Unbiased Delayed Feedback Label Correction for Conversion Rate Prediction

Yifan Wang
DCST, BNRist, Tsinghua University
Beijing, China
yf-wang21@mails.tsinghua.edu.cn

Peijie Sun
DCST, BNRist, Tsinghua University
Beijing, China
sun.hfut@gmail.com

Min Zhang*
DCST, BNRist, Tsinghua University
Beijing, China
z-m@tsinghua.edu.cn

Qinglin Jia
Noah's Ark Lab, Huawei
Beijing, China
jiaqinglin2@huawei.com

Jingjie Li
Noah's Ark Lab, Huawei
Beijing, China
lijingjie1@huawei.com

Shaoping Ma
DCST, BNRist, Tsinghua University
Beijing, China
msp@tsinghua.edu.cn

ABSTRACT

Conversion rate prediction is critical to many online applications such as digital display advertising. To capture dynamic data distribution, industrial systems often require retraining models on recent data daily or weekly. However, the delay of conversion behavior usually leads to incorrect labeling, which is called delayed feedback problem. Existing work may fail to introduce the correct information about false negative samples due to data sparsity and dynamic data distribution. To directly introduce the correct feedback label information, we propose an Unbiased delayed feedback Label Correction framework (ULC), which uses an auxiliary model to correct labels for observed negative feedback samples. Firstly, we theoretically prove that the label-corrected loss is an unbiased estimate of the oracle loss using true labels. Then, as there are no ready training data for label correction, counterfactual labeling is used to construct artificial training data. Furthermore, since counterfactual labeling utilizes only partial training data, we design an embedding-based alternative training method to enhance performance. Comparative experiments on both public and private datasets and detailed analyses show that our proposed approach effectively alleviates the delayed feedback problem and consistently outperforms the previous state-of-the-art methods.

CCS CONCEPTS

• **Information systems → Online advertising**; **Recommender systems**.

KEYWORDS

CVR prediction; Delayed Feedback; Feedback Label Correction

ACM Reference Format:
Yifan Wang, Peijie Sun, Min Zhang, Qinglin Jia, Jingjie Li, and Shaoping Ma. 2023. Unbiased Delayed Feedback Label Correction for Conversion Rate Prediction. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3580305.3599536

*Corresponding author

1 INTRODUCTION

Predicting the probability of users clicking or converting on ads or items is critical to many online applications, such as digital display advertising and recommender systems. Take online advertising as an example. Generally, ad delivery platforms provide advertisers with several optional billing models, such as Cost Per thousand iMpressions (CPM), Cost Per Click (CPC) and Cost Per Acquisition (CPA), in which CPA is preferred as the conversion is closer to advertisers' profits. For the CPA model, predicting the click rate and conversion rate of users to the placed advertisements is the key to achieving more revenue, which are also known as the Click-Through Rate (CTR) and Conversion Rate (CVR) prediction tasks. These two tasks have received increasing attention from industry and academia in recent years [5, 30, 32].

Model freshness is important for CTR and CVR prediction models as user interests change dynamically. A common strategy to keep fresh in the industry is to retrain the model daily or weekly on all collected data. This simple strategy can be effective for CTR prediction. However, the delay of conversion behavior makes it challenging to ensure the freshness of CVR models, which is called *Delayed Feedback Problem.* Unlike click behavior happening quickly within minutes of impression, conversions occur much more slowly after days, sometimes taking up to weeks [2]. This leads that the ground truth of recently clicked but unconverted samples is unknown as they may convert in the future.

A vanilla solution is to treat all these unconverted samples as negative feedback, which will cause some positive samples (i.e., real conversions) to be mislabeled, leading to the false negative problem. These mislabeled samples can significantly damage the performance of the CVR prediction model as they are important to the model freshness. Another obvious solution is to wait for a long time until the labels are accurate enough. However, this means that the data is old, which conflicts with the purpose of keeping models fresh. Thus, the delayed feedback problem reflects a trade-off between model freshness and label correctness. Therefore, handling fresh unconverted data with unknown labels is an important challenge for CVR prediction.

Existing methods for the delayed feedback problem can be classified into two types based on the problem setting: online training

[3, 4, 7, 10, 11, 24, 25, 31] and offline training [2, 16, 27, 28]. In the online setting, when new user behaviors are logged, the model is immediately updated on the new data to keep it fresh. In the offline setting, the model is retrained daily or weekly on all collected data to ensure freshness. Industrial systems often choose the appropriate training setting based on their business requirements [4]. The methods under different settings are quite different, and here we only focus on the offline training.

To our knowledge, DFM [2] first studied the delayed feedback problem. They propose to explicitly model delay time with delay distribution assumptions. However, the actual delay may not obey the assumptions, which leads to its suboptimal performance. Recent work [26, 27] in offline learning attempts to construct unbiased estimates of the oracle loss that uses true labels to alleviate the delayed feedback problem. However, although these methods are theoretically unbiased, we argue that they may fail to introduce the correct information about false negative samples, as they do not construct the correct samples corresponding to these mislabeled samples.

In this paper, we aim to address the delayed feedback problem in CVR prediction through label correction. We theoretically prove that if the label of the observed unconverted samples can be corrected to its probability of being a false negative sample, the label-corrected loss will be an unbiased estimate of the oracle loss that uses true labels. Compared to existing unbiased methods for delayed feedback, the advantage of label correction is that it directly complements the information of the correct samples corresponding to the false negative samples. Further, we attempt to train a label correction (LC) model to correct the labels for the observed unconverted samples. If the LC model is accurate enough, the delayed feedback problem can be well addressed.

However, how to train an accurate LC model is non-trivial. Above all, there exist no ready training data for the LC model. We use a counterfactual labeling method to construct artificial training data by setting a counterfactual deadline. Nevertheless, counterfactual labeling utilizes only partial training data. To enhance the performance of the LC model, we further designed an alternative learning method based on embedding transfer. To demonstrate the effectiveness of our method, we conduct extensive experiments on the public and private datasets. Experimental results show that our method can effectively alleviate the delayed feedback problem. Our main contributions can be summarized as follows.

- To the best of our knowledge, it is the first work in the offline training setting to use an unbiased label correction approach to solve the delayed feedback problem of CVR prediction.
- We give a theoretical analysis of unbiased label learning and propose an alternative learning framework for CVR prediction to meet the delayed feedback challenge.
- Comparative experimental results on both public and private datasets demonstrate the effectiveness of our proposed framework.

2 RELATED WORK

2.1 CVR Prediction

The CVR prediction task shares many similarities with the widely studied CTR prediction task. They both predict the probability of a user performing a certain behavior on an ad or an item. Besides, their inputs are generally the same. Generally, the model structure designed for the CTR task can also be applied to CVR prediction. Thus, existing research on CVR prediction focuses more on the differences between CVR and CTR.

There are three main challenges for CVR prediction. First, the data for the CVR task are often more sparse than the CTR task. Existing research mitigates this problem through multi-task learning [9, 14] and pre-training [18]. Second, CVR prediction suffers from selection bias. The CVR prediction model is trained on click samples but infers for all exposure samples during inference. Differences in exposure distribution and click distribution lead to selection bias, which existing work addresses through entire sample space modeling [13, 19, 22, 23], inverse propensity score [1, 29], and doubly robust methods [6, 15]. Third, conversions do not happen as immediately as clicks, with some conversions taking days or even a week. This could result in some positive samples that have not yet converted being incorrectly treated as negative samples. Existing studies address it by delay time modeling [2, 21, 28] or importance sampling [4, 27], which we will detail in Session 2.2. In this work, we focus on the third challenge, the delayed feedback problem, and leave the extension of our method to other problems for future work.

2.2 Delayed Feedback

Here we only focus on the delayed feedback problem in the offline setting.

To our knowledge, the delayed feedback problem was first studied by DFM [2]. DFM models the delay time explicitly. It assumes that the delay time obeys an exponential distribution and then optimizes the maximum likelihood of the currently observed data labels. [28] extends this approach further by using a non-parametric approach to modeling delay time. A drawback of the above methods is that they try to optimize the observed conversion information instead of directly optimizing the true conversion information.

In contrast to explicitly modeling delay time, recent work [26, 27] attempts to address the delay feedback problem by constructing unbiased estimates of the oracle loss that uses true labels. FSIW [27] leverages importance sampling to construct an unbiased loss. Intuitively, it increases the weight of observed positive samples and decreases the weight of potentially negative samples as these samples may be mislabeled. Besides, nnDF [26] assumes that the labels of samples before a time window are accurate and then uses these samples to correct for the biased loss of the whole training data. A drawback of the above methods is that, despite their theoretical guarantee of unbias, they might fail to introduce information about the correct positive sample for each specific false negative sample. For FSIW, it only reduces the weight of the mislabeled samples but cannot introduce the information of the corrected sample, i.e., the weight of the corresponding correct sample is still zero. For nnDF, it does not process recent samples, and therefore cannot introduce information about the correct samples among them. This problem is worse when the data distribution has changed recently. As the information about the false negative samples may differ from the past observed positive samples, only using the observed positive samples cannot complement the correct information about the fresh false negative samples.

Unlike the above state-of-art approaches, we propose to correct the label for each observed negative sample so that the information of the correct sample can be introduced directly. Besides, it can also be proved theoretically that the label-corrected loss is an unbiased estimate of the oracle loss.

3 PRELIMINARIES

3.1 Notations

In online advertising platforms, the user behaviors for the display ads are logged to train the CVR prediction model. Suppose we collect training data $\mathcal{D}$ at timestamp T, i.e., we can obtain all the user behaviors and corresponding features before T. Let $\mathcal{D} = \{(x_i, v_i, e_i, cts_i, cvt_i), i = 1, 2, ...\}$. The notation i denotes the i-th sample. Each sample represents a click record of users. For the i-th sample $(x_i, v_i, e_i, cts_i, cvt_i)$, x_i denotes the feature information of this sample. cts_i denotes the click timestamp. v_i is a binary value that denotes whether the clicked ad has a further conversion before the observed timestamp T. If $v_i = 1$, cvt_i will record the corresponding conversion timestamp. Otherwise, cvt_i is empty. e_i denotes the time elapsed from cts_i to T, i.e., $T - cts_i$.

Let c_i denote whether the i-th sample will finally lead to a conversion. Note that we cannot wait forever for the possible conversion to happen. In practice, a long time window w_a is applied depending on the specific scenario, e.g., one month for Criteo [2]. Only conversions within the time window after clicks are considered valid. In other words, if $e_i \geq w_a$, then $v_i = c_i$. If $e_i < w_a$, c_i is unknown. Thus, c_i is not included in the training data $\mathcal{D}$. For test data, we can wait enough time to obtain c_i for evaluation.

For easy reading, the notations are summarized in Table 1.

Table 1: Notations and Explanations of Variables

Notation	Explanation
T	the training data collection timestamp
x_i	the feature vector of i-th sample
v_i	the observed conversion label of i-th sample
e_i	the elapsed time of i-th sample
c_i	the true conversion label of i-th sample
cts_i	the click timestamp of i-th sample
cvt_i	the conversion timestamp of i-th sample

3.2 Task Formulation

The conversion rate is defined as the probability of the final conversion for a clicked ad, i.e., $p_{CVR} = p(c_i = 1|x_i)$. The CVR prediction task under delayed feedback is aimed to use the training data $\mathcal{D}$ collected at T to predict p_{CVR} for the clicked ads after T.

Note that training samples clicked before $T - w_a$ (i.e., $e_i \geq w_a$) can be fed directly into the model without any processing as their labels are correct. Since the core issue for delayed feedback is how to handle the fresh data with unknown labels, we omit these data in the rest of this paper for simplicity, which does not influence the correctness of our proof and method.

3.3 Vanilla and Oracle Loss

Next, we introduce the two basic loss functions in the delayed feedback problem. Note that CVR prediction is essentially a binary classification problem. Generally, the cross-entropy loss is adopted for training the CVR model. Let $f(\cdot;\theta)$ denote the CVR model with trainable parameters θ. Suppose we can now foresee the future and obtain an ideal dataset $\mathcal{D}^*$, which contains c_i for each sample. Then the cross-entropy loss can be written as:

$$\mathcal{L}_{oracle}(\mathcal{D}^*) = \frac{1}{|\mathcal{D}^*|}\sum_{i=1}^{|\mathcal{D}^*|}\left(c_i \log f(x_i;\theta) + (1-c_i)\log(1-f(x_i;\theta))\right), \tag{1}$$

Equation (1) is called the oracle loss $\mathcal{L}_{oracle}$ as we suppose the final conversion label c_i for each click record is available.

However, in practice, we cannot obtain the oracle label c_i for each sample at the data collection timestamp T. If we ignore the delayed feedback and replace the oracle label c_i with the observed label v_i, we can get the vanilla loss $\mathcal{L}_{vanilla}$ for CVR model training:

$$\mathcal{L}_{vanilla}(\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\left(v_i \log f(x_i;\theta) + (1-v_i)\log(1-f(x_i;\theta))\right). \tag{2}$$

Note that some samples may convert after the data collection timestamp T. Obviously, the vanilla loss will incorrectly treat some positive samples as negative samples, which will damage the performance of CVR prediction model.

4 UNBIASED LABEL CORRECTION FOR DELAYED FEEDBACK PROBLEM

4.1 Overall Framework

We propose an Unbiased delayed feedback Label Correction framework (ULC), which aims to address the delay feedback problem in CVR prediction through label correction. The key idea is that delayed feedback leads to and only leads to incorrect labels. If we are able to identify all the incorrect labels and correct them, we can directly calculate the oracle loss.

Fig.1 illustrates the overall framework of ULC, which consists of a label correction (LC) model and a CVR prediction model. The LC model is designed to predict the probability that an observed unconverted training sample will finally convert, which is used to calculate our proposed label-corrected loss for CVR model training. We prove in Section 4.2 that if the LC model is accurate enough, the label-corrected loss is an unbiased estimate of the oracle loss.

The next question is how to learn an accurate LC model. As there is no ready training data for the LC model, we leverage counterfactual labeling to generate training data. It constructs the artificial data by imagining a counterfactual data collection time $T' < T$, the details of which will be introduced in Section 4.3. However, counterfactual labeling suffers from some problems, such as inadequate utilization of the whole training data, which we analyze in Section 4.4. To mitigate this problem, we further apply alternative training to re-train these two models, enhancing the performance by transferring the underlying representation of the CVR model to the LC model.

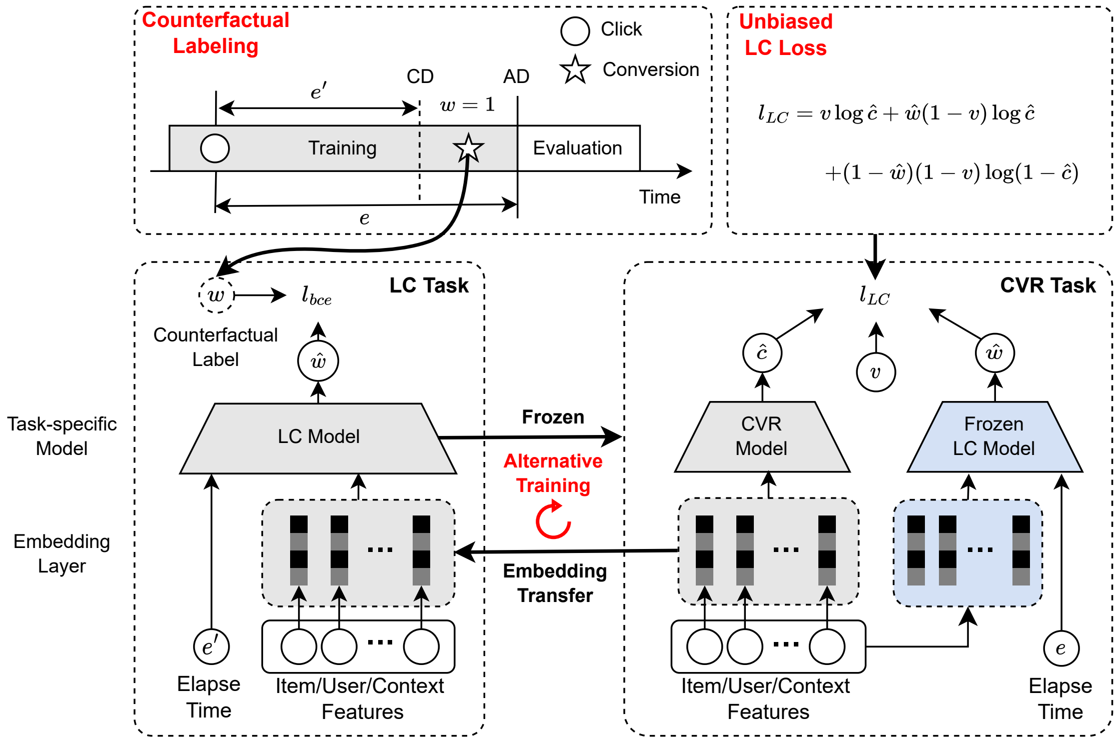

Figure 1: Illustration for our proposed framework ULC.

Next, we elaborate on our framework from unbiased loss, data generation with counterfactual labeling, and alternative training, respectively.

4.2 Unbiased Loss via Label Correction

If we have a label correction model $g(\cdot;\psi)$, which can predict the probability w_i of a observed non-conversion sample i to be a final conversion sample, i.e., $P(c_i = 1|x_i, e_i, v_i = 0)$. Then we can directly correct the sample label for the observed negative samples and get the following label-corrected (LC) loss:

$$\mathcal{L}_{LC}(\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}[v_i \log f(x_i;\theta) + w_i(1-v_i)\log f(x_i;\theta) + (1-w_i)(1-v_i)\log(1-f(x_i;\theta))] \quad (3)$$

The above loss is unbiased to the oracle loss if we have an ideal label correction model, as shown in the following theorem.

Theorem 1. *If an ideal label correction model is satisfied, i.e., $w_i = P(c_i = 1|x_i, e_i, v_i = 0)$, then the LC loss is unbiased to the oracle loss.*

Proof. For simplicity, we denote $\log f(x;\theta)$ as l_θ and $\log(1 - f(x;\theta))$ as $l_{-\theta}$.

$$\begin{aligned}
E[\mathcal{L}_{LC}] &= \iint p(x,e)[p(v=1|x,e)l_\theta + p(c=1|v=0,x,e)p(v=0|x,e)l_\theta \\
&\quad + p(c=0|v=0,x,e)p(v=0|x,e)l_{-\theta})]dedx \\
&= \iint p(x,e)[p(v=1,c=1|x,e)l_\theta + p(c=1,v=0|x,e)l_\theta \\
&\quad + p(c=0,v=0|x,e)l_{-\theta})]dedx \\
&= \iint p(x,e)[p(c=1|x,e)l_\theta + p(c=0|x,e)l_{-\theta})]dedx \\
&= \int p(x)[p(c=1|x)l_\theta + p(c=0|x)l_{-\theta})]dx \\
&= E[\mathcal{L}_{oracle}]
\end{aligned}$$

□

The advantage of LC loss over the previous unbiased loss (e.g., FSIW and nnDF) is that it directly complements the information of the correct sample corresponding to the false negative samples, i.e., $w_i(1-v_i)\log f(x_i;\theta)$. The existing unbiased losses complement the corresponding information in indirect ways, which is strongly influenced by data sparsity and data dynamics. For example, in practice, FSIW complements the correct information by increasing the weights of observed positive samples similar to the false negative sample. However, for some fresh false negative samples, there may not exist similar observed positive samples due to the sparsity and dynamics of CVR data. In this case, these indirect methods cannot effectively supplement the corresponding positive sample information, which is important for the delayed feedback problem as false negative samples are often fresh. In contrast, the LC loss can adequately solve this problem as it directly corrects the label and complements the corresponding correct information.

The remaining problem is how to train an accurate LC model, which we introduce in the next section.

4.3 Data Generation with Counterfactual Labeling

For the LC model, there is no ready training data. Note that we need samples with $v_i = 0$ & $c_i = 1$ as positive samples and with $v_i = 0$ & $c_i = 0$ as negative samples. However, there are only samples with $v_i = 1$ & $c_i = 1$ and samples with $v_i = 0$ in the original data. To train the LC model, we need to construct artificial samples.

We leverage a counterfactual method to generate training data for the LC model. First, we imagine that the training data was collected at a *counterfactual deadline* (CD) before the training data's *actual deadline* (AD), i.e., T. The time interval τ between the CD and the AD is a hyperparameter. Second, the samples that are clicked but have not converted before the CD are collected as training data, together with e' as the elapsed time of these samples at the CD. Third, we treat the samples with conversion between CD and AD as positive samples, i.e., $w = 1$, and others as negative samples, i.e., $w = 0$. Obviously, there exist some samples converting after AD are ignored. Nevertheless, as τ increases, the proportion of these samples keeps getting smaller. The subsequent experiments demonstrate that even a relatively short τ can effectively alleviate the delayed feedback problem.

After data generation, we can train the LC model via the classical binary cross-entropy loss. Then the LC model is frozen and utilized to infer w in the above LC loss. Note that the elapsed time e at the AD is used instead of e' when inferring for LC loss. The detailed data generation procedure is shown in Algorithm 1 (lines 2-14).

4.4 Alternative Training

Although the training data required for the LC model can be constructed by counterfactual labeling, this method still has some drawbacks. First, data generation only leverages partial training data, i.e., samples that are clicked before CD and converted after CD, which may result in the suboptimal performance of LC model. Second, the LC model also suffers somewhat from delayed feedback. Some potential positive samples that have a long delay and convert after AD may be mistreated as negative samples. Next, we propose an alternative learning based approach to alleviate the first problem. The solution to the second problem we leave to future work.

Note that the CVR prediction model is trained on the whole data, and the conversion rate $P(c = 1|x)$ is similar to the label correction rate $P(c = 1|x, e, v = 0)$. We suppose that the bottom representation (i.e., the embedding layer in Fig.1) learned by the CVR model may facilitate the learning of the LC model and alleviate the first problem mentioned above. Therefore, we adopt an alternative learning paradigm. After training the CVR prediction model, the bottom representation of the LC model is initialized using the bottom representation of the CVR prediction model, and then the LC model is retrained. The retrained LC model can be further used for the retraining of the CVR model. The alternative training procedure is shown in Algorithm 1.

Algorithm 1 Alternative training with data generation

```
Input: training data $\mathcal{D} = \{(x_i, v_i, e_i, cts_i, cvt_i)\}$, $T$ is the timestamp
    when the data $\mathcal{D}$ are collected, where $x_i$ is the feature vector,
    $v_i$ is the observed conversion label, $e_i$ is elapsed time since the
    click timestamp $cts_i$, $cvt_i$ is the conversion timestamp. $\tau$ is a
    hyperparameter denoting the time interval between CD and
    AD. $n$ is the rounds of alternative training.
Output: CVR prediction model $M_{cvr}$
 1: Initialize CVR prediction model $M_{cvr}$ and LC model $M_{lc}$
 2: // start to generate data $\mathcal{D}_{lc}$ for LC model training
 3: $\mathcal{D}_{lc} = \emptyset$
 4: for $i = 1$ to $|\mathcal{D}|$ do
 5:     if $cts_i < T - \tau$ and ($v_i == 0$ or $cvt_i > T - \tau$) then
 6:         if $cvt_i > T - \tau$ then
 7:             Label the sample $i$ as $w_i = 1$
 8:         else
 9:             Label the sample $i$ as $w_i = 0$
10:         end if
11:         Insert the sample $(x_i, e_i - \tau, w_i)$ to $\mathcal{D}_{lc}$
12:     end if
13: end for
14: // finish data generation, start to alternative training
15: for $i = 1$ to $n + 1$ do
16:     Initialize $M_{cvr}$
17:     Initialize $M_{lc}$ except bottom embeddings
18:     Training $M_{lc}$ on $\mathcal{D}_{lc}$ until converge
19:     Training $M_{cvr}$ on $\mathcal{D}$ using LC loss until converge
20:     Transfer the bottom embeddings from $M_{cvr}$ to $M_{lc}$
21: end for
22: return CVR prediction model $M_{cvr}$
```

There are some alternatives compared to alternative training with embedding transfer. For example, joint learning is also a common learning paradigm that enables the LC model to leverage the knowledge of the CVR model. Moreover, in addition to utilizing the learned representation of the CVR model, another easily thought of option is to leverage its prediction. It is possible to mine the mislabeled samples in the training data for the LC model using the CVR prediction model as these potential positive samples might have a high predicted CVR. We also conduct experiments and compare these alternatives in experiment section 5.4.

5 EXPERIMENTS

To validate the effectiveness of our proposed method, we conduct a series of experiments to answer the following research questions:

- **RQ1:** How does ULC perform on the CVR prediction task compared to the state-of-the-art methods?
- **RQ2:** How do the label correctness and data freshness of counterfactual labeling affect the performance of ULC?
- **RQ3:** In addition to embedding-based alternative training, how do other common schemes such as joint learning perform?
- **RQ4:** How does the ULC model perform on samples with different delay time?

5.1 Dataset and Settings

5.1.1 Datasets. To our knowledge, there exists only one public dataset [2] widely used in the research of the delayed feedback problem in the offline setting. Other public CVR datasets do not have noticeably delayed feedback or lack enough temporal information. Following the common settings in previous work [2, 27] of using one public and one private dataset, we also introduce a collected private production dataset.

Criteo dataset. This public dataset contains clicks and the corresponding conversions from Criteo live traffic data. Each sample corresponds to a single click and is described by several categorical features and continuous features, with the corresponding conversion information, if any. It also includes the timestamps of the click and the possible conversion behavior. We use this dataset's last 23 days of data to conduct our experiments. Following previous work [2, 27], three consecutive weeks of data are leveraged as training data, data of the 22nd day is used for validation and the last day is for the testing. Note that this dataset tracks conversion behavior for each click sample, so the ground truth c_i is available for testing. For validation, we assume that c_i is unknown and use the label-corrected loss for parameter selection. The processed dataset includes 6,363,085 click samples with a conversion rate of 0.2294.

Production dataset. This dataset is collected from a real production platform with game advertising. In-game payments are treated as conversions. Specifically, we collected and sampled one-month consecutive user feedback logs. The data format is similar to Criteo, and the last two days are used for validation and testing, respectively. The dataset includes over 2,400,000 click samples with a conversion rate of about 0.005. Statistics are shown in Table 2.

Table 2: Statistics of the datasets.

Name	Days	#clicks	CVR	Name	Days	#clicks	CVR
Criteo	23	6,363,085	0.2294	**Production**	30	2,400,000	0.0050

5.1.2 Evaluation Metrics. We adopt three metrics that are widely used in CVR prediction tasks [4, 24]. The first metric is area under ROC curve (**AUC**) that measures the pairwise ranking performance of the CVR prediction model. The second is area under the precision-recall curve (**PRAUC**)[24], which also measures the pairwise ranking performance. The third one is the log loss (**LL**), which measures the accuracy of the absolute value of the CVR prediction.

To further analyze the benefits gained by solving the delayed feedback problem, we calculate the relative improvements (**RI**) to the maximum gain (i.e., the improvement of the oracle model over the vanilla model) on the above three metrics. For method f, the relative improvements on metric $M(\cdot)$ is defined as $\frac{M(f)-M(Vanilla)}{M(Oralce)-M(Vanilla)}$. Then we can obtain **RI-AUC**, **RI-PRAUC** and **RI-LL**.

5.1.3 Compared Methods. The following state-of-the-art methods are our baselines for solving delayed feedback in CVR prediction:

- **Vanilla**: a CVR model trained with the observed conversion label. This is the lower bound of possible improvements.
- **Oracle**: a CVR model trained with the ground truth label instead of observed labels. This is the upper bound of possible improvements.
- **DFM**[2]: a CVR model trained using delayed feedback loss.
- **FSIW**[27]: a CVR model trained using FSIW loss and pre-trained auxiliary models.
- **nnDF**[26]: a CVR model trained using the nnDF loss.
- **ULC(ours)**: a CVR model alternately trained with the LC model and the LC loss.

The above methods can be applied to different CVR models. Due to the column anonymity of the Criteo dataset, we cannot use the models that rely on user modeling. We consider the following classical models that focus on feature interactions as backbones:

- **MLP**: the classical fully connected neural networks.
- **DeepFM** [5]: a model combining the factorization machines and deep neural networks.
- **AutoInt** [17]: a model using multi-head self-attention to learn the high-order feature interactions automatically.
- **DCNV2** [20]: a model using deep and cross networks to learn effective explicit and implicit feature crosses.

5.1.4 Implementation Details. The embedding size is 64 for all the methods. The MLP model in all the backbones is a simple three-layer model with hidden units [256,256,128] and Leaky ReLU activation. For AutoInt, the layer number is 3, the number of heads is 2, and the attention size is 64. For DCNV2, we use the stacked structure and one cross-layer. Adam [8] is used as the optimizer, and the learning rate is tuned in the range of [1e-3, 5e-4, 1e-4] with L2 regularization tuned in [0, 1e-7, 1e-6, 1e-5, 1e-4]. The batch size is set to 1024 for all the methods except nnDF. Given that the nnDF approach cannot apply to batch-wise training, we set the batch size to the size of the whole training set. For a fair comparison, we consistently use the MLP as the auxiliary model for all methods that rely on auxiliary models. The additional hyperparameters for the baselines are fine-tuned. Early stopping is applied to obtain the best parameters. We repeat each experiment 5 times with different random seeds and report the average results and make the statistical tests. [1]

5.2 Overall Performance: RQ1

From Table 3, we can observe that our proposed method ULC outperforms all the baselines and achieves state-of-the-art performance on all the backbones. There are some further observations. First, the oracle model works significantly better than the vanilla model, which validates that the delayed feedback problem indeed hurts the performance of CVR model. Second, FSIW performs significantly better than the DFM and Vanilla models, which is consistent with previous studies [27]. However, nnDF is significantly weaker than Vanilla method. It is because nnDF loss requires global dependence computation and can only be optimized using full training data when updating, which leads to a weaker performance than batch-wise optimization methods. Third, compared to the best baseline, our method shows a significant improvement of 0.76% in the AUC metric, 1.02% in the PRAUC metric, and 1.85% in the LL metric on average across the four backbones, which demonstrates the effectiveness of our proposed method.

We further analyze the benefits gained by solving the delayed feedback problem. As shown in Table 3, our method narrows the

[1]The codes can be found at https://github.com/yfwang2021/ULC. A mindspore version: https://gitee.com/mindspore/models/tree/master/research/recommend/ULC.

Table 3: Performance comparisons of the proposed model with baseline models on the public Criteo dataset. The best results are in boldface, and the best baselines are underlined. The superscripts ** indicate $p \leq 0.01$ for the t-test of ULC vs. the best baseline. ↑ means the higher the better, and ↓ is for the lower the better.

Backbone	Method	AUC ↑	PRAUC ↑	LL ↓	RI-AUC ↑	RI-PRAUC ↑	RI-LL ↑
MLP	Vanilla	0.8208	0.6356	0.4505	0.000	0.000	0.000
	Oracle	0.8441	0.6595	0.4025	1.000	1.000	1.000
	DFM	0.8264	0.6398	0.4378	0.2403	0.1757	0.2645
	FSIW	<u>0.8335</u>	<u>0.6465</u>	<u>0.4178</u>	<u>0.5450</u>	<u>0.4560</u>	<u>0.6812</u>
	nnDF	0.6859	0.4169	0.5969	-5.789	-9.150	-3.05
	ULC(ours)	**0.8403****	**0.6543****	**0.4104****	**0.8369****	**0.7824****	**0.8354****
DeepFM	Vanilla	0.822	0.6379	0.4451	0.000	0.000	0.000
	Oracle	0.8442	0.66	0.4023	1.000	1.000	1.000
	DFM	0.8266	0.6416	0.4319	0.2072	0.1674	0.3084
	FSIW	<u>0.8326</u>	<u>0.6451</u>	<u>0.4195</u>	<u>0.4774</u>	<u>0.3257</u>	<u>0.5981</u>
	nnDF	0.6994	0.4332	0.596	-5.522	-9.262	-3.525
	ULC(ours)	**0.8393****	**0.6525****	**0.4104****	**0.7792****	**0.6606****	**0.8107****
AutoInt	Vanilla	0.8232	0.6383	0.4468	0.000	0.000	0.000
	Oracle	0.8442	0.6601	0.4025	1.000	1.000	1.000
	DFM	0.8276	0.6412	0.4306	0.2095	0.1330	0.3656
	FSIW	<u>0.8329</u>	<u>0.646</u>	<u>0.4192</u>	<u>0.4619</u>	<u>0.3532</u>	<u>0.6230</u>
	nnDF	0.6903	0.4214	0.6123	-6.328	-9.949	-3.735
	ULC(ours)	**0.8388****	**0.6517****	**0.4114****	**0.7428****	**0.6146****	**0.7990****
DCNV2	Vanilla	0.8229	0.6386	0.4454	0.000	0.000	0.000
	Oracle	0.8447	0.6606	0.4018	1.000	1.000	1.000
	DFM	0.8272	0.6417	0.4325	0.1972	0.1409	0.2958
	FSIW	<u>0.8328</u>	<u>0.6461</u>	<u>0.4174</u>	<u>0.4541</u>	<u>0.3409</u>	<u>0.6422</u>
	nnDF	0.6867	0.423	0.6101	-6.247	-9.8	-3.777
	ULC(ours)	**0.8391****	**0.6519****	**0.4105****	**0.7431****	**0.6045****	**0.8004****

gap between Vanilla and Oracle by 77.55% in the AUC metric, 66.55% in the PRAUC metric, and 83.13% in the LL metric on average across the four backbones. Compared to the best baseline, our method shows a significant improvement of 60.3% in the RI-AUC metric, 81.43% in the RI-PRAUC metric, and 27.76% in the RI-LL metric on average across the four backbones. This shows that our method can effectively alleviate the delayed feedback problem.

Fig.2 shows the offline performance on the production dataset. For limited space, we only present the results with MLP as the backbone. It is clearly observed that our proposed method alleviates the delayed feedback problem and outperforms the two best baselines.

To guarantee the reproducibility of our work, and also due to the page limitation, we make the following further detailed analyses on the public-available dataset.

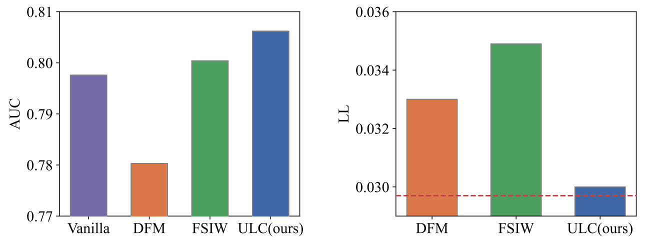

Figure 2: Performance comparisons of the proposed method with the top two baselines on the private dataset. The backbone model is MLP. The red dotted line in the right figure denotes Oracle.

5.3 Analysis on Counterfactual Labeling: RQ2

In counterfactual labeling, only the samples converted between CD and AD are treated as positive samples (i.e., $w = 1$), which leads to some samples converted after AD being mislabeled as negative samples. A long time interval between CD and AD can improve label correctness of counterfactual labeling but reduce data freshness as only clicked data before CD are utilized. We further analyze the effect of different time intervals.

Experimental results using different time intervals are shown in Fig.3. First, increasing the time interval can effectively increase the recall of counterfactual labeling on positive samples. Second, the best τ on the Criteo dataset is around a week. Besides, smaller or larger τ will reduce the performance of CVR model. Smaller τ leads to more mislabeled samples in the training data of LC model, which in turn leads to lower performance of CVR model. Larger τ values, while reducing the mislabeled samples, will make the training data of the LC model older, which leads to its inability to correct well for false negative samples in the CVR training data, since these false negative samples are relatively fresh.

5.4 Effectiveness of Alternative Training: RQ3

In addition to embedding-based alternative training, there are also some other design schemes that enable the LC model to exploit the

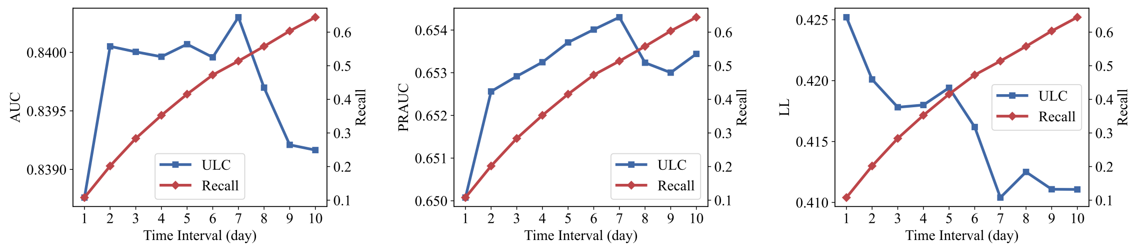

Figure 3: Effect of different time intervals between counterfactual deadline (CD) and actual deadline (AD) on the Criteo dataset with MLP as the backbone. The blue line represents the performance of ULC and the red line represents the recall of counterfactual labeling on positive samples. Larger recall means fewer mislabels in the training data of LC model.

knowledge of CVR prediction model. We conduct experiments on these schemes.

5.4.1 Joint Learning Strategy. An obvious solution is to jointly train the LC model and the CVR model, which also enables the LC model to utilize the information learned by the CVR model. We use a simple shared-bottom structure [12] to validate the effectiveness of this scheme. The joint loss is a linear weighting of the LC model loss and the CVR model loss.

5.4.2 Prediction-based Alternative Training Strategy. In alternative training, in addition to using the learned representation of the CVR model, another easily thought of option is to leverage its prediction. Note that in Section 4.3, we mention that some potential positive samples that have a long delay and convert after AD may be mislabeled as negative samples during counterfactual labeling. To alleviate this problem, we consider using the prediction of the CVR model to mine these potentially positive samples. Intuitively, samples with high predicted CVR are more likely to be potentially positive samples. Thus, we design three simple strategies to process the training data of the LC model: (i) hard strategy, i.e., negative samples ($w = 0$) with predicted CVR above a predefined threshold are treated as positive samples ($w = 1$); (ii) soft strategy, i.e., using predicted CVR as the label for each negative sample; (iii) drop strategy, i.e., dropping negative samples with predicted CVR above a predefined threshold from the training data of the LC model, as the labels of these samples are not reliable.

5.4.3 Comparisons on Different Strategies. The results of the above strategies on Criteo with MLP as backbone are shown in Fig.4. Results on AUC are similar to PRAUC and hence omitted. We have the following observations: (i) using a simple joint learning scheme cannot improve performance. Instead, there is a large loss of performance. The reason is that the inaccurate LC model at the early training stage will mislead the CVR model, which in turn affects the subsequent training. (ii) the three prediction-based strategies cannot improve the performance of the CVR model and even cause a slight degradation. The potential positive samples after AD have a higher delay than τ, and the number of these samples is very small compared to the number of true negative samples (about 1:50 ratio). Using only predicted CVR cannot effectively discover these samples; instead, it introduces noise.

5.4.4 Sensitivity of Alternative Training Rounds. We further analyze the effect of alternative training rounds on the performance of ULC. n controls the rounds of alternative training, and $n = 0$ means no alternative training. We conduct experiments using different values of n on the Criteo dataset with MLP as the backbone. As shown in Fig.5, alternative training once can significantly improve the performance of the CVR prediction model, which validates the effectiveness of alternative training. Besides, one round is enough, and more rounds have little impact, which is reasonable since the first round that changes the initialization of the LC model from random to the embeddings of CVR model brings more significant changes than the subsequent rounds. Note that even without alternative learning, the performance of ULC is still significantly better than the best baseline, which reflects the effectiveness of using

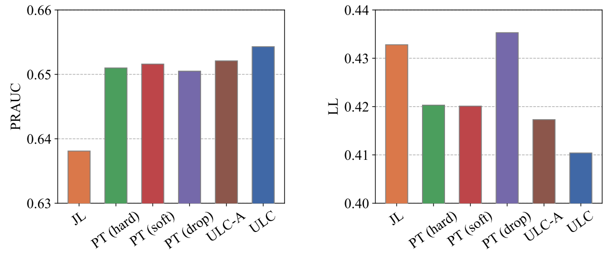

Figure 4: Comparisons on different training strategies. Dataset: Criteo. Backbone: MLP. "ULC-A": Proposed ULC without alternative training. "JL": Joint learning. "PT": Prediction-based alternative training.

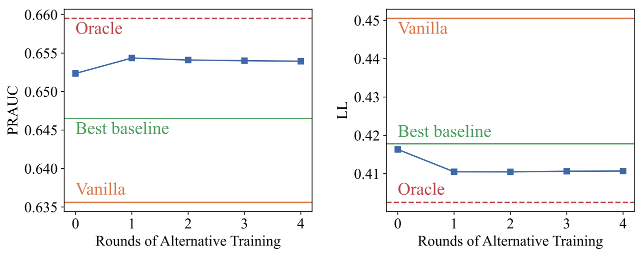

Figure 5: Performance w.r.t. the alternative training rounds. Dataset: Criteo. Backbone: MLP. Note that 0 on the x axis means no alternative training.

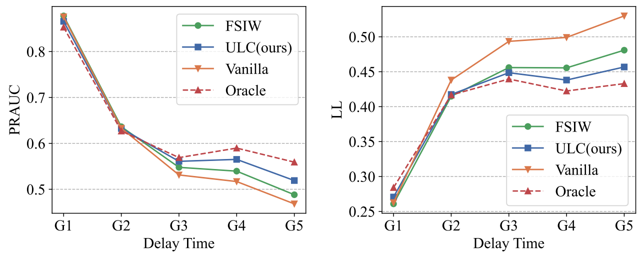

Figure 6: Performance w.r.t. test samples with different delay time on the Criteo dataset with MLP as the backbone. A larger value of the x axis means a longer delay time. Results on AUC are similar to PRAUC and hence omitted.

the LC model for label correction. Results on AUC are similar to PRAUC and hence omitted.

5.5 Analysis on Different Delay Time: RQ4

The delay time is an important property of delayed feedback. We further analyze the performance of CVR model and LC model on samples with different delay time.

5.5.1 CVR performance on different delay time. For a fair comparison, we divide the positive samples in the test set into five groups in ascending order based on their delay time. Each group has the same number of positive samples. Then, each group is combined with all the negative samples in the test set to form test sets with different delay times. In this way, the number of positive and negative samples is the same for different test sets. Further, since the log loss is sensitive to the conversion rate, to ensure that the conversion rate in the test set is consistent with the original test set, we duplicate five copies of each positive sample.

Experiment results on the Criteo dataset with MLP as backbone are shown in Fig.6. We have the following observations: (i) for the Oracle model without the delayed feedback problem, its performance decreases somewhat as the sample delay time increases, which indicates that samples with a long delay time are more likely to be hard samples. (ii) as the sample delay time increases, the Oracle model performs increasingly better than Vanilla, which is because positive samples are more likely to be false negative samples as the delay time increases. (iii) our method significantly outperforms the Vanilla model and the best baseline on samples with high delays (e.g., G3, G4, and G5), and our boost increases as the delay time increases, which reflects the effectiveness of our method.

An interesting phenomenon is that the Vanilla model performs better than the Oracle model on samples with short delays (G1). It may be because samples with short delays have a higher percentage of observed positive samples than actual positive samples. Further analysis can be found in Appendix.

5.5.2 LC performance on different delay time. We further analyze the performance of the LC model on samples with different delay time. Similarly, we divide the false negative samples in the training data into five groups in ascending order based on their delay time. Each group has the same number of false positive samples. Then, as the goal of the LC model is to distinguish between false negative samples and true negative samples, each group is combined with all the true negative samples in the training data to form evaluation data with different delay time.

Table 4: Label correction performance of ULC w.r.t. samples with different delay time. G5 is the group with the longest delay, and G1 has the shortest delay. ↑ means the higher the better, and ↓ is for the lower the better.

Metrics	G1	G2	G3	G4	G5
AUC ↑	0.8698	0.8350	0.8117	0.7896	0.7811
PRAUC ↑	0.1397	0.0757	0.0545	0.0434	0.0398
LL ↓	0.0549	0.0584	0.0604	0.0621	0.0628

Experiment results on the Criteo dataset are shown in Table 4. We have the following observations: (i) AUC ranges from 0.7811 to 0.8698 at different delay time, which reflects that the LC model can effectively recognize false negative samples from all negative samples. (ii) the performance of the LC model decreases as the delay time of the false negative samples increases. There are two possible reasons for this. First, samples with longer delays are more likely to be hard samples. Second, in counterfactual labeling, false negative samples with longer delays are more likely to convert after AD and be recognized as true negative examples, which damages the performance of LC model.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a framework ULC to address the delayed feedback problem in the offline setting via unbiased label correction. The key idea is that delayed feedback leads to and only leads to incorrect labels. If the incorrect labels can be effectively corrected, the delayed feedback problem can be well addressed. ULC uses an additional LC model to guide the CVR prediction model for unbiased label correction and enhances the performance through alternative training. We prove theoretically that the label-corrected loss in our method is an unbiased estimate of the oracle loss. Comparative experiments on both public and private datasets and detailed analyses show that ULC effectively alleviates the delayed feedback problem and consistently outperforms the previous state-of-the-art methods.

For future work, we are interested in the following points. First, using multiple and dynamic counterfactual deadlines is likely to exploit training data more effectively. Second, given that samples with a long delay time are more likely to be hard samples, we would like to design approaches to enhance the model performance on long-delay samples. Third, we are interested in the combination of our method to selection bias in CVR prediction.

ACKNOWLEDGMENTS

This work is supported by the Natural Science Foundation of China (Grant No.U21B2026), the fellowship of China Postdoctoral Science Foundation (No.2022TQ0178) and Huawei (Huawei Innovation Research Program). We also thank MindSpore for the partial support of this work, which is a new deep learning computing framework.

REFERENCES

[1] Wentian Bao, Hong Wen, Sha Li, Xiao-Yang Liu, Quan Lin, and Keping Yang. 2020. GMCM: Graph-Based Micro-Behavior Conversion Model for Post-Click Conversion Rate Estimation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 2201–2210. https://doi.org/10.1145/3397271.3401425

[2] Olivier Chapelle. 2014. Modeling Delayed Feedback in Display Advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, New York, USA) *(KDD '14)*. Association for Computing Machinery, New York, NY, USA, 1097–1105. https://doi.org/10.1145/2623330.2623634

[3] Yu Chen, Jiaqi Jin, Hui Zhao, Pengjie Wang, Guojun Liu, Jian Xu, and Bo Zheng. 2022. Asymptotically Unbiased Estimation for Delayed Feedback Modeling via Label Correction. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 369–379. https://doi.org/10.1145/3485447.3511965

[4] Siyu Gu, Xiang-Rong Sheng, Ying Fan, Guorui Zhou, and Xiaoqiang Zhu. 2021. Real Negatives Matter: Continuous Training with Real Negatives for Delayed Feedback Modeling. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 2890–2898. https://doi.org/10.1145/3447548.3467086

[5] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne, Australia) *(IJCAI'17)*. AAAI Press, 1725–1731.

[6] Siyuan Guo, Lixin Zou, Yiding Liu, Wenwen Ye, Suqi Cheng, Shuaiqiang Wang, Hechang Chen, Dawei Yin, and Yi Chang. 2021. Enhanced Doubly Robust Learning for Debiasing Post-Click Conversion Rate Estimation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 275–284. https://doi.org/10.1145/3404835.3462917

[7] Yuyao Guo, Haoming Li, Xiang Ao, Min Lu, Dapeng Liu, Lei Xiao, Jie Jiang, and Qing He. 2022. Calibrated Conversion Rate Prediction via Knowledge Distillation under Delayed Feedback in Online Advertising. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 3983–3987. https://doi.org/10.1145/3511808.3557557

[8] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6980

[9] Shunsuke Kitada, Hitoshi Iyatomi, and Yoshifumi Seki. 2019. Conversion Prediction Using Multi-Task Conditional Attention Networks to Support the Creation of Effective Ad Creatives. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2069–2077. https://doi.org/10.1145/3292500.3330789

[10] Sofia Ira Ktena, Alykhan Tejani, Lucas Theis, Pranay Kumar Myana, Deepak Dilipkumar, Ferenc Huszár, Steven Yoo, and Wenzhe Shi. 2019. Addressing Delayed Feedback for Continuous Training with Neural Networks in CTR Prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems* (Copenhagen, Denmark) *(RecSys '19)*. Association for Computing Machinery, New York, NY, USA, 187–195. https://doi.org/10.1145/3298689.3347002

[11] Haoming Li, Feiyang Pan, Xiang Ao, Zhao Yang, Min Lu, Junwei Pan, Dapeng Liu, Lei Xiao, and Qing He. 2021. Follow the Prophet: Accurate Online Conversion Rate Prediction in the Face of Delayed Feedback. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 1915–1919. https://doi.org/10.1145/3404835.3463045

[12] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. 2018. Modeling Task Relationships in Multi-Task Learning with Multi-Gate Mixture-of-Experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1930–1939. https://doi.org/10.1145/3219819.3220007

[13] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 1137–1140. https://doi.org/10.1145/3209978.3210104

[14] Junwei Pan, Yizhi Mao, Alfonso Lobos Ruiz, Yu Sun, and Aaron Flores. 2019. Predicting Different Types of Conversions with Multi-Task Learning in Online Advertising. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2689–2697. https://doi.org/10.1145/3292500.3330783

[15] Yuta Saito. 2020. Doubly Robust Estimator for Ranking Metrics with Post-Click Conversions. In *Proceedings of the 14th ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 92–100. https://doi.org/10.1145/3383313.3412262

[16] Yuta Saito, Gota Morisihta, and Shota Yasui. 2020. Dual Learning Algorithm for Delayed Conversions. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 1849–1852. https://doi.org/10.1145/3397271.3401282

[17] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 1161–1170. https://doi.org/10.1145/3357384.3357925

[18] Yumin Su, Liang Zhang, Quanyu Dai, Bo Zhang, Jinyao Yan, Dan Wang, Yongjun Bao, Sulong Xu, Yang He, and Weipeng Yan. 2021. An Attention-Based Model for Conversion Rate Prediction with Delayed Feedback via Post-Click Calibration. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence* (Yokohama, Yokohama, Japan) *(IJCAI'20)*. Article 487, 7 pages.

[19] Hao Wang, Tai-Wei Chang, Tianqiao Liu, Jianmin Huang, Zhichao Chen, Chao Yu, Ruopeng Li, and Wei Chu. 2022. ESCM2: Entire Space Counterfactual Multi-Task Model for Post-Click Conversion Rate Estimation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 363–372. https://doi.org/10.1145/3477495.3531972

[20] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-Scale Learning to Rank Systems. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 1785–1797. https://doi.org/10.1145/3442381.3450078

[21] Yanshi Wang, Jie Zhang, Qing Da, and Anxiang Zeng. 2020. Delayed Feedback Modeling for the Entire Space Conversion Rate Prediction. *CoRR* abs/2011.11826 (2020). arXiv:2011.11826 https://arxiv.org/abs/2011.11826

[22] Hong Wen, Jing Zhang, Fuyu Lv, Wentian Bao, Tianyi Wang, and Zulong Chen. 2021. Hierarchically Modeling Micro and Macro Behaviors via Multi-Task Learning for Conversion Rate Prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 2187–2191. https://doi.org/10.1145/3404835.3463053

[23] Hong Wen, Jing Zhang, Yuan Wang, Fuyu Lv, Wentian Bao, Quan Lin, and Keping Yang. 2020. Entire Space Multi-Task Modeling via Post-Click Behavior Decomposition for Conversion Rate Prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 2377–2386. https://doi.org/10.1145/3397271.3401443

[24] Jia-Qi Yang, Xiang Li, Shuguang Han, Tao Zhuang, De-Chuan Zhan, Xiaoyi Zeng, and Bin Tong. 2021. Capturing Delayed Feedback in Conversion Rate Prediction via Elapsed-Time Sampling. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021.* AAAI Press, 4582–4589. https://ojs.aaai.org/index.php/AAAI/article/view/16587

[25] Jia-Qi Yang and De-Chuan Zhan. 2022. Generalized Delayed Feedback Model with Post-Click Information in Recommender Systems. In *NeurIPS*. http://papers.nips.cc/paper_files/paper/2022/hash/a7f90da65dd41d699d00e95700e6fa1e-Abstract-Conference.html

[26] Shota Yasui and Masahiro Kato. 2022. Learning Classifiers under Delayed Feedback with a Time Window Assumption. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 2286–2295. https://doi.org/10.1145/3534678.3539372

[27] Shota Yasui, Gota Morishita, Fujita Komei, and Masashi Shibata. 2020. A Feedback Shift Correction in Predicting Conversion Rates under Delayed Feedback. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 2740–2746. https://doi.org/10.1145/3366423.3380032

[28] Yuya Yoshikawa and Yusaku Imai. 2018. A Nonparametric Delayed Feedback Model for Conversion Rate Prediction. *CoRR* abs/1802.00255 (2018). arXiv:1802.00255 http://arxiv.org/abs/1802.00255

[29] Wenhao Zhang, Wentian Bao, Xiao-Yang Liu, Keping Yang, Quan Lin, Hong Wen, and Ramin Ramezani. 2020. Large-Scale Causal Approaches to Debiasing Post-Click Conversion Rate Estimation with Multi-Task Learning. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 2775–2781. https://doi.org/10.1145/3366423.3380037

[30] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021. Deep Learning for Click-Through Rate Estimation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, Zhi-Hua Zhou (Ed.). ijcai.org, 4695–4703. https://doi.org/10.24963/ijcai.2021/636

[31] Xiao Zhang, Haonan Jia, Hanjing Su, Wenhan Wang, Jun Xu, and Ji-Rong Wen. 2021. Counterfactual Reward Modification for Streaming Recommendation with Delayed Feedback. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 41–50. https://doi.org/10.1145/3404835.3462892

[32] 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 *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1059–1068. https://doi.org/10.1145/3219819.3219823

A APPENDIX

A.1 Further Analysis on Different Delay Time

An interesting phenomenon in Figure.6 is that the Vanilla model performs better than the Oracle model on samples with short delays (G1). It may be because samples with short delays have a higher percentage of observed positive samples than actual positive samples. Specifically, the proportion of G1 to the observed positive samples is 25.8%, which is higher than that of G1 to the true positive samples, 21.4%. Thus, the Vanilla model focuses more on these short-delay samples and learns better about them. We analyze the training samples with different delay time in the same way as above. As shown in Fig.7, the Vanilla model indeed learns better on the short-delay samples of training data than the Oracle model. Moreover, we can also observe that the samples with a longer delay time on the training set have a larger training loss for the Oracle model. As the number of true positive samples is the same for these groups, this also indicates that samples with longer delays are more likely to be hard samples.

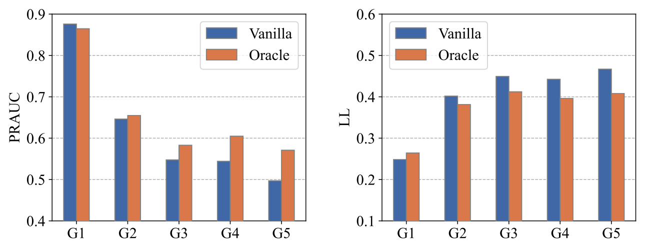

Figure 7: Performance w.r.t. training samples with different delay time on the Criteo dataset with MLP as the backbone. A larger value of the x axis means a longer delay time. Results on AUC are similar and hence omitted.

A.2 Case Study

Here are two concrete cases on the Criteo dataset with MLP as the backbone. For each test sample, we find the ten most similar samples in the training set, as the label correctness of these training samples might have a strong impact on the prediction of the test sample. We calculate similarity using the L2 distance of sample embeddings in our model. As shown in Table 5 and 6, it can be found that our method effectively corrects the labels of false negative samples without wrongly categorizing true negative samples as positive ones. Consequently, the prediction accuracy of the test sample is enhanced.

Table 5: The first concrete case on the Criteo dataset with MLP as the backbone. The false negatives are in boldface. Our method effectively corrects the labels of false negative samples (5063626, 5630493). Meanwhile, the true negative samples (1738500, 904961, 3478664) are not corrected to positive samples.

Test Sample ID	Label	Vanilla Pred.	ULC(ours) Pred.
186833	1	0.081	0.787
Ten Most Similar Training Samples	**True Label**	**Observed Label**	**Corrected Label**
5063626	**1**	**0**	**0.906**
4136975	1	1	1
3726489	1	1	1
1738500	0	0	0.000
5630493	**1**	**0**	**0.821**
904961	0	0	0.023
1980991	1	1	1
3424953	1	1	1
3478664	0	0	0.044
5645863	1	1	1

Table 6: The second concrete case on the Criteo dataset with MLP as the backbone. The false negatives are in boldface. Our method effectively corrects the labels of false negative samples (5782400, 5796674, 5801703). Meanwhile, the true negative samples (remaining part) are not corrected to positive samples.

Test Sample ID	Label	Vanilla Pred.	ULC(ours) Pred.
146531	1	0.061	0.790
Ten Most Similar Training Samples	**True Label**	**Observed Label**	**Corrected Label**
5782400	**1**	**0**	**0.538**
5796674	**1**	**0**	**0.677**
5801703	**1**	**0**	**0.616**
2648493	0	0	0.002
4545673	0	0	0.012
686762	0	0	0.000
214715	0	0	0.000
5184723	0	0	0.007
602570	0	0	0.000
5382003	0	0	0.013