, the policy makes decisions by considering if the expected revenue is large enough. Therefore, the logging policy of our CPC system can be considered as under the following deterministic setting.

$$O_l = \mathbb{I}(\text{ECPM}_l > t), \tag{20}$$

where ECPM_l is the expected revenue of the event l, and t is a dynamic threshold. This policy deterministically selects the events with high propensity scores to be displayed. In this case, the events with low propensity scores are absent in the labeled set. The deterministic logging policy makes theory and practice of the IPS method inapplicable to practical CTR prediction.

Secondly, for the direct method and the corresponding part in the doubly robust method, we need an imputation model $\sigma(\cdot)$ to generate labels for non-displayed events. However, as we have shown that $P_U \neq P_L$ in Section 2.2, many existing works [4, 16] find that imputation models learnt from displayed events will propagate the selection bias to the non-displayed events. Then both the direct and the doubly robust methods cannot give unbiased predictions.

The third issue is that for estimators obtained by solving optimization problems like (17) and (19), they involve $O(\hat{L})$ costs of time and space. However, $\hat{L}$ is extremely large considering all displayed and non-displayed events. Therefore, to have a practical framework for large-scale counterfactual CTR prediction, any $O(\hat{L})$ cost of storage and computation should be avoided.

4 A PRACTICAL FRAMEWORK FOR COUNTERFACTUAL CTR PREDICTION

To address issues mentioned in Section 3.4, we propose a practical framework for counterfactual CTR prediction. Our framework extends the doubly robust method by dropping the propensity scores to address the issue of using a deterministic logging policy. We call it as the propensity-free doubly robust method. To further improve the robustness of our framework, we propose a new way to obtain the imputation model by taking the advantage of the small but unbiased set S_t defined in Section 2.2. Finally, by connecting the CTR prediction with recommender systems, we propose an efficient training algorithm so that the prohibitive $O(\hat{L})$ cost can be trickily avoided if some conditions are satisfied.

4.1 Propensity-free Doubly Robust Method

A recent work [16] describes an analogous scenario that many commercial statistical machine translation systems only deterministically provide the best translation result to users and then collect
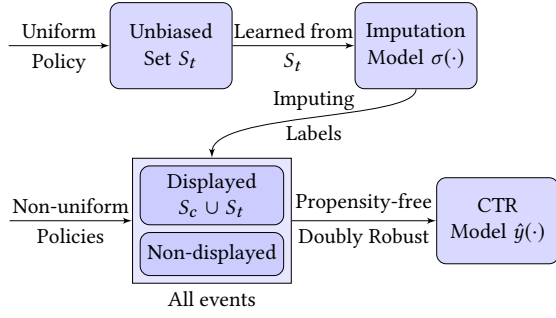
Figure 4: A diagram showing components of our framework

corresponding feedbacks. Then they find that in their scenario, the doubly robust method still can perform well even under a deterministic logging policy. Their analysis provides a reason behind the result that the direct method part of doubly robust imputes labels for non-displayed events, which remedies the absence of unlabeled events and improves the generalization ability of the model. This result motivates us to investigate if the similar idea can be extended on CTR prediction. Specifically, we follow [16] to set $z_l = 1$ for all displayed events logged by a deterministic policy. Then we modify (19) to have the following problem of the propensity-free doubly robust method.

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \sum_{l=1}^{L} \Big(\ell(y_l, \hat{y}_l) - \omega \ell(\sigma_l, \hat{y}_l) \Big) + \omega \sum_{l=1}^{\hat{L}} \ell(\sigma_l, \hat{y}_l), \quad (21)$$

where ω is used as a hyper-parameter to balance the IPS part and the direct method part.

Recall that the concept of the doubly robust method is that the bias can be eliminated if either one of the two sub-estimators is unbiased. Compared with (19), we can see that the IPS part in (21) under the setting of $z_l = 1$ is essentially reduced to the naive binary classification problem (4), which definitely cannot give unbiased predictions. Therefore, we rely on the direct method part to get unbiased predictions. However, if we follow existing works of the doubly robust method (e.g., [4, 16]) to learn $\sigma(\cdot)$ from displayed events, the carried selection bias will be propagated to non-displayed events through the imputation model. In this case, the direct method part can not give unbiased predictions either. In order to avoid this situation, we design a new approach to obtain the imputation model in the next section.

4.2 Unbiased Imputation Models

Recall that in Section 2.2, to help realize the severity of the selection bias, we collect a labeled set S_t through deploying a uniform policy on a small percentage of our online traffic. Though S_t is very small, it still contains some gold-standard unbiased information. We aim to leverage S_t in our framework.

Past studies that have used S_t include, for example, [2, 26]. By considering both S_c and S_t as training sets, they jointly learn two models respectively from the two sets in a multi-task fashion and conduct predictions by combining the two models. Our approach differs from theirs on the usage of S_t. As illustrated in Figure 4, we learn the imputation model $\sigma(\cdot)$ from S_t and apply the model to

impute labels of all displayed and non-displayed events; see the last term in (21).

Note that in our framework, learning the imputation model and learning the CTR model are two separate tasks. Different from the task of learning the CTR model which is generally complex and requires a large training set, the task of learning the imputation model is much more flexible. We can decide the imputation model and the corresponding feature set depending on the size of S_t. That is, when S_t is large enough, the task can be as complex as that of learning the CTR model. But when the size of S_t is limited, the task can be reduced to impute labels with a single value inferred from S_t (e.g., the average CTR in S_t). Through the experimental results shown in Section 5.5, we will see that unbiased imputation models help our framework to obtain good performances even if S_t is very small.

4.3 Efficient Training

By imputing labels for all possible events in the direct method part, (21) becomes a standard classification problem with $\hat{L}$ instances. Because $\hat{L}$ can be extremely large for a real-world CPC system, any cost of $O(\hat{L})$ will make the training process infeasible. Therefore, many existing optimization techniques, if directly applied, are unable to efficiently solve (21). For example, the stochastic gradient (SG) method popularly used for CTR prediction [6, 14] may face difficulties in the process of sampling $O(\hat{L})$ instances.[1] To overcome the difficulties, we connect CPC systems with a related but different field, called recommender systems, where an analogous issue has been well-studied.

In a typical recommender system, we observe part of a rating matrix $Y \in \mathcal{R}^{m \times n}$ of m users and n items. For each observed (i, j) pair, Y_{ij} indicates the rating of item j given by user i. The goal is to predict the rating of an item not rated by a user yet. Following the idea, if we consider an event as a (request, ad) pair, an advertising system is naturally a recommender system. As illustrated in Figure 5, the L displayed events can be considered as L observed (request, ad) entries in a rating matrix $Y \in \mathcal{R}^{m \times n}$, where m is the total number of requests and n is the total number of ads. According to whether the ad was clicked by the user or not, these observed entries can be grouped into the following two sets

$$\Omega^+ = \{(i, j) \mid \text{ad } j \text{ was displayed and clicked in request } i\},$$

$$\Omega^- = \{(i, j) \mid \text{ad } j \text{ was displayed but not clicked in request } i\}.$$

By assigning ratings to observed entries as

$$Y_{ij} = \begin{cases} 1 & (i, j) \in \Omega^+, \\ -1 & (i, j) \in \Omega^-, \end{cases}$$

and replacing the index l with the index (i, j), we can re-write (21) as

$$\begin{aligned} \min_{\mathcal{W}} \quad & \lambda R(\mathcal{W}) + \sum_{(i,j) \in \Omega} \Big(\ell(Y_{ij}, \hat{Y}_{ij}) - \omega \ell(A_{ij}, \hat{Y}_{ij}) \Big) \\ & + \omega \sum_{i=1}^{m} \sum_{j=1}^{n} \ell(A_{ij}, \hat{Y}_{ij}), \end{aligned} \quad (22)$$

---

[1]The failure of SG on handling similar scenarios has been reported in several past works [30, 32].

Figure 5: An illustration of a rating matrix of m requests and n ads, where L events are displayed. The symbols '✓', '✗', and '?' indicate the event is clicked, not clicked, and not displayed, respectively.

where

$$\Omega = \Omega^+ \cup \Omega^-$$

is the set of displayed pairs with $|\Omega| = L$, $\hat{L} = mn$, $\hat{Y} \in \mathcal{R}^{m \times n}$ is the output matrix with $\hat{Y}_{ij} = \hat{y}(\boldsymbol{x}^{i,j})$, and $A \in \mathcal{R}^{m \times n}$ is the matrix of imputed labels with

$$A_{ij} = \sigma(\boldsymbol{x}^{i,j}). \tag{23}$$

Under the above setting, $\hat{L} = mn$ indicates the total number of possible events. The second term in (22) involves $O(mn) = O(\hat{L})$ operations, which are prohibitive when m and n are large. The same $O(mn)$ difficulty occurs in a topic called "recommender systems with implicit feedbacks" [10, 11, 30, 32]. It describes a scenario where a rating matrix Y includes both positive and negative entries. However, only a subset Ω^+ including some positive entries have been observed. An approach to handle this scenario is to treat all unobserved pairs as a negative rating r, such as 0 or -1 in [31, 32]. Let $A \in \mathcal{R}^{m \times n}$ be an artificial rating matrix with

$$A_{ij} = \begin{cases} 1 & (i, j) \in \Omega^+, \\ r & (i, j) \notin \Omega^+. \end{cases} \tag{24}$$

To find a model fitting all A_{ij} values, the cost of $O(mn)$ arises in solving the following problem with mn instances.

$$\min_{\mathcal{W}} \quad \lambda R(\mathcal{W}) + \sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} \ell(A_{ij}, \hat{Y}_{ij}). \tag{25}$$

where C_{ij} is a cost parameter. To avoid the $O(mn)$ cost, existing studies [1, 30–32] impose some resctrictions on terms over values not in Ω^+ and change (25) to

$$\begin{aligned} \min_{\mathcal{W}} \quad & \lambda R(\mathcal{W}) + \sum_{(i,j) \in \Omega^+} C_{ij} \ell(1, \hat{Y}_{ij}) + \sum_{(i,j) \notin \Omega^+} \bar{C}_{ij} \bar{\ell}(r, \hat{Y}_{ij}) \\ = & \lambda R(\mathcal{W}) + \sum_{(i,j) \in \Omega^+} \left(C_{ij} \ell(1, \hat{Y}_{ij}) - \bar{C}_{ij} \bar{\ell}(r, \hat{Y}_{ij}) \right) \\ & + \sum_{i=1}^{m} \sum_{j=1}^{n} \bar{C}_{ij} \bar{\ell}(r, \hat{Y}_{ij}), \end{aligned} \tag{26}$$

where the following conditions should apply.

- The cost parameters $\bar{C}_{ij}$ in (26) cannot be arbitrary values. In practice, a constant value is often considered.
- A simple loss function $\bar{\ell}(\cdot, \cdot)$ is needed. In most existing works the squared loss is considered.

- The model function $\hat{y}(\cdot)$ can not be an arbitrary one. Specifically, given a feature vector $\boldsymbol{x}^{i,j}$, $\hat{y}(\boldsymbol{x}^{i,j})$ can be factorized into multiple parts, each of which is only related to i or j and convex in the corresponding parameters. For example, a modified FFM model proposed in [32] satisfies this condition.

Optimization methods have been proposed so that the $O(mn)$ term in the overall cost can be replaced with a smaller $O(m) + O(n)$ term. The main reason why $O(mn)$ can be eliminated is because results on all i and all j can be obtained by using values solely related to i and j, respectively. A conceptual illustration is in the following equation.

$$\sum_{i=1}^{m} \sum_{j=1}^{n} (\cdots) = \sum_{i=1}^{m} (\cdots) \quad \times \text{ or } + \quad \sum_{j=1}^{n} (\cdots). \tag{27}$$

By considering the similarity between (22) and (26), with the corresponding restrictions in mind, we propose the following final realization of the framework (21):

$$\begin{aligned} \min_{\mathcal{W}} \quad & \lambda R(\mathcal{W}) + \sum_{(i,j) \in \Omega} \left(\ell(Y_{ij}, \hat{Y}_{ij}) - \omega(A_{ij} - \hat{Y}_{ij})^2 \right) \\ & + \omega \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij} - \hat{Y}_{ij})^2. \end{aligned} \tag{28}$$

For (i, j) in Ω, we allow general convex loss functions in the original IPS part to maintain the connection to conventional CTR training, while in the direct method part, we follow the above restriction for $\bar{\ell}$ in (26) to consider the squared loss. We then compare the two $O(mn)$ terms in (26) and (28). By respectively excluding $\bar{C}_{ij}$ and ω, ours considers a more complicated setting of A in (23), where all mn entries can be various rather than only a constant value in (26). There remains a cost of $O(mn)$ caused by A in (28). Inspired by the restriction of $\hat{y}(\cdot)$ that is used to have $\hat{Y}_{ij} = \hat{y}(\boldsymbol{x}^{i,j})$, to extend existing optimization methods for (26) on our proposed framework, an additional condition should be considered:

- The imputation model function $\sigma(\cdot)$ can not be an arbitrary one. Specifically, given a feature vector $\boldsymbol{x}^{i,j}$, $\sigma(\boldsymbol{x}^{i,j})$ can be factorized into multiple parts, each of which is only related to i or j but not required to be convex in parameters. Thus the choice of $\sigma(\cdot)$ is broader than that of $\hat{y}(\cdot)$. For example, the multi-view deep learning approach proposed in [5] satisfies this condition.

In our realization, we implement various imputation models from simple to complex, all of which satisfy the above condition; see details in Section 5.2.2.

To use (28), we consider the FFM model for $\hat{y}(\boldsymbol{x}^{i,j})$ and develop an optimization method. Details are in Section ?? of supplementary materials.

5 EXPERIMENTS

We begin by presenting our experimental settings including data sets and parameter selections. Then, we design and conduct a series of experiments to demonstrate the effectiveness and robustness of our proposed framework in Section 4.

5.1 Data Sets

We consider the following data sets for experiments, where statistics are described in Table 2.
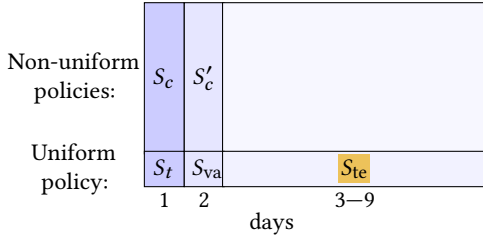
Figure 6: An illustration of data segmentation according to training, validation, and test periods.

Table 2: Data statistics

Data set	m	n	D	$|S_c|$	$|S_t|$	$|S_{va}|$	$|S_{te}|$
Yahoo! R3	14,877	1,000	15,877	176,816	2,631	2,775	48,594
CPC	21.7M	140	1.4M	42.8M	0.35M	0.31M	2.0M

- **CPC**: This is a large-scale data set for CTR prediction, which includes events of nine days from a real-world CPC system. We consider a two-slot scenario so that for each request, the two ads with the highest expected revenues from 140 candidates are displayed. To have training, validation and test sets under uniform and non-uniform policies, as illustrated in Figure 6, we segment the data to six subsets. The five subsets used in our experiments are described below.
  – S_t, S_c defined in Section 2.2, are used separately or together as the training set depending the approach.
  – S_{va} is the set obtained in the validation period under the uniform logging policy. As an unbiased set, it is used by all methods for the parameter selection.
  – S_{te} is the set obtained in the test period under the uniform logging policy. As an unbiased set, it is used to check the test performance of all methods.
  – S_c' is the set of events logged by non-uniform policies in the validation period. For methods that have involved S_c for training in the validation process, after parameters are selected, S_c' is included for training the final model to predict S_{te}.
  For the feature set, we collect user profiles, contextual information (e.g., time, location) of each request and side information of each advertisement. All are categorical features.
- **Yahoo! R3** [20]: The data set, from a music recommendation service, includes a training and a test set of rated (user, song) pairs in a rating matrix R. For the training set, users deliberately selected and rated the songs by preference, which can be considered as a stochastic logging policy by following [27, 29]. For the test set, users were asked to rate 10 songs randomly selected by the system. Because the logging policy of the test set is completely random, the ratings contained in the test set can be considered as the ground truth without the selection bias. In Section ?? of supplementary material, we give details of modifying the problem to simulate a deterministic logging policy, which is what our framework intends to handle.

5.2 Considered Approaches

The considered approaches in our comparison fall into two main categories.

5.2.1 Conventional Approaches for CTR Prediction. This category includes approaches that solve the binary classification problem in (4) with the following CTR model.

- FFM: This is the CTR model reviewed in Section 2.1. We consider the slightly modified formulation proposed in [32].

For the training set, we consider the following three combinations of S_c and S_t.

- S_c: This is a large set including events logged by non-uniform policies; see Section 2.2. This simulates the setting of most systems for CTR prediction.
- S_t: This is a small set including events logged by a uniform policy; see Section 2.2. Because the set is free from the selection bias, some works (e.g., [19]) deploy a model trained on S_t, though an issue is that $|S_t|$ is small.
- $S_c \cup S_t$: This is the set of all logged events.

5.2.2 Counterfactual Learning Approaches. This category includes the following approaches, where all of them consider FFM as the CTR model and $S_c \cup S_t$ as the training set.

- IPS: The approach was reviewed in Section 3.2. For the estimation of propensity scores, we follow [27] to consider the following Naive Bayes propensity estimator.

$$z_l \approx P(O_l = 1 \mid y_l = y) \approx \frac{P(y, O = 1)}{P(y)}, \quad (29)$$

where $y = \{-1, 1\}$ is the label, $P(y, O = 1)$ is the ratio of events labeled as y in the displayed set, and $P(y)$ is the ratio of the events labeled as y in a selection-bias free set. We estimated them by respectively using $S_c \cup S_t$ and S_t.
- CausE: The "causal embedding" approach [2] jointly learns two CTR models $\hat{y}_c(\cdot)$ and $\hat{y}_t(\cdot)$ respectively from S_c and S_t. They solve the following optimization problem.

$$\min_{\mathcal{W}_c, \mathcal{W}_t} \quad \frac{1}{|S_c|} \sum_{(y, \mathbf{x}) \in S_c} \ell(y, \hat{y}_c(\mathbf{x})) + \frac{1}{|S_t|} \sum_{(y, \mathbf{x}) \in S_t} \ell(y, \hat{y}_t(\mathbf{x}))$$
$$+ \lambda_c R(\mathcal{W}_c) + \lambda_t R(\mathcal{W}_t) + \lambda_{\|t-c\|} \|\mathcal{W}_c - \mathcal{W}_t\|_2^2,$$

where $\mathcal{W}_c$ and $\mathcal{W}_t$ are are parameters of $\hat{y}_c(\cdot)$ and $\hat{y}_t(\cdot)$, respectively, and the last term controls the discrepancy between $\mathcal{W}_c$ and $\mathcal{W}_t$. The original study in [2] only supports matrix factorization models. We make an extension to support more complex models (e.g., FFM). For the prediction, we follow [2] to use only $\hat{y}_c(\cdot)$.
- New: This approach is a realization of our proposed framework in (28), where we use the logistic loss defined in (5) as $\ell(\cdot, \cdot)$. We use a modified FFM model in [32] for the CTR model $\hat{y}(\cdot)$. It satisfies the conditions needed in Section 4.3. For the imputation model $\sigma(\cdot)$, we consider three settings.
  – avg: This is a simple setting is to impute labels with the average CTR in S_t. We use the following logit function to transform a probability to a model output in $[-\infty, \infty]$.

$$\sigma(\mathbf{x}^{i,j}) = \log(\frac{\overline{\text{CTR}}}{1 - \overline{\text{CTR}}}), \forall i, j$$

Table 3: **A performance comparison of different approaches. We show the relative improvements over the** average (S_c) **on test NLL and AUC scores. The best approach is bold-faced.**

Metric Data set Approach	NLL	AUC	NLL	AUC
	Yahoo! R3		**CPC**	
average (S_c)	+0.0%	+0.0%	+0.0%	+0.0%
average (S_t)	**+79.1**%	+0.0%	+29.6%	+0.0%
FFM (S_c)	-7.7%	+36.4%	-16.9%	+32.2%
FFM (S_t)	-20.7%	+4.6%	+32.4%	+42.8%
FFM $(S_c \cup S_t)$	+0.2%	+36.4%	+10.6%	+34.6%
IPS	+62.2%	+23.2%	+28.5%	+31.2%
CausE	+6.8%	+37.8%	+25.3%	+39.4%
New (avg)	**+79.1**%	+51.8%	+32.9%	+48.6%
New (item-avg)	+76.8%	**+54.2**%	+32.4%	+48.2%
New (complex)	-0.2%	+37.4%	**+33.4**%	**+50.6**%

where $\overline{\text{CTR}}$ is the average CTR from events in S_t.

– item-avg: This is an item-wise extension of avg as

$$\sigma(\boldsymbol{x}^{i,j}) = \log(\frac{\overline{\text{CTR}}_j}{1 - \overline{\text{CTR}}_j}), \forall i$$

where $\overline{\text{CTR}}_j$ is the average CTR of ad j from events in S_t.

– complex: This is a complex setting to learn $\sigma(\cdot)$ from S_t by solving (4) with the modified FFM model in [32].

We further include two constant predictors as baselines.

- average (S_c): This is a constant predictor defined as

$$\hat{y}(\boldsymbol{x}^{i,j}) = \log(\frac{\overline{\text{CTR}}}{1 - \overline{\text{CTR}}}), \forall i, j$$

where $\overline{\text{CTR}}$ is the average CTR from events in S_c.

- average (S_t): It is the same as average (S_c), except that the average CTR from events in S_t is considered.

For evaluation criteria, we consider NLL and AUC respectively defined in (8) and (9) and report relative improvements over the method average (S_c) as

$$\frac{criterion_{approach} - criterion_{\text{average }(S_c)}}{|criterion_{\text{average }(S_c)}|},$$

where *criterion* is either NLL or AUC, and *approach* is the approach being measured. Because the performance of average (S_c) is unchanged in all experiments, results presented in different tables can be comparable. Details of our parameter selection procedure are in supplementary materials.

5.3 Comparison on Various Approaches

By comparing models in Table 3, we have the following observations.

- For two constant predictors, NLL scores of average (S_t) are much better than those of average (S_c) estimated from the biased S_c set. The reason is that the distribution of S_t is consistent with that of the test set. However, because average (S_t) is still a constant predictor, its AUC scores are the same as those of average (S_c).

Table 4: **The performance when** S_t **becomes smaller sets.** $|S_t|$ **indicates the size of** S_t **as the percentage of total data. We present the relative improvements over the** average (S_c) **on test NLL and AUC scores. The CPC set is considered and the best approach is bold-faced.**

Metric $|S_t|$ Approach	NLL	AUC	NLL	AUC
	0.1%		0.01%	
average (S_t)	+29.2%	+0.0%	+28.7%	+0.0%
FFM (S_t)	+28.6%	+18.6%	+26.5%	+1.2%
FFM $(S_c \cup S_t)$	-12.5%	+29.6%	-16.9%	+32.0%
IPS	+26.1%	+26.8%	+18.7%	+23.6%
CausE	+25.6%	+37.4%	+22.4%	+31.6%
New (avg)	**+33.5**%	**+48.4**%	+30.4%	**+47.2**%
New (item-avg)	+31.5%	+46.6%	+26.7%	+46.0%
New (complex)	+32.2%	+44.6%	**+32.5**%	+43.0%

- For FFM (S_t), it performs much better than FFM (S_c) and FFM $(S_c \cup S_t)$ on **CPC** set. This result confirms our conjecture in Section 2.2 that conventional approaches considering only displayed events selected by non-uniform policies may cause a strong selection bias. However, the result of FFM (S_t) on **Yahoo! R3** set is not ideal. The reason is that the feature set of **Yahoo! R3** is too sparse by having only two identifier features[2] and S_t is too small to learn an FFM model.

- IPS has good NLL scores, but its AUC scores are not competitive. In (29) to estimate propensity scores, the set S_t is used. This may explain why the obtained NLL is similar to that by average (S_t). On the other hand, the unsatisfactory AUC results may be because the Naive Bayes estimator used in (29) is too simple.

- For CausE, it performs better than FFM (S_c) and FFM $(S_c \cup S_t)$ but worse than FFM (S_t) on the **CPC** set. The reason might be that models learnt from S_t are better than those learnt from S_c, but we follow [2] to use only $\hat{y}_c(\cdot)$ for prediction.

- Our proposed framework performs significantly better than others, though an exception is New (complex) on the **Yahoo! R3** set. The reason is analogous to the explanation of the bad result of FFM (S_t) on **Yahoo! R3** set where S_t is too small. In contrast, for the **CPC** set, because S_t is large enough to learn a complex imputation model, New (complex) performs the best.

5.4 Impact of Unbiased Imputation Models Learnt from S_t

In Section 4.2, we proposed unbiased imputation models because past works [4, 16] find that learning from biased data sets will propagate the carried selection bias to non-displayed events and worsens the performance. We conduct an experiment to verify this conjecture and the importance of using an unbiased imputation model. Detailed results are in Section **??** of supplementary materials.

---

[2]See details of the **Yahoo! R3** set in supplementary materials

5.5 Impact of the Size of S_t

To examine the robustness of the approaches relying on S_t to learn an unbiased imputation model, we shrink S_t to see how they perform on the **CPC** set. For experiments in Section 5.3, S_t includes 1% of the data. We now select two subsets respectively having 0.1% and 0.01% of total data. These subsets are then used as S_t for experiments, though each model's parameters must be re-selected. The set S_c remains unchanged.

The result is in Table 4. By comparing the third column of Table 3, which shows the result of using 1% data as S_t, we observe that with the shrinkage of S_t, the performance of all approaches gradually becomes worse. However, our proposed framework still performs significantly better than others. In particular, for other approaches AUC values are strongly affected by the size of S_t, but ours remain similar. Among the three settings of our framework, the simple setting New (avg) of using average CTR in S_t as the imputation model is competitive. This might be because for small S_t, training a complicated imputation model is not easy.

6 CONCLUSION

In this work, through an A/B test on our system, we show that conventional approaches for CTR prediction considering only displayed events may cause a strong selection bias and inaccurate predictions. To alleviate the bias, we need to conduct counterfactual learning by considering non-displayed events. We point out some difficulties for applying counterfactual learning techniques in real-world advertising systems.

(1) For many advertising systems, the thresholding operation (20) is applied as a deterministic logging policy, but existing approaches may require stochastic policies.
(2) Learning imputation models from displayed events causes the selection bias being propagated to non-displayed events.
(3) The high cost of handling all displayed and non-displayed events causes some approaches impractical for large systems.

To overcome these difficulties, we propose a novel framework for counterfactual CTR prediction. We introduce a propensity-free doubly robust method. Then we propose a setting to obtain unbiased imputation models in our framework. For efficient training we link the proposed framework to recommender systems with implicit feedbacks and adapt algorithms developed there. Experiments conducted on real-world data sets show our proposed framework outperforms not only conventional approaches for CTR prediction but also some existing counterfactual learning methods.

REFERENCES

[1] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. 2017. A generic coordinate descent framework for learning from implicit feedback. In *WWW*.
[2] Stephen Bonner and Flavian Vasile. 2018. Causal Embeddings for Recommendation. In *RecSys*.
[3] Olivier Chapelle, Eren Manavoglu, and Romer Rosales. 2015. Simple and scalable response prediction for display advertising. *ACM TIST* 5, 4 (2015), 61:1–61:34.
[4] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly robust policy evaluation and learning. In *ICML*.
[5] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *WWW*.
[6] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *IJCAI*.
[7] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. 2014. Practical Lessons from Predicting Clicks on Ads at Facebook. In *ADKDD*.
[8] James J. Heckman. 1979. Sample selection bias as a specification error. *Econometrica* 47, 1 (1979), 153–161.
[9] Daniel G. Horvitz and Donovan J. Thompson. 1952. A Generalization of Sampling Without Replacement From a Finite Universe. *J. Amer. Statist. Assoc.* 47 (1952), 663–685.
[10] Cho-Jui Hsieh, Nagarajan Natarajan, and Inderjit Dhillon. 2015. PU Learning for Matrix Completion. In *ICML*.
[11] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *ICDM*.
[12] Machael Jahrer, Andreas Töscher, Jeong-Yoon Lee, Jingjing Deng, Hang Zhang, and Jacob Spoelstra. 2012. Ensemble of Collaborative Filtering and Feature Engineered Model for Click Through Rate Prediction. In *KDD Cup 2012 Workshop*.
[13] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. 2017. Field-aware factorization machines in a real-world online advertising system. In *WWW*.
[14] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *RecSys*.
[15] Himabindu Lakkaraju, Jon Kleinberg, Jure Leskovec, Jens Ludwig, and Sendhil Mullainathan. 2017. The selective labels problem: Evaluating algorithmic predictions in the presence of unobservables. In *KDD*.
[16] Carolin Lawrence, Artem Sokolov, and Stefan Riezler. 2017. Counterfactual Learning from Bandit Feedback under Deterministic Logging: A Case Study in Statistical Machine Translation. In *EMNLP*.
[17] Damien Lefortier, Adith Swaminathan, Xiaotao Gu, Thorsten Joachims, and Maarten de Rijke. 2016. Large-scale validation of counterfactual learning methods: A test-bed. In *NIPS Workshop on Inference and Learning of Hypothetical and Counterfactual Interventions in Complex Systems*.
[18] Roderick J. A. Little and Donald B. Rubin. 2002. *Statistical Analysis with Missing Data* (second ed.). Wiley-Interscience.
[19] David C. Liu, Stephanie Rogers, Raymond Shiau, Dmitry Kislyuk, Kevin C. Ma, Zhigang Zhong, Jenny Liu, and Yushi Jing. 2017. Related Pins at Pinterest: The Evolution of a Real-World Recommender System. In *WWW*.
[20] Benjamin M. Marlin and Richard S. Zemel. 2009. Collaborative Prediction and Ranking with Non-random Missing Data. In *RecSys*.
[21] H. Brendan McMahan, Gary Holt, D. Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hrafnkelsson, Tom Boulos, and Jeremy Kubica. 2013. Ad Click Prediction: a View from the Trenches. In *KDD*.
[22] Yusuke Narita, Shota Yasui, and Kohei Yata. 2019. Efficient Counterfactual Learning from Bandit Feedback. In *AAAI*.
[23] Yanru Qu, Bohui Fang, Weinan Zhang, Ruiming Tang, Minzhe Niu, Huifeng Guo, Yong Yu, and Xiuqiang He. 2018. Product-Based Neural Networks for User Response Prediction over Multi-Field Categorical Data. *ACM TOIS* 37 (2018), 5:1–5:35.
[24] Steffen Rendle. 2010. Factorization machines. In *ICDM*.
[25] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ADs. In *WWW*.
[26] Nir Rosenfeld, Yishay Mansour, and Elad Yom-Tov. 2017. Predicting counterfactuals from large historical data and small randomized trials. In *WWW*.
[27] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations As Treatments: Debiasing Learning and Evaluation. In *ICML*.
[28] Adith Swaminathan and Thorsten Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *JMLR* 16 (2015), 1731–1755.
[29] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *RecSys*.
[30] Hsiang-Fu Yu, Mikhail Bilenko, and Chih-Jen Lin. 2017. Selection of Negative Samples for One-class Matrix Factorization. In *SDM*.
[31] Hsiang-Fu Yu, Hsin-Yuan Huang, Inderjit S. Dihillon, and Chih-Jen Lin. 2017. A Unified Algorithm for One-class Structured Matrix Factorization with Side Information. In *AAAI*.
[32] Bowen Yuan, Meng-Yuan Yang, Jui-Yang Hsia, Hong Zhu, Zhirong Liu, Zhenhua Dong, and Chih-Jen Lin. 2019. *One-class Field-aware Factorization Machines for Recommender Systems with Implicit Feedbacks*. Technical Report. National Taiwan University.
[33] Weinan Zhang, Tianxiong Zhou, Jun Wang, and Jian Xu. 2016. Bid-aware gradient descent for unbiased learning with censored data in display advertising. In *KDD*.
[34] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *KDD*.