Unbiased Ad Click Prediction for Position-aware Advertising Systems

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

Yaxu Liu
National Taiwan University
d08944012@csie.ntu.edu.tw

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

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

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

ABSTRACT

Click-through rate (CTR) prediction is a core problem of building advertising systems. In many real-world applications, because an ad placed in various positions has different click probabilities, the position information should be considered in both training and prediction. For such position-aware systems, existing approaches learn CTR models from clicks/not-clicks on historically displayed events by leveraging the position information in different ways. In this work, we explain that these approaches may give a heavily biased model. We first point out that in position-aware systems, two different types of selection biases coexist in displayed events. Secondly, we explain that some approaches attempting to eliminate the position effect from clicks/not-clicks may possess an additional bias. Finally, to obtain an unbiased CTR model for position-aware systems, we propose a novel counterfactual learning framework. Experiments confirm both our analysis on selection biases and the effectiveness of our proposed counterfactual learning framework.

CCS CONCEPTS

• **Information systems → Computational advertising**.

KEYWORDS

CTR prediction, Selection bias, Counterfactual learning

ACM Reference Format:
Bowen Yuan, Yaxu Liu, Jui-Yang Hsia, Zhenhua Dong, and Chih-Jen Lin. 2020. Unbiased Ad Click Prediction for Position-aware Advertising Systems. In *Fourteenth ACM Conference on Recommender Systems (RecSys '20), September 22–26, 2020, Virtual Event, Brazil.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3383313.3412241

1 INTRODUCTION

With the development of online advertising in recent decades, compensation methods in advertising systems have been evolved from simply charging by the number of displays to charging by explicit behaviours on displayed ads. A typical example is cost-per-click (CPC) systems, where advertisers are only charged if their ads are clicked by customers. When a customer visits a publisher page providing K positions to display ads, a request carrying the information of the publisher and the customer will be sent to the system. Once the system receives the request, the following two procedures are conducted.

- Retrieval: The system selects K ads from an inventory of n ads.
- Placement: The system assigns positions for the retrieved K ads and places them.

We use a tuple (i, j, k) to denote an event that in request i, ad j is placed in position k. After these events are displayed, the system will receive signals if a customer clicks one of these events. Then the system gains revenue from the corresponding advertiser.

To maximize the revenue, the system must select the most valuable events to display. Therefore, the system values event (i, j, k) with the following expected revenue

$$P_{ijk} \times \text{bid}_j, \tag{1}$$

where bid_j is the price paid by advertisers if the ad j is clicked by customers, and P_{ijk} is the probability of event (i, j, k) being clicked by the customer. For each request i, to maximize the expected revenue, the system solves the following optimization problem to retrieve and place K ads

$$\max_{S_i} \sum_{(i,j,k)\in S_i} P_{ijk} \times \text{bid}_j, \tag{2}$$

where S_i is a sequence of events $(i, j_1, 1), \cdots, (i, j_K, K)$ indicating the K ads displayed in K positions. We give a simplified example in Figure 1, which shows that for any request i, five ads bid for three positions. According to the solution to (2), the system retrieves ad "1","2", and "5" and then places them in position "2", "1" and "3", respectively. Thus displayed events are $(i, 2, 1)$, $(i, 1, 2)$ and $(i, 5, 3)$. Among these displayed events, event $(i, 2, 1)$ is clicked, while the other two are not clicked.

The estimation of P_{ijk}, also referred to as click-through rate (CTR) prediction, is the core problem of building advertising systems, which generally consists of the following two tasks

- Learning a CTR model to estimate P_{ijk} via machine learning techniques
- Conducting an offline evaluation to estimate the CTR model's online performance before it is deployed

Typically CTR models are learned from clicks/not-clicks on historically displayed events. Many past works (e.g., [2, 13]) assume their systems are position-independent, so whether the event

RecSys '20, September 22–26, 2020, Virtual Event, Brazil

ACM ISBN 978-1-4503-7583-2/20/09…$15.00
https://doi.org/10.1145/3383313.3412241

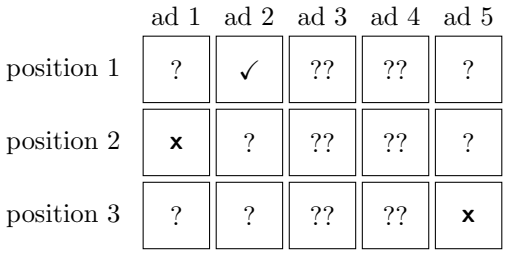


Figure 1: An illustration of a scenario of five ads biding for three positions. For displayed events, the symbols '✓', and 'x' indicate the event is clicked, and not clicked, respectively. For any non-displayed event (i, j, k), the symbol '??' indicates ad j is not retrieved, and the symbol '?' indicates ad j is retrieved but not placed in position k.

(i, j, k) is clicked or not only depends on the information included in request i and ad j, and the position k has no effect. With this assumption, clicks/not-clicks on displayed events are observations from the following position-independent click probability,

$$P_{ijk} = \Pr(\text{click=1} \mid \text{request} = i, \text{ad} = j). \tag{3}$$

Then they learn CTR models as an estimator of (3).

In many real-world applications, an ad placed in various positions has different click probabilities. Thus clicks/not-clicks on displayed events are observations from the following position-aware click probability.

$$P_{ijk} = \Pr(\text{click=1} \mid \text{request} = i, \text{ad} = j, \text{position} = k). \tag{4}$$

Such position-aware systems are the focus in this work. Existing approaches fall into the following two categories.

• Positional approach: This approach learns an estimator of (4) as the CTR model, where position information is included in input features

• Non-positional approach: The positional approach may not be practically viable, because (2) is a difficulty assignment problem. By assuming that (4) can be decomposed to a position-independent part $\tilde{P}_{ij}$ and a position-dependent part, (2) can be easily solved. This approach learns an estimator of $\tilde{P}_{ij}$ as the CTR model.

However, in this work, we argue that existing approaches for position-aware systems are biased. For position-independent systems, it is known that the selection bias in historically displayed events is a serious issue. Some recent works (e.g., [3, 12, 21]) have proposed solutions, but few studies have considered this issue in position-aware systems. Our first contribution is to investigate the selection bias in position-aware systems. Specifically, we point out that two different types of selection biases coexist, so the situation is more complicated than that of position-independent systems.

Our second contribution is to point out that CTR models obtained by the non-positional approach may posses an additional type of biases. The reason is that because clicks and not-clicks on displayed events are observations from P_{ijk} rather than $\tilde{P}_{ij}$, CTR models are learned and evaluated indirectly.

Finally, to address the above issues, our third contribution is to derive a new counterfactual learning framework to consider both displayed events and non-displayed events. The idea is related to [21] for position-independent scenarios. In [21], by ignoring the position information, non-displayed events are those of non-retrieved ads. Examples are ads "3", and "4" in Figure 1, where events are marked by "??". However, for position-aware systems, we should additionally consider events of retrieved ads with non-placed positions. Examples in Figure 1 are events of retrieved ads "1", "2", and "5" marked by "?". We prove that under the proposed framework, for both positional and non-positional approaches, learning and evaluation are unbiased. A related topic is the unbiased learning to rank from clicks [8, 19], but we will explain in Section 5 that these techniques are not applicable to position-aware advertising systems which require the point-wise learning for click probability due to the existence of bid in (2). Therefore, to the best of our knowledge, this is the first time that a solution of unbiased CTR prediction for position-aware systems is offered.

Table 1: Main notation.

(i, j, k)	(request, ad, position) event
m, n, K	numbers of requests, ads, positions
$\bar{C}, \mathcal{E}$	indicator random variables of click and examination
$(\boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}})$	random variables for feature vectors of a request, an ad, and a position
S, S^+, S^-	sets of displayed events and clicked/not-clicked displayed events
C	a tensor including $C_{ijk} = 0/1$ as a not-click/click observation of event (i, j, k)
$\sigma(\cdot)$	a function transforming real values to $[0, 1]$ as probabilities
$\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{p}$	feature vector of a request, an ad and a position
$g(\cdot), f(\cdot), a(\cdot)$	positional/non-positional CTR models and imputation model

The paper is organized as follows. In Section 2, we review existing settings for position-independent and position-aware systems. In Section 3, we first review the issue of selection bias in position-independent systems. We then extend the discussion into position-aware systems and novelly derive two types of selection biases. In Section 4, we propose our solutions of unbiased CTR prediction for position-aware systems. Related works are given in Section 5. In Section 6, a series of experiments confirm our contributions. Finally Section 7 concludes this work. A list of notations used in this work is in Table 1. Supplementary materials and experiments code are at https://www.csie.ntu.edu.tw/~cjlin/papers/debiases/.

2 ADVERTISING SYSTEMS: A REVIEW AND EVOLUTION TO POSITION-AWARE SCENARIO

We review position-independent and position-aware systems.

2.1 Position-independent Systems

To estimate CTR, most approaches assume that the position k has no effect and the following click probability depends only on the information included in request i and ad j.

$$P_{ijk} \equiv P_{ij} = \Pr(\bar{C} = 1 \mid \boldsymbol{\mathcal{U}} = \boldsymbol{u}_i, \boldsymbol{\mathcal{V}} = \boldsymbol{v}_j), \tag{5}$$

where $\bar{C}$ is the random variable indicating whether the event is being clicked, $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{V}}$ are random variables respectively indicating feature vectors of a request and an ad, and $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ are feature vectors of request i and ad j, respectively. From displayed events

(denoted as S) with clicks and not-clicks respectively as positive (denoted as S^+) and negative instances (denoted as S^-), a CTR model $y(\boldsymbol{u}, \boldsymbol{v})$ is learnt by solving a binary classification problem.

$$\min_{\mathcal{W}} \quad \sum_{(i,j,k)\in S^+} \ell(1, \hat{Y}_{ij}) + \sum_{(i,j,k)\in S^-} \ell(0, \hat{Y}_{ij}) + \lambda \mathcal{R}(\mathcal{W}), \quad (6)$$

where $\hat{Y}_{ij} \equiv y(\boldsymbol{u}_i, \boldsymbol{v}_j)$, $\mathcal{W}$ is the set of parameters in the function $y(\boldsymbol{u}, \boldsymbol{v})$, $\mathcal{R}(\mathcal{W})$ is the regularization term with the parameter λ decided by users, and $\ell(a, b)$ is any loss function. For example, the following logistic loss is commonly used.

$$\ell(a, b) = -ab + \log(1 + e^b). \quad (7)$$

To achieve good CTR, many past works have focused on developing the model $y(\boldsymbol{u}, \boldsymbol{v})$, which can range from simple ones (e.g., logistic regression [2, 15]) to sophisticated ones (e.g., factorization machine [9, 14], deep learning [5]).

In the prediction phase, because from (5) the click probability is position-independent, the problem (2) is reduced to

$$\max_{S_i} \sum_{(i,j,\_)\in S_i} P_{ij} \times \text{bid}_j. \quad (8)$$

Solving (2) becomes very easy by simply selecting the K ads with the largest values of $P_{ij} \times \text{bid}_j$. In practice, we consider the predicted $y(\boldsymbol{u}_i, \boldsymbol{v}_j)$ and use $\sigma(y(\boldsymbol{u}_i, \boldsymbol{v}_j))$ to transform $y(\boldsymbol{u}_i, \boldsymbol{v}_j)$ into a probability value in $[0, 1]$ as an estimator of P_{ij} (e.g., the sigmoid function). We then sort

$$\sigma(y(\boldsymbol{u}_i, \boldsymbol{v}_j)) \times \text{bid}_j,$$

of all n ads and display the top K ads. Thus handling a request can be done within $\mathbb{O}(n \log n)$ cost.

Before deploying a machine learning model, usually we need offline evaluation to select models and tune parameters such as λ in (6). For this purpose, we usually split displayed events to two parts:

- One part is treated as the training set S.
- The other part is the test set S_{te} for offline evaluation.

The success of offline evaluation relies on that the test set is independent of the training set. For CTR prediction, the progressive validation [13] is commonly used. That is, we split the displayed events in a chronological way and the latest period is treated as the test set S_{te}.

For evaluation criterion, two metrics are commonly used. Suppose $|S_{\text{te}}|$ is the number of events in the test set S_{te}. The first criterion is the logarithmic loss (LogLoss):

$$\frac{1}{|S_{\text{te}}|}\Big(\sum_{(i,j,k)\in S^+_{\text{te}}} \log(1 + e^{-y(\boldsymbol{u}_i, \boldsymbol{v}_j)}) + \sum_{(i,j,k)\in S^-_{\text{te}}} \log(1 + e^{y(\boldsymbol{u}_i, \boldsymbol{v}_j)})\Big). \quad (9)$$

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

$$\frac{\sum_{(i,j,k)\in S^+_{\text{te}}} \text{Rank}_{i,j} - \frac{|S^+_{\text{te}}|(|S^+_{\text{te}}|+1)}{2}}{|S^+_{\text{te}}||S^-_{\text{te}}|}, \quad (10)$$

where $|S^+_{\text{te}}|$, $|S^-_{\text{te}}|$ are respectively the number of positive instances and negative instances, and $\text{Rank}_{i,j}$ is the rank of a positive event (i, j, k) in S_{te}, obtained through ordering predicted values from $y(\boldsymbol{u}, \boldsymbol{v})$.

2.2 Position-aware Advertising Systems

In contrast to the position-independent systems reviewed in Section 2.1, in many applications the click probability of an event is position-aware. That is, in each request, an ad placed in various positions has different click probabilities and the assumption in (5) no longer holds. To incorporate position information, we review two approaches that learn an estimator $g(\cdot)$ of position-aware CTR in (4) by solving the following problem.

$$\min_{\mathcal{W}} \sum_{(i,j,k)\in S^+} \ell\left(1, \hat{C}_{ijk}\right) + \sum_{(i,j,k)\in S^-} \ell\left(0, \hat{C}_{ijk}\right) + \lambda \mathcal{R}(\mathcal{W}), \quad (11)$$

where $\hat{C}_{ijk} \equiv g(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k)$, and $\boldsymbol{p}_k$ is the feature vector of position k. Because the two approaches differ on whether the inferred CTR model depends on the position information, we call them respectively as positional approach and non-positional approach.

2.2.1 Positional Approach. A positional approach directly applies $g(\cdot)$ as the CTR model. In the training stage, because clicks and not-clicks have been observed , the task is to simply solve the binary classification problem in (11). However, a bottleneck of this approach occurs in the prediction stage. The complexity in deciding which and where ads to be displayed is extremely high. Specifically, once $g(\cdot)$ is deployed online for handling the request i, among P^n_K permutations, we must choose one so that the sum of

$$\sigma(\hat{C}_{ijk}) \times \text{bid}_j \quad (12)$$

over K events is maximized. Such an assignment problem is a difficult combinational optimization issue. For any system with large n, handling each request in real-time is prohibitive.

2.2.2 Non-positional Approach. To avoid the prohibitive cost in the prediction stage, the non-positional approach is proposed [1, 15]. Let $\boldsymbol{\mathcal{P}}$ and $\mathcal{E}$ be respectively the random variable of a position's feature vector, and the indicator of whether the ad is examined or not. The idea is to assume that (4) can be expressed as

$$\Pr(\bar{C} = 1 \mid \boldsymbol{\mathcal{U}} = \boldsymbol{u}_i, \boldsymbol{\mathcal{V}} = \boldsymbol{v}_j, \boldsymbol{\mathcal{P}} = \boldsymbol{p}_k) = \tilde{P}_{ij} \times \beta_k, \quad (13)$$

where we factor out a position-independent term

$$\tilde{P}_{ij} \equiv \Pr(\bar{C} = 1 \mid \boldsymbol{\mathcal{U}} = \boldsymbol{u}_i, \boldsymbol{\mathcal{V}} = \boldsymbol{v}_j, \mathcal{E} = 1), \quad (14)$$

being the click probability of ad j given it has been examined by the customer, and

$$\beta_k \equiv \Pr(\mathcal{E} = 1 \mid \boldsymbol{\mathcal{P}} = \boldsymbol{p}_k) \quad (15)$$

is the probability of any ad placed at position k being examined. Without loss of generality, we number the positions such that

$$\beta_1 \geq \beta_2 \geq \cdots \geq \beta_K. \quad (16)$$

Under the assumption in (13), the expected revenue of placing ad j at position k is changed from (1) to $\tilde{P}_{ij} \times \text{bid}_j \times \beta_k$. From (16), for any retrieved K ads, we must order them such that $\tilde{P}_{ij_1} \times \text{bid}_{j_1} \geq \cdots \geq \tilde{P}_{ij_k} \times \text{bid}_{j_k}$ in order to maximize the objective value in (2). Then problem (2) is reduced to find and sort K values of $\tilde{P}_{ij} \times \text{bid}_j$ such that the inner product with $\beta_1, \cdots, \beta_k$ is maximized. Clearly, the solution is to retrieve the K ads with the largest $\tilde{P}_{ij} \times \text{bid}_j$. Therefore, in the training stage we learn a CTR model $f(\cdot)$ so that $\sigma(f(\boldsymbol{u}_i, \boldsymbol{v}_j))$ is an estimate of $\tilde{P}_{ij}$ in (14). Once $f(\cdot)$ is deployed online for handling request i, we sort

$$\sigma(f(\boldsymbol{u}_i, \boldsymbol{v}_j)) \times \text{bid}_j \quad (17)$$

of all n ads and orderly place the top K ads from the first position to the Kth position, which can be done within $\mathbb{O}(n \log n)$. This setting effectively addresses the high cost of using (12) to select events for display.

Though using (17) rather than (12) helps to develop an efficient display algorithm, it brings a new difficulty in the training stage for learning $f(\cdot)$. For most systems, observations from $\tilde{P}_{ij}$ and β_k can not be collected, because we do not know if an ad is examined or not. In contrast, if (4) instead of (14) is considered, collected clicks and not-clicks are available observations. To overcome the difficulty, some recent works [6, 11, 23] propose to infer $f(\cdot)$ from $g(\cdot)$. Specifically, they follow the positional approach to solve the problem (11), but consider a customized $g(\cdot)$, which can be decomposed into two parts: $f(\cdot)$ and $h(\cdot)$. Then $\sigma(f(\cdot))$ is taken as the estimator of (14), while $\sigma(h(\cdot))$ is the estimator of (15). For example, in [6], they propose

$$\sigma(g(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k)) = \sigma\left(f(\boldsymbol{u}_i, \boldsymbol{v}_j)\right) \sigma\left(h(\boldsymbol{p}_k)\right). \tag{18}$$

Another line of works (e.g., [11, 23]) consider the following combination.

$$g(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k) = f(\boldsymbol{u}_i, \boldsymbol{v}_j) + h(\boldsymbol{p}_k). \tag{19}$$

2.2.3 Offline Evaluation. As we mentioned in Section 2.1, before deploying a CTR model, we need an offline evaluation to estimate the potential online performance. For the positional approach, because collected clicks and not-clicks of displayed events in S_{te} have been observed, we can compute LogLoss and AUC scores respectively defined in (9) and (10) but replace $y(\boldsymbol{u}_i, \boldsymbol{v}_j)$ with position-aware predictions $g(\boldsymbol{u}_i, \boldsymbol{v}_j\ \boldsymbol{p}_k)$, for any event (i, j, k) in S_{te}.

For the non-positional approach aiming to estimate (14), as we lack the corresponding observations, we can not directly evaluate the CTR model $f(\cdot)$ from clicks/not-clicks of displayed events in S_{te}. Instead, existing works (e.g., [6, 23]) consider $g(\cdot)$ as the surrogate to evaluate LogLoss and AUC scores. We then deploy a $f(\cdot)$ extracted from $g(\cdot)$ that can result in the best LogLoss or AUC score. However, this indirect way brings a new potential problem that will be discussed in Section 4.2.

2.2.4 Discussion. From the review in this section, we can see that to construct a position-aware system, both positional and non-positional approaches have their own merits. We note that in the training phase, both learn some $g(\cdot)$ functions as an estimator of (4) through solving the problem in (11). Unfortunately, we will discuss in Section 3 that solving (11) with clicks/not-clicks on past displayed events may give a heavily biased estimator of (4).

3 SELECTION BIAS IN CTR PREDICTION

In this section, we first review the issue of selection bias in the position-independent systems. Then we point out that for the position-aware systems, this issue also occurs and becomes more complicated.

3.1 Selection Bias in Position-independent Systems

Let $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}})$ be the distribution to generate events and the corresponding clicks or not-clicks. The probability value from the CTR model $y(\boldsymbol{u}, \boldsymbol{v})$ obtained by solving the following expected risk minimization should be an optimal estimation of (5).

$$\min_y R(y), \text{ where } R(y) = \int \ell(Y, y(\boldsymbol{u}, \boldsymbol{v})) d\Pr(Y, \boldsymbol{u}, \boldsymbol{v}). \tag{20}$$

In practice, because $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}})$ is unknown, we conduct empirical risk minimization by minimizing the average loss of sampled results. Assume that m different requests have occurred. Then from (20), we should learn $y(\boldsymbol{u}, \boldsymbol{v})$ from all mn observations Y_{ij}, for $i = 1, \cdots, m$, $j = 1, \cdots, n$, which are observations of the click result $\bar{C}$ under $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$. Thus we have a standard binary classification problem of mn training instances $(Y_{ij}, \boldsymbol{u}_i, \boldsymbol{v}_j)$, $\forall (i, j)$. However, because the previously deployed CTR model only displays a subset S of events, only Y_{ij} for $(i, j, k) \in S$ are revealed. Thus Y_{ij} satisfies

$$Y_{ij} = \begin{cases} 1 & (i, j, k) \in S^+, \\ 0 & (i, j, k) \in S^-, \\ \text{unrevealed} & (i, j, k) \notin S. \end{cases} \tag{21}$$

Because we lack Y_{ij} for non-displayed events, as mentioned in Section 2.1, most approaches solve (6) only with S by ignoring the non-displayed events. However, many works (e.g., [16, 21]) have pointed out that because the distribution of displayed events in S is biased to that of the overall mn events, the model obtained by (6) is a biased estimator of the probability defined in (5). Specifically, let O be an indicator of whether an event is displayed, and $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, O = 1)$ be the joint distribution of clicks, request, and ads being displayed. The expected risk minimization is changed to

$$\min_y \bar{R}(y), \text{ where } \bar{R}(y) = \int \ell(Y, y(\boldsymbol{u}, \boldsymbol{v})) d\Pr(Y, \boldsymbol{u}, \boldsymbol{v}, O = 1). \tag{22}$$

Because (22) can be reformulated as

$$\bar{R}(y) = \int \ell(Y, y(\boldsymbol{u}, \boldsymbol{v})) \frac{d\Pr(Y, \boldsymbol{u}, \boldsymbol{v}, O = 1)}{d\Pr(Y, \boldsymbol{u}, \boldsymbol{v})}, \tag{23}$$

from (20), whether $\bar{R}(\cdot)$ is an unbiased expected risk depends on if events have the same chance to be displayed. That is, if there exists a constant $\Delta > 0$ such that

$$\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, O = 1) = \Delta \Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}), \tag{24}$$

then

$$\bar{R}(y(\boldsymbol{u}, \boldsymbol{v})) = \int \ell(Y, y(\boldsymbol{u}, \boldsymbol{v})) \Delta d\Pr(Y, \boldsymbol{u}, \boldsymbol{v}) = \Delta R(y), \tag{25}$$

so (20) and (22) return the same optimal model. To investigate (24), we first express two sides respectively as

$$\begin{aligned} &\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, O = 1) = \Pr(\bar{C} \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, O = 1) \Pr(\boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, O = 1), \\ &\text{and } \Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}) = \Pr(\bar{C} \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}) \Pr(\boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}). \end{aligned} \tag{26}$$

Because the tendency of a customer click can not be affected by if the ad is displayed, given an event, we should have

$$\Pr(\bar{C} \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, O = 1) = \Pr(\bar{C} \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}). \tag{27}$$

By using (27) to cancel terms in the division of the two expressions in (26) and through $\Pr(\boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, O = 1) = \Pr(\boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}) \Pr(O = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}})$, the task of investigating (24) is reduced to checking if the distribution of an event being displayed

$$\Pr(O = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}) \tag{28}$$

is uniform. For position-independent systems, whether event (i, j, k) can be included in S depends on if ad j is retrieved. To maximize the revenue, most real-world systems retrieve ads with higher expected revenue into S, so (28) is non-uniform and (24) does not hold, nor does (25). Then $R(\cdot)$ is not proportional to $\bar{R}(\cdot)$, so $\bar{R}(\cdot)$ is a biased expected risk of $y(\boldsymbol{u}, \boldsymbol{v})$ and the model obtained by solving (6) leads to a biased estimator of (5). The bias caused by the non-uniform (28) is referred to as the selection bias and we say the selected set S carries the selection bias. Analysis in works such as [21] has pointed out that a biased model may overestimate the expected revenue of ads which had less opportunity to appear in S and wrongly display too many of them in future requests. For the offline evaluation, the set S_{te} obtained from displayed events in the test period also carries the selection bias. Thus criteria such as LogLoss and AUC computed on S_{te} are biased and cannot give an accurate performance estimation.

3.2 Two Types of Selection Bias in Position-aware Systems

Now we extend our discussion of the selection bias to position-aware systems, where clicks and not-clicks are affected by positions. For a displayed event (i, j, k) in request i, we only know whether ad j placed in position k is clicked or not, but the click information of this ad placed in other positions is unknown. To incorporate the position information, we extend Y in (21) into a tensor $C \in \{0, 1\}^{m \times n \times K}$ where each C_{ijk} is the click/not-click observation of the random variable $\bar{C}$. Only for any event included in S, the corresponding C_{ijk} can be revealed. Thus C_{ijk} satisfies

$$C_{ijk} = \begin{cases} 1 & (i, j, k) \in S^+, \\ 0 & (i, j, k) \in S^-, \\ \text{unrevealed} & (i, j, k) \notin S. \end{cases} \tag{29}$$

By the discussion of (28) in Section 3.1, in a biased setting, the distribution of an event being included in S,

$$\Pr(O = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}}), \tag{30}$$

is non-uniform. Thus similar to the failure of (24), in position-aware systems, $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}}, O = 1)$ is not proportional to $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}})$. As a result, the learnt $g(\cdot)$ from solving (11) is a biased estimator of (4). However, different from the situation of position-independent systems where only the retrieval procedure is involved, for position-aware systems, event (i, j, k) is included in S only if the placement procedure puts the retrieved ad j at position k. From this view, by assuming the retrieval procedure is independent to the position information, we can expand (30) as

$$\begin{aligned} (30) &= \Pr(\text{retrieved=1}, \text{placed@}\boldsymbol{\mathcal{P}}\text{=1} \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}}) \\ &= \Pr(\text{retrieved} = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}) \\ &\quad \times \Pr(\text{placed@}\boldsymbol{\mathcal{P}} = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}}, \text{retrieved} = 1), \end{aligned} \tag{31}$$

In (31), the retrieval distribution,

$$\Pr(\text{retrieved} = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}), \tag{32}$$

is the same as (28) for position-independent systems, but the following placement distribution is additionally involved for position-aware systems

$$\Pr(\text{placed@}\boldsymbol{\mathcal{P}} = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}}, \text{retrieved} = 1). \tag{33}$$

Therefore, besides the bias caused by the non-uniform (32) discussed in Section 3.1, we expect the non-uniform (33) in the placement procedure introduces some other bias. To systematically investigate this, our next analysis is conducted from two aspects:

- We extend the analysis given in [21] to position-aware systems by assuming that (33) is uniform.
- To explore the bias newly introduced by the placement procedure, we assume (32) is uniform.

To make (33) uniform, we assume the retrieved K ads are placed in random positions. In this scenario, the system can be seen as a combination of K unrelated position-independent systems and each has a single position. From this view, the analysis given in Section 3.1 can be easily extended to position-aware systems. Given a fixed position k, for an ad more frequently displayed in this position, more associated events are included in S and the clicked/not-clicked results are revealed in C. Then the observed click probability of this ad placed in position k is closer to the unbiased one obtained from the fully-labeled C. In contrast, for an ad infrequently displayed in this position, it was chosen because of the higher expected revenue predicted by the previous model. The observed click probability tends to be higher than the unbiased one. The overestimation of such ads may carry on through the learning of $g(\cdot)$ by solving (11).

Next, to understand the new bias caused by the non-uniform (33) in the placement procedure, we design a hypothetical example. Consider a position-independent system where in each request, two ads are displayed in two positions. Because both ads are retrieved,

$$\Pr(\text{retrieved} = 1 \mid \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}) = 1.$$

Thus no bias is introduced in the retrieval procedure. We assume that a previous model was deployed for handling requests. For each request, the model tends to place the ad with a higher click probability in the first position, while the one with a lower click probability in the second position. Apparently, the observed click probability of an ad placed in the first position is significantly higher than if it is placed in the second position. This example indicates that for any position-aware scenario, even if the bias discussed in Section 3.1 is avoided through applying the uniform (32), the non-uniform (33) results in a new bias. The estimator $g(\cdot)$ obtained by the positional approach should overestimate click probability of each ad placed in front positions while underestimate click probability of each ad placed in last positions.

Moreover, for the non-positional approach that extracts $f(\cdot)$ from $g(\cdot)$, if the bias due to (33) is ignored, we may excessively blame the examination probability in (15) for the difference of click probabilities among various positions. Specifically, recall that in the hypothetical example given above, we assume a position-independent system, whose probability in (15) of two positions should satisfy

$$\Pr(\mathcal{E} = 1 \mid \boldsymbol{\mathcal{P}} = \boldsymbol{p}_1) = \Pr(\mathcal{E} = 1 \mid \boldsymbol{\mathcal{P}} = \boldsymbol{p}_2) = 1.$$

Thus an unbiased estimator $h(\cdot)$ of (15) should predict the same value for these two positions. However, we have mentioned that the bias leads to a large gap between

$$\Pr(\bar{C} = 1 \mid \boldsymbol{\mathcal{U}} = \boldsymbol{u}_i, \boldsymbol{\mathcal{V}} = \boldsymbol{v}_j, \boldsymbol{\mathcal{P}} = \boldsymbol{p}_1) \text{ and}$$
$$\Pr(\bar{C} = 1 \mid \boldsymbol{\mathcal{U}} = \boldsymbol{u}_i, \boldsymbol{\mathcal{V}} = \boldsymbol{v}_j, \boldsymbol{\mathcal{P}} = \boldsymbol{p}_2)$$

observed from displayed events, which misleads us to consider this system to be position-dependent. If we learn $g(\cdot)$ by solving (11) with the set carrying the bias, the $h(\cdot)$ extracted from the learnt $g(\cdot)$ will wrongly digest the gap. As a result, both extracted $f(\cdot)$ and $h(\cdot)$ should be biased and cause the loss of revenue.

To summarize, in this section, we point out two different types of selection biases in position-aware systems. According to how they occur, we formally term them respectively as

• Retrieval bias: The bias caused by the non-uniform (32). That is, in each request, some ads are non-uniformly retrieved.

• Placement bias: The bias caused by the non-uniform (33). That is, the retrieved ads are non-uniformly placed in positions.

The coexistence of two types of selection biases makes the situation much more complicated than that of position-independent systems. Next, we propose a novel framework of unbiased CTR prediction for position-aware systems.

4 UNBIASED CTR PREDICTION FOR POSITION-AWARE SYSTEMS

Up to now, for the position-aware systems, we can conclude that due to the retrieval bias and the placement bias discussed in Section 3.2, both the positional approach and the non-positional approach reviewed in Section 2.2 meet two difficulties:

• CTR models learnt from the biased training set S cannot give accurate predictions.

• Offline evaluation conducted on a biased test set S_{te} gives misleading performance estimation on CTR models.

In this section, we propose our solutions to overcome these difficulties respectively for the two approaches.

4.1 Unbiased Positional Approach

We first discuss an unbiased solution for the positional approach, which aims to estimate (4). Following the discussion in Section 3, to avoid the selection bias, we should consider all mnK training instances $(C_{ijk}, \boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k)$ drawn from $\Pr(C, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{p})$ and learn an unbiased estimator of (4) through solving the following problem.

$$\min_{\mathcal{W}} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K} \ell(C_{ijk}, \hat{C}_{ijk}) + \lambda\mathcal{R}(\mathcal{W}). \tag{34}$$

However, in practice only a subset S of events are displayed, so from (29), C_{ijk} of non-displayed events are unrevealed. Further, instances $(C_{ijk}, \boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k)$ in S may not follow the true probability distribution $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}})$, so solving the binary classification problem in (11) leads to a biased model. The research "counterfactual learning" [18] aims to obtain an unbiased model by (34) or its modifications under the given partially labeled and possibly biased instances. Existing approaches are based on either one (or some) of the following ideas:

• Randomization: A straightforward idea is to collect an unbiased training set S through randomly retrieving K ads and then placing them on random positions. Because both (32) and (33) become uniform distributions, $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}}, O = 1)$ is proportional to $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}})$. Therefore, the optimization problem (11) is the empirical risk minimization of the expected risk and the resulting model is unbiased. In other words, instead of drawing all mnK instances from $\Pr(\bar{C}, \boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{P}})$, we sample fewer instances and solve an approximation of (34). The number of instances collected through the random strategy is crucial. It cannot be too small to give enough converage. On the other hand, in contrast to the greedy strategy discussed in Section 2.2 to maximize the expected revenue, the random strategy may cause a significant loss of revenue. Thus collecting a large S by the random strategy is often impractical.

• Imputation of non-displayed events [4, 17]: As $C_{ijk}, \forall(i,j,k) \notin S$ are not available, the idea is to estimate these values by learning an imputation model $a(\cdot)$ using displayed events in S. Assume $a(\cdot)$ outputs $A_{ijk} \equiv a(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k)$ as labels for non-displayed events. The following problem is solved

$$\min_{\mathcal{W}} \sum_{(i,j,k)\in S} \ell(C_{ijk}, \hat{C}_{ijk}) + \sum_{(i,j,k)\notin S} \ell(A_{ijk}, \hat{C}_{ijk}) + \lambda\mathcal{R}(\mathcal{W}). \tag{35}$$

However, the imputation model $a(\cdot)$ learnt from a biased set S may propagate the bias to non-displayed events, so CTR models from (35) may still be biased.

• Inverse-propensity scores (IPS) of displayed events [7, 16]: From (30), define $z_{ijk} = \Pr(O = 1 \mid \boldsymbol{\mathcal{U}} = \boldsymbol{u}_i, \boldsymbol{\mathcal{V}} = \boldsymbol{v}_j, \boldsymbol{\mathcal{P}} = \boldsymbol{p}_k)$ referred to as the propensity score of event (i,j,k). We can have a 0/1 random variable $\tilde{O}_{ijk}$ that follows a Bernoulli distribution as $\tilde{O}_{ijk} \sim \text{Bern}(z_{ijk})$ to control whether an event (i,j,k) can be displayed. Because

$$\mathbb{E}_{\tilde{\boldsymbol{O}}}\Big[\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K} \frac{\ell(C_{ijk}, \hat{C}_{ijk})}{z_{ijk}} \tilde{O}_{ijk}\Big] = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K} \ell(C_{ijk}, \hat{C}_{ijk}),$$

from the observation of $\tilde{O}_{ijk}$ satisfying $\tilde{O}_{ijk} = 0, \forall(i,j,k) \notin S$, we can consider the following problem

$$\min_{\mathcal{W}} \sum_{(i,j,k)\in S} \frac{\ell(C_{ijk}, \hat{C}_{ijk})}{z_{ijk}} + \lambda\mathcal{R}(\mathcal{W}) \tag{36}$$

to get an unbiased estimator of P_{ijk}. Unfortunately, it is known that accurately estimating z_{ijk} is not an easy task. This requires that whether event (i,j,k) can be displayed is decided by a random variable $\tilde{O}_{ijk}$, but as pointed out in [21], for most advertising systems, displayed events are selected according to the expected revenue. Such a deterministic display algorithm makes the IPS method impractical.

Through a tradeoff between these ideas and their shortcomings, we extend the framework proposed in [21] to position-aware systems, which combines the three ideas listed above and solves the following problem.

$$\begin{aligned}\min_{\mathcal{W}} \sum_{(i,j,k)\in S} & \frac{\ell(C_{ijk}, \hat{C}_{ijk}) - \omega\bar{\ell}(A_{ijk}, \hat{C}_{ijk})}{z_{ijk}} \\ & + \omega\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K} \bar{\ell}(A_{ijk}, \hat{C}_{ijk}) + \lambda\mathcal{R}(\mathcal{W}),\end{aligned} \tag{37}$$

where (35) and (36) are combined by the doubly robust method [4] in the first two terms, which are respectively termed as the IPS part and the imputation part, and $\bar{\ell}(\cdot)$ is the loss function for the imputation part. Moreover, because the number of $(i,j,k) \notin S$ is huge and the imputed labels A_{ijk} of these non-displayed events are less reliable than observed C_{ijk} of those displayed events in S, a small user-specified parameter ω is applied to balance the two parts. To accurately impute labels of non-displayed events in the imputation part, we utilize the random strategy on a small

proportion of traffic to collect a minor but unbiased subset S_{unbiased} of S. Through solving (11) with S_{unbiased}, we can learn an unbiased imputation model $a(\cdot)$, which outputs $A_{ijk} \equiv a(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k)$ as the imputed labels for non-displayed events. For z_{ijk} in the IPS part, with the tuning of ω, we follow [10, 21] to use a constant.

With the inclusion of non-displayed events, (37) can be seen as a standard classification problem with mnK instances. Because both m and n can be tremendous in real-world systems, any optimization algorithm applied to solve (37) may easily involve a prohibitive cost proportional to $\mathbb{O}(mnK)$. For position-independent systems, the training of fully-labeled problems has been well-studied in recent works (e.g., [20–22]) by imposing some restrictions on the selection of the model function and the loss function $\bar{\ell}(\cdot)$ of the imputation part. These past studies consider a two-dimensional label matrix without the position dimension, but here we have a tensor C. We make an extension by considering the following settings.

- The squared loss is applied for $\bar{\ell}(\cdot)$ as $\bar{\ell}(a, b) = (a - b)^2$
- The CTR model function $g(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{p})$ and the imputation model $a(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{p})$ can be decomposed to multiple parts respectively related to $\boldsymbol{u}$,$\boldsymbol{v}$ or $\boldsymbol{p}$ only.

Under these settings efficient optimization methods can be developed so all operations involving the $\mathbb{O}(mnK)$ cost can be factorized into a series of operations with a smaller $\mathbb{O}(m) + \mathbb{O}(n) + \mathbb{O}(K)$ cost. In our realization for experiments in Section 6, we propose tensor factorization machine (TFM) as both the CTR model $g(\cdot)$ and the imputation model $a(\cdot)$. It is an extension of factorization machines [14]. Given any $\boldsymbol{u}_i \in \mathbb{R}^{D_u}$, $\boldsymbol{v}_j \in \mathbb{R}^{D_v}$ and $\boldsymbol{p}_k \in \mathbb{R}^{D_p}$, TFM finds three embedding matrices $Q_u \in \mathbb{R}^{D_u \times d}$, $Q_v \in \mathbb{R}^{D_v \times d}$, and $Q_p \in \mathbb{R}^{D_p \times d}$ to represent information of request i, ad j and position k as three d-dimensional vectors, $Q_u^T \boldsymbol{u}_i$, $Q_v^T \boldsymbol{v}_j$, and $Q_p^T \boldsymbol{p}_k$, respectively. Here d is a user-specified parameter. Then TFM uses $\mathbf{1}_d$, a vector of all ones, to sum up the component-wise product of the three vectors as the following output value

$$\mathbf{1}_d^T \left((Q_u^T \boldsymbol{u}_i) \odot (Q_v^T \boldsymbol{v}_j) \odot (Q_p^T \boldsymbol{p}_k) \right), \tag{38}$$

where $\odot$ is the Hadamard product. We apply a block coordinate descent method to efficient solve (37) without the $\mathbb{O}(mnK)$ cost; see more details in supplementary materials.

For the offline evaluation, some displayed events in a separate time period should be considered as the test set. For our setting, recall a small portion of traffic is used to collect an unbiased subset through the random strategy. This set can be used as the test set S_{te} to give an unbiased offline evaluation.

4.2 Unbiased Non-positional Approach

The non-positional approach finds $f(\cdot)$ and $h(\cdot)$ that are decomposed from $g(\cdot)$. Naturally we can extract them from an unbiased estimator $g(\cdot)$ obtained through techniques discussed in Section 4.1. However, subsequently we point out a new issue occurred in the non-positional approach.

We mentioned in Section 2.2 that under the decomposition of the probability value in (13), the click random variable $\tilde{C}$ is observed, but neither $\tilde{P}_{ij}$ nor β_k is directly available. A new issue is that even if an unbiased $g(\cdot)$ is obtained, the resulting $f(\cdot)$ can still be a heavily biased estimator of (14). Take the setting (19) to decompose $g(\cdot)$ as an example and assume $f^*(\cdot)$ and $h^*(\cdot)$ are respectively unbiased estimators of (14) and (15). Given any $g(\cdot)$, there exists an arbitrary constant c_1 such that for any $\boldsymbol{u}_i, \boldsymbol{v}_j$, and $\boldsymbol{p}_k$,

$$g(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k) = f^*(\boldsymbol{u}_i, \boldsymbol{v}_j) + h^*(\boldsymbol{p}_k) = (f^*(\boldsymbol{u}_i, \boldsymbol{v}_j) - c_1) + (h^*(\boldsymbol{p}_k) + c_1).$$

If $c_1 \neq 0$, we explain that $f(\cdot) = f^*(\cdot) - c_1$ may be a biased estimator. For any request i, computing (17) by $f^*(\cdot)$ and $f(\cdot)$ respectively as

$$\sigma(f^*(\boldsymbol{u}_i, \boldsymbol{v}_j)) \times \text{bid}_j, \forall j \text{ and } \sigma(f^*(\boldsymbol{u}_i, \boldsymbol{v}_j) - c_1) \times \text{bid}_j, \forall j$$

may lead to different orders of the n ads. Thus results by $f(\cdot)$ are different from $f^*(\cdot)$. In addition, an unbiased $f(\cdot)$ can not be detected through offline evaluation because, from Section 2.2.3, existing non-positional approaches can only evaluate $g(\cdot)$ rather than $f(\cdot)$.

Recall that the goal of the non-positional approach is that through the assumption in (13), we can learn a non-positional CTR model $f(\cdot)$ for an efficient display algorithm. We show that this purpose can be accomplished without requiring $f(\cdot)$ to be an unbiased estimator of $\tilde{P}_{ij}$ in (14). For the explanation, we begin with a simplified scenario so all positions share a positive constant α as the examination probability (15). From (13), the click probability of event (i, j, k) satisfies

$$\alpha \tilde{P}_{ij} = P_{ijk}. \tag{39}$$

Under this setting, for any request i, retrieving K ads with the largest values respectively of

$$\alpha \tilde{P}_{ij} \times \text{bid}_j, \forall j, \text{ and } \tilde{P}_{ij} \times \text{bid}_j, \forall j, \tag{40}$$

leads to the same solution of the assignment problem discussed in Section 2.2. So from clicks/not-clicks we can learn an estimator of P_{ijk} in (39) as the non-positional CTR model $f(\cdot)$. Though $f(\cdot)$ obtained in this way estimates $\alpha \tilde{P}_{ij}$ rather than $\tilde{P}_{ij}$, through (40) it accomplishes the original purpose of the non-positional approach.

Now we switch back to discuss the general scenario, where (39) does not hold and different positions have various examination probabilities β_k. The following theorem (proof is in supplementary materials) shows that an unbiased $f(\cdot)$ can be learnt by solving (34).

Theorem 4.1. *Assume the following conditions hold.*

(1) *A fully-labeled C is avaiable*
(2) *The logistic loss in (7) is applied as the loss function $\ell(\cdot)$*

By considering $\hat{C}_{ijk} = f(\boldsymbol{u}_i, \boldsymbol{v}_j)$ and solving (34), there exists an α such that the resulting $f(\cdot)$ is an unbiased estimator of $\alpha \tilde{P}_{ij}$.

From Theorem 4.1, because $\hat{C}_{ijk} = f(\boldsymbol{u}_i, \boldsymbol{v}_j)$ in solving (37), the function $h(\cdot)$ in the non-positional approach can be ignored. For the offline evaluation, we follow the setting in Section 4.1 to have a small portion of traffic served by the random strategy to compose an unbiased subset as S_{te}. We prove in supplementary materials that for any non-positional CTR model $f(\boldsymbol{u}, \boldsymbol{v})$ to be evaluated, the obtained LogLoss and AUC on S_{te} are unbiased metrics of estimating (39).

5 RELATED WORKS

Besides the several works on CTR prediction and the selection bias reviewed earlier, in supplementary materials we discuss a related topic "unbiased learning-to-rank (LTR) from clicks" [8, 19]. However, due to the different roles of position and the difference between ranking learning and point-wise learning, their ideas and derived approaches are quite different from ours.

6 EXPERIMENTS

In this section, after presenting our experiment setup including datasets, evaluation criteria and parameter selection, we conduct a series of experiments to investigate the following research questions (RQs).

• RQ1: How do the retrieval bias and the placement bias degrade the performance of CTR prediction for position-aware systems?
• RQ2: Can our proposed unbiased positional approach accurately estimate (4)?
• RQ3: Can our proposed unbiased non-positional approach accurately estimate (39)?
• RQ4: Can our proposed unbiased offline evaluation for the non-position approachal give accurate performance estimation on non-positional CTR models?

6.1 Experiment Setup

We consider two data sets for experiments. They are generated from the "Outbrain Click Prediction" competition,[1] and "KKBox's Music Recommendation Challenge" competition.[2] From the original set we construct a fully-observed set with instances in three time windows. For Outbrain, we select 8.4M requests and 300 ads. For KKBox, we select 2.6M requests and 100 ads. Details of data preprocessing is in supplementary materials.

To get the training set S, similar to practical CTR systems, we use a model built on data in the first window to select events in the second window. From the discussion in Sections 2.2, 3.2 and 4, for each request, we select K events by the following strategies .

• **Greedy**: This strategy, described in Section 2.2, retrieves the top-K ads according to the expected revenue. For the placement, the retrieved K ads are orderly placed in K positions; see Section 3.2. The generated S carries both the retrieval bias and placement bias.
• **RG**: This strategy randomly retrieves K ads and orderly places them according to the expected revenue. From Section 3.2, the generated S solely carries the placement bias, as the retrieval bias has been eliminated by the uniform (32).
• **GR**: In contrast to **RG**, this strategy retrieves the top-K ads with the highest expected revenue but places them in random positions. From Section 3.2, this setting eliminates the placement bias by the uniform (33), so the generated S solely carries the retrieval bias.
• **Random**: This strategy randomly retrieves K ads and places them in random positions. Because both (32) and (33) are uniform, the generated S is unbiased.
• ϵ-**greedy**: This strategy is a compromise of **Greedy** and **Random** introduced in Section 4.1. For most requests, ads are generated by the **Greedy** strategy, while for a small ratio ϵ of requests, we use the **Random** strategy to form an unbiased S_{unbiased} as a subset of S. For ϵ, we consider 0.01 and 0.1.

Next we discuss the generation of labels (clicks and not-clicks) in S, which now simulates a position-aware system. We consider

$$C_{ijk} = Y_{ij}E_{ijk}, \quad \forall (i,j,k) \in S, \tag{41}$$

where Y_{ij} is the ground-truth observation of $\tilde{P}_{ij}$ in (14), and under the given (i,j), E_{ijk} is an observation from $B(\beta_k)$. In our experiments, we set $\beta_k = 0.5^{k-1}$. From C_{ijk}, we can split S into S^+ and S^-. Finally, by the random strategy described above and the C_{ijk} setting in (41), we construct and unbiased test set S_{te} from events in the last time window, which are independent from the training set S.

[1] https://www.kaggle.com/c/outbrain-click-prediction
[2] https://www.kaggle.com/c/kkbox-music-recommendation-challenge

Table 2: Data statistics.

Data Set	Outbrain		KKBox	
	$\lvert S^+\rvert$	$\lvert S^-\rvert$	$\lvert S^+\rvert$	$\lvert S^-\rvert$
Greedy	1.34M	83.70M	0.28M	25.54M
RG	0.19M	84.85M	0.11M	25.71M
GR	0.68M	84.36M	0.17M	25.65M
Random	0.06M	84.98M	0.05M	25.77M
0.01-greedy	1.32M	83.71M	0.28M	25.54M
0.1-greedy	1.21M	83.83M	0.25M	25.57M

Besides the S and S_{te} sets generated above, to evaluate results in Section 4.2 on non-positional approaches, we prepare another version of S and S_{te} as follows.

• $S(\alpha)$: We select events by the **Random** strategy. For the generation of label C_{ijk}, the setting in (41) is applied but positions are assumed to share the same examination probability α. Thus E_{ijk} is the observation from $B(\alpha)$ and α is computed by $\frac{1}{K}\sum_{k=1}^{K}\beta_k$.
• $S_{\text{te}}(\alpha)$: By the same setting for $S(\alpha)$, we select events by the **Random** strategy and apply E_{ijk}, the observation from $B(\alpha)$, to generate C_{ijk}.

The statistics of the generated S sets are in Table 2. Note that $|S^+|$, the number of clicked events in S, significantly vary according to the strategy to retrieve and place ads. For example, S^+ of the **Greedy** strategy is much larger than **Random**. The reason is that from the prior knowledge by the previous model, the **Greedy** strategy tends to select events with a high click probability, while the **Random** strategy randomly selects events without considering their click probabilities. For the similar reason, though the **RG** and **GR** strategies employ randomization solely in the retrieval or placement procedure, their $|S^+|$ are much smaller than **Greedy**. Note that as $|S^+|$ also reflects the revenue gained during collecting S, the gap between $|S^+|$ collected by **Greedy** and $|S^+|$ by other strategies can be seen as the revenue loss caused by employing randomization. Clearly, while **Random** (or **RG**, **GR**) leads to an unbiased or (less biased) model for future predictions, the revenue loss in the current time period may be huge. In contrast, for the strategy ϵ-**greedy**, through the collection and the effective use of a small unbiased subset S_{unbiased}, the revenue loss due to randomization is negligible.

For evaluation criteria, we consider LogLoss and AUC respectively defined in (9) and (10). For parameter selection, details are in supplementary materials. We finally train a model with S by using the selected parameters, and report LogLoss and AUC by evaluating S_{te}.

6.2 Deep Understanding of Two Types of Selection Bias through Visualization

Details of this analysis are in supplementary materials.

6.3 Comparison on Various Positional Approaches

To investigate RQ2, we compare various positional approaches, where all of them consider the TFM in (38) as the positional CTR

model $g(\cdot)$. We group these approaches into two main categories. The first category includes four approaches solving (11) with different biased training set S. According to how S is generated in Section 6.1, we term them respectively as TFM-Greedy, TFM-RG, TFM-GR, and TFM-EE (ϵ). The second category includes approaches in Section 4.1 that aim to eliminate biases.

• TFM-Random: This approach simply solves (11) using an unbiased S generated by the **Random** strategy.

• TFM-CF (ϵ): This approach is a realization of our proposed framework in (37). The set S collected by the ϵ-**greedy** strategy is used so that it includes a small unbiased subset S_{unbiased}. The imputation model $a(\cdot)$ is obtained by solving (11) with the TFM model on the subset S_{unbiased}. We then have $A_{ijk} = a(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{p}_k)$ and solve (37) to obtained a CTR model. For the propensity score z_{ijk}, we follow [21] to simply set $z_{ijk} = 1$.

After the above models are trained, we evaluate them on the test set S_{te}; see details in Section 6.1. By comparing results of different positional approaches in Table 3, we have the following observations.

• The approach TFM-Greedy, which learns $g(\cdot)$ with S carrying both the retrieval and the placement biases, performs the worst. This observation confirms our conjecture that the two types of selection bias significantly degrade the performance of the learnt $g(\cdot)$.

• For TFM-RG and TFM-GR, through respectively eliminating the retrieval bias and the placement bias, their performance has improved. In particular, the improvement of TFM-RG is more remarkable. The reason is as we mentioned in Section 6.2, the overestimation effect caused by the retrieval bias is more dominant.

• The approach TFM-Random performs the best. The reason is as we mentioned in Section 4.1, learning from the unbiased S leads $g(\cdot)$ to be an unbiased estimator of (4).

• For TFM-EE (ϵ) and TFM-CF (ϵ), by including a small but unbiased subset S_{unbiased} in S collected by ϵ-**greedy**, their performance is significantly better than TFM-Greedy. With the increase of the ratio ϵ, the improvement enlarges. Under the same ϵ, our proposed TFM-CF (ϵ) is remarkably better than TFM-EE (ϵ). The reason is that besides displayed events in S, TFM-CF (ϵ) incorporates non-displayed events by imputing their labels.

• The performance of TFM-CF (ϵ) is almost as good as the best approaches TFM-Random and TFM-RG. However, as we mentioned in Section 6.1, the revenue loss is much smaller due to employing randomization on only a small subset S_{unbiased}. By trading off the performance improvement against the revenue loss, TFM-CF (ϵ) is the most competitive approach.

6.4 Comparison on Various Approaches for Learning Non-positional CTR Models

We investigate RQ3 through comparing various approaches for learning non-positional CTR models. Following the discussion in Section 2.2.2, the CTR model $f(\cdot)$ is now non-positional. Thus the TFM model considered earlier should be reduced to standard factorization machine (FM) [14], but here we consider an extension called the field-aware FM (FFM) [9, 22]. We group non-positional approaches into three categories. The first category includes an ideal approach.

Table 3: A comparison of test scores by using different positional approaches. The best approach is bold-faced.

Data Set	KKBox		Outbrain	
Metric	LL	AUC	LL	AUC
TFM-Greedy	0.049	0.779	0.102	0.766
TFM-RG	0.012	0.869	0.005	0.915
TFM-GR	0.025	0.765	0.062	0.807
TFM-EE (0.01)	0.018	0.749	0.011	0.839
TFM-EE (0.1)	0.013	0.854	0.005	0.875
TFM-Random	**0.012**	**0.875**	**0.004**	**0.948**
TFM-CF (0.01)	0.013	0.858	0.005	0.888
TFM-CF (0.1)	0.013	0.864	0.005	0.918

• FFM-Ideal: This approach learns $f(\cdot)$ through solving (6) with $S(\alpha)$. As mentioned in Section 6.1, because $S(\alpha)$ has the ground-truth of (39) and is free of selection biases, the performance of the learnt $f(\cdot)$ can be considered as the optimal benchmark.

The scecond category includes two existing approaches reviewed in Section 2.2.2. They extract $f(\cdot)$ from a customized $g(\cdot)$ learnt through solving (11) by the following settings.

• FFM-Greedy-A: This approach follows [11, 23] to consider the combination in (19).

• FFM-Greedy-P: This approach follows [6] to consider the combination in (18).

For $h(\cdot)$, we apply $h(\boldsymbol{p}_k) = b_k$ for both approaches, where b_k is a real-valued parameter of position k. The training set S is the one collected by the **Greedy** strategy because that is the standard setting ignoring the issue of selection biases. The third category includes unbiased approaches discussed in Section 4.2.

• FFM-Random: By using S collected by the **Random** strategy, this approach obtains an unbiased estimator of (39) through solving (6).

• FFM-CF (ϵ): By using S collected under the **Random** strategy, this is a realization of our proposed framework in (37), where S is collected by ϵ-**greedy** to have a small unbiased subset S_{unbiased}. For learning the imputation model $a(\cdot)$, we consider the FFM and solve (6) by using the unbiased subset S_{unbiased}. We then have $A_{ijk} \equiv a(\boldsymbol{u}_i, \boldsymbol{v}_j)$ and solve (37) with $\hat{C}_{ijk} = f(\boldsymbol{u}_i, \boldsymbol{v}_j)$. Similar to the TFM-CF (ϵ) approach in Section 6.3, we set $z_{ijk} = 1$ in (37).

While non-positional approaches were introduced to address the impracticability of positional approaches on large-scale position-aware systems, it is important to check if they are really superior to systems without considering position information at all. To this end, we include the following position-independent systems as a baseline for comparison: FFM-Greedy, and FFM-EE (ϵ). They vary according to how S is collected. The model function $y(\boldsymbol{u}, \boldsymbol{v})$ is learned by solving (6).

For evaluation, we consider two test sets S_{te} and $S_{\text{te}}(\alpha)$; see details of these sets in Section 6.1. A purpose is to check for RQ4 whether results on S_{te}, which are obtained in practice, are consistent with those by using $S_{\text{te}}(\alpha)$ that has the ground truth of (39).

From the results presented in Table 4, we have the following observations.

• By comparing the scores on S_{te} and $S_{\text{te}}(\alpha)$, the small difference justifies the result in Section 4.2 that evaluating LogLoss and AUC on S_{te} can an unbiased evaluation of estimating (39).

• The bad performance of FFM-Greedy, which simulates the standard setting of position-independent systems, confirms the need to

Table 4: A performance comparison of different approaches of estimating (14). We show scores of evaluating on two test sets S_{te}, and $S_{\text{te}}(\alpha)$. The best approach is bold-faced.

Data Set	KKbox				Outbrain			
Test Set	S_{te}		$S_{\text{te}}(\alpha)$		S_{te}		$S_{\text{te}}(\alpha)$	
Metric	LL	AUC	LL	AUC	LL	AUC	LL	AUC
FFM-Ideal	**0.014**	**0.726**	**0.014**	**0.729**	**0.004**	**0.914**	**0.004**	**0.914**
FFM-Greedy-A	0.057	0.449	0.057	0.448	0.213	0.615	0.213	0.615
FFM-Greedy-P	0.102	0.472	0.102	0.473	0.155	0.583	0.155	0.584
FFM-Random	**0.014**	**0.729**	**0.014**	**0.733**	**0.004**	**0.912**	**0.004**	**0.912**
FFM-CF (0.01)	0.015	0.634	0.015	0.642	0.006	0.780	0.006	0.785
FFM-CF (0.1)	0.014	0.677	0.014	0.682	0.005	0.865	0.005	0.864
Baseline: position-independent systems								
FFM-Greedy	0.028	0.493	0.028	0.493	0.046	0.611	0.046	0.612
FFM-EE (0.1)	0.016	0.531	0.016	0.530	0.008	0.683	0.008	0.682
FFM-EE (0.1)	0.015	0.585	0.015	0.581	0.006	0.767	0.006	0.766

consider the position information and the selection bias.

• Compared to FFM-Greedy, the improvement of FFM-Greedy-A and FFM-Greedy-P is minor. From the discussion in Section 3, the reason is that because the selection bias leads to a biased $g(\cdot)$, the extracted $f(\cdot)$ is also biased and gives poor predictions.

• The approach FFM-Ideal and FFM-Random are the best. The performance of FFM-Ideal is intuitive, as it directly learns $f(\cdot)$ from the unbiased $S(\alpha)$ containing ground-truth of (39). The small difference between FFM-Random and FFM-Ideal confirms our result given in Theorem 4.1 that the $f(\cdot)$ obtained by the unbiased solution of the problem (34) is an unbiased estimator of (39).

• For FFM-CF (ϵ) and FFM-EE (ϵ) , leveraging the unbiased small subset S_{unbiased} leads to competitive performances. By incorporating the non-displayed events through imputing their labels, FFM-CF (ϵ) performs significantly better than FFM-EE (ϵ) under the same ϵ. This observation is consistent with the comparison between TFM-EE (ϵ) and TFM-CF (ϵ) in Section 6.3.

• Finally, as we have commented on the approach TFM-CF (ϵ) in Section 4.1, the revenue loss of FFM-CF (ϵ) is much smaller than FFM-Random because of considering only a small subset to employ randomization. Thus our proposed FFM-CF (ϵ) approach is practically useful for position-aware systems.

7 CONCLUSION

In this work, we focus on CTR prediction for position-aware advertising systems, where because an ad placed in various positions has different click probabilities, the position information should be considered in both training and prediction. Existing approaches for position-aware systems broadly fall into two categories: positional approach and non-positional approach, which learn CTR models from clicks/not-clicks on historically displayed events by leveraging the position information in different ways. However, through pointing out the following issues, we argue that these existing approaches are biased.

• For each request, some ads are non-uniformly retrieved and then these retrieved ads are non-uniformly placed in positions. This causes that two types of selection biases, the retrieval bias and the placement bias, coexist in displayed events.

• For the non-positional approaches attempting to estimate (14), CTR models are learned and evaluated indirectly. An additional bias may occur.

To address the above issues, we derive a novel counterfactual learning framework. We prove that under the proposed framework, for both positional and non-positional approaches, learning and evaluation are unbiased. Experiments confirm both our analysis on selection biases and the effectiveness of our proposed counterfactual learning framework.

REFERENCES

[1] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X. Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual reasoning and learning systems: The example of computational advertising. *JMLR* 14 (2013), 3207–3260.

[2] Olivier Chapelle, Eren Manavoglu, and Romer Rosales. 2015. Simple and scalable response prediction for display advertising. *ACM TIST* 5, 4 (2015), 61:1–61:34.

[3] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. 2019. Top-k off-policy correction for a REINFORCE recommender system. In *WSDM*.

[4] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly robust policy evaluation and learning. In *ICML*.

[5] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *IJCAI*.

[6] Huifeng Guo, Jinkai Yu, Qing Liu, Ruiming Tang, and Yuzhou Zhang. 2019. PAL: a position-bias aware learning framework for CTR prediction in live recommender systems. In *RecSys*.

[7] Daniel G. Horvitz and Donovan J. Thompson. 1952. A Generalization of Sampling Without Replacement From a Finite Universe. *J. Amer. Statist. Assoc.* 47 (1952), 663–685.

[8] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *WSDM*.

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

[10] Carolin Lawrence, Artem Sokolov, and Stefan Riezler. 2017. Counterfactual Learning from Bandit Feedback under Deterministic Logging: A Case Study in Statistical Machine Translation. In *EMNLP*.

[11] Xiaoliang Ling, Weiwei Deng, Chen Gu, Hucheng Zhou, Cui Li, and Feng Sun. 2017. Model ensemble for click prediction in bing search ads. In *WWW*.

[12] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Ji Yang, Minmin Chen, Jiaxi Tang, Lichan Hong, and Ed H Chi. 2020. Off-policy Learning in Two-stage Recommender Systems. In *WWW*.

[13] H. Brendan McMahan, Gary Holt, D. Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hrafnkelsson, Tom Boulos, and Jeremy Kubica. 2013. Ad Click Prediction: a View from the Trenches. In *KDD*.

[14] Steffen Rendle. 2010. Factorization machines. In *ICDM*.

[15] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ADs. In *WWW*.

[16] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations As Treatments: Debiasing Learning and Evaluation. In *ICML*.

[17] Harald Steck. 2010. Training and Testing of Recommender Systems on Data Missing Not at Random. In *KDD*.

[18] Adith Swaminathan and Thorsten Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *JMLR* 16 (2015), 1731–1755.

[19] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *SIGIR*.

[20] Hsiang-Fu Yu, Hsin-Yuan Huang, Inderjit S. Dihillon, and Chih-Jen Lin. 2017. A Unified Algorithm for One-class Structured Matrix Factorization with Side Information. In *AAAI*.

[21] Bowen Yuan, Jui-Yang Hsia, Meng-Yuan Yang, Hong Zhu, Chihyao Chang, Zhenhua Dong, and Chih-Jen Lin. 2019. Improving Ad Click Prediction by Considering Non-displayed Events. In *CIKM*.

[22] Bowen Yuan, Meng-Yuan Yang, Jui-Yang Hsia, Hong Zhu, Zhirong Liu, Zhenhua Dong, and Chih-Jen Lin. 2019. *One-class Field-aware Factorization Machines for Recommender Systems with Implicit Feedbacks.* Technical Report. National Taiwan Univ.

[23] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. 2019. Recommending what video to watch next: a multitask ranking system. In *RecSys*.

Supplementary Materials for "Unbiased Ad Click Prediction for Position-aware Advertising Systems"

A EFFICIENT TRAINING OF TENSOR FACTORIZATION MACHINE

As we mentioned in Section 4.1, to solve (37) without $\mathbb{O}(mnK)$ costs, the CTR model and imputation model should satisfy some conditions. In this section, we take the TFM model defined in (38) as an example to present our efficient training algorithm. Specifically, we consider the TFM model as both CTR and imputation models, which is a multi-block convex function. Therefore, we can apply block coordinate descent (CD) method to solve (37). At each cycle of the block CD method, we sequentially solve convex sub-problems of Q_u, Q_v and Q_p. The overall procedure is summarized in Algorithm 1.

If we consider to update the block Q_u, then the convex sub-problem is to solve

$$\min_{Q_u} \sum_{(i,j,k)\in S} \frac{\ell(C_{ijk}, \hat{C}_{ijk}) - \omega(A_{ijk} - \hat{C}_{ijk})^2}{z_{ijk}} + \omega \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}(A_{ijk} - \hat{C}_{ijk})^2 + \frac{\lambda}{2}\|Q_u\|_{\mathrm{F}}^2, \tag{A.1}$$

where $l2$ regularization is considered such that $R(Q_u) = \|Q_u\|_{\mathrm{F}}^2/2$. The objective function in (A.1) is a general convex function, so some differentiable optimization techniques such as gradient descent, Quasi-Newton or Newton are needed. All these methods must calculate the gradient (i.e., the first derivative) or the Hessian-vector product (where Hessian is the second-order derivative), but the $\mathbb{O}(mnK)$ cost is the main concern.

Following [20–22], for easy analysis, we write (A.1) to a vector form

$$f(\tilde{\boldsymbol{w}}),$$

where $\tilde{\boldsymbol{w}} = \mathrm{vec}(Q_u)$ and $\mathrm{vec}(\cdot)$ stacks the columns of a matrix. We consider Newton methods to solve the sub-problem.

An iterative procedure is conducted in a Newton method. Each Newton iteration minimizes a second-order approximation of the function to obtain a updating direction $\boldsymbol{s}$

$$\min_{\boldsymbol{s}} \nabla f(\tilde{\boldsymbol{w}})^T \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T \nabla^2 f(\tilde{\boldsymbol{w}})\boldsymbol{s}. \tag{A.2}$$

Because $f(\tilde{\boldsymbol{w}})$ is convex, the direction $\boldsymbol{s}$ can be obtained by solving the following linear system

$$\nabla^2 f(\tilde{\boldsymbol{w}})\boldsymbol{s} = -\nabla f(\tilde{\boldsymbol{w}}). \tag{A.3}$$

We follow [20, 22] to solve (A.2) by an iterative procedure called the conjugate gradient method (CG). The main reason of using CG is that $\nabla^2 f(\tilde{\boldsymbol{w}})$ is too large to be stored, and CG addresses this issue by that each CG step mainly involves the product between $\nabla^2 f(\tilde{\boldsymbol{w}})$ and a vector $\tilde{\boldsymbol{v}}$

$$\nabla^2 f(\tilde{\boldsymbol{w}})\tilde{\boldsymbol{v}}, \tag{A.4}$$

which, with the problem structure, may be conducted without explicitly forming $\nabla^2 f(\tilde{\boldsymbol{w}})$.

To ensure the convergence of the Newton method, after obtaining an updating direction $\boldsymbol{s}$, a line search procedure is conducted to find a suitable step size θ. Then we update the $\tilde{\boldsymbol{w}}$ by

$$\tilde{\boldsymbol{w}} \leftarrow \tilde{\boldsymbol{w}} + \theta \boldsymbol{s}.$$

We follow the standard backtracking line search to check a sequence $1, \beta, \beta^2, \cdots$ and choose the largest θ such that the sufficient decrease of the function value is obtained

$$f(\tilde{\boldsymbol{w}} + \theta\boldsymbol{s}) - f(\tilde{\boldsymbol{w}}) \le \theta\nu\nabla f^T(\tilde{\boldsymbol{w}})\boldsymbol{s}. \tag{A.5}$$

where $\beta, \nu \in (0, 1)$ are pre-specified constants.

A.1 Algorithm Details

To discuss techniques for addressing the issue of $\mathbb{O}(mnK)$ complexity, let

$$\tilde{\boldsymbol{w}} = \mathrm{vec}(Q_u)$$

and re-write (A.1) as

$$f(\tilde{\boldsymbol{w}}) = \frac{\lambda}{2}\|\tilde{\boldsymbol{w}}\|_2^2 + L^+(\tilde{\boldsymbol{w}}) + 2\omega L^-(\tilde{\boldsymbol{w}}), \tag{A.6}$$

where

$$L^+(\tilde{\boldsymbol{w}}) = \sum_{(i,j,k)\in S} \frac{1}{z_{ijk}}(\ell(C_{ijk}, \hat{C}_{ijk}) - \omega(A_{ijk} - \hat{C}_{ijk})^2) \tag{A.7}$$

$$L^-(\tilde{\boldsymbol{w}}) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}(A_{ijk} - \hat{C}_{ijk})^2.$$

Then the gradient and Hessian-vector product can be respectively re-written as

$$\nabla\tilde{f}(\tilde{\boldsymbol{w}}) = \lambda\tilde{\boldsymbol{w}} + \nabla L^+(\tilde{\boldsymbol{w}}) + 2\omega\nabla L^-(\tilde{\boldsymbol{w}}), \text{ and}$$
$$\nabla^2\tilde{f}(\tilde{\boldsymbol{w}})\tilde{\boldsymbol{v}} = \lambda\tilde{\boldsymbol{v}} + \nabla^2 L^+(\tilde{\boldsymbol{w}})\tilde{\boldsymbol{v}} + 2\omega\nabla^2 L^-(\tilde{\boldsymbol{w}})\tilde{\boldsymbol{v}}.$$

A.1.1 The Computation of $\nabla\tilde{f}(\tilde{\boldsymbol{w}})$. Let $\boldsymbol{w}_i = Q_u^T\boldsymbol{u}_i$, $\boldsymbol{h}_j = Q_v^T\boldsymbol{v}_j$ and $\boldsymbol{z}_k = Q_p^T\boldsymbol{p}_k$. When updating $\mathrm{vec}(Q_u)$, the value $\hat{C}_{ijk}$ can be written as

$$\hat{C}_{ijk} = \mathbf{1}_d^T\left((Q_u^T\boldsymbol{u}_i)\odot(Q_v^T\boldsymbol{v}_j)\odot(Q_p^T\boldsymbol{p}_k)\right) = (\boldsymbol{u}_i^T Q_u)(\boldsymbol{h}_j\odot\boldsymbol{z}_k)$$
$$= \mathrm{vec}(Q_u)^T(\boldsymbol{u}_i\otimes(\boldsymbol{h}_j\odot\boldsymbol{z}_k)) = \tilde{\boldsymbol{w}}^T\mathrm{vec}(\boldsymbol{u}_i(\boldsymbol{h}_j\odot\boldsymbol{z}_k)^T), \tag{A.8}$$

and its derivative $\hat{C}'_{ijk}$ with respect to $\tilde{\boldsymbol{w}}$ is $\mathrm{vec}(\boldsymbol{u}_i(\boldsymbol{h}_j\odot\boldsymbol{z}_k)^T)$. Then $\nabla L^+(\tilde{\boldsymbol{w}})$ can be computed as

$$\sum_{(i,j,k)\in S}\frac{1}{z_{ijk}}(\ell'(C_{ijk}, \hat{C}_{ijk}) - 2\omega(\hat{C}_{ijk} - A_{ijk}))\hat{C}'_{ijk}$$
$$= \mathrm{vec}\Big(\sum_{(i,j,k)\in S}\frac{1}{z_{ijk}}(\ell'(C_{ijk}, \hat{C}_{ijk}) - 2\omega(\hat{C}_{ijk} - A_{ijk}))(\boldsymbol{u}_i(\boldsymbol{h}_j\odot\boldsymbol{z}_k)^T)\Big), \tag{A.9}$$

where $\ell'(C, \hat{C})$ indicates the derivative with respect to $\hat{C}$, and $\nabla L^-(\tilde{\boldsymbol{w}})$ can be computed as

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}(\hat{C}_{ijk} - A_{ijk})\hat{C}'_{ijk}$$
$$= \mathrm{vec}\Big(\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K}(\hat{C}_{ijk} - A_{ijk})(\boldsymbol{u}_i(\boldsymbol{h}_j\odot\boldsymbol{z}_k)^T)\Big). \tag{A.10}$$

```
Given initial Q_u, Q_v and Q_p;
while stopping condition is not satisfied do
    Update Q_u via solving (37) by fixing Q_v and Q_p;
    Update Q_v via solving (37) by fixing Q_u and Q_p;
    Update Q_p via solving (37) by fixing Q_u and Q_v;
end
```

Algorithm 1: Block CD method for solving (37)

The $\nabla L^+(\tilde{\boldsymbol{w}})$ evaluation in (A.9) requires the summation of $\mathbb{O}(|S|)$ terms. With $|S| \ll mnK$, $\nabla L^+(\tilde{\boldsymbol{w}})$ can be easily calculated but the bottleneck is on $\nabla L^-(\tilde{\boldsymbol{w}})$, which sums up $\mathbb{O}(mnK)$ terms.

In order to deal with the $\mathbb{O}(mnK)$ cost in (A.10), we develop an efficient computation for $\nabla L^-(\tilde{\boldsymbol{w}})$ as follows. From (A.8),

$$\begin{aligned}
&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \boldsymbol{u}_i(\hat{C}_{ijk} - A_{ijk})(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T \\
=&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \boldsymbol{u}_i(\boldsymbol{w}_i^T(\boldsymbol{h}_j \odot \boldsymbol{z}_k) - A_{ijk})(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T \\
=&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \boldsymbol{u}_i(\boldsymbol{w}_i^T(\boldsymbol{h}_j \odot \boldsymbol{z}_k))(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T \qquad \text{(A.11)} \\
&- \sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K A_{ijk}\boldsymbol{u}_i(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T. \qquad \text{(A.12)}
\end{aligned}$$

Let

$$U = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m]^T \in \mathbb{R}^{m \times D_u}, W = [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_m]^T \in \mathbb{R}^{m \times d},$$
$$H = [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_n]^T \in \mathbb{R}^{n \times d} \text{ and } Z = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_K]^T \in \mathbb{R}^{K \times d}.$$

The term in (A.11) can be computed as

$$\begin{aligned}
&\sum_{i=1}^m (\boldsymbol{u}_i\boldsymbol{w}_i^T)(\sum_{j=1}^n \sum_{k=1}^K (\boldsymbol{h}_j \odot \boldsymbol{z}_k)(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T) \\
=&(U^TW)((H^TH) \odot (Z^TZ)), \qquad \text{(A.13)}
\end{aligned}$$

which can be computed within $\mathbb{O}(\text{nnz}(U)d + (m+n+K)d^2)$. For the term in (A.12), because the TFM imputation model is used, the imputed label have an identical structure with $\hat{C}_{ijk}$, which can be expressed as

$$A_{ijk} = \hat{\boldsymbol{w}}_i^T(\hat{\boldsymbol{h}}_j \odot \hat{\boldsymbol{z}}_k) \qquad \text{(A.14)}$$

Let

$$\hat{W} = [\hat{\boldsymbol{w}}_1, \cdots, \hat{\boldsymbol{w}}_m]^T, \hat{H} = [\hat{\boldsymbol{h}}_1, \cdots, \hat{\boldsymbol{h}}_n]^T \text{ and } \hat{Z} = [\hat{\boldsymbol{z}}_1, \cdots, \hat{\boldsymbol{z}}_K]^T.$$

By substituting (A.14), the term in (A.12) can be computed as

$$\begin{aligned}
&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \hat{\boldsymbol{w}}_i^T(\hat{\boldsymbol{h}}_j \odot \hat{\boldsymbol{z}}_k)\boldsymbol{u}_i(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T \\
=&\sum_{i=1}^m (\boldsymbol{u}_i\hat{\boldsymbol{w}}_i^T) \sum_{j=1}^n \sum_{k=1}^K (\hat{\boldsymbol{h}}_j \odot \hat{\boldsymbol{z}}_k)(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T) \\
=&(U^T\hat{W})((\hat{H}^TH) \odot (\hat{Z}^TZ)), \qquad \text{(A.15)}
\end{aligned}$$

which can be computed within $\mathbb{O}(\text{nnz}(U)d + (m+n+K)d^2)$.

A.1.2 The Computation of $\nabla^2 \tilde{f}(\tilde{\boldsymbol{w}})\tilde{\boldsymbol{v}}$. From (A.9), we give the derivation of $\nabla^2 L^+(\tilde{\boldsymbol{w}})$ as

$$\begin{aligned}
&\nabla\left[\sum_{(i,j,k)\in S} \frac{1}{z_{ijk}}(\ell'(C_{ijk}, \hat{C}_{ijk}) - 2\omega(\hat{C}_{ijk} - A_{ijk}))\hat{C}'_{ijk}\right] \\
=&\sum_{(i,j,k)\in S} \frac{1}{z_{ijk}}(\ell''(C_{ijk}, \hat{C}_{ijk}) - \omega)\hat{C}'_{ijk}(\hat{C}'_{ijk})^T,
\end{aligned}$$

where $\ell''(C, \hat{C})$ indicates the second derivative with respect to $\hat{C}$. Let $M \in \mathbb{R}^{D_u \times d}$ be a matrix satisfying $\text{vec}(M) = \tilde{\boldsymbol{v}}$. Then $\nabla^2 L^+(\tilde{\boldsymbol{w}})\text{vec}(M)$ can be computed as

$$\begin{aligned}
&\sum_{(i,j,k)\in S} \frac{1}{z_{ijk}}(\ell''(C_{ijk}, \hat{C}_{ijk}) - \omega)\hat{C}'_{ijk}(\hat{C}'_{ijk})^T\text{vec}(M) \\
=&\sum_{(i,j,k)\in S} \frac{1}{z_{ijk}}(\ell''(C_{ijk}, \hat{C}_{ijk}) - \omega)\hat{C}'_{ijk}(\boldsymbol{u}_i^TM)(\boldsymbol{h}_j \odot \boldsymbol{z}_k) \\
=&\sum_{(i,j,k)\in S} \frac{1}{z_{ijk}}\text{vec}\Big[\ell''(C_{ijk}, \hat{C}_{ijk}) - \omega) \\
&\qquad \boldsymbol{u}_i(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T((\boldsymbol{u}_i^TM)(\boldsymbol{h}_j \odot \boldsymbol{z}_k))\Big], \qquad \text{(A.16)}
\end{aligned}$$

Analogous to (A.9), because $\nabla^2 L^+(\tilde{\boldsymbol{w}})\tilde{\boldsymbol{v}}$ only requires the summation of $\mathbb{O}(|S|)$ cost, it can be easily calculated without any $\mathbb{O}(mnK)$ cost.

The derivation of $\nabla^2 L^-(\tilde{\boldsymbol{w}})$ is

$$\begin{aligned}
&\nabla\left[\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K (\hat{C}_{ijk} - A_{ijk})\hat{C}'_{ijk}\right] \\
=&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \hat{C}'_{ijk}(\hat{C}'_{ijk})^T.
\end{aligned}$$

For $\nabla^2 L^-(\tilde{\boldsymbol{w}})\tilde{\boldsymbol{v}}$, to avoid the $\mathbb{O}(mnK)$ cost, we develop the following computation.

$$\begin{aligned}
&\nabla^2 L^-(\tilde{\boldsymbol{w}})\text{vec}(M) \\
=&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \hat{C}'_{ijk}(\hat{C}'_{ijk})^T\text{vec}(M) \\
=&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \hat{C}'_{ijk}((\boldsymbol{u}_i^TM)(\boldsymbol{h}_j \odot \boldsymbol{z}_k)) \\
=&\sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^K \text{vec}(\boldsymbol{u}_i(\boldsymbol{u}_i^TM)(\boldsymbol{h}_j \odot \boldsymbol{z}_k)(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T) \\
=&\text{vec}(\sum_{i=1}^m (\boldsymbol{u}_i(\boldsymbol{u}_i^TM) \sum_{j=1}^n \sum_{k=1}^K (\boldsymbol{h}_j \odot \boldsymbol{z}_k)(\boldsymbol{h}_j \odot \boldsymbol{z}_k)^T) \\
=&\text{vec}(U^T((UM)((H^TH) \odot (Z^TZ)))), \qquad \text{(A.17)}
\end{aligned}$$

which can be computed within $\mathbb{O}(\text{nnz}(U)d + (m+n+K)d^2)$.

A.1.3 Line Search. To check the sufficient decrease condition in (A.5), for the current iterate $\tilde{w}$ and a direction $\boldsymbol{s}$, we must calculate

$$f(\tilde{\boldsymbol{w}} + \theta\boldsymbol{s}) - f(\tilde{\boldsymbol{w}}). \qquad \text{(A.18)}$$

Because

$$\frac{1}{2}\lambda\|\tilde{\boldsymbol{w}}\| + L^-(\tilde{\boldsymbol{w}})$$

is quadratic, (A.18) is equivalent to

$$\begin{aligned}
&L^+(\tilde{\boldsymbol{w}} + \theta\boldsymbol{s}) - L^+(\tilde{\boldsymbol{w}}) + \theta\nabla(\frac{1}{2}\lambda\|\tilde{\boldsymbol{w}}\|^2 + L^-(\tilde{\boldsymbol{w}}))^T\boldsymbol{s} \\
&\quad + \frac{1}{2}\theta^2\boldsymbol{s}^T\nabla^2(\frac{1}{2}\lambda\|\tilde{\boldsymbol{w}}\|^2 + L^-(\tilde{\boldsymbol{w}}))\boldsymbol{s} \\
=&L^+(\tilde{\boldsymbol{w}} + \theta\boldsymbol{s}) - L^+(\tilde{\boldsymbol{w}}) + \theta(\lambda\tilde{\boldsymbol{w}} + \nabla L^-(\tilde{\boldsymbol{w}}))^T\boldsymbol{s} \\
&\quad + \frac{1}{2}\theta^2\boldsymbol{s}^T(\lambda I + \nabla^2 L^-(\tilde{\boldsymbol{w}}))\boldsymbol{s}.
\end{aligned} \qquad \text{(A.19)}$$

The value $L^+(\tilde{\boldsymbol{w}})$ is available from the previous iteration. We then calculate the following two values so that (A.19) can be easily obtained for any θ.

$$(\lambda\tilde{\boldsymbol{w}} + \nabla L^-(\tilde{\boldsymbol{w}}))^T\boldsymbol{s} \qquad \text{(A.20)}$$

and

$$\boldsymbol{s}^T(\lambda I + \nabla^2 L^-(\tilde{\boldsymbol{w}}))\boldsymbol{s}. \qquad \text{(A.21)}$$

Besides, we also have to cache the right hand side of (A.5), which is

$$\nabla f(\tilde{\boldsymbol{w}})^T \boldsymbol{s}.$$

For the value in (A.21) we have

$$(\lambda I + \nabla^2 L^-(\tilde{\boldsymbol{w}}))\boldsymbol{s} = \lambda \boldsymbol{s} + \nabla^2 L^-(\tilde{\boldsymbol{w}})\boldsymbol{s},$$

and the second term can be conducted by the procedure in (A.17).

With the above cached values, to evaluate the function-value difference at each new θ, we only need to calculate

$$L^+(\tilde{\boldsymbol{w}} + \theta \boldsymbol{s}). \tag{A.22}$$

From (A.7), new $\hat{C}_{ijk}$ values are needed. That is, from (A.8),

$$\hat{C}_{ijk} + \theta \boldsymbol{u}_i^T \text{vec}^{-1}(\boldsymbol{s})(\boldsymbol{h}_j \odot \boldsymbol{z}_k) \quad \forall (i,j,k) \in S, \tag{A.23}$$

where $\text{vec}^{-1}(\boldsymbol{s})$ converts $\boldsymbol{s}$ to a matrix in a column-wise setting and has the same dimension as Q_u. We can avoid calculating (A.23) for any new θ by applying a similar trick. Specifically, with the cached variables

$$\Delta_{ijk} = \boldsymbol{u}_i^T \text{vec}^{-1}(\boldsymbol{s})(\boldsymbol{h}_j \odot \boldsymbol{z}_k),$$

we can calculate (A.22) by

$$\sum\nolimits_{(i,j,k)\in S} \ell(C_{ijk}, \hat{C}_{ijk} + \theta\Delta_{ijk}) - \omega(A_{ijk} - (\hat{C}_{ijk} + \theta\Delta_{ijk}))^2$$

with the complexity proportional to $\mathbb{O}(|S|)$.

B ADDITIONAL DETAILS OF UNBIASED NON-POSITIONAL APPROACH

B.1 Proof of Theorem 4.1

Proof. Let $\tilde{C}_{ijk} \sim \text{Bern}(P_{ijk})$ be the random variable indicating whether event (i,j,k) is clicked, $\tilde{\mathcal{Y}}_{ij} \sim \text{Bern}(\tilde{P}_{ij})$ be the random variable indicating whether an examined event (i,j,k) is clicked, and $\tilde{\mathcal{E}}_k$ be the random variable indicating whether the ad placed in the position k is examined or not. Because each click/not-click C_{ijk} is an observation of $\tilde{C}_{ijk} = \tilde{\mathcal{Y}}_{ij}\tilde{\mathcal{E}}_k$ and the logistic loss is used, we have

$$\begin{aligned} &\mathbb{E}[\sum\nolimits_{i=1}^m \sum\nolimits_{j=1}^n \sum\nolimits_{k=1}^K \ell\left(\tilde{C}_{ijk}, f(\boldsymbol{u}_i, \boldsymbol{v}_j)\right)] \\ =&\mathbb{E}[\sum\nolimits_{i=1}^m \sum\nolimits_{j=1}^n \sum\nolimits_{k=1}^K -\tilde{\mathcal{Y}}_{ij}\tilde{\mathcal{E}}_k f(\boldsymbol{u}_i, \boldsymbol{v}_j)]. \end{aligned} \tag{B.1}$$

For the simplified scenario of assuming a constant α as the examination probability, the random variable $\tilde{\mathcal{E}}_k$ follows a Bernoulli distribution $\text{Bern}(\alpha)$. The value of (B.1) is

$$K \sum\nolimits_{i=1}^m \sum\nolimits_{j=1}^n -\tilde{P}_{ij} f(\boldsymbol{u}_i, \boldsymbol{v}_j)\alpha. \tag{B.2}$$

For the general scenario of assuming β_k as the examination probability of position k, the random variable $\tilde{\mathcal{E}}_k$ follows a Bernoulli distribution $\text{Bern}(\beta_k)$. The value of (B.1) is

$$K \sum\nolimits_{i=1}^m \sum\nolimits_{j=1}^n -\tilde{P}_{ij} f(\boldsymbol{u}_i, \boldsymbol{v}_j)(\frac{1}{K}\sum\nolimits_{k=1}^K \beta_k). \tag{B.3}$$

If we set

$$\alpha = \frac{1}{K}\sum\nolimits_{k=1}^K \beta_k, \tag{B.4}$$

then solving (34) leads to an unbiased estimator of $\alpha\tilde{P}_{ij}$. Then though (40), $f(\cdot)$ behaves the same as an unbiased estimator of $\tilde{P}_{ij}$. □

B.2 Unbiased Evaluation

In this section, we prove that for any non-positional CTR model $f(\boldsymbol{u}, \boldsymbol{v})$ to be evaluated on a test set S_{te} collected from a small portion of traffic served by the random strategy, the obtained LogLoss and AUC on S_{te} are unbiased metrics of estimating (39).

B.2.1 LogLoss. Because $f(\boldsymbol{u}, \boldsymbol{v})$ does not consider the position information, we omit this information to construct the following set

$$\Omega = \{(i,j) \mid (i,j,k) \in S_{\text{te}}\}. \tag{B.5}$$

Let $\mathcal{D}_{ij} = 1$ or 0 to indicate whether pair (i,j) is selected to S_{te}^+. We can write (9) as

$$\frac{1}{|S_{\text{te}}|}\sum\nolimits_{(i,j)\in\Omega} -\mathcal{D}_{ij} f(\boldsymbol{u}_i, \boldsymbol{v}_j) + \log(1 + e^{f(\boldsymbol{u}_i, \boldsymbol{v}_j)}) \tag{B.6}$$

Because the position of pair (i,j) in Ω is uniformly assigned from $1, \cdots, K$ with the probability $1/K$, $\mathcal{D}_{ij}$ is a random variable sampled from $\text{Bern}(\frac{1}{K}\sum_{k=1}^K P_{ijk})$. In the simplified scenario of (14), where all positions share the same constant α as the examination probabilities, $\mathcal{D}_{ij}$ is sampled from a Bernoulli distribution as

$$\mathcal{D}_{ij} \sim \text{Bern}(\alpha\tilde{P}_{ij}). \tag{B.7}$$

For the case of the general scenario, where β_k is various over K positions, $\mathcal{D}_{ij}$ is sampled from a Bernoulli distribution as

$$\mathcal{D}_{ij} \sim \text{Bern}(\frac{1}{K}\sum\nolimits_{k=1}^K \beta_k \tilde{P}_{ij}). \tag{B.8}$$

We have

$$\mathbb{E}_{\mathcal{D}}[(\text{B.6})] = \frac{1}{|S_{\text{te}}|}\sum\nolimits_{(i,j)\in\Omega} -\mathbb{E}[\mathcal{D}_{ij}] f(\boldsymbol{u}_i, \boldsymbol{v}_j). \tag{B.9}$$

From (B.4), the two results of computing (B.9) with $\mathcal{D}_{ij}$ respectively generated by (B.7) and (B.8) are equivalent. Therefore, LogLoss computed on an unbiased test set S_{te} is an unbiased evaluation of estimating (39) with certain α.

B.2.2 AUC. From the same random variable $\mathcal{D}_{ij}$ and the set Ω defined above, we can express (10) as

$$\frac{\sum_{(i,j)\in\Omega} \mathcal{D}_{ij}\text{Rank}_{i,j} - \frac{\gamma(\gamma+1)}{2}}{\gamma(|S_{\text{te}}| - \gamma)}, \tag{B.10}$$

where $\text{Rank}_{i,j}$ is the rank of pair (i,j) in all events in Ω in the descending order according to prediction values by $f(\cdot)$, and $\gamma = \sum_{(i,j)\in\Omega} \mathcal{D}_{ij}$. We then compute the expectation of (B.10) with regard to $\mathcal{D}_{ij}$ as follows

$$\frac{\sum_{(i,j)\in\Omega} \mathbb{E}[\mathcal{D}_{ij}]\text{Rank}_{i,j} - \frac{\mathbb{E}[\gamma](\mathbb{E}[\gamma]+1)}{2}}{\mathbb{E}[\gamma](|S_{\text{te}}| - \mathbb{E}[\gamma])}, \tag{B.11}$$

where

$$\mathbb{E}[\gamma] = \sum\nolimits_{(i,j)\in\Omega} \mathbb{E}[\mathcal{D}_{ij}].$$

By considering the two ways of generating $\mathcal{D}_{ij}$ in (B.7) and (B.8), respectively, if we apply (B.4), the results of (B.11) computed with $\mathcal{D}_{ij}$ generated by (B.7) and (B.8) are equivalent. Therefore, AUC computed on an unbiased test set S_{te} is an unbiased evaluation of estimating (39) with certain α.

C ADDITIONAL DETAILS OF RELATED WORKS

Due to the following properties, the ideas and derived approaches of unbiased learning-to-rank (LTR) are quite different from ours.

• The role of position: for LTR systems, the learnt models predict the relevance between a query and a document, which should be position-independent. From this perspective, each click/not-click is the feedback of the relevance but carries the bias caused by positions, which is also referred to as the position bias and should be eliminated in this context. However, as we mentioned in Section 1, the goal of our considered position-aware ad systems is to estimate P_{ijk}, which is intrinsically dependent on position k. Thus the learnt CTR model can be position-dependent. For example, in the positional approach reviewed in Section 2.2.1, the CTR model $g(\cdot)$ includes the position information in the input features. It is worth noting that though the non-positional approach in Section 2.2.2 learns a position-independent CTR model $f(\cdot)$, its motivation is to accelerate the display algorithm rather than to eliminate the position bias.

• Ranking learning vs point-wise learning: To eliminate the position bias from clicks, similar to the non-positional approach, some approaches of unbiased LTR consider the relation in (13) but term $\tilde{P}_{ij}$ and β_k as the relevance to learn and the position bias to eliminate, respectively. Because LTR systems rank documents according to learnt relevance, these works (e.g., [8, 19]) take such a property into account. For example, by only requiring the relevant documents to be ranked ahead of not-clicked ones, they do not have to disambiguate whether each not-clicked event is unexamined or irrelevant. Thus the position bias on not-clicked events can be ignored and the unbiasedness discussed in these works is limited to ranking relevant documents ahead of others. However, for the non-positional approach in our case, due to the existence of bids in the expected revenue, it is required to conduct point-wise learning for $\tilde{P}_{ij}$ or $\alpha\tilde{P}_{ij}$. Therefore, existing approaches for unbiased LTR are not applicable for position-aware ad systems. In contrast, our proposed approach in Section 4.2 to learn unbiased point-wise $\alpha\tilde{P}_{ij}$ can naturally give unbiased ranking of $\tilde{P}_{ij}$ for LTR systems.

In addition, for LTR systems, because clicks are collected on search results provided by formerly deployed models, the issue of the selection bias discussed in 3.2 also occurs. However, this issue is neglected in most approaches for LTR.

D ADDITIONAL DETAILS OF EXPERIMENTS

D.1 Data Preprocessing

For our experiments, we select m requests and n ads to construct a fully-observed set

$$G = \{(i, j) \mid i = 1, \cdots, m \text{ and } j = 1, \cdots, n\}.$$

The corresponding labels are denoted as a dense matrix $Y \in \{0, 1\}^{m\times n}$, where Y_{ij} is assumed to be the observation from (14) satisfying

$$Y_{ij} = \begin{cases} 1 & (i, j) \text{ is clicked in the original set,} \\ 0 & (i, j) \text{ is not-clicked or unobserved in the original set.} \end{cases} \tag{D.1}$$

To have the training set S and the test set S_{te} satisfying the scenario of position-aware systems with two types of selection bias, we orderly split G by requests into three subset G_1, G_2 and G_3 in ratios of 10%, 80%, and 10%, respectively. They are considered to be fully-labeled sets of events for three consecutive windows.

We first generate the training set S by using G_1 and G_2 of the first two windows. Specifically, by referring to Y_{ij},$\forall(i, j) \in G_1$, we split G_1 into G_1^+ and G_1^- as the sets of clicked/not-clicked events collected in the first window, with that we learn a $f_1(\cdot)$ by solving (11). The purpose of $f_1(\cdot)$ is to act as a CTR model deployed during the second window. In the second window, we select some events from G_2 into the training set S by simulating different retrieval and placing strategies. To accomplish this, for any request i, we must order ads according to their expected revenue. By using $f_1(\cdot)$ we calculate the expected revenue as

$$\sigma(f_1(\boldsymbol{u}_i, \boldsymbol{v}_j)) \times \text{bid}_j, \quad \forall(i, j) \in G_2,$$

where because we focus on CTR prediction and the ad auction is out of scope of this work, we simply assume $\text{bid}_j = 1,\forall j$. Then by considering different strategies listed in Section 6.1, S is generated. Finally, G_3 is used to generate S_{te}.

For any event (i, j, k) in S and S_{te}, besides the feature vectors $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ available from the original data sets, we convert k into a vector $\boldsymbol{p}_k$ by applying one-hot encoding technique.

D.2 Implementation

For training approaches compared in Section 6.3, we implement the algorithm proposed in Section A, where we set $\omega = 0$ for all approaches except TFM-CF (ϵ). The dense operations (e.g., matrix-matrix and matrix-vector products) are implemented using Intel® Math Kernel Library (MKL).

For approaches compared in Section 6.4, we implement FFM-Ideal, FFM-Greedy-A, FFM-Greedy-P, FFM-Random, FFM-Greedy, and FFM-EE (ϵ) by PyTorch, where we apply Adam as the optimizer with initial rates 0.001 and batch size 5,120. For FFM-CF (ϵ), we use the implementation in [21].

D.3 Parameter Selection

For parameter selections, we orderly split S by requests into S_{tr} and S_{va} in ratios of 80% and 20% respectively and conduct grid searches by training multiple models on the corresponding training set of each approach. We tune the following parameters.

• λ: the $l2$ regularization coefficient for each approach.
• d: the number of latent factors for each approach.
• ω: the weight of the imputation part.
• T: the max number of training iterations.

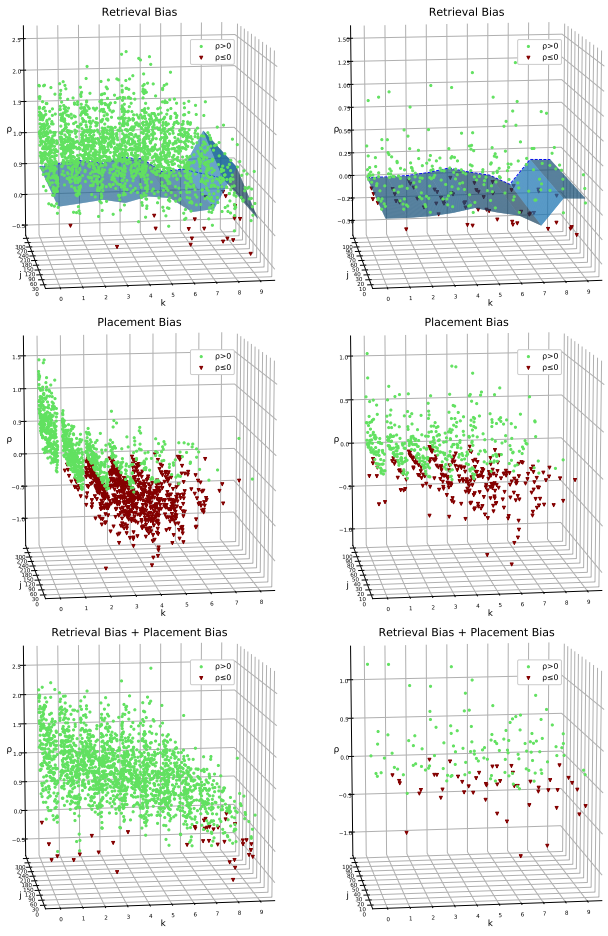
The search range of each parameter is listed in Table I. We then choose parameters and an iteration $\in \{1, \cdots, T\}$ achieving the highest AUC on the validation set. For building the final model to predict the test set, we should include data in the validation period.

D.4 Deep Understanding of Two Types of Selection Bias through Visualization

We visualize the effect caused by the retrieval bias, the placement bias and both biases. To begin we calculate CTR values of the unbiased S obtained from the **Random** strategy; see details in Section 4.1. Then we compare them with CTR values of some biased S sets, which were generated using the **GR**, **RG**, and **Greedy** strategies.

Table I: Search range of parameters considered in different approaches.

	TFM-Greedy, TFM-Random, TFM-RG, TFM-GR, TFM-EE	TFM-CF	FFM-Ideal, FFM-Greedy, FFM-Random, FFM-EE	FFM-CF	FFM-Greedy-A	FFM-Greedy-P
λ	$4^{\{-2,\cdots,2\}} \times 10^{-6}$	$4^{\{-2,\cdots,2\}} \times 10^{-6}$	$10^{\{-7,-6,-5\}}$	$4^{\{-2,\cdots,3\}}$	$10^{\{-7,-6,-5\}}$	$10^{\{-7,-6,-5\}}$
d	2^5	2^5	2^5	2^5	2^5	2^5
ω	-	$16^{\{-4,\cdots,0\}}$	-	$16^{\{-4,\cdots,0\}}$	-	-
T	50	50	20	50	20	20



(a) Outbrain (b) KKbox

Figure I: A demonstration of the effect of the retrieval bias and the placement bias; see the definition of ρ in (D.2)

Specifically, we check the CTR value for events associated with any (ad j, position k) pair and denote it as $\text{CTR}_{j,k}$. Then points $(j, k, \rho_{j,k})$ from the Outbrain and KKbox datasets are presented in Figure I, where $\rho_{j,k}$ measures the CTR difference between a biased S and an unbiased S:

$$\rho_{j,k} = \log\left(\frac{\text{CTR}_{j,k} \text{ observed from a biased } S}{\text{CTR}_{j,k} \text{ observed from } S \text{ collected by } \textbf{Random}}\right). \tag{D.2}$$

A positive or a negative $\rho_{j,k}$ respectively indicate $\text{CTR}_{j,k}$ observed in this biased S is higher or lower than the unbiased one, and the CTR model learnt from the biased S will probably overestimate or underestimate the corresponding P_{ijk}. From the results presented in Figure I, we have the following observations.

- Retrieval bias: From the first column of Figure I, we can observe most points are with $\rho_{j,k} > 0$. This confirms our analysis given in Section 3.2 that because any ad j is only retrieved when it has higher expected revenue, its $\text{CTR}_{j,k}$ observed from the biased S is always higher than the unbiased one. Besides, we use a blue surface to highlight points with the largest numbers of requests. Their corresponding $\rho_{j,k}$ are close to 0, indicating that $\text{CTR}_{j,k}$, even from a biased S, is similar to $\text{CTR}_{j,k}$ of an unbiased S. This observation confirms that the more associated events are included, the more associated C_{ijk} are revealed, so the observed $\text{CTR}_{j,k}$ is more accurate.
- Placement bias: From the second column of Figure I, a clear trend is that in the order of positions, $\text{CTR}_{j,k}$ gradually changes from higher to lower than the unbiased ones. The reason is that as we mentioned in Section 3.2, the retrieved ads with higher expected revenue tend to be placed in the front positions, while those with lower expected revenue are placed in the back positions.
- Both biases: From the third column of Figure I, the effects of two types of selection biases are mixed together. By comparing with the ranges of ρ shown in the first and second columns, because the overestimation effect caused by the retrieval bias is more serious than the underestimation effect caused by the placement bias, the retrieval bias is more dominant on the overall trend for both data sets.