Unbiased Learning to Rank in Feeds Recommendation

Xinwei Wu[1,6], Hechang Chen[2,6] ✉, Jiashu Zhao[3], Li He[4], Dawei Yin[5], Yi Chang[2,6] ✉
[1]College of Computer Science and Technology, Jilin University
[2]School of Artificial Intelligence, Jilin University, [3]Wilfrid Laurier University, [4]JD.com, [5]Baidu Inc.
[6]Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin Univeriity
wuxw18@mails.jlu.edu.cn,chenhc@jlu.edu.cn,jzhao@wlu.ca
heli223@jd.com,yindawei@acm.org,yichang@jlu.edu.cn

ABSTRACT

In feeds recommendation, users are able to constantly browse items generated by never-ending feeds using mobile phones. The implicit feedback from users is an important resource for learning to rank, however, building ranking functions from such observed data is recognized to be biased. The presentation of the items will influence the user's judgements and therefore introduces biases. Most previous works in the unbiased learning to rank literature focus on position bias (i.e., an item ranked higher has more chances of being examined and interacted with). By analyzing user behaviors in product feeds recommendation, in this paper, we identify and introduce context bias, which refers to the probability that a user interacting with an item is biased by its surroundings, to unbiased learning to rank. We propose an *Unbiased Learning to Rank with Combinational Propensity (ULTR-CP) framework* to remove the inherent biases jointly caused by multiple factors. Under this framework, a context-aware position bias model is instantiated to estimate the unified bias considering both position and context biases. In addition to evaluating propensity score estimation approaches by the ranking metrics, we also discuss the evaluation of the propensities directly by checking their balancing properties. Extensive experiments performed on a real e-commerce data set collected from JD.com verify the effectiveness of context bias and illustrate the superiority of $ULTR-CP$ against the state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Learning to rank**; *Information retrieval*; • **Computing methodologies** → **Learning from implicit feedback**.

KEYWORDS

Feeds Recommendation; Learning to Rank; Unbiased Learning

ACM Reference Format:
Xinwei Wu[1,6], Hechang Chen[2,6] ✉, Jiashu Zhao[3], Li He[4], Dawei Yin[5], Yi Chang[2,6] ✉. 2021. Unbiased Learning to Rank in Feeds Recommendation. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441751

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

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441751

1 INTRODUCTION

Recommender systems have become an increasingly attractive way for customers to discover interesting products on e-commerce sites, such as Amazon, JD, Alibaba, and Walmart [14, 31]. Recently, feeds recommendation has emerged, where customers are able to constantly browse products generated by never-ending feeds using mobile phones. During their interactions with product streams, customers can click on the products of interests, or skip uninterested items and keep scrolling. This customer click data is essentially free and abundant, which is usually leveraged for learning to rank (LTR) models by treating the clicked customer-product pairs as positives and non-clicked pairs as negatives. However, this treatment of training data can be heavily biased [6, 20, 28], since the relevance is not the only factor that influences the customers' behaviors. For example, customers are more likely to click on higher-ranked products (i.e., position bias), and they might only click on products to which they have previously been exposed (i.e., a type of presentation bias). In order to accurately estimate the relevance between a product and a customer from click behaviors, we need to identify and remove the bias factors in LTR.

To eliminate this bias, recent unbiased LTR approaches treat click bias as a counterfactual effect and debias user feedback by weighting each click with their inverse propensity weights (IPW) [20, 35, 36]. However, most previous literatures on unbiased LTR focus on the position bias [3, 4, 10, 36] (i.e., an item ranked higher has more chances of being examined and interacted with).

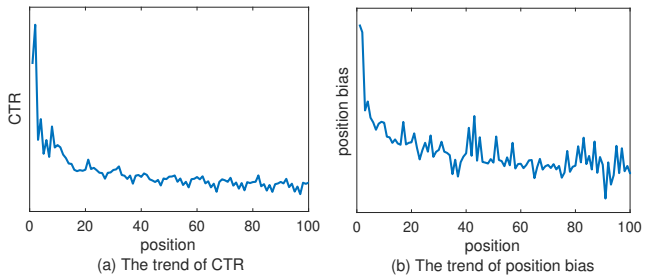
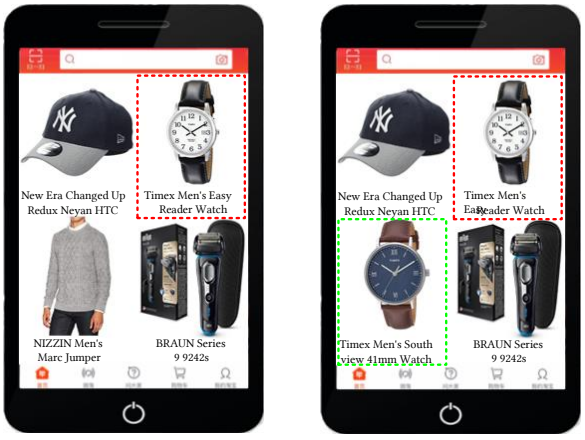


Figure 1: CTR and the position bias obtained from 20 days user click log across the portal.

Unlike web search where users locate and click web pages in top positions (e.g., ten or twenty), customers using feeds recommendation tend to view more products (hundreds, or even more) by scrolling down. We observe from Figure 1(a) that as the user browses more deeply, the click-through rate (CTR) significantly drops for items in high-ranked positions (e.g., around position 20), whereas it stays in a similar range beyond position 20. Here we only show the tendency of CTR due to privacy reasons. This implies that position bias might not always be significant. Figure 1(b) shows position bias (obtained by the method in [36]) in feeds recommendation. This verifies that position bias will become trivial when the position goes beyond 20. That is, position bias does not have much effect when the customers browse deeply. Position bias is one of the various presentation biases, depending on specific scenarios. In feeds recommendation, e-commerce customers usually compare products before making decisions. Specifically, in mobile feeds recommendation, four grids with pictures are usually displayed on the same phone screen, as shown in Figure 2. In this example, the Timex watch is surrounded by non-similar products in Figure 2(a), whereas in Figure 2(b), it is placed close to a similar product (i.e., another Timex watch). For a target product, its surrounding products refer to those appear on the same screen. We say it is a similar product when its cosine similarity with the target product is larger than a given threshold. Otherwise, we say it is a non-similar product. Our experiment shows that customers behave quite differently under different contexts in relation to the product, the CTR of target products surrounded by non-similar products is consistently higher than that of target products surrounded by similar products. The detailed explanations of similar product and the existence of context bias will be given in Section 6.2.

We refer to such a phenomenon as context bias. Intuitively, we know that context bias is also significant, and becomes even more important than position bias when the user browses more deeply. Unfortunately, existing work on unbiased LTR [3, 20, 35, 36] mainly focuses on position bias and neglects other biases, e.g. context bias.

To address this problem, in this paper we identify and introduce context bias, which refers to the probability that a user interacting with an item is biased by its surroundings, to unbiased LTR. First, an *Unbiased LTR with Combinational Propensity (ULTR−CP) framework* is proposed to remove the unified bias composed of multiple types of biases. Then, the proposed context bias is instantiated and estimated within the framework. We assume the probability of a customer clicking on a product depends on three factors: examination, relevance, and comparison. Since the relevance between a product and a user request is difficult to obtain directly, we leverage the regression-based expectation-maximization (EM) algorithm [36] to estimate these three factors. Finally, a unified bias, composed of both the position and context biases, is estimated and further eliminated in LTR in feeds recommendation. In unbiased LTR, the propensity weighting is usually indirectly evaluated on the LTR task by ranking metrics such as precision, recall, and mean reciprocal rank (*MRR*), etc. Under such settings, the ranking metrics are highly dependent on the ranking models adopted. In order to understand the estimated propensity weightings more straightforwardly, we also discuss the evaluation of the propensities directly by checking their balancing properties. In particular, we establish a connection



(a) A Timex watch surrounded by non-similar products

(b) A Timex watch with another Timex watch surrounded

Figure 2: A target product surrounded by non-similar products v.s. similar products: the CTR of target products surrounded by non-similar products is consistently higher than that of target products surrounded by similar products.

between balancing checking in causal inference and feeds recommendation. A better propensity estimation should better balance the covariates between the treated and controlled sub-populations for similar propensity scores. The main contributions of this work are summarized as follows:

- In the scenario of e-commerce feeds recommendation, we identify and estimate context bias. To the best of our knowledge, this is the first study to eliminate context bias in LTR for feeds recommendation.
- Theoretically, a unified unbiased LTR framework, *ULTR−CP*, is introduced to integrate multiple types of biases. Context bias is instantiated under this framework.
- Experimental results on real data sets show improvements of the proposed methods against the state-of-the-arts. The propensity scores are evaluated both directly via balancing checking and indirectly via ranking metrics[1].

2 RELATED WORK

In this section, we briefly review research related to our study, i.e., unbiased learning to rank. There are two kinds of approaches, online and offline, to learn unbiased LTR systems. Online LTR [27, 38] interact with users directly and performs randomization, enabling them to deal with several biases, while this approach is beyond the scope of this paper so for more information, please refer to [17, 25]. Offline LTR take users' feedback, such as user click log, as relevance indicators to learn a ranking model. Though users' feedback is easy to obtain and in low cost [20], they often contain different types of biases [23, 39], including position bias [18, 19], selection bias [28, 35], trust bias [1, 26], among others. For example, selection bias

[1]https://github.com/flamewei123/Unbaised-LTR-in-Feeds-Recommendation-WSDM21.

problem is that queries are under-sampled to different extents and thus biased when click data is collected to learn ranking functions, which injects a presentation bias in the collected data [35].

In order to learn an unbiased ranker from biased user feedback, researchers have proposed click models to extract unbiased relevance feedback [8, 11, 12]. The main procedures are to make assumptions about the causes of the bias, construct a probabilistic model for the bias, and then use statistical inference to get unbiased estimation of relevance. Another solution is to construct unbiased learning algorithm [1, 3–5, 15, 20, 35, 36]. Wang et al. [35] adopted inverse propensity weighting approach in debiasing method. Joachims et al. [20] proposed an IPW framework to correct position bias, and prove the method can reach ideal expectations. Wang et al. [36] used a regression-based EM method to solve the sparsity of query and document pair without randomized tests. Then dual learning algorithm is proposed by Ai et al. [4] to estimate propensities while training the ranker. Most unbiased LTR works belong to the pointwise category [24, 37]. Hu et al. [15] introduced a method that jointly estimates position bias and train a ranker by using pairwise LTR function. Our work is partially inspired by [36], where regression-based EM method is proposed to estimate position bias in personal search. We extend this idea to estimate a more general combinational bias in the scenario of e-commerce feeds recommendation, and instantiated with a novel bias, i.e., context bias.

3 ULTR-CP FRAMEWORK

To unify different types of biases, we propose an unbiased LTR with combinational propensity ($ULTR-CP$) framework. Under this framework, multiple biases that affect users' behaviors are aligned. Here 'customer' refers specifically to 'user' on e-commerce sites. In this work, we will use 'customer' and 'user' interchangeably.

3.1 Unbiased Learning to Rank

The IPW is a commonly used technique to correct the bias in many applications, and now is being applied to unbiased LTR [35]. Here we use IPW to estimate the unified bias, which includes all the possible biases occurred during the user interactions, including but not limited to the position bias and context bias. In this section, we formally define the unified unbiased LTR problem where different types of biases can be considered in different scenarios.

Given a user request u_i, we denote with $r_i(x)$ the relevance of item x for user request u_i. The relevances can be used to compute the loss $l(\boldsymbol{x}|u_i, r_i)$ of any ranking $\boldsymbol{x}$. Following previous work, we assume that the relevances are binary, i.e., $r_i(x) \in \{0, 1\}$. We define the performance measure $l(\boldsymbol{x}|u_i, r_i)$ as follows.

$$l(\boldsymbol{x}|u_i, r_i) = \sum_{x \in \boldsymbol{x}} M(rank(x|\boldsymbol{x})) \cdot r_i(x), \tag{1}$$

where $M(\cdot)$ can be any weighting function that depends on the rank $rank(x|\boldsymbol{x})$ of item x in ranking $\boldsymbol{x}$. For instance, setting $M(rank) = rank$ gives the sum of relevant ranks metric (also called average rank), and $M(rank) = -\frac{1}{\log(1+rank)}$ gives the DCG metric. For a ranking system f that returns ranking $f(u, X)$, the risk is

$$\mathcal{L}(f) = \int l(f(u, X)|u, r) \mathrm{d}P(u, r), \tag{2}$$

where X is the set of products recommended for user request u (i.e., the recalled products for u). We suppress the dependence of f on X and u in $\mathcal{L}(f)$.

We first assume that there is a given ranking list $\bar{\boldsymbol{x}}_i$. On one hand, the relevance signals for top-ranked results are more likely to be observed than those for bottom-ranked results in feeds recommendation. On the other hand, the probability of observing the relevance signal of a certain item will be influenced by its surroundings. Let O_i denote the 0/1 vector indicating which relevance values are revealed. We have $O_i \sim P(O|u_i, \bar{\boldsymbol{x}}_i, r_i, B_i)$, where B_i is a set of bias factors which could cause different types of biases, e.g., the rank for position bias [36], the figures for marketing bias [34], and the context for context bias (which we will explain and study in latter sections), etc. For each element of O_i, we denote $Q(O_i(x) = 1|u_i, \bar{\boldsymbol{x}}_i, r_i, B_i)$ the marginal probability of revealing the relevance $r_i(x)$ of the item x for user request u_i, if the user was presented the ranking $\bar{\boldsymbol{x}}_i$ with set B_i. We call this probability value a *combinational propensity* of the observation.

With the help of IPW technique, we can obtain an unbiased estimate of $l(\boldsymbol{x}|u_i, r_i)$ for any new ranking $\boldsymbol{x}$. The IPW loss for a unified unbiased ranking model is defined as

$$l_{IPW}(\boldsymbol{x}|u_i, \bar{\boldsymbol{x}}_i, O_i) = \sum_{x:O_i(x)=1} \frac{M(rank(x|\boldsymbol{x})) \cdot r_i(x)}{Q(O_i(x) = 1|u_i, \bar{\boldsymbol{x}}_i, r_i, B_i)}. \tag{3}$$

Assume there is a sample of N user requests u_i. Then, the empirical risk of a system is the average of the IPW loss over user requests:

$$\hat{\mathcal{L}}_{IPW}(f) = \frac{1}{N} \sum_{i=1}^{N} \sum_{x:O_i(x)=1} \frac{M(rank(x|f(u_i, X_i))) \cdot r_i(x)}{Q(O_i(x) = 1|u_i, \bar{\boldsymbol{x}}_i, r_i, B_i)}.$$

The goal of unified unbiased LTR is to find the best scoring function $\hat{f}$. A commonly used learning strategy is empirical risk minimization (ERM) [33], i.e., $\hat{f} = \arg\min_f \{\hat{\mathcal{L}}_{IPW}(f)\}$. Please note that only the clicked items contribute to empirical risk (i.e., $O_i(x) = 1 \wedge r_i(x) = 1$). For simplicity, the clicked item is assumed to be relevant, which can be further relaxed.

3.2 Combinational Propensity Weighting

The unified bias (i.e., combinational propensity) discussed in the previous section includes all possible presentation biases, and in this section, we discuss how to estimate it. The notation $Q(O(x) = 1|u, \bar{\boldsymbol{x}}, r, B)$ is a combinational propensity of all bias factors in B when a ranking $\bar{\boldsymbol{x}}$ is presented to user request u. We assume that different types of biases are independent of each other (the independence test with Chi-square test is included in Section 6.2). Let's take position bias and presentation bias as an intuitive example. When the user browses top pages, the presentation of product pictures will affect the user's behavior. Similarly, when users browse deeply, users will also be influenced by the product pictures in a similar way. Therefore, combinational propensity weight can be defined as the product of propensities for all considered bias factors.

$$Q(O(x) = 1|u, \bar{\boldsymbol{x}}, r, B) \equiv \prod_{b \in B} Q(O(x) = 1|u, \bar{\boldsymbol{x}}, r, b). \tag{4}$$

Specifically, we aim to study position bias and context bias in this work, which is a special case of the unified unbiased LTR problem.

Then the combinational propensity becomes

$$Q(O(x)=1|u,\bar{\mathbf{x}},r,B)=Q(O(x)=1|u,\bar{\mathbf{x}},r,b_1)\cdot Q(O(x)=1|u,\bar{\mathbf{x}},r,b_2),$$

where $Q(O(x)=1|u,\bar{\mathbf{x}},r,b_1)$ and $Q(O(x)=1|u,\bar{\mathbf{x}},r,b_2)$ represent the propensity of position bias and context bias, respectively. We will discuss how to estimate them in Section 4 in detail.

3.3 Unbiased Ranking

The ranking problem can be optimized by utilizing a LTR approach with combinational propensity. First, combinational propensity $Q(O(x)=1|u,\bar{\mathbf{x}},r,B)$ is estimated using a probabilistic model (e.g., click model) from a user implicit feedback data set. Denote $\varphi(u,x)$ as the feature vector of a user request and an item. Then, given the estimated combinational propensity and the training data set $\{\varphi(u,x)\}$, a ranking scoring function $f_w(u,x)$ can be learnt. The relevance signals are not directly obtained from the click signals, but they are adjusted by the combinational propensity. The debiased rank of x in training becomes $rank(x|f_w(u,x))$, where w is a parameter for the ranking scoring function. The ranking scoring function can be instantiated in conventional ranking models, such as LambdaMART [7], SVM-Rank [22], etc., or deep neural network based approaches.

4 AN INSTANTIATION OF ULTR-CP FRAMEWORK

In this section, we provide an instantiation of the proposed $ULTR-CP$ framework, which debiases the effects of position and context for feeds recommendation. We first propose a context-aware position bias model, then introduce the estimation of context-aware propensity and give the unbiased SVM-rank subsequently.

4.1 Context-Aware Position Bias Model

We first propose a context-aware position bias model by assuming that the probability of a customer clicking on a product depends on three factors, i.e., examination, relevance, and comparison, which are denoted by E, R, and H, respectively. Consider formulating the probability of a product x being clicked as follows:

$$P(C=1|u,x,k,z)=P(E=1|k)\cdot P(R=1|u,x)\cdot P(H=1|z). \quad (5)$$

The symbol C denotes the observed click Bernoulli variable, which indicates whether it is clicked or not, $P(E=1|k)$ represents the probability of a customer examining the product at position k and $P(R=1|u,x)$ represents the probability that product x is relevant to user request u. Given context z, $P(H=1|z)$ denotes the probability of whether the user's attention on item x is kept when comparing it with its surroundings. We identify the context of a target item by judging whether there exists similar items in its surroundings, which is specifically described in Section 6.2.

Different from the existing studies which only consider examination (E) and relevance (R) [9], the proposed context-aware position bias model considers the impact of with-similar context on user click behaviors. For simplicity, we use the following shorthand notations for derivation:

$$\alpha_k=P(E=1|k),\ \beta_{u,x}=P(R=1|u,x),\ \gamma_z=P(H=1|z).$$

In this model, we have equivalence between O and $E\wedge H$, i.e., $Q(O(x)=1|u,\bar{\mathbf{x}},r,B)=P(E(x)=1|k)\cdot P(H(x)=1|z)$. In this case, the set B consists of the rank factor k and the context factor z. This is consistent with previous independent assumption in Section 3.2.

We denote click logs as $\pi=\{(c,u,x,k,z)\}$, which records click event c from user request u on item x under the condition of position k and context z. Then, the log likelihood of generating π is $\sum_{(c,u,x,k,z)\in\pi} c\log\alpha_k\beta_{u,x}\gamma_z+(1-c)\log(1-\alpha_k\beta_{u,x}\gamma_z)$. Given the click logs, the EM algorithm can find the parameters that maximize the log likelihood. In the following section, we will describe the details of parameter estimation.

4.2 Context-Aware Propensity Estimation

It is difficult to obtain the relevance between recommended products and customer's interests (i.e., $\beta_{u,x}$), since the exposed product list for every user request is entirely different even for multiple requests from the same customer. This leads to a non-trivial direct estimation of parameters α_k, $\beta_{u,x}$, and γ_z using the standard EM algorithm. Therefore, based on the idea of regression-based EM algorithm [36], we propose to estimate the parameters simultaneously by estimating $\beta_{u,x}$ with a regression function. In the following, we first describe the standard EM algorithm, then introduce the modified Maximization step in detail.

Expectation Step: In each iteration, the distribution of variables E, R, and H are estimated, given click logs π and parameters α_k, $\beta_{u,x}$ and γ_z from the previous iteration in the Maximization step.

$$\begin{aligned}
&P(E=1,R=1,H=1|C=1,u,x,k,z)=1,\\
&P(E=1,R=1,H=0|C=0,u,x,k,z)=\frac{\alpha_k\beta_{u,x}(1-\gamma_z)}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(E=1,R=0,H=1|C=0,u,x,k,z)=\frac{\alpha_k(1-\beta_{u,x})\gamma_z}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(E=1,R=0,H=0|C=0,u,x,k,z)=\frac{\alpha_k(1-\beta_{u,x})(1-\gamma_z)}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(E=0,R=1,H=1|C=0,u,x,k,z)=\frac{(1-\alpha_k)\beta_{u,x}\gamma_z}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(E=0,R=1,H=0|C=0,u,x,k,z)=\frac{(1-\alpha_k)\beta_{u,x}(1-\gamma_z)}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(E=0,R=0,H=1|C=0,u,x,k,z)=\frac{(1-\alpha_k)(1-\beta_{u,x})\gamma_z}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(E=0,R=0,H=0|C=0,u,x,k,z)=\frac{(1-\alpha_k)(1-\beta_{u,x})(1-\gamma_z)}{1-\alpha_k\beta_{u,x}\gamma_z}.
\end{aligned} \quad (6)$$

Therefore, we have the following results:

$$\begin{aligned}
&P(E=1|C=0,u,x,k,z)=\frac{\alpha_k\beta_{u,x}(1-\gamma_z)}{1-\alpha_k\beta_{u,x}\gamma_z}+\frac{\alpha_k(1-\beta_{u,x})}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(R=1|C=0,u,x,k,z)=\frac{\alpha_k\beta_{u,x}(1-\gamma_z)}{1-\alpha_k\beta_{u,x}\gamma_z}+\frac{(1-\alpha_k)\beta_{u,x}}{1-\alpha_k\beta_{u,x}\gamma_z},\\
&P(H=1|C=0,u,x,k,z)=\frac{\gamma_z(1-\alpha_k\beta_{u,x})}{1-\alpha_k\beta_{u,x}\gamma_z}.
\end{aligned}$$

Maximization Step: Given the above probabilities, parameters α_k, $\beta_{u,x}$, and γ_z can be updated accordingly.

$$\begin{aligned}
\alpha_k&=\frac{\sum_{c,u,x,k',z}\mathbb{I}_{k'=k}\cdot[c+(1-c)\cdot P(E=1|c,u,x,k,z)]}{\sum_{c,u,x,k',z}\mathbb{I}_{k'=k}},\\
\beta_{u,x}&=\frac{\sum_{c,u',x',k,z}\mathbb{I}_{u'=u,x'=x}[c+(1-c)\cdot P(R=1|c,u,x,k,z)]}{\sum_{c,u',x',k,z}\mathbb{I}_{u'=u,x'=x}},\\
\gamma_z&=\frac{\sum_{c,u,x,k,z'}\mathbb{I}_{z'=z}[c+(1-c)\cdot P(H=1|c,u,x,k,z)]}{\sum_{c,u,x,k,z'}\mathbb{I}_{z'=z}}.
\end{aligned} \quad (7)$$

Algorithm 1: Context-Aware Unbiased LTR

Input: $\pi = \{(c, u, x, k, z)\}$, $\{\varphi(u, x)\}$, λ, $\{\alpha_k\}$, $\{\beta_{u,x}\}$, $\{\gamma_z\}$
Output: $\{\alpha_k\}$, $g(\varphi(u, x))$, $\{\gamma_z\}$, and $f_w(u, x)$

Let $G(\varphi) = 0$
repeat
  Compute the hidden probability by Eq. (6)
  Let R={}
  for all $(c, u, x, k, z) \in \pi$ **do**
    Sample $r \in \{0, 1\}$ from $P(R = 1|c, u, x, k, z)$
    $R = R \bigcup (\varphi(u, x), r)$
  end for
  $G(\varphi) = GBDT(G(\varphi), R)$
  Update $\{\alpha_k\}$, $\{\gamma_z\}$ by Eq. (7)
  Update $\{\beta_{u,x} = g(\varphi(u, x))\}$
until Convergence
return $\{\alpha_k\}$, $g(\varphi)$ and $\{\gamma_z\}$
for the i-th $(u, x, k, z) \in \pi$ **do**
  Compute $\eta_i = \alpha_k \cdot \gamma_z$
end for
$f_w(u, x)$ =Unbiased SVM-rank$(\varphi(u, x), \{\eta_i\}, \lambda)$
return $f_w(u, x)$

However, due to the sparseness and noisiness of the data, we learn a function instead to estimate the relevance $\beta_{u,x} = g(\varphi(u, x))$ in the Maximization step [36], where $\varphi(u, x)$ is a feature vector of user request u and item x. The goal is to find a regression function $g(\varphi)$ to maximize the likelihood given the estimation from the Expectation step. We sample relevance label $r \in \{0, 1\}$ based on probability $P(R = 1|c, u, x, k, z)$ for each $(c, u, x, k, z) \in \pi$. Then, we obtain a training set $\{(\varphi, r)\}$, which can be used to train $g(\varphi)$ by maximizing the log-likelihood $\sum_{\{(\varphi,r)\}} r \log g(\varphi) + (1 - r) \log(1 - g(\varphi))$, where $g(\cdot)$ is the Sigmoid function $g(\varphi) = \frac{1}{1+\exp(-G(\varphi))}$. The function $G(\varphi)$ can be learnt by any regression models. We follow the previous work [1] and use Gradient Boosted Decision Tree (GBDT) model in this work. Please refer to Algorithm 1.

4.3 Unbiased SVM-rank

Given the combinational propensity estimated by regression-based EM, we optimize the ranking problem via SVM-rank with linear kernels [2]. SVM-rank learns a scoring function $f_w(u, x) = w \cdot \varphi(u, x)$, where w is a weight vector learnt from the training data. Unbiased SVM-rank [20] optimizes the following objective function:

$$\hat{w} = \arg\min_{w,\xi} \frac{1}{2} \|w\|^2 + \frac{\lambda}{n} \sum_{i=1}^{n} \frac{1}{\eta_i} M\left(\sum_{x \in X_i} \xi_{ix} \right)$$
$$s.t. \forall x \in X_1 \backslash \{x_1\} : w \cdot [\varphi(u_1, x_1) - \varphi(u_1, x)] \geq 1 - \xi_{1x},$$
$$\dots$$
$$\forall x \in X_n \backslash \{x_n\} : w \cdot [\varphi(u_n, x_n) - \varphi(u_n, x)] \geq 1 - \xi_{nx},$$
$$\forall i \forall x : \xi_{ix} \geq 0.$$

where $\|\cdot\|$ denotes L_2 norm, λ is a regularization parameter, ξ is a slack variable, and η_i is the combinational propensity where $i \in \{1, 2, \dots, n\}$ denotes the i-th click. Finally, unbiased SVM-Rank is able to learn a ranking scoring function from an implicit feedback data set, in which all types of biases have been corrected by the combinational propensity weight. In the experiment, we consider optimizing average rank, i.e., setting $M(rank) = rank$.

5 PROPENSITY CHECKING FOR LTR

Previous unbiased LTR approaches evaluate the estimation by the final ranking performance. However, we can see from Algorithm 1 that the propensity estimation approach (inner steps 1-16) and the LTR model (inner step 17) are independent, and the propensity weightings are served as input for the LTR model. In fact, any LTR approach could be adopted here with the training samples adjusted by propensity scores. Researchers have found that a slight mis-specification of the propensity score model can result in substantial bias of the estimated treatment effects [21, 32]. Therefore, we argue that it is also crucial to check the estimated propensity scores directly, so that the propensity estimation evaluation does not rely on the performance of the LTR approaches. Inspired by the properties of propensity score, in this section, we introduce balance checking to compare the estimation of the propensity score for different models under the scenario of LTR.

5.1 Balancing Property

First, let's briefly review causal inference using a simple example. One may want to analyze the effect of participation in a job training program on individual earnings. There are three basic elements for causal inference: treatment, outcome, and covariates. The goal is to analyze the causal effect of treatment on outcome. For this example, the treatment is whether to participate in a job training program or not, the outcome is individual earnings, and the covariates of a person could be age, years of education, etc. Generally speaking, causal effects involve a comparison of the outcome actually observed with other potential outcomes that could have been observed had the treatment taken on a different level (but that are not observed in fact). However, as the data we have usually come from observational studies but not randomized trials, there will be bias in the estimation of treatment effects with observational data sets. Propensity-based approaches have been widely used in causal inference from observational studies [16, 30], such as in medical trials and the evaluation of economic policy interventions.

In causal inference study, the propensity score represents the conditional probability of assignment to a particular treatment given a vector of observed covariates [30]. The propensity score could be regarded as a balancing score, which is used to group treatment and control units so that direct comparisons are more meaningful. Thus, the propensity score satisfies the following balancing property: the treatment assignment and the observed covariates are conditionally independent given the propensity score, that is

$$covariates \perp\!\!\!\perp T \mid P(T = 1|covariates),$$

where T is treatment, and $P(T = 1|covariates)$ is the propensity score. Simply speaking, the conditional distribution of covariates given $P(T = 1|covariates)$ should be the same for treatment ($T = 1$) and control ($T = 0$) units. For unbiased LTR, the propensity score needs to be estimated. Naturally, if the estimated propensity scores can better balance the covariates, then the propensity estimation approach is more appropriate for the designated ranking problem.

The corresponding balancing checking method will be discussed in detail in Section 5.2.

5.2 Balancing Checking Method

In this subsection, we describe the detailed balancing checking method for feeds recommendation. We aim to find estimates of the propensity score that can balance the covariates between the treatment and control sub-populations. The balancing property can be investigated by stratifying the units into blocks with propensity scores of similar values, and then testing independence of treatment and covariates within each resulting block [13]. For each covariate, we can test whether the means for the treatment and control are different in all blocks. If the difference of the means is smaller, then the covariates achieve more balance, which indicates propensity estimation is more appropriate for the problem.

To connect with the concepts in causal inference, our outcome in feeds recommendation is the relevance r of a product. The treatment is $O \in \{0, 1\}$, i.e., whether the relevance of a product x is revealed. The goal is to analyze the causal effect of treatment O on outcome r. Our propensity score is defined as $Q(O(x) = 1|u, \bar{x}, r, B)$. Thus, the covariates among B are those factors that have influences on propensity. For example, the position k is a covariate for position bias model [36], since propensity only depends on the rank of product x in ranking list $\bar{x}$. To check the balancing property of the estimated propensity score for the proposed ULTR-CP framework, we first divide the products into several subsets, where products within one subset have propensity scores of similar values. Then, for each covariate among B, we test whether its mean values for the treatment (the clicked products) and the controls (the non-clicked products) are different in all subsets. Taking the position bias model as an example, for products with similar propensity, we compare the average ranks of the clicked and the non-clicked products. Now we are ready to compare the estimation of propensity scores under any two models. The degree of balance of covariates is used to distinguish which propensity estimation is more appropriate, i.e., a smaller difference of the average covariate values means a better balance for the estimated propensity score.

6 EVALUATION

In this section, we first describe the experimental settings including data set, baselines, and evaluation metrics, then conduct experiments with the aim of answering the following research questions: **RQ1:** Does context bias exist in feeds recommendation applications? **RQ2:** How does our proposed approach perform compared with the state-of-the-art bias estimation methods? **RQ3:** How does the effectiveness of position bias and context bias vary with different ranking positions? **RQ4:** Does the estimated propensity score satisfy the balancing property? **RQ5:** Whether the proposed model is stable enough to be applied in practice?

6.1 Experimental Settings

6.1.1 Data Set. We collect data from JD.com, which is one of the largest E-commerce sites in the world. When a customer accesses JD.com, a list of recommended products is given to the customer and this list is extended when the customer browses more products. The logged data set includes the entire list and the clicked products. We collected 171,226,678 records with 735 features on 9,105,498 products for 16 consecutive days in September 2019.

The dataset is split into training and testing datasets by date, where data from the first 7 days are used for bias estimation and training, and data from the last 9 days are used for testing. We take the top 100 records exposed to a customer request. The experiments are conducted by predicting online recommendation results on each day in the testing time period.

6.1.2 Baselines. We compare our proposed context-aware position bias model ($ULTR-CP$) with the following three baselines:

- Naive SVM-Rank ($SVM-rank$) [22]: a conventional LTR model based on kernel functions, which does not consider any bias and is shown to be effective for large and sparse data. It is adopted as the ranking model for all unbiased learning approaches.
- Context-based unbiased LTR ($ULTR-cxt$): a comparison model to verify the performance of individual context bias.
- Position-based unbiased LTR ($ULTR-pos$) [36]: a state-of-the-art unbiased LTR model, which only considers position bias when calculating product rankings.

6.1.3 Evaluation Metric. We use a weighted Mean Reciprocal Rank (MRR) [36], which is the inverse rank of the clicked item of the i-th recommendation list weighted by v_i, to evaluate the ranking effectiveness. A higher MRR indicates better ranking performance. The formulation of weighted MRR is

$$\frac{1}{\sum_{i=1}^{N}\frac{1}{v_i}}\sum_{i=1}^{N}\frac{1}{v_i}\cdot\frac{1}{Rank_i}.$$

The weight v in weighted MRR affects model performance [1]. When $v = 1$, it becomes the original MRR. The results under *naive MRR* are not referential to evaluate the unbiased LTR models, since it does not consider the inherent biases in the test data. Previous work has estimated the weight v by online randomization experiments [36] or by EM estimated propensity [5]. Since online randomization experiments will cause great loss in e-commerce,

we study the following different estimations of v as the metric weights:

(i) $CP-based\ MRR$ uses the propensity estimated by our proposed context-aware position bias model;

(ii) $Pos-based\ MRR$ exploits the propensity estimated by position based model [36];

(iii) $CTR-based\ MRR$ is firstly proposed in our paper to use the statistical CTR as a more objective approach.

6.2 Verification of Context Bias (RQ1)

We analyze the CTR distributions of the recommended products exposed within different contexts to answer **RQ1**. Then, we verify the independence between position bias and context bias.

6.2.1 Definition. Context bias in this paper refers to the probability of a user interacting with an item is biased with the displacement of its surroundings. Particularly, in our dataset, 4 products are displayed on a cell phone screen at the same time. For a target item, its surrounding is defined as the set of the other items on the same screen. We represent each item by a 50-dimension embedding vector

Table 1: The distribution of items in different contexts and positions

Contexts	Positions										
	[1,10]	[11,20]	[21,30]	[31,40]	[41,50]	[51,60]	[61,70]	[71,80]	[81,90]	[91,100]	total
with-similar-item	12903	3418	1995	1397	1193	808	636	566	451	352	23719
without-similar-item	340145	184179	130154	98745	77842	62323	50963	42555	35583	29993	1052482
total	353048	187597	132149	100142	79035	63131	51599	43121	36034	30345	1076201

(estimated by item information using *word2vec* [29]), then compute the cosine similarity between the target item and the surrounding items. If there exists at least one item with high similarity to the target item, we say the target item is in a with-similar-item context. Otherwise, we say it is in a without-similar-item context. Here the similarity threshold is set to be 0.9.

6.2.2 Existence of Context Bias . Using our training set up, we compare the products in with-similar-item contexts with the products in without-similar-item contexts to observe the existence of context bias. Figure 3 shows the CTRs of these two groups at different positions averaged over every 10 positions. For the sake of privacy, the absolute values of the CTRs are not shown in the figure, but we can still observe the tendency very clearly. The CTRs for the products in without-similar-item contexts are consistently higher than those products in with-similar-item contexts. It illustrates that context bias does exist in e-commerce recommendation. Moreover, the rank of a product has a strong effect on customers' behaviors in relation to products in the top positions. When customers continuously viewing products into deep positions in feeds recommendation, context bias remains obvious while position bias becomes weaker.

6.2.3 Independence of Context Bias and Position Bias. To verify our assumption in Section 3.2, we test the independence of position bias and context bias by Chi-square test. Table 1 is a contingency table generated from the training set, showing the frequency of products in different contexts and positions. The p-value of Chi-square test is less than 0.01, which indicates that the position bias and context bias are independent from each other.

6.3 Overall Performance (RQ2)

In this section, we compare the proposed $ULTR-CP$ with several state-of-the-art baseline models to answer **RQ2**.

Since the performance of an unbiased LTR approach is affected by the weight of *MRR*, we adopt *MRR* with different weights to evaluate the effectiveness of the models from various perspectives, including $CP-based\ MRR$, $pos-based\ MRR$, and $CTR-based\ MRR$. The CTR is obtained from random sampling statistics of 16-day data, which can be treated as the ground truth. Our testing period consists of 9 consecutive days, and we record the results averaged on each of the given day to show the stability of the models.

As shown in Table 2, for different *MRR*, $ULTR-CP$ achieves the best performance over all the baselines, including the state-of-the-art $ULTR-pos$ model. $ULTR-cxt$ is worse than $ULTR-pos$ and $ULTR-CP$, which indicates using context bias alone cannot improve ranking method effectively. The results under $CTR-based\ MRR$ are more convincing, because the value of CTR is obtained through real data statistics, which is more objective. Overall, we can conclude that the proposed context bias estimation approach is able to capture this new type of bias in the real-world e-commerce recommendation application.

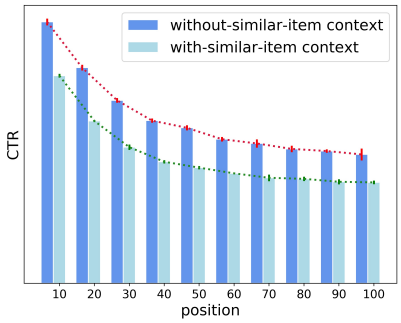


Figure 3: CTRs for products in two different contexts (averaged over every 10 positions, e.g., [1,10],[11, 20],...), the values of CTRs are hidden for privacy.

6.4 Effectiveness over Different Positions (RQ3)

We investigate the effects of context bias over different positions to answer **RQ3**.

Most of the previous LTR models focus on the top-ranked positions, whereas we aim to improve the customer experience in relation to feeds recommendation even when customers browse a large number of products. Figure 4 shows the re-ranked results for the clicked products averaged over every 10 positions on the entire testing data set, where the x-axis represents the original positions of the clicked products and the y-axis denotes the positions after being re-ranked by the ULTR methods. A lower value indicates a better performance, and percentages on bars represent improvements of $ULTR-CP$ re-ranked positions over $ULTR-pos$ re-ranked positions. From Figure 4, we can observe that $ULTR-pos$ performs better than $ULTR-CP$ for products in the top 20 positions, while $ULTR-CP$ consistently outperforms $ULTR-pos$ for deeper positions. It is also consistent with the findings in Figure 1, which shows that the top 20 positions are more affected by position bias, whereas position bias does not have much influence on products in the deeper positions. Therefore, considering both context bias and position bias results in a better ranking performance, especially for products in deep positions in feeds recommendation.

6.5 Propensity Checking (RQ4)

We check the balancing property of estimated propensity scores under $ULTR-CP$ and $ULTR-pos$, respectively, to answer **RQ4**.

Table 2: Comparison of different weights based *MRR* between *ULTR – CP* and baseline models. The best number per row is bolded and * means statistically significance compared with SVM-rank based on t-test (p-values < 0.05).

	Date	09-17	09-18	09-19	09-20	09-21	09-22	09-23	09-24	09-25
CP – based MRR	*SVM – rank*	0.3689	0.3694	0.3647	0.3597	0.3628	0.3532	0.3523	0.3583	0.3607
	ULTR – cxt	0.3693	0.3696	0.3650	0.3599	0.3636	0.3542	0.3533	0.3565	0.3614
	ULTR – pos	0.3784*	0.3782*	0.3737*	0.3695*	0.3725*	0.3623*	0.3627*	0.3653*	0.3702*
	ULTR – CP	**0.3790***	**0.3786***	**0.3747***	**0.3708***	**0.3733***	**0.3634***	**0.3642***	**0.3663***	**0.3713***
Pos – based MRR	*SVM – rank*	0.3647	0.3657	0.3608	0.3554	0.3588	0.3594	0.3484	0.3514	0.3561
	ULTR – cxt	0.3645	0.3653	0.3606	0.3550	0.3590	0.3498	0.3489	0.3517	0.3581
	ULTR – pos	0.3748*	0.3747*	0.3704*	0.3653*	0.3687*	0.3587*	0.3590*	0.3612*	0.3660*
	ULTR – CP	**0.3748***	**0.3748***	**0.3709***	**0.3661***	**0.3691***	**0.3593***	**0.3600***	**0.3617***	**0.3666***
CTR – based MRR	*SVM – rank*	0.3437	0.3449	0.3399	0.3347	0.3378	0.3285	0.3280	0.3308	0.3349
	ULTR – cxt	0.3436	0.3444	0.3395	0.3342	0.3380	0.3289	0.3285	0.3310	0.3351
	ULTR – pos	0.3581*	0.3577*	0.3538*	0.3487*	0.3518*	0.3418*	0.3426*	0.3447*	0.3490*
	ULTR – CP	**0.3582***	**0.3579***	**0.3538***	**0.3494***	**0.3520***	**0.3423***	**0.3434***	**0.3451***	**0.3496***

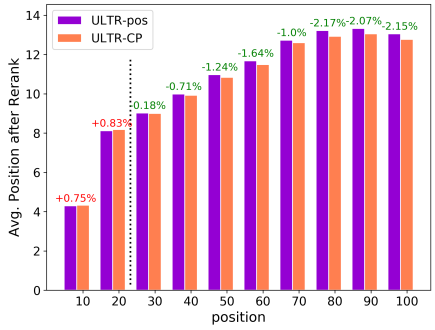


Figure 4: Re-ranked positions for *ULTR – pos* v.s. *ULTR – CP* (averaged over every 10 positions, e.g., [1, 10],[11, 20],...).

Given a propensity estimation model, we first divide the propensity scores into m blocks according to the range of their values. The propensity scores are ordered and then be divided evenly into m intervals, so that the propensity scores within each block are similar to each other. The larger the value of m, the closer the propensity score values in the same block. First, the products in the data set are divided into the same number of blocks according to their propensity scores. Second, within each block, we separate the products as treatment (clicked) and control groups (non-clicked). We then calculate the relative difference between the average ranks in these two groups. From Table 3, we observe that the relative differences in average rank under *ULTR – CP* are always smaller than that of *ULTR – pos*, which indicates the estimated propensity score under *ULTR – CP* achieves more balance. Moreover, with an increase of m, the relative differences in average rank under *ULTR – CP* and *ULTR – pos* both decrease correspondingly. Since there are fewer products in one block for smaller granularity, the difference becomes smaller. The value of m does not affect the relative comparisons between these two propensity estimation approaches. In summary, the estimated propensity score under *ULTR – CP* is more appropriate since the rank covariate achieves more balance than that of *ULTR – pos*.

Table 3: The relative differences of averaged rank values for *ULTR – CP* and *ULTR – pos* under different granularity.

m	5	10	15	20	25
ULTR – CP	0.73%	0.32%	0.53%	0.45%	0.36%
ULTR – pos	1.66%	1.05%	0.70%	0.65%	0.46%

6.6 Model Robustness Analysis (RQ5)

In this section, we aim to investigate the robustness of the proposed *ULTR – CP* model concerning the penalty factor C in SVM-Rank, so as to answer **RQ5**.

Specifically, we change the value of factor C, which has the greatest impact on performance, and then calculate the CTR-based *MRR* of the *ULTR – CP* model and the *ULTR – pos* model. As shown in Figure 5, the *ULTR – CP* outperforms the *ULTR – pos* in all settings when the value of parameter C changes from 1 to 100 under different granularity. Since SVM-rank has the worst performance which is far from the above two methods, so we did not show it in the figure. The experimental results demonstrate that the proposed *ULTR – CP* model, considering both position bias and context bias, is stable and robust to outperform *ULTR – pos*.

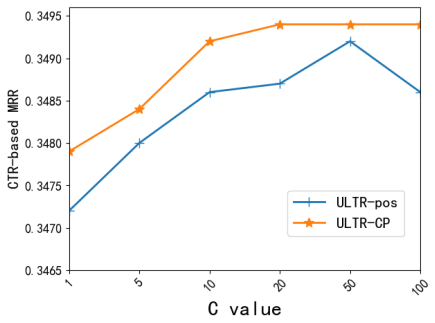


Figure 5: Performance of *ULTR – CP* and *ULTR – pos* models with different values of the penalty factor in SVM-Rank.

7 CONCLUSIONS

In this paper, through a preliminary analysis, we identify that multiple types of biases exist in the scenario of e-commerce feeds recommendation. To eliminate multiple biases simultaneously, the $ULTR$–CP framework is proposed, where combinational propensity weight is estimated for the unified bias. Specifically, we instantiate the $ULTR-CP$ framework in a context-aware position bias model. In this model, we assume that in addition to position and relevance, the user clicks are also determined by its surrounding context. Extensive experiments on a real e-commerce dataset verify the existence of context bias and show that $ULTR-CP$ outperforms the state-of-the-arts noticeably. The balancing checking results also directly confirm the advantage of introducing context bias. Moreover, the context bias concept and the proposed $ULTR-CP$ framework have a potential inspiration for other applications as well, e.g., web search ranking, whole page optimization, etc.

8 ACKNOWLEDGMENTS

This work is partially supported by National Natural Science Foundation of China through grants No.61902145, No.61976102 and No.U19A2065. We also acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

[1] A. Agarwal, X. Wang, C. Li, M. Bendersky, and M. Najork. 2019. Addressing Trust Bias for Unbiased Learning-to-Rank. In *Proceedings of the 28th international conference on World wide web*. 4–14.

[2] Aman Agarwal, Ivan Zaitsev, and Thorsten Joachims. 2018. Counterfactual Learning-to-Rank for Additive Metrics and Deep Models. *arXiv preprint arXiv:1805.00065* (2018).

[3] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating Position Bias without Intrusive Interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, pp: 474–482.

[4] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W Bruce Croft. 2018. Unbiased learning to rank with unbiased propensity estimation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 385–394.

[5] Grigor Aslanyan and Utkarsh Porwal. 2018. Direct Estimation of Position Bias for Unbiased Learning-to-Rank without Intervention. *Computing Research Repository* abs/1812.09338 (2018).

[6] Ricardo Baeza-Yates. 2016. Data and algorithmic bias in the web. In *Proceedings of the 8th ACM Conference on Web Science*. ACM, pp: 1–1.

[7] Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11, 23-581 (2010), 81.

[8] Olivier Chapelle and Ya Zhang. 2009. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web*. ACM, pp: 1–10.

[9] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 7, 3 (2015), pp: 1–115.

[10] Andrew Collins, Dominika Tkaczyk, Akiko Aizawa, and Joeran Beel. 2018. Position bias in recommender systems for digital libraries. In *Proceedings of the International Conference on Information*. Springer, pp: 335–344.

[11] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*. ACM, 87–94.

[12] Georges Dupret and Benjamin Piwowarski. 2008. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. 331–338.

[13] Leonardo Grilli and Carla Rampichini. 2011. Propensity scores for the estimation of average treatment effects in observational studies. *Training Sessions on Causal Inference, Bristol* (2011), pp: 28–29.

[14] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, Lixin Zou, Yiding Liu, and Dawei Yin. 2020. Deep Multifaceted Transformers for Multi-Objective Ranking in Large-Scale E-Commerce Recommender Systems. Association for Computing Machinery. https://doi.org/10.1145/3340531.3412697

[15] Ziniu Hu, Yang Wang, Qu Peng, and Hang Li. 2019. Unbiased LambdaMART: An Unbiased Pairwise Learning-to-Rank Algorithm. In *Proceedings of the 28th international conference on World wide web*. pp:2830–2836.

[16] Guido W Imbens and Donald B Rubin. 2015. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press.

[17] Rolf Jagerman, Harrie Oosterhuis, and Maarten de Rijke. 2019. To Model or to Intervene: A Comparison of Counterfactual and Online Learning to Rank from User Interactions. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. pp: 15–24.

[18] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems* 25, 2 (2007), pp: 7.

[19] Thorsten Joachims, Laura A Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR*, Vol. 5. pp: 154–161.

[20] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, pp: 781–789.

[21] Joseph DY Kang, Joseph L Schafer, et al. 2007. Demystifying double robustness: A comparison of alternative strategies for estimating a population mean from incomplete data. *Statistical science* 22, 4 (2007), pp: 523–539.

[22] Ching-Pei Lee and Chih-Jen Lin. 2014. Large-scale linear ranksvm. *Neural computation* 26, 4 (2014), pp: 781–817.

[23] Kristina Lerman and Tad Hogg. 2014. Leveraging position bias to improve peer recommendation. *PloS one* 9, 6 (2014), e98914.

[24] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. *Foundations and Trends in Information Retrieval* 3, 3, pp: 225–331.

[25] Claudio Lucchese, Franco Maria Nardini, Rama Kumar Pasumarthi, Sebastian Bruch, Michael Bendersky, Xuanhui Wang, Harrie Oosterhuis, Rolf Jagerman, and Maarten de Rijke. 2019. Learning to Rank in Theory and Practice: From Gradient Boosting to Neural Networks and Unbiased Learning. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. pp: 1419–1420.

[26] Maeve O'Brien and Mark T Keane. 2006. Modeling result–list searching in the World Wide Web: The role of relevance topologies and trust bias. In *Proceedings of the 28th annual conference of the cognitive science society*, Vol. 28. Citeseer, pp: 1881–1886.

[27] Harrie Oosterhuis and Maarten de Rijke. 2018. Differentiable Unbiased Online Learning to Rank. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. pp: 1293–1302.

[28] Z. Ovaisi, R. Ahsan, Y. Zhang, K. Vasilaky, and E. Zheleva. 2020. Correcting for Selection Bias in Learning-to-rank Systems. In *Proceedings of the 29th international conference on World wide web*. 1863–1873.

[29] Xin Rong. 2014. word2vec Parameter Learning Explained. *Computer Science* (2014).

[30] Paul R Rosenbaum and Donald B Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (1983), pp: 41–55.

[31] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2000. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*. ACM, pp: 158–167.

[32] Jeffrey A Smith and Petra E Todd. 2005. Does matching overcome LaLonde's critique of nonexperimental estimators? *Journal of econometrics* 125, 1-2 (2005), pp: 305–353.

[33] V Vapnik. 1998. Statistical Learning Theory. *Wiley, Chichester* GB (1998).

[34] Mengting Wan, Jianmo Ni, Rishabh Misra, and Julian McAuley. 2020. Addressing marketing bias in product recommendations. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. ACM.

[35] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, pp: 115–124.

[36] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, pp: 610–618.

[37] Fen Xia, Tie Yan Liu, Jue Wang, Wensheng Zhang, and Li Hang. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*.

[38] Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*. pp: 1201–1208.

[39] Hua Zheng, Dong Wang, Qi Zhang, Hang Li, and Tinghao Yang. 2010. Do clicks measure recommendation relevancy?: an empirical user study. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, pp: 249–252.