A Feedback Shift Correction in Predicting Conversion Rates under Delayed Feedback

Shota Yasui*
yasui_shota@cyberagent.co.jp
Cyberagent, Inc.
Shibuya, Tokyo

Gota Morishita*
morishit_gota@cyberagent.co.jp
Cyberagent, Inc.
Shibuya, Tokyo

Komei Fujita
fujita_komei@cyberagent.co.jp
Cyberagent, Inc.
Shibuya, Tokyo

Masashi Shibata
shibata_masashi@cyberagent.co.jp
Cyberagent, Inc.
Shibuya, Tokyo

ABSTRACT

In display advertising, predicting the conversion rate, that is, the probability that a user takes a predefined action on an advertiser's website, such as purchasing goods is fundamental in estimating the value of displaying the advertisement. However, there is a relatively long time delay between a click and its resultant conversion. Because of the delayed feedback, some positive instances at the training period are labeled as negative because some conversions have not yet occurred when training data are gathered. As a result, the conditional label distributions differ between the training data and the production environment. This situation is referred to as a *feedback shift*. We address this problem by using an importance weight approach typically used for covariate shift correction. We prove its consistency for the feedback shift. Results in both offline and online experiments show that our proposed method outperforms the existing method.

CCS CONCEPTS

• **Information systems** → **Display advertising**; • **Computing methodologies** → **Learning under covariate shift**.

KEYWORDS

Delayed Feedback, Conversion Prediction, Importance Weight

1 INTRODUCTION

Over the last decade, programmatic advertisement (ad) buying through real-time auction has become common in performance

*Both authors contributed equally to this research.

display advertising. Advertisers have been offered several payment options, such as paying per impression (CPM), paying per click (CPC), and paying per conversion (CPA). The CPA option is preferred by advertisers because they would rather pay for a conversion which is more likely to lead to profits. Thus, we focus on a CPA model in which advertisers pay only if a user performs a predefined action on their website after clicking on the advertisement. In this payment model, accurately predicting a conversion rate (CVR) is essential in estimating the value of an ad impression. However, there is a delay between a click and its resultant conversion. It takes some time for a conversion to occur following a click. In the production environment, the training data are collected right before training a model. Therefore, some conversions are not observed yet for samples observed near the training timing. This leads to mislabeling some samples in training data. Consequently, there is a discrepancy between the conditional label distribution of the training data and that of the test data because the test data are tracked for a sufficiently long period to ensure accurate labeling. It is possible to wait for a fixed time window before assigning a label to ad clicks and then train on the data. However, as discussed in [2], a shorter window increases the likelihood of positive training samples being mislabeled as negative. Training samples are, at the least, as old as a window length; thus a longer window tends to generate a stalled model because of a large shift over time resulting from factors, such as seasonality and changes to ad campaigns.

Work has already been completed regarding the delayed feedback issue. To the best of our knowledge, this issue was first addressed in [2]. [2] assumes that the delay distribution is exponential and proposes two models: one predicts the CVR and the other predicts the delay in conversion. Both models are jointly trained via the expectation-maximization (EM) algorithm or the gradient descent optimization. [25] extends this approach and proposes use of a non-parametric model for delay distribution estimation.

Whereas the previous studies focus on the CVR prediction task, [12] examines the click-through-rate (CTR) prediction for a video ad. On their platform, there is a severe feature distribution shift that necessitates training a model online on fresh data, giving rise to the delayed feedback problem.

In this study, we regard the delayed feedback as a data shift in which there is a disparity between the label distribution in the training and test data although the feature distributions remain the same. We term this situation a *feedback shift*. This concept is closely

related to the covariate shift, wherein there is a disparity between the feature distribution in the training and test data whereas the conditional label distributions remain the same[18]. Similar to the covariate shift, the feedback shift also results in an inconsistent empirical loss function that degrades the performance of a model. To address the feedback shift, we propose the importance weighting (IW) approach, which is a well-known solution to the covariate shift [18]. As with [20], the importance weighted estimate of the loss under *feedback shift* is also consistent, as is shown subsequently. However, because the IW requires the test data distribution which is unavailable, we have to estimate the IW, and then train a CVR model using the estimated IW.

We conducted two offline and one online experiments. For offline experiments, we used different datasets: a public dataset, the conversion logs dataset provided by Criteo[1] and an in-house dataset provided by Dynalyst[2]. Using the public data, we conducted the first experiment to demonstrate the effectiveness of our proposed method. In the second experiment, we used the offline in-house data, and incorporated the IW approach into the field-aware factorization machines (FFM) [10]; subsequently we evaluated the derived method, hereafter referred to as FFMIW, on the offline in-house data, and demonstrated its superiority to the FFM under a specific circumstance. Finally, based on the offline result, we decided to conduct an online A/B test to confirm the effectiveness of our proposed method in the production system.

2 RELATED WORK

Some works on CVR prediction ignore the delayed feedback problem[1, 3, 9, 14, 16]. Following the publication of [2], the delayed feedback has been a focus of constant attention. Some recent studies have attempted to solve it [12, 17, 22, 25]. Furthermore, [2] states that the delayed feedback problem is related to positive-unlabeled(PU) Learning.

We considered the delayed feedback problem as a feedback shift. The feedback shift can be addressed using an IW approach, similar to the covariate shift correction. Covariate shift, also known as sample selection bias, has been extensively studied [5, 7, 18, 21, 26]. Another similar concept is termed the label shift, wherein there is a disparity between test and training label distributions ,which is a general case of feedback shift[13, 19, 27].

The delayed feedback in the bandit algorithm has been researched[8, 15, 23, 24]. Whereas the objective in the bandit problem is to sequentially make decisions in order to minimize the cumulative regret, our goal is to predict the CVR in order to derive a bid price in ad auction. It is also necessary to consider the delay of the reward feedback in the bandit problem. [8, 23, 24] updates the reward estimator just at the moment that the reward is observed.

3 PRE-ANALYSIS

We analyze the Criteo and Dynalyst datasets to show that the delay exists in reality.

First, we calculated the CVR at different time intervals in the Criteo dataset. Figure 1 shows that there is a delay between a click and its resultant conversion. While 30% of the conversions occur

---

[1]https://labs.criteo.com/2013/12/conversion-logs-dataset/
[2]http://www.dynalyst.io/

less than 1 hour after the click, one-half of them occur after one day. Additionally, 13% of them occur after two weeks. These long delays imply that a large portion of the samples in the training data are incorrectly labeled; thus, the label distributions in the training and test data are different.

As discussed in Section 1, one of the ways to circumvent mislabeling arising from the delay is to wait for a fixed time window before assigning a label to the samples. However, as shown in Figure 1, to correctly label approximately 90% of samples, at least two weeks is required. If the fixed time window is two weeks, samples that are collected less than two weeks before would be unlabeled. Therefore, they would not be used to train the model to predict the CVR. However, this causes a data shift between the training and test data because of the frequent changes to ad campaigns. For example, 11.3% of the traffic came from new campaigns [2]. In addition to the changes to ad campaigns, the arrival of new products and special events affect the CVR. For example, in the Dynalyst dataset, an upward trend is observed during a specific period in Figure 2. Therefore, merely waiting for a fixed time window is not an optimal way to eliminate the effect of delayed feedback in CVR prediction. Although we can avoid the data shift by using fresh data, the fresh data are likely to be mislabeled due to the delayed feedback as discussed in Introduction. It is key to use fresh data and eliminate the effect of the delay.
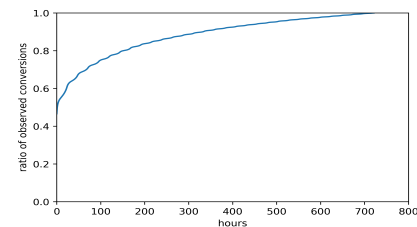

Figure 1: Criteo Dataset: Cumulative distribution of the delay between the click and its conversion.
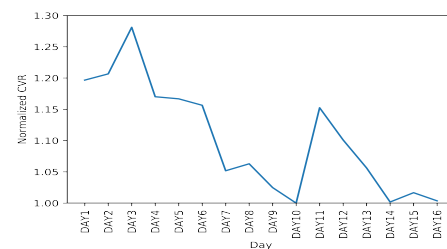
Figure 2: Dynalyst Dataset: Normalized observed CVR.

Secondly, we calculated the empirical probability density functions of the delay in the Criteo and Dynalyst datasets. There is a 24-hour cyclicality in the Criteo dataset because people browse and surf the internet more at a certain times of the day[2]. The cyclicality patterns of different campaigns are varied, as shown in Figure 3, 4. [2] ignores the periodicity and assumes the delay follows the

exponential distributions. However, our proposed method using the IW can capture the oscillating shapes of a delay distribution if a flexible model is used to estimate the IW. This leads to improved CVR prediction.
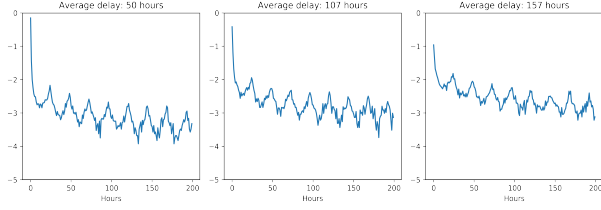

Figure 3: Criteo Dataset: Probability density functions of the delays between clicks and conversions for three different campaigns chosen among many campaigns that have different average delays.
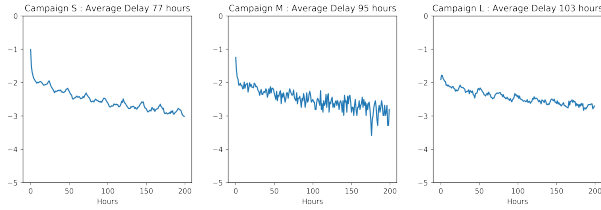

Figure 4: Dyanlyst dataset: Probability density functions of the delays between clicks and conversions for three selected campaigns with different average delays.

4 DELAYED FEEDBACK

The structure of the delayed feedback in the CVR prediction is described in this section. We show that we can see this problem as a feedback shift where the training and test conditional label distributions are different.

4.1 Delayed Feedback Formulation

We define some random variables as follows.

- X: $\mathcal{X}$-valued random variable of features;
- Y: $\{0, 1\}$-valued random variable indicating whether a conversion occurs during the training term if $Y = 1$;
- C: $\{0, 1\}$-valued random variable indicating whether a conversion occurs if $C = 1$.
- S: $\{0, 1\}$-valued random variable indicating whether a sample is correctly labeled if $S = 1$ in the training data.
- D: $\mathbb{R}$-valued random variable of the delay, which is a gap between a click and its resultant conversion. If $C = 0$, it is not defined;
- E: $\mathbb{R}$-valued random variable of the elapsed time between a click and the training time

,where $\mathcal{X} \subset \mathbb{R}^d$. In summary, the data structure is (X, Y, E, D, C, S). Because C and S are unobservable, the training data consist of samples (x_i, y_i, e_i, d_i) where the lower case letter variables correspond

to the realization of random variables. Note that when $y_i = 0$, then d_i is empty.

The samples that are labeled as $Y = 1$ in the training data are true positive ($C = 1$). In other words, these samples are correctly labeled($S = 1$) because $Y = C$. Formally, $Y = 1 \Leftrightarrow S = 1, C = 1$. In the delayed feedback, however, some positive samples ($C = 1$) are mislabeled ($S = 0$) when their elapsed time E is shorter than their delay D. Formally, $E < D \Rightarrow S = 0$. Hence, they are labeled as $Y = 0$ although $C = 1$. For instance, the samples observed right before the end of the training dataset are yet to be converted but would eventually be. Therefore, the negative samples in the training data consists of false and true ones. Formally, $Y = 0 \Leftrightarrow C = 0$ or $S = 0$.

Based on the discussion above, the relation of the conditional distributions of Y and C are as follows:

$$P(Y = 1|X = x) = P(C = 1|X = x)P(S = 1|C = 1, X = x), \quad (1)$$

$$P(Y = 0|X = x) = P(C = 0|X = x) + P(S = 0, C = 1|X = x). \quad (2)$$

The first equation (1) implies that the conditional probability $Y = 1$ is equal to the conditional probability that a conversion occurs and is correctly labeled. The second equation (2) implies that the conditional probability $Y = 0$ is equal to the conditional probability that a conversion either does not occur or is mislabeled.

In the CVR prediction, our objective is to estimate $P(C = 1|X)$ where it is impossible to observe C in the training data. Therefore, we have to use Y to train a CVR prediction model. We regard the delayed feedback as a problem in which there is a discrepancy between the conditional label distribution in the training $P(Y|X)$ and test $P(C|X)$ datasets although the feature distributions $P(X)$ remain the same. We refer to this situation as a feedback shift.

4.2 Problem Formulation

Let $L(x, y, \hat{y}) : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ be a loss function. For simplicity, we assume that a model is parametric. Let $\hat{f}(x, \theta)$ denote a model trained to predict C where $\theta \in \Theta \subset \mathbb{R}^b$ is a parameter. The generalization error G, that is, the expected test error over the training samples, is denoted by (3)

$$G \equiv \mathbb{E}_{(x, c) \sim (X, C)} \left[L \left(x, c; \hat{f}(x, \theta) \right) \right]. \quad (3)$$

Let the optimal parameter

$$\theta^* \in \arg\min_{\theta \in \Theta} G.$$

Our aim is to estimate θ^* to obtain the CVR predictor. Typically, θ^* is estimated using the empirical risk minimization (ERM).

c_i is required to calculate the empirical risk; however, it is not available as it is not observed during the training period. If y_i is used instead, the empirical risk is defined as follows:

$$\hat{G}^{(n)} \equiv \frac{1}{n} \sum_{i=1}^{n} L \left(x_i, y_i; \hat{f}(x_i, \theta) \right). \quad (4)$$

Minimizing the above empirical risk provides an estimator $\hat{\theta}_{ERM}$ that is consistent when c_i and y_i are extracted from the same conditional label distribution[18]:

$$\hat{\theta}_{ERM}^{(n)} \equiv \arg\min_{\theta \in \Theta} \hat{G}^{(n)}. \quad (5)$$

In terms of the delayed feedback, there is a feedback shift that causes the discrepancy between the test distribution from which c_i is drawn and the training distribution from which y_i is drawn. Hence, the ERM estimator is not consistent for most cases. This is represented as follows:

$$\lim_{n\to\infty} \hat{\theta}_{ERM}^{(n)} \neq \theta^*.$$

Specifically, a CVR predictor would be prone to downward bias under the feedback shift because from (1) and (2) we have $P(Y = 1|X = x) \leq P(C = 1|X = x)$.

5 IMPORTANCE WEIGHT (IW) APPROACH

To obtain the consistent ERM estimator, we introduce an feedback shift importance weight (FSIW). First, we provide its theoretical background of FSIW, following which we propose a method for estimating it.

5.1 Theoretical Background

The loss weighted using the FSIW is defined below:

$$\hat{G}_{IW}^{(n)} \equiv \frac{1}{n} \sum_{i=1}^{n} \frac{P(C = y_i|X = x_i)}{P(Y = y_i|X = x_i)} L\left(x_i, y_i; \hat{f}(x_i, \theta)\right). \quad (6)$$

Under the feedback shift, the loss weighted using the FSIW is consistent as follows.

THEOREM 5.1. *The loss weighted using the FSIW is consistent under the feedback shift, that is,*

$$\lim_{n\to\infty} \hat{G}_{IW}^{(n)} = \mathbb{E}_{(x,c)\sim(X,C)}\left[L\left(x, y; \hat{f}(x_i, \theta)\right)\right]. \quad (7)$$

PROOF. As the feature distribution does not change, and because of the law of large numbers, the following holds:

$$\lim_{n\to\infty} \hat{G}_{IW}^{(n)} = \mathbb{E}_{(x,y)\sim(X,Y)}\left[\frac{P(C = y|X = x)}{P(Y = y|X = x)} L\left(x, y; \hat{f}(x, \theta)\right)\right]$$

$$= \iint \frac{P(C = y|X = x)}{P(Y = y|X = x)} L\left(x, y; \hat{f}(x, \theta)\right) P(X = x)P(Y = y|X = x)dydx$$

$$= \iint P(C = y|X = x)L\left(x, y; \hat{f}(x, \theta)\right) P(X = x)dydx$$

$$= \iint L\left(x, c; \hat{f}(x, \theta)\right) P(C = c|X = x)P(X = x)dcdx$$

$$= \mathbb{E}_{(x,c)\sim(X,C)}\left[L\left(x, c; \hat{f}(x, \theta)\right)\right].$$

$\square$

Note that this proof does not assume a specific model, a loss function, or any parameter learning method. Thus, the approach is valid for many models and algorithms.

Since (6) is consistent with the true loss (3), we can obtain a consistent estimator of θ^* by minimizing it. However, it would be impossible to directly estimate FSIW itself because its numerator is $P(C = y|X = x)$, which is what we finally wish to predict. Fortunately, in the delayed feedback case, we can indirectly estimate the FSIW. In the next section, we will explain how to estimate the FSIW in this situation.

5.2 Estimation of FSIW

From (1) and (2), we obtain

$$\frac{P(C = 1|X = x)}{P(Y = 1|X = x)} = \frac{1}{P(S = 1|C = 1, X = x)} \quad (8)$$

$$\frac{P(C = 0|X = x)}{P(Y = 0|X = x)} = 1 - \frac{P(S = 0, C = 1|X = x)}{P(Y = 0|X = x)} \quad (9)$$

Therefore, instead of directly estimating the FSIW $\frac{P(C|X=x)}{P(Y|X=x)}$, we separately estimate the reciprocal of the probability of the occurrence of true positives $P(S = 1|C = 1, X = x)$ and the probability of the occurrence of true negatives $(1 - \frac{P(S=0,C=1|X=x)}{P(Y=0|X=x)})$ using the elapsed time after a click as well as the other features.

To estimate these two probabilities, we prepare training data by artificially creating a situation in which delayed feedback occurs using the following steps. First, we set a hypothetical deadline τ referred to as a *counterfactual deadline*. Secondly, we discard samples that are clicked later than the counterfactual deadline. If the counterfactual deadline is sufficiently long, $Y = C$; thus we assume so. In the Experiment Section, we analyze the performance of the proposed method in an instance where this assumption is violated. Furthermore, we create a new elapsed time e_i' between a click timestamp and the counterfactual deadline. Thirdly, we label these remaining samples S. In the case where $Y = 1$, we label S according to whether the samples are converted before or after the counterfactual deadline. When $Y = 0$, we set $S = 1$. Consequently, we obtain the *artificial dataset* that has labels S.

In estimating $P(S = 1|C = 1, X = x)$, we only use the samples that are converted in the artificial dataset because $C = 1$ is the same as $Y = 1$ in this data. After training a model M using these samples, we make predictions of S on the original training data with $Y = 1$ including the data used to train the model to estimate $P(S = 1|C = 1, X = x)$. To estimate $(1 - \frac{P(S=0,C=1|X=x)}{P(Y=0|X=x)})$, we use samples with $Y = 0$ and those with $S = 0$ in the artificial dataset. For the training and prediction process, the same procedure is conducted as the estimation of $P(S = 1|C = 1, X = x)$. Note that the original elapsed time e_i is used instead of e_i' during prediction. The detailed procedure is shown in Algorithm 1.

6 EXPERIMENT

We conducted two different experiments. First, we evaluated our proposed method on the Criteo dataset[3] comparing it to the state-of-the-art method to show that the proposed method is more efficient.

Secondly, we incorporate the IW approach into the FFM (FFMIW) and evaluate the derived method, FFMIW on the offline in-house dataset that has three campaigns. Based on the offline evaluation, we conducted an A/B test on one of the campaigns to test the effectiveness of the proposed method.

6.1 Public Dataset: Criteo Dataset

6.1.1 Dataset and Metrics. We use the Criteo dataset used in [2] to evaluate the proposed method. The experimental process is identical to that of [2] to equally compare the method. The feature engineering, such as feature crossing and feature hashing is the same as that of [2]. It also includes the timestamps of the clicks

---

[3]https://labs.criteo.com/2013/12/conversion-logs-dataset/

Algorithm 1 Pseudo code of FSIW estimation

Input: train data $\mathcal{D} = \{(x_i, y_i, e_i, d_i, ts_i)\}$, a counterfactual deadline τ, model to estimate FSIW M, and T is a timestamp when the data $\mathcal{D}$ are collected, where x_i is a feature vector, y_i is an observed label, e_i is elapsed time since a click timestamp, d_i is a delay which is a gap between a click timestamp and a CV timestamp, and ts_i is a click timestamp.

Output: FSIW

1: $\mathcal{D}_{iw}^1$, $\mathcal{D}_{iw}^0 = \phi$
2: **for** $i = 1$ to *number of samples* **do**
3:     **if** $ts_i < T - \tau$ and $y_i = 1$ **then**
4:        **if** $ts_i + d_i < T - \tau$ **then**
5:           Label the sample i as $s_i = 1$
6:        **else**
7:           Label the sample i as $s_i = 0$
8:        **end if**
9:        Insert the sample $(x_i, e_i - \tau, s_i)$ to $\mathcal{D}_{iw}^1$
10:     **end if**
11:     **if** $ts_i < T - \tau$ **and** $(ts_i + d_i >= T - \tau$ **or** $y_i = 0)$ **then**
12:        **if** $y_i = 0$ **then**
13:           Label the sample i as $s_i = 1$
14:        **else**
15:           Label the sample i as $s_i = 0$
16:        **end if**
17:        Insert the sample $(x_i, e_i - \tau, s_i)$ to $\mathcal{D}_{iw}^0$
18:     **end if**
19: **end for**
20: Train M on $\mathcal{D}_{iw}^1$ and predict s on $\mathcal{D}$ with the label $y = 1$
21: Train M on $\mathcal{D}_{iw}^0$ and predict s on $\mathcal{D}$ with the label $y = 0$
22: **return** the reciprocal of the prediction for $y = 1$ and the prediction for $y = 0$, which are FSIW.

---

and those of the conversions, if any. Additionally, we include the elapsed time following a click, as a feature to estimate FSIW. We divide the original dataset into seven datasets as follows. For each dataset, the training data starts at a specific time, and ends 3 weeks from that point. The test data begin immediately after the training data, ending in 1 day[4]. For the next dataset, the training data start the day after the starting point of the previous training data, and so forth.

The metrics used to evaluate our method are log loss (LL); area under the precision-recall curve (PR-AUC); and normalized log loss (NLL) that is the log loss normalized by that of the naive predictor that always predicts the average CVR of the training set. The PR-AUC is a more commonly used metric because it is more sensitive to skewed data than the AUC, and generally conversion log data are skewed[12]. In display advertising, the predicted probabilities are important because they are directly used to compute the value of an impression that is equal to a bid in an ad auction. Therefore, the LL and NLL are more important than PR-AUC. Moreover, the LL heavily depends on the mean of the label in the training dataset, that is, a CVR in the context of this setting. Hence, the NLL is a more effective metric for assessing the performance of the CVR predictors than the LL because the NLL is less sensitive to the background CVR[6].

---

[4]The test data were tracked for 30 days; thus samples in the test data are tracked long enough to ensure that they are accurately observed.

6.1.2 FSIW Estimation and Hyperparameters. In this experiment, we estimate the FSIW separately as mentioned in Section 5.2, using the LightGBM [11]. We use the different hyperparamter settings for these two estimates. For estimating $P(S = 1|C = 1, X = x)$, the learning rate is 0.01, number of leaves is 64, and maximum depth is 6. For estimating $1 - \frac{P(S=0, C=1|X=x)}{P(Y=0|X=x)}$, the learning rate is 0.01, number of leaves is 63, and maximum depth is 6. The early stopping technique was applied for both settings to decide the number of trees. Note that a seven-day counterfactual deadline is set. We use the estimated FSIW as a sample weight when training the model to predict the CVR.

For learning a *naive* logistic regression (LR) model that is only trained on the training data without taking the delay into consideration, we use the code provided with the Criteo dataset in [2]. The reported LL in [2] is reproduced when the L2 regularization parameter is 100. We applied this setting to all LR models to predict the CVR during this experiment to establish a fair basis of comparison of the methods.

	LR	DFM	LR-FSIW
LL	0.4076	0.3989	**0.3928***
PR-AUC	0.6345	0.6481	0.6482
NLL	25.21	27.33	**28.02***

Table 1: Average metrics . LR: Logistic Regression without any consideration; DFM: Delayed Feedback Model proposed in [2]; LR-FSIW: Logistic regression with FSIW. * means statistical significance in comparison to DFM.

6.1.3 Result. We compare our suggested model to the DFM[2] and LR. We use LR equipped with the FSIW as our proposed model (LR-FSIW). The result is provided in Table 1. In the rest of the study, we used bootstrap [4] to calculate a 95% confidence interval. Our proposed method improves the LL by 1.5% and the NLL by 2.5% compared to the DFM and these improvements are statistically significant. However, there is no statistically significant difference in PR-AUC.

To demonstrate the feasibility of our proposed method, we measured the training time of the two models for seven datasets. The time reported is the total amount of time. Whereas it takes approximately 140 h to train the DFM, the proposed method requires approximately 2.1 h[5].

6.1.4 Counterfactual deadline. In the Criteo dataset, an observational period which is a duration of tracking samples to fix their labels is 30 days. It would be natural to set a 30-day counterfactual deadline. However, the deadline is so long that the data to be used to estimate the FSIW become obsolete. Hence, it is necessary to shorten the counterfactual deadline, however, there is no means of setting the counterfactual deadline in a natural manner. Therefore, we vary the counterfactual deadline from one to seven days to evaluate the performance. The result, as shown in Figure 5, confirms the stability of our proposed method.

---

[5]DFM is implemented in Cython. https://github.com/CyberAgent/delayedFeedback. We used N1 series with 8 CPU cores in Google Cloud Platform Compute Engine here.

Shota Yasui, Gota Morishita, Komei Fujita, and Masashi Shibata
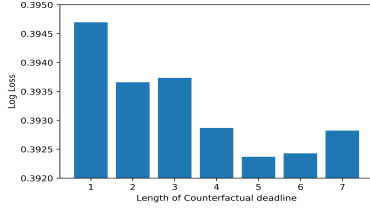
Figure 5: LL of different counterfactual deadline lengths

		LL	PR-AUC	NLL
Campaign L	FFM	0.3523	0.1612	1.7197
	FFMIW	**0.3500**	**0.1660**	**2.304***
Campaign M	FFM	0.2409	0.0808	0.2160
	FFMIW	**0.2401**	**0.0828**	**0.3771**
Campaign S	FFM	0.4026	0.2055	2.9953
	FFMIW	**0.3967**	**0.2058**	**3.361**

Table 2: Average metrics of FFM and FFMIW for 3 different campaigns. * denotes statistical significance.

6.2 Dataset: Dynalyst Dataset

6.2.1 Dataset and Metrics. The experiments are performed using the in-house data provided by Dynalyst. The dataset has three campaigns, the candidates on which our proposed method would be tested online.

In our production environment, the observational period of each campaign is different because it is decided by different advertisers. For instance, Campaign L has a 1-week-long observational period, Campaign M has a 3-day-long observational period, and Campaign S has a 1-day-long observational period. The data has eight categorical features and five numerical features that are categorized. Although we use the hashing trick, the dimensionality is 10^5. The datasets are divided into 16 sets, as in the previous experiment. The difference is that there is a 1-day validation set between the training and test data. Each dataset has a 13-day training set, 1-day validation set, and 1-day test set.

6.2.2 FSIW Estimation and Hyperparameters. In our production environment, we create a model for each campaign. Thus, we trained three models and evaluated them for each campaign.

The hyperparameters setting used to estimate the FSIW are the same, as in the previous experiment. In this experiment, the observational period is designated the counterfactual deadline. For example, the observational period in Campaign L is seven days. Therefore, a seven-day counterfactual deadline is set. Because samples collected 7 days before are correctly labeled, it is reasonable for the counterfactual deadline to be set in such a manner.

For learning a CVR predictor, we used the FFM [10]. The hyperparameters are as follows. There are four latent factors, and the regularization parameter is 0.00002; these settings are identical for the FFM and FFMIW. In addition, we decide the number of iterations by applying early stopping using the validation set; the validation set is weighted according to the FSIW when we train FFMIW. Finally, we combined the training and the validation set, following which trained the model.

6.2.3 Result. We compare the FFM and FFMIW. We evaluated 16 sets, and report the average metrics in Table 2 for three different campaigns. In all the campaigns, the FFMIW seems to be better than the FFM. However, only the difference in the NLL of Campaign L is statistically significant. As discussed in Section 6.1.1, the NLL is the most important metric in the production environment; thus we concluded that the FFMIW outperformed the FFM in Campaign L only.

This is because Campaign S and M have relatively shorter observational period, which makes the delay less influential.

6.3 Online Experiment

According to our offline experiment results, the proposed method outperforms the existing method in Campaign L; thus we conducted 14 days of A/B testing for the FFM and FFMIW in Campaign L.

We trained the FFM and FFMIW once a day with a 14-day-long training data. The number of iterations and the hyperparameters are decided in the same way as described in Section 6.2. During the A/B test period, we equally randomly selected one of the two models to predict the CVR every time a bid request arrives. Note that this A/B test is applied to approximately one million advertising impressions in this period.

The results are listed in Table 3. We observed that there is a statistically significant increase in the number of conversions(CV) and the consumed costs compared to the FFM. This is because since the FFM ignores the delayed feedback and its predicted values are thus subject to downward bias. However, the FFMIW considers the delay, and thus, places relatively higher bids. This results in the higher costs and the acquisition of more CV. The CPA was 2% lower. However, it was not a statistically significant difference. These results indicate that the FFMIW incurred more costs, and performed similarly to the FFM in obtaining a conversion. For the CPA model, a higher cost is suitable if a CPA is the same or lower. This is because advertisers are able to conduct a larger campaign, and the total sales of the product increases. Therefore, the FFMIW outperformed the FFM in our production environment.

CV	Cost	CPA
+31%*	+28%*	-2%

Table 3: Online relative comparison of FFM and FFMIW. The values shown are the relative change in the FFMIW against the FFM. * denotes statistical significance.

7 CONCLUSION

In this study, we tackle on the delayed feedback in CVR prediction by using the IW technique, which constructs the consistent empirical loss. We empirically show that our proposed method performs better than the existing methods in the Criteo dataset. Furthermore, we incorporate the IW approach into the FFM, and compare its performance to that of the FFM without the IW using the in-house dataset. Finally, we conducted an online A/B test to confirm the effectiveness of our proposed method in the production system.

REFERENCES

[1] Deepak Agarwal, Rahul Agrawal, Rajiv Khanna, and Nagaraj Kota. 2010. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In *KDD*.

[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*. ACM, 1097–1105.

[3] Kuang chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past erformance data. In *KDD*.

[4] Bradley Efron and Trevor Hastie. 2016. *Computer Age Statistical Inference: Algorithms, Evidence, and Data Science* (1st ed.). Cambridge University Press, New York, NY, USA.

[5] Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten M. Borgwardt, and Bernhard Schölkopf. 2008. 1 Covariate Shift by Kernel Mean Matching.

[6] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bin 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@KDD*.

[7] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. 2006. Correcting Sample Selection Bias by Unlabeled Data. In *NIPS*.

[8] Pooria Joulani, András György, and Cs. Szepesvari. 2013. Online Learning under Delayed Feedback. In *ICML*.

[9] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. 2017. Field-aware Factorization Machines in a Real-world Online Advertising System. *ArXiv* abs/1701.04099 (2017).

[10] Yu-Chin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *RecSys*.

[11] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *NIPS*.

[12] Sofia Ira Ktena, Alykhan Tejani, Matthias Bethge, Pranay K. Myana, Deepak Dilipkumar, Ferenc Huszár, Steven Yoo, and Wenzhe Shi. 2019. Addressing Delayed Feedback for Continuous Training with Neural Networks in CTR prediction. *ArXiv* abs/1907.06558 (2019).

[13] Zachary Chase Lipton, Yu-Xiang Wang, and Alexander J. Smola. 2018. Detecting and Correcting for Label Shift with Black Box Predictors. In *ICML*.

[14] Aditya Krishna Menon, Krishna Prasad Chitrapura, Sachin Garg, Deepak Agarwal, and Nagaraj Kota. 2011. Response prediction using collaborative filtering with hierarchies and side-information. In *KDD*.

[15] Ciara Pike-Burke, Shipra Agrawal, Cs. Szepesvari, and Steffen Grünewälder. 2017. Bandits with Delayed, Aggregated Anonymous Feedback. In *ICML*.

[16] Rómer Rosales, Haibin Cheng, and Eren Manavoglu. 2012. Post-click conversion modeling and analysis for non-guaranteed delivery display advertising. In *WSDM*.

[17] Abdollah Safari, Rachel MacKay Altman, and Thomas M. Loughin. 2017. Display advertising: Estimating conversion probability efficiently.

[18] Hidetoshi Shimodaira. 2000. Improving predictive inference under covariate shift by weighting the loglikelihood function.

[19] Amos J. Storkey. 2013. When Training and Test Sets are Different: Characterising Learning Transfer.

[20] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Muller. 2007. Covariate Shift Adaptation by Importance Weighted Cross Validation. In *Journal of Machine Learning Research 8*. 985–1005.

[21] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul von Bünau, and Motoaki Kawanabe. 2007. Direct Importance Estimation with Model Selection and Its Application to Covariate Shift Adaptation. In *NIPS*.

[22] Marcelo Tallis and Pranjul Yadav. 2018. Reacting to Variations in Product Demand: An Application for Conversion Rate (CR) Prediction in Sponsored Search. *2018 IEEE International Conference on Big Data (Big Data)* (2018), 1856–1864.

[23] Claire Vernade, Olivier Cappé, and Vianney Perchet. 2017. Stochastic Bandit Models for Delayed Conversions. *CoRR* abs/1706.09186 (2017).

[24] Claire Vernade, Alexandra Carpentier, Giovanni Zappella, Beyza Ermis, and Michael Brueckner. 2018. Contextual Bandits under Delayed Feedback. *ArXiv* abs/1807.02089 (2018).

[25] Yuya Yoshikawa and Yusaku Imai. 2018. A Nonparametric Delayed Feedback Model for Conversion Rate Prediction. In *arXiv*. arXiv.

[26] Bianca Zadrozny. 2004. Learning and evaluating classifiers under sample selection bias. In *ICML*.

[27] Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. 2013. Domain Adaptation under Target and Conditional Shift. In *ICML*.