Counterfactual Learning to Rank using Heterogeneous Treatment Effect Estimation

Mucun Tian, Chun Guo, Vito Ostuni, Zhen Zhu
Pandora Media LLC
Oakland, CA
{mtian,cguo,vostuni,zzhu}@pandora.com

ABSTRACT

Learning-to-Rank (LTR) models trained from implicit feedback (e.g. clicks) suffer from inherent biases. A well-known one is the position bias — documents in top positions are more likely to receive clicks due in part to their position advantages. To unbiasedly learn to rank, existing counterfactual frameworks first estimate the propensity (probability) of missing clicks with intervention data from a small portion of search traffic, and then use inverse propensity score (IPS) to debias LTR algorithms on the whole data set. These approaches often assume the propensity only depends on the position of the document, which may cause high estimation variance in applications where the search context (e.g. query, user) varies frequently. While context-dependent propensity models reduce variance, accurate estimations may require randomization or intervention on a large amount of traffic, which may not be realistic in real-world systems, especially for long tail queries. In this work, we employ heterogeneous treatment effect estimation techniques to estimate position bias when intervention click data is limited. We then use such estimations to debias the observed click distribution and re-draw a new de-biased data set, which can be used for any LTR algorithms. We conduct simulations with varying experiment conditions and show the effectiveness of the proposed method in regimes with long tail queries and sparse clicks.

CCS CONCEPTS

• **Information systems → Learning to rank**.

KEYWORDS

position bias estimation, inverse propensity scoring, heterogeneous treatment effect

ACM Reference Format:
Mucun Tian, Chun Guo, Vito Ostuni, Zhen Zhu. 2020. Counterfactual Learning to Rank using Heterogeneous Treatment Effect Estimation. In *Proceedings of ACM SIGIR Workshop on eCommerce (SIGIR eCom'20).* ACM, New York, NY, USA, 9 pages.

1 INTRODUCTION

Learning-to-rank (LTR) models have been widely used in information retrieval and recommender systems. These models are often trained in the offline setting with implicit feedback (e.g. clicks) collected from production systems. While implicit feedback is an attractive training source (e.g. abundant, privacy preserving), its inherent biases hinder the effectiveness of learning-to-rank [21]. One such bias is the position bias. To mitigate position bias, traditional approaches have gone into modeling the bias-aware relevance [13, 14, 19]. However, accurately inferring individual relevance requires each query-document pair repeating multiple times at multiple positions, which is not realistic in many search systems. Instead of modeling individual relevance, recent counterfactual frameworks [4, 15, 22, 34] attempt to estimate the examination probability under Position-Based Model (PBM) [27] and use the estimation as inverse propensity score (IPS) to weight pairwise or listwise ranking. While IPS weighting provides an unbiased LTR under PBM, it has several limitations.

First, all the existing approaches follow a direct method to estimate propensities, which requires the same set of query-document pairs appearing in at least two different positions. This can be implemented by randomizing top-N [34], swapping pairs [22, 35], or integrating multiple loggers [4, 15]. For long tail queries, however, we rarely observe their intervention counterparts even with randomization, causing biased propensity estimations on these long tail queries. This could potentially break the unbiasedness of IPS weighted LTR. Second, the examination propensity can only be estimated on observed clicks when using the PBM model. In other words, we cannot infer from un-clicked documents whether these are irrelevant or not examined. Therefore, applying IPS into pointwise learning is not as effective as pairwise or listwise ones [35]. Third, IPS only weights clicked documents in the empirical loss function. Novel items (e.g. new music releases, products) for which we haven't observed any clicks, due to the lack of exposure to users, are treated as negative examples under IPS weighted LTR. This undermines LTR's ability to promote novel documents.

In this paper, we employ heterogeneous treatment effect (HTE) estimation methods [7, 23] to address these limitations. We first estimate causal effects of click probabilities between two positions. Based on these estimations, we then debias click distributions of observational data and draw clicks for unbiased LTR. Finally, we evaluate the effectiveness of the proposed method under varying experiment conditions using semi-synthetic data simulated from the Microsoft Learning to Rank dataset [26]. The objective of this work is twofold: i) compare the proposed heterogeneous treatment effect methodology to existing ones, ii) evaluate the effectiveness of the proposed method on long tail queries.

Estimating heterogeneous treatment effect does not require intervention data, nor the click model. Instead, it utilizes "collaborative information" in the feature space, allowing the position bias estimation for long tail queries. Under the unconfoundedness assumption

SIGIR eCom'20, July 30, 2020, Virtual Event, China

[29], the estimator is unbiased for any given context [7], so drawing from debiased click distributions can provide reliable clicks for documents with unknown relevance information in observational data.

2 RELATED WORK

There are several lines of research on estimating and/or debiasing click data for LTR. One approach is to *infer relevance with heuristics or modeling*, including *SkipAbove* [20], *Position-Based Model* (PBM) [27], the *Cascade* Model [14], and other extensions [13]. These approaches attempt to derive the absolute or relative relevance by taking into account users' search behavior. While relative relevance is found to be more accurate on average, LTR trained from this relevance is likely to reverse the presented order without additional heuristics [21, 22].

Another approach is *Online Learning* [31, 36]. Online learning is robust to bias and noise, but it learns from randomization data. This can hurt users' experience during the initial deployment stage [18].

Counterfactual LTR frameworks [2, 22, 34] seek to estimate how likely a document is to be examined and use the inverse propensity score (IPS) to weight pairwise or listwise LTR. Counterfactual LTR does not need randomization in the learning process and is proven to be unbiased under PBM [22]. However, it's sensitive to selection bias and noise [18]. Some attempts were made to keep the IPS estimator "doubly robust", such as adding an imputation term (regression on the complete data) [32], inclusion propensity (the propensity of a new document being exposed to users) [9], or noise-aware parameters [3]. The robustness of these estimators rely on the accurate imputation or noise modeling. These approaches also often assume the examination propensity only depends on the position. Several techniques have been proposed to estimate the context-dependent propensity [10, 15, 34], yet all of these require intervention data. There is another line of work that jointly estimates the relevance and the position bias [5, 16, 35] on observational data. But coupling the relevance and the bias together without controlling for either one of them calls into question the unbiasedness of the estimator [4, 15].

Recent advances in *heterogeneous treatment effect estimation* provide promising techniques for identifying individual treatment effect in observational studies. This line of work follows the potential outcomes framework [17, 30] to estimate the treatment effect using Robinson [28] transformation under the unconfoundedness assumption [29]. Künzel et al. [23] introduced meta-learners that indirectly predict the heterogeneous treatment effect using imputation on unobserved outcomes. Athey and Imbens [6] proposed recursive partitioning — namely *causal trees* — to assess heterogeneity in the treatment effect. A following work by Wager and Athey [33] developed *causal forests* to consistently estimate the true treatment effect. These tree-based methods require manually-designed criteria for parameter tuning due to the fundamental problem of causal inference — we observed an individual either in treatment or control group, but not both. Nie and Wager [24] proposed *R-learner* that separates confounding factors from the treatment effect estimator, enabling the traditional cross-validation for goodness-of-fit. This motivates *generalized random forests*'s [7] parameter tuning.

3 METHODS

In this section, we briefly revisit the existing counterfactual LTR framework, then introduce the heterogeneous treatment effect estimation and describe our experiment protocol.

3.1 Counterfactual LTR

Counterfactual LTR frameworks [2, 22, 34] assume that documents at higher positions are more likely to be examined by a user than ones at lower positions. Therefore, observed clicks are missing with certain propensities (probabilities) for documents at position k. Given these propensities, we can use inverse propensity score (IPS) technique to weight the positive examples for unbiased LTR.

3.1.1 Position-Based Propensity Estimation. The propensity is often unknown in practice. To estimate it, existing literature follows the position-based model (PBM) [27], which assumes the observed click $C \in \{0, 1\}$ depends on the examination $E \in \{0, 1\}$ and the relevance $R \in \{0, 1\}$ in the following way,

$$P(C = 1|X = x, P = k) \coloneqq P(E = 1|P = k) \cdot P(R = 1|X = x),$$

where $X \in \mathbb{R}^t$ is the feature vector that encodes the query, user and document, and $P \in \{1, ..., K\}$ is the document position. This assumes the examination E only depends on the position. We can also relax the examination to be context-dependent as

$$P(C = 1|X = x, P = k) \coloneqq P(E = 1|X = x, P = k) \cdot P(R = 1|X = x).$$

We then can fit the examination $\hat{f}_p(x, k)$ and the average relevance $\hat{f}_r(x)$ models with the intervention data to consistently estimate the propensity by minimizing the cross-entropy loss [15, 35],

$$\hat{f}_p(\cdot), \hat{f}_r(\cdot) \coloneqq \arg\min_{f_p, f_r} -\Bigg\{ \sum_i^N \sum_k^K \Big[y_k^i \log\Big(f_p(x^i, k) \cdot f_r(x^i)\Big) + (1 - y_k^i) \log\Big(1 - f_p(x^i, k) \cdot f_r(x^i)\Big)\Big]\Bigg\}, \tag{1}$$

where $i = 1, ..., N$ represents a unique tuple of (query, document, position) and y_k^i is the click rate of a unique query-document pair at position k.

3.1.2 IPS Weighted LTR. IPS weighting is found to be effective to pairwise or listwise LTR [35]. In the IPS weighted pairwise LTR setting, we have N of lists with size K, and we want to learn a score function $\hat{f}(\cdot)$ from the following loss,

$$\hat{f}(\cdot) \coloneqq \arg\min_f \left\{ \frac{1}{N} \sum_i^N \sum_k^K \frac{c_k^i}{p_k^i} \mathcal{L}(f(x_1^i), ..., f(x_K^i)) \right\},$$

where $c_k^i \in \{0, 1\}$ is the click of the kth document in a ranking list, p_k^i is the propensity score at position k, and $\mathcal{L}(\cdot)$ is the pairwise loss that approximates or bounds to a ranking metric (e.g. DCG, Relevance Rank) [2, 22].

3.1.3 Propensity Model Implementation. We select the contextual-dependent position-based model (CPBM) [15] as the contextual

propensity model[1]. Specifically, the examination model is implemented with a 3-layer neural network; the input $X \in \mathbb{R}^t$ corresponds to the context feature with size t, and the output $f_p(x,k) \in \mathbb{R}^K$ corresponds to the top-K positions to be estimated. The average relevance is also modeled by a 3-layer neural network; the input $X \in \mathbb{R}^t$ corresponds to the context feature with size t, and the output $f_r(x,k,k') \in \mathbb{R}^{K\times K}$ corresponds to the intervention sets for the $K \times K$ position pairs. An intervention set is composed of documents that appear at at least two different positions. To take into account the fact that the average relevance given the context $X = x$ for intervention documents at positions (k,k') is equal to the average relevance at positions (k',k), the output layer is the arithmetic mean of the previous output and its transpose, making the final output a symmetric matrix (see [15] for the details).

3.2 Heterogeneous Treatment Effect Estimation

Another way to achieve unbiased LTR is to place every document to the first position and collect the data for offline training. This is obviously impractical, so we seek to unbiasedly estimate the conditional incremental effect — how much is the increase of the click probability if a document would have been in the first position had it been in position $k, k \neq 1$. With the estimation, we can compensate the click probability of the document at position k during the offline training.

3.2.1 Heterogeneous Treatment Effect Estimation. To estimate the conditional incremental effect, we employ the potential outcome framework [30] to formulate this problem. In LTR, we observe N of i.i.d. examples $(X^i, Y^i, P^i), i = 1, ..., N$, where $X^i \in \mathbb{R}^t$ is query-document feature, $Y^i \in \mathbb{R}$ is the observed outcome (e.g. click, grade), and $P^i \in \{0,1\}$ is the treatment variable, indicating whether a document is observed in position k ($k \neq 1$) or 1; $P^i = 1$, if the position is 1; $P^i = 0$, if the position is k. We assume there are potential outcomes $\{Y(1), Y(0)\}$, corresponding to the treatment or control group, so $Y^i = Y^i(1)$, if $P^i = 1$; otherwise, $Y^i = Y^i(0)$. We then want to estimate the conditional average treatment effect (CATE) between position 1 and k,

$$\tau_k^*(x) \coloneqq \mathbb{E}[Y(1) - Y(0) \mid X = x].$$

To estimate CATE, we assume unconfoundedness given any specific context [29],

$$\{Y(1), Y(0)\} \perp P \mid X = x. \tag{2}$$

Intuitively, this assumes data points surrounding a specific context, $X = x$, are missing at random so that we can estimate $\tau_k^*(x)$ without bias.[2]

3.2.2 Click Distribution Correction and Data Resampling. With the estimator $\hat{\tau}_k(x)$ in hand, we compute a potential click rate at position 1 for each unique query-document pair $(X^i = x)$ observed at position k by

$$\hat{CTR}_1(X^i) = CTR_k^{obs}(X^i) + \hat{\tau}_k(X^i), \tag{3}$$

where $CTR_k^{obs}(X^i)$ is the observed click rate of query-document pair with feature X^i at position k.

Based on (3), we re-draw N^{obs} clicks/non-clicks for each unique query-document pair $(X^i = x)$ from

$$\hat{C}_{X^i} \sim \text{Bernoulli}(\theta^i),\ \theta^i = \hat{CTR}_1(X^i),$$

where N^{obs} is the number of query-document pair $(X^i = x)$ observed in the training data. We truncate θ^i to the range $[0,1]$ before sampling clicks.

In this way, we reconstruct clicks as if we would have been putting each document in the first position. A nice property of our approach is that relevant documents without any observed clicks due to position bias can now become positive training examples after the correction and resampling. This is particularly useful for long tail queries and long results lists. However, this is not achievable using existing counterfactual framework since IPS only weights the clicked documents.

3.2.3 Heterogeneous Treatment Effect Implementation. In the following we describe the two methods we used to estimate the heterogeneous treatment effect $\tau_k(x)$: *causal forests* [7, 33] and *X-Learner* [23].

Causal forests [33] estimates the heterogeneous treatment effect at each leaf node L by

$$\begin{aligned}\hat{\tau}_k(x) = &\frac{1}{|\{i : P^i = 1, X^i \in L\}|} \sum_{\{i:P^i=1, X^i\in L\}} Y^i \\ &- \frac{1}{|\{i : P^i = 0, X^i \in L\}|} \sum_{\{i:P^i=0, X^i\in L\}} Y^i.\end{aligned} \tag{4}$$

Causal forests [33] recursively solves (4) and selects a cut in the feature space that maximizes the difference of $\hat{\tau}_k(x)$ between two child nodes. To speed up the tree building process, [7] uses the gradient method to optimize a linear approximation of the difference (see [7] for the details). We train $K-1$ causal forests to estimate the bias in top-K positions.

X-Learner [23] estimates the heterogeneous treatment effect by fitting models on the imputed outcomes. It has three steps:

(1) Fit two regression models, $\hat{\mu}_0(x)$ and $\hat{\mu}_1(x)$, to estimate the average outcomes of the treatment and control group, $\mu_0(x) = \mathbb{E}[Y(0)|X = x]$ and $\mu_1(x) = \mathbb{E}[Y(1)|X = x]$, respectively.
(2) Impute the individual treatment effects in the treatment and control group by $\tilde{D}_0^i \coloneqq \hat{\mu}_1(X_0^i) - Y_0^i$ and $\tilde{D}_1^i \coloneqq Y_1^i - \hat{\mu}_0(X_1^i)$ and fit two regression models, $\hat{\tau}_k^0(x)$ and $\hat{\tau}_k^1(x)$, to estimate the imputed treatment effects, $\tau_k^0(x) = \mathbb{E}[\tilde{D}_0|X = x]$ and $\tau_k^1(x) = \mathbb{E}[\tilde{D}_1|X = x]$, respectively.
(3) Estimate the heterogeneous treatment effect by

$$\hat{\tau}_k(x) = g(x)\hat{\tau}_k^0(x) + (1 - g(x))\hat{\tau}_k^1(x),$$

where $g(x) \in [0,1]$ is often set to the propensity of treatment assignment, that is $g(x) = P(P = 1|X = x)$.

[1] This was implemented using Tensorflow [1].
[2] In practice, this assumption can be met by conducting randomization experiments. For example, we map each unique query to a random seed, and then we randomly shuffle the top-K list based on this random seed. In this way, the ranking is still deterministic, but documents' positions now are independent of clicks they will receive. This reduces the harm to the user experience compared to full randomization.

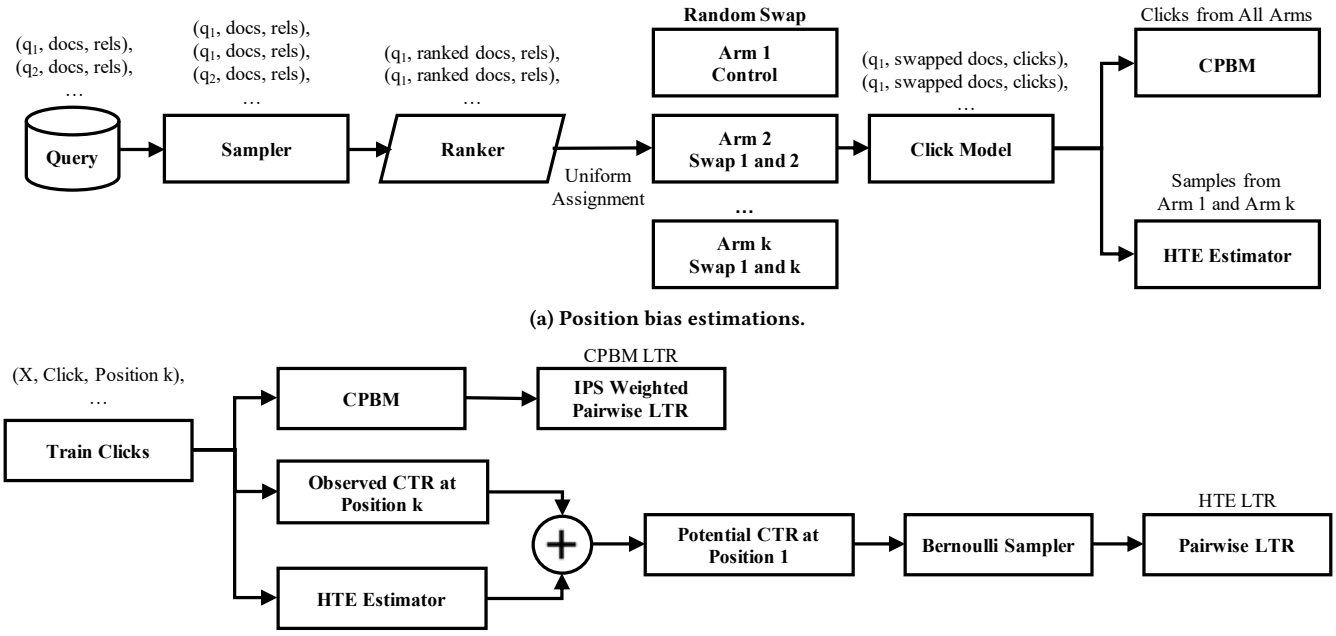

Figure 1: Simulation architecture.

Similarly, we train $K-1$ X-Learners for top-K positions using Causal ML [11]. We select the tree boosting method [12] as the base regression models.

3.3 Simulation Protocol

To evaluate the accuracy of position bias estimations and the ranking effectiveness, we simulate the entire search system, intervention experiments, and model training and evaluation. Figure 1 shows our simulation architecture. We begin by generating intervention clicks and estimating position bias (Figure 1a), and then apply bias estimations to unbiased learning to rank (Figure 1b). In the following we detail the different steps and components of the simulation architecture.

3.3.1 Query Sampler. We use the Microsoft Learning-to-rank data set [26] as the query corpus since it provides the real-world context features and human annotated 5-grade relevance, $rel \in \{0, 1, 2, 3, 4\}$. The data set contains 31K unique queries and their corresponding candidate documents, and it is split into train (60%), validation (20%), and test sets (20%). To simulate the long tail and popular search queries, we draw uniformly from the query corpus with different sample sizes. We also train a linear pairwise ranking with 1% of the train and validation queries to simulate the production ranker. For each incoming query, the production ranker outputs a ranked list for further experiments.

3.3.2 Intervention Simulation. Recent work collects intervention clicks from randomization [22, 34, 35] or multiple loggers [4, 15]. For the purpose of comparison between the proposed method with the existing propensity estimation methods, we adopt an intervention simulation that is similar to the randomized swap [22, 35]. We randomly assign each ranked query list to one of K arms to create intervention rankings for position bias estimations. In the control group (Arm-1), incoming query lists remain unchanged; in Arm-k, we always swap the document at the first position with the one at the kth position. The random swap assignment creates not only the intervention sets but also the unconfoundedness equation (2), so that we can compare the two methods without violating their own assumptions.

3.3.3 Click Model. Before generating clicks for query lists, we follow [22] to binarize the 5-grade relevance by setting $R = 1$, if $rel = 3, 4$, and $R = 0$, if $rel = 0, 1, 2$, and truncate lists to top-10. We then model the contextual examination by $P(E = 1|X = x, P = k) := \frac{1}{k^{\max(w \cdot x + 1, 0)}}$ [15], w is drawn uniformly from $[-1, 1)$. To select the context features, we trained a random forests [8] from the train and validation queries with normalized features and binary relevance, and we selected the top-10 important features as our context features[3]. We also add click noise to the final click model by $P(C = 1|E = 1, R = 1) = 1; P(C = 1|E = 1, R = 0) = 0.1$ [15, 18, 22], modeling that a user can mistakenly click an irrelevant document after examining the document.

3.3.4 Position Bias Estimation. To feed CPBM and HTE Estimator (i.e. Causal Forests and X-Learner), we process the simulation clicks in the two following ways:

- CPBM, merge clicks from all arms, select query-document pairs shown at least two positions (intervention sets) [4, 15],

[3]These context features encode the dependency between relevance and the context. Reducing the feature number in simulation also make the examination model more stable to produce enough variations in examination values.

and estimate the examination propensity using the method described in 3.1.
- HTE Estimator, combine clicks between Arm-1 and Arm-k, sample one position randomly from intervention sets for each unique query-document pair shown at position 1 and/or k, compute click rate at the sampled position, and estimate the heterogeneous treatment effect between position 1 and k with the method described in 3.2.

From the data process, we can see that not all the interventions are required for fitting HTE estimators. We train models on the simulation clicks generated from the train and validation sets of the query corpus. The training features are the same as the ones selected in the click model described in section 3.3.3. After this phase, we have now trained CPBM and HTE Estimator. These models will be used in the next phase in Figure 1b.

3.3.5 Unbiased Learning-to-rank. Figure 1b illustrates the architecture of unbiased LTR. We implemented two types of LTR models for the comparison. The main steps are:

- CPBM LTR, estimate the examination propensity by CPBM and use the inverse propensity to weight the pairwise LTR.
- HTE LTR, compute click rate at the observed position, estimate the heterogeneous treatment effects with HTE estimators trained in the previous phase, compute the potential click rate at position 1 by the sum of the observed click rate and the treatment effect estimated, draw click/non-click by the Bernoulli distribution with the parameter of the potential click rate at position 1, and train the pairwise LTR.

The LTR models were implemented using the *tensorflow-ranking* [25] library. We used a linear scoring function, pairwise hinge loss and L2 regularization. We used all features in the query corpus to train LTR models. Hyper-parameter tuning was conducted on the validation sets.

3.3.6 Evaluation Metrics. We evaluate the accuracy of the position bias estimation by computing the Root Mean Square Error (RMSE) between the estimated $\hat{\tau}_k(X_i)$ and the true $\tau_k^*(X_i)$ on queries in the test set for top-10 positions. CPBM does not output $\hat{\tau}_k(x)$ directly. Instead, it predicts the examination probability, $\hat{f}_p(x,k)$, and the average relevance, $\hat{f}_r(x,k,k')$, separately. We compute $\hat{\tau_k}(X_i)$ under CPBM by

$$\hat{\tau_k}(X_i) = \hat{f_p}(X_i,1)\cdot\hat{f_r}(X_i,1,k) - \hat{f_p}(X_i,k)\cdot\hat{f_r}(X_i,k,1).$$

To evaluate ranking effectiveness we computed nDCG@10 on the test set and used binary relevance. We rerun 3 times of the entire simulation experiment, including simulation click generation, position bias estimations, and LTR training and evaluations. [4]

4 RESULTS

In this section, we detail and analyze the experiment results comparing CPBM and the proposed methodology based on heterogeneous treatment effect estimation methods.

[4]Full experiment code is publicly available at https://github.com/KimuraTian/sigir-eCom20-counterfactual-ltr-using-hte.

4.1 Position Bias Estimation

Table 1 shows RMSE of heterogeneous treatment effect estimations on the test set; models with the best RMSE are in bold. The RMSE gives us a picture of how accurate the estimator is able to capture the heterogeneity of the position bias. Among the three estimation methods, X-Learner has the largest estimation error. This may be because X-Learner is often more efficient when there is imbalance between the treatment and control groups or structural assumptions on heterogeneous treatment effects [23]. However, our experiment has balanced treatment and control groups, and the structure of the true relevance is unknown. This may suggest that future work will consider the ratio of treatment to control group and base learners for specific applications when picking HTE estimators. Under the extremely sparse condition (the percentage of training queries = 1% and avg. searches per query = 5), causal forests method exhibits the smallest estimation errors. This is to be expected, since causal forests can utilize the similarities in the context feature spaces, reducing the high sample variance faced by CPBM due to the lack of intervention clicks. As we increase the density of intervention data points, CPBM has the best estimation accuracy. We conjecture this is related to the fact that causal forests uses the average value of the data points in the leaf node to make predictions, which introduces prediction noise when there is higher feature heterogeneity in the leaf node. When a large amount of intervention clicks is available, all the three algorithms tend to converge to a comparable level of estimation accuracy.

Figure 2 shows RMSE of heterogeneous treatment effect estimations for true relevant ($R = 1$) and irrelevant ($R = 0$) documents. For relevant documents, CPBM achieves the best RMSE except for the extremely sparse condition (avg. searches/query = 5 and percentage of training queries = 1%), while both HTE estimators are comparable to each other. For irrelevant documents, Causal Forests has the best estimation accuracy. As our simulation modeled click noise, heterogeneous treatment effects for irrelevant documents are probabilities of misclicks. Although a good estimator should accurately capture heterogeneous treatment effects for both relevant and irrelevant documents, we speculate that the ranking performance of HTE LTR is less sensitive to the estimation noise of irrelevant documents compared to IPS weighted LTR, when click noise is not high. We think this is also related to the fact that misclicked irrelevant documents contribute to a small proportion of the clicked documents (e.g. maximum 0.1 under our simulation). Furthermore, in our HTE LTR approach the debiased clicks are generated by resampling, while the IPS weighting may amplify misclicks.

4.2 Unbiased LTR

Figure 3 shows the box plot of ranking accuracy results while Table 2 shows ranking accuracy measured with nDCG@10. Overall, HTE LTR approaches (Causal Forests LTR and X-Learner LTR) outperform the IPS weighted LTR baseline (CPBM LTR). There are two possible reasons: i) noisy propensity estimations cause extremely high IPS when propensity scores are close to 0, which further explodes the weights of IPS weighted LTR loss function, making learning process unstable; ii) the Bernoulli sampling can create positive clicks for documents without any observation clicks. To mitigate the high variance problem encountered by IPS weighted LTR, we

Table 1: Position bias estimations RMSE.

	Avg. Searches/Query = 5				Avg. Searches/Query = 10				Avg. Searches/Query = 25				Avg. Searches/Query = 50			
Percentage of Training Queries	1%	10%	50%	100%	1%	10%	50%	100%	1%	10%	50%	100%	1%	10%	50%	100%
CPBM	0.360	**0.256**	**0.239**	**0.237**	0.325	**0.249**	**0.239**	**0.238**	**0.304**	**0.246**	**0.238**	**0.237**	**0.295**	**0.245**	**0.239**	**0.237**
Causal Forests	**0.335**	0.276	0.255	0.249	**0.317**	0.270	0.251	0.246	0.310	0.266	0.250	0.245	0.307	0.265	0.249	0.244
X-Learner	0.374	0.315	0.275	0.260	0.350	0.302	0.265	0.253	0.341	0.291	0.260	0.249	0.341	0.286	0.256	0.247

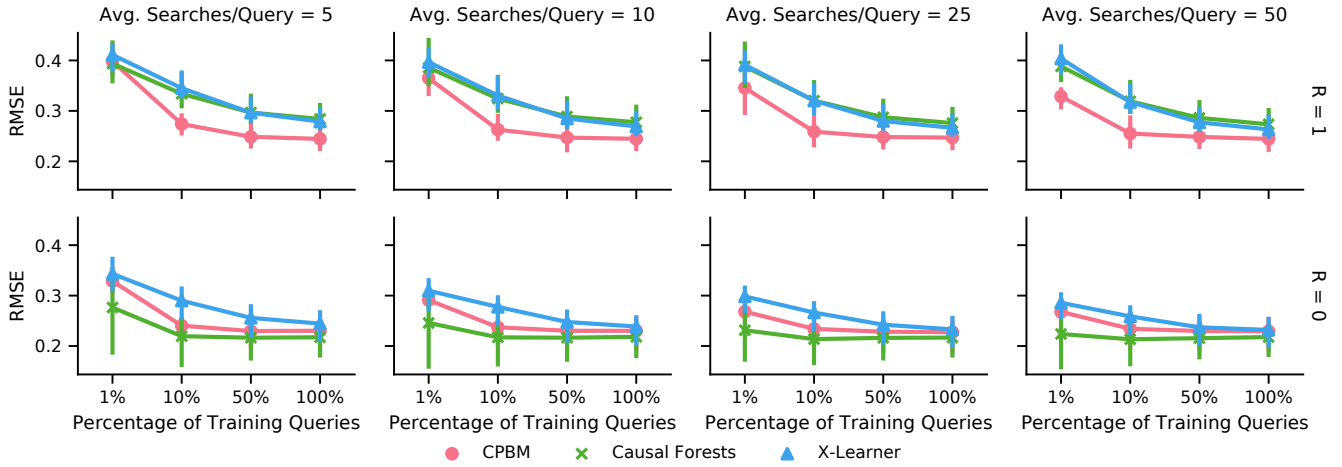

Figure 2: Position bias estimations RMSE, as $\sqrt{\frac{1}{N}\sum_i\sum_k(\hat{\tau}_k(X^i)-\tau_k^*(X^i))^2}$. The columns and rows are the average searches per query and the true relevance of documents, respectively. X-axis represents the percentage of training queries used in the intervention simulation.

trained a IPS weighted LTR with truncated propensity estimations in the range $[0.01, 1)$ (CPBM Clipped IPS LTR). CPBM Clipped IPS LTR improved the ranking performance over the baseline in some cases (e.g. avg. searches/query = 10 and percentage of training queries = 10%, 50%, 100%, avg. searches/query = 25 and percentage of training queries = 1%, 10%), but it does not beat HTE LTR approaches. When clicks become abundant (i.e. avg. searches/query = 50), the performance of IPS weighted LTRs is significantly improved. For example, CPBM LTR beats Causal Forests LTR when 100% of training queries are used for the propensity estimation. This may suggest that in applications where a large amount of intervention clicks is available, CPBM LTR is preferred as it requires simpler pre-processing. But when the lack of clicks is a main concern (e.g. personal file search, long tail queries), heterogeneous treatment effect estimation LTR methods may be better alternatives. Within the HTE LTR group, Causal Forests LTR and X-Learner LTR have similar ranking performance (cases when $p > 0.05$,[5] the percentage of training queries = 1%, 10%, 100% and avg. searches per query = 5, the percentage of training queries = 10% and avg. searches/query = 10).

4.3 Position Bias Results Analysis

The core idea of counterfactual LTR is to estimate the bias in observational data to debias the training data and learn a better LTR model. To better understand the impact of the bias estimation on the ranking performance, we look at the distribution of corrected CTR and observed CTR in the training data. Figure 4 illustrates distribution of click rate for relevant documents and irrelevant documents. True Model CTR is the click probability defined in our click model (encoded with position bias); Observation CTR is observation CTR computed from simulation clicks; Causal Forests Theta Truncated and X-Leaner Theta Truncated are corrected CTR (θ of the Bernoulli sampling); CPBM IPS Weighted Observation CTR is the multiplication of IPS estimated by CPBM and observation CTR; the horizontal line marks $CTR = 0.5$. For relevant documents, we expect corrected CTR is as close as to 1 since these are true relevant. For irrelevant documents, we expect corrected CTR to be as low as 0 and not over-estimate the false CTR. By looking at the bottom part of Figure 4, we can see that the observation CTR almost covers the range $[0, 1]$ (high position bias), but it approaches to 1 after CTR adjustments. HTE LTR methods exhibit this property for all simulation conditions, while IPS weighted LTR only works well when there are more clicks available (small variance when avg. searches/query = 25, 50). On the other hand (top part of Figure 4), we can see that HTE estimators, while having larger estimation variance on irrelevant documents, can still confine the corrected CTR of the majority of irrelevant documents under a certain level (e.g. 0.5), making the noisy estimation less detrimental to LTR. As long as relevant documents get sampled much more often than

[5]T-test with unequal variance on data from three runs.

Table 2: Ranking performance measured by nDCG@10. Notation $*$, $**$, and $***$ mean statistically significant with $p < 0.5$, $p < 0.01$, and $p < 0.001$, respectively, compared to CPBM LTR.

	Avg. Searches/Query = 5				Avg. Searches/Query = 10				Avg. Searches/Query = 25				Avg. Searches/Query = 50			
Percentage of Training Queries	1%	10%	50%	100%	1%	10%	50%	100%	1%	10%	50%	100%	1%	10%	50%	100%
CPBM LTR	0.77	0.80	0.78	0.76	0.78	0.77	0.77	0.76	0.77	0.77	0.79	0.77	0.80	0.81	0.82	0.82
CPBM Clipped IPS LTR	0.77	0.78***	0.78	0.75**	0.77***	0.78***	0.79***	0.78***	0.78*	0.80***	0.77***	0.77	0.83***	0.79***	0.82	0.79***
Causal Forests LTR	0.82***	0.81***	0.81***	0.81***	**0.82*****	0.81***	0.81***	0.82***	0.83***	0.82***	0.82***	**0.84*****	0.81***	**0.85*****	**0.84*****	0.79***
X-Learner LTR	**0.82*****	**0.82*****	**0.82*****	**0.81*****	0.81***	**0.81*****	**0.82*****	**0.84*****	**0.84*****	**0.84*****	**0.85*****	0.82***	**0.84*****	0.83***	0.83**	**0.86*****

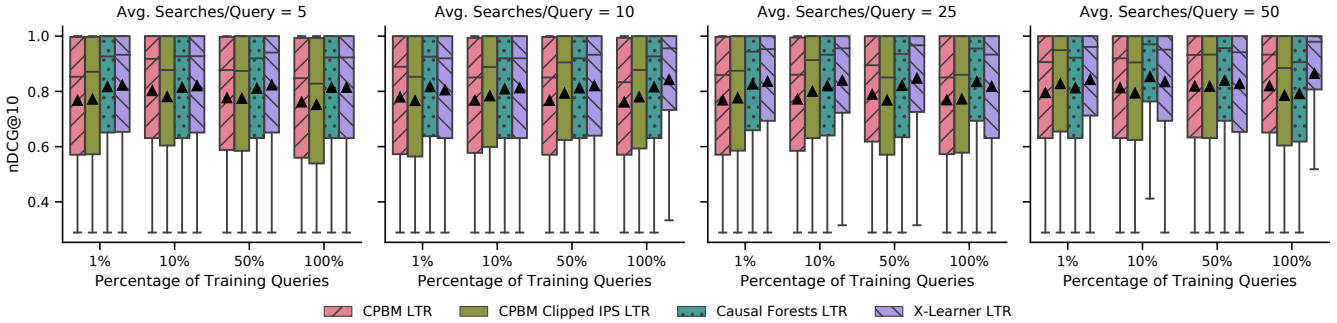

Figure 3: Ranking performance of unbiased LTRs.

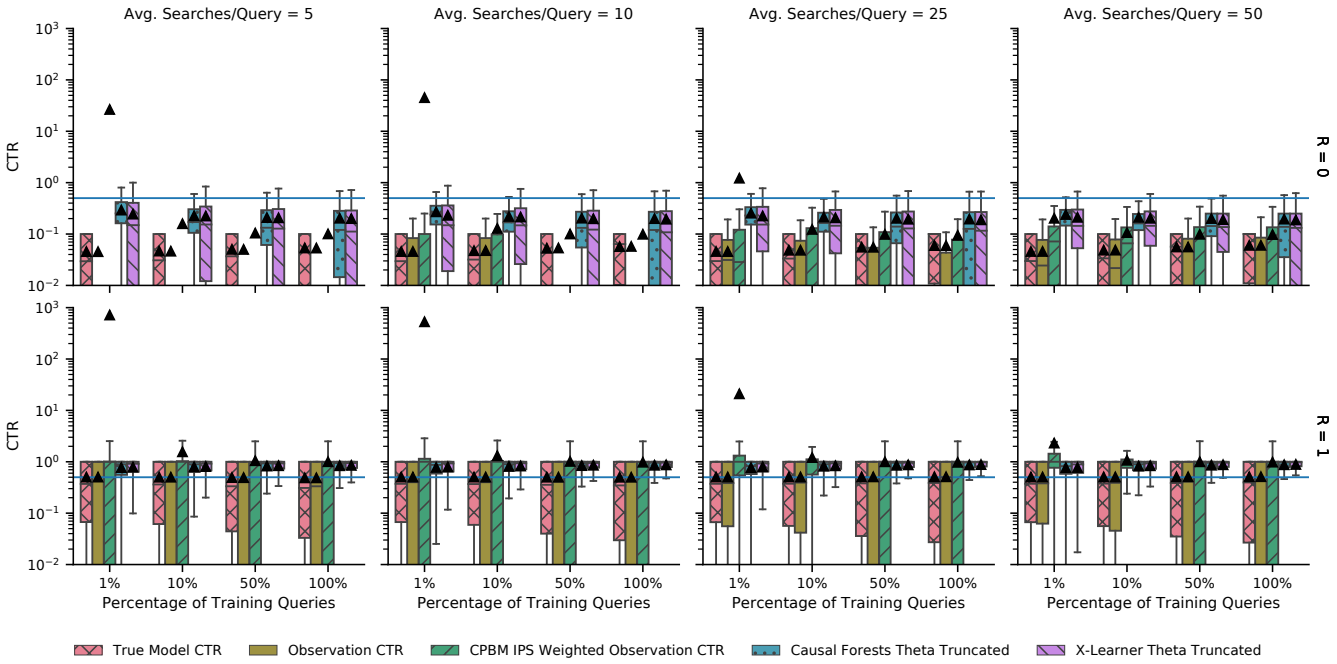

Figure 4: Distribution of click rate in the training data.

irrelevant documents, pairwise LTR can still differentiate relevant documents with irrelevant ones effectively.

We also investigate how counterfactual LTR helps with debiasing observational data. Specifically, we focus on two cases: False Negative (Relevant documents without clicks) and False Positive (Misclicks) in observational data. Figure 5 and Figure 6 show the distributions. From Figure 5, HTE LTR recovers the majority of unobserved relevant documents through the click distribution correction. However, IPS weighted LTR does not help much, since it only weights observed clicks (always 0 for unobserved ones). For

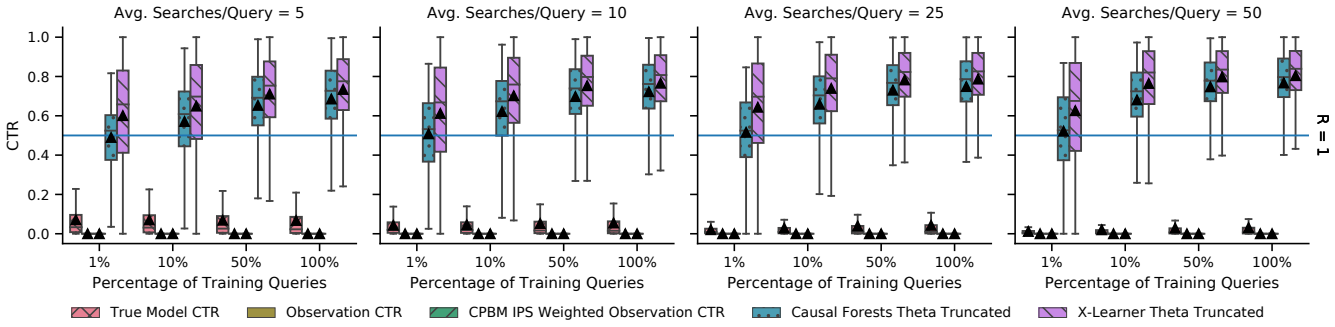

Figure 5: Distribution of click rate for unobserved true relevant documents.

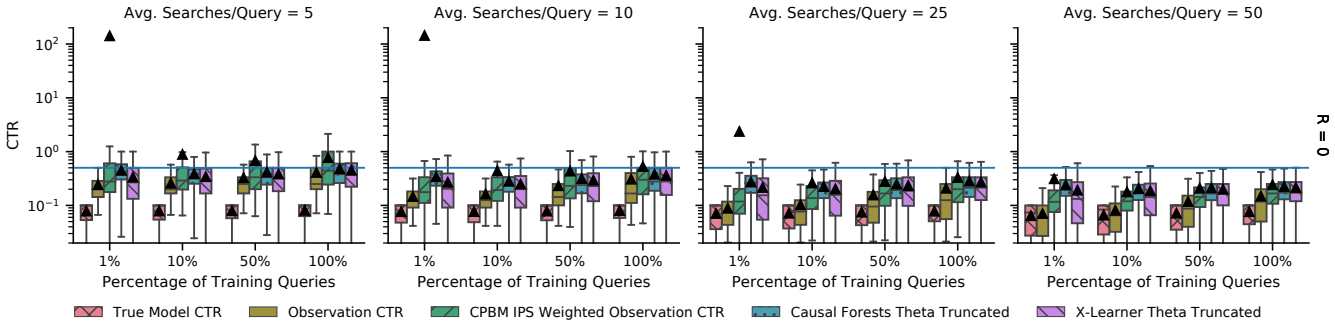

Figure 6: Distribution of click rate for mis-clicked true irrelevant documents.

the false positive case (Figure 6), large estimation variance (e.g. extremely high means when avg.searches/query <=25 and percentage of training queries = 1%) could be detrimental to IPS weighted LTR as it may severely amplify the noise.

5 CONCLUSIONS AND FUTURE WORK

In this work we show how heterogeneous treatment effect estimation techniques can be used to address the position bias in search results ranking. To utilize estimated incremental causal effects for unbiased LTR, we drew clicks from debiased click distributions of observational data. We compared the proposed method with an existing contextual position-based model [15] under varying simulation conditions. Our results showed that the heterogeneous treatment effect estimation method is particularly effective for long tail queries with high click sparsity.

The usage of sampled clicks is not limited to pairwise LTR. It would be interesting to see how it performs when applied to other LTR methods. There is a variety of estimation methods for heterogeneous treatment effect, such as T-learner [23], R-learner [24]. In the future, we can explore the multivariate extension of R-learner [24] so that we only need to build a single model for estimating bias in multiple positions instead of multiple models with the binary treatment indicator. The treatment position we chose in this work is the first position, we conjecture that the estimation could be improved if we extend to other anchor positions when there is huge imbalance of clicks across different positions. The simulation of user search behavior we used in this work (uniform sampling) is simple, future work may capture complex user behavior using other sampling distribution (e.g. Pareto distribution).

ACKNOWLEDGMENTS

We thank Tao Ye, Jenny Lin, Oliver Bembom, Filip Korzeniowski, Ali Goli, Oscar Celma, and the Pandora Science Team, for their support.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.

[2] Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. 2019. A General Framework for Counterfactual Learning-to-Rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 5–14. https://doi.org/10.1145/3331184.3331202

[3] Aman Agarwal, Xuanhui Wang, Cheng Li, Michael Bendersky, and Marc Najork. 2019. Addressing Trust Bias for Unbiased Learning-to-Rank. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 4–14. https://doi.org/10.1145/3308558.3313697

[4] 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* (Melbourne VIC, Australia) *(WSDM '19)*. Association for Computing Machinery, New York, NY, USA, 474–482. https://doi.org/10.1145/3289600.

3291017
[5] 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* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 385–394. https://doi.org/10.1145/3209978.3209986
[6] Susan Athey and Guido Imbens. 2016. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences of the United States of America* 113, 27 (2016), 7353–7360. https://www.jstor.org/stable/26470691
[7] Susan Athey, Julie Tibshirani, and Stefan Wager. 2019. Generalized random forests. *Ann. Statist.* 47, 2 (04 2019), 1148–1178. https://doi.org/10.1214/18-AOS1709
[8] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[9] Ben Carterette and Praveen Chandar. 2018. Offline Comparative Evaluation with Incremental, Minimally-Invasive Online Feedback. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 705–714. https://doi.org/10.1145/3209978.3210050
[10] Praveen Chandar and Ben Carterette. 2018. Estimating Clickthrough Bias in the Cascade Model. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 1587–1590. https://doi.org/10.1145/3269206.3269315
[11] Huigang Chen, Totte Harinen, Jeong-Yoon Lee, Mike Yung, and Zhenyu Zhao. 2020. CausalML: Python Package for Causal Machine Learning. arXiv:cs.CY/2002.11631
[12] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785
[13] 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), 1–115. https://doi.org/10.2200/S00654ED1V01Y201507ICR043 arXiv:https://doi.org/10.2200/S00654ED1V01Y201507ICR043
[14] 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* (Palo Alto, California, USA) *(WSDM '08)*. Association for Computing Machinery, New York, NY, USA, 87–94. https://doi.org/10.1145/1341531.1341545
[15] Zhichong Fang, Aman Agarwal, and Thorsten Joachims. 2019. Intervention Harvesting for Context-Dependent Examination-Bias Estimation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 825–834. https://doi.org/10.1145/3331184.3331238
[16] Ziniu Hu, Yang Wang, Qu Peng, and Hang Li. 2019. Unbiased LambdaMART: An Unbiased Pairwise Learning-to-Rank Algorithm. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 2830–2836. https://doi.org/10.1145/3308558.3313447
[17] Guido W. Imbens and Donald B. Rubin. 2015. *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction.* Cambridge University Press. https://doi.org/10.1017/CBO9781139025751
[18] 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* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 15–24. https://doi.org/10.1145/3331184.3331269
[19] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Edmonton, Alberta, Canada) *(KDD '02)*. Association for Computing Machinery, New York, NY, USA, 133–142. https://doi.org/10.1145/775047.775067
[20] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Salvador, Brazil) *(SIGIR '05)*. Association for Computing Machinery, New York, NY, USA, 154–161. https://doi.org/10.1145/1076034.1076063
[21] 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 Trans. Inf. Syst.* 25, 2 (April 2007), 7–es. https://doi.org/10.1145/1229179.1229181
[22] 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* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 781–789. https://doi.org/10.1145/3018661.3018699
[23] Sören R. Künzel, Jasjeet S. Sekhon, Peter J. Bickel, and Bin Yu. 2019. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the National Academy of Sciences* 116, 10 (2019), 4156–4165. https://doi.org/10.1073/pnas.1804597116 arXiv:https://www.pnas.org/content/116/10/4156.full.pdf
[24] Xinkun Nie and Stefan Wager. 2017. Quasi-Oracle Estimation of Heterogeneous Treatment Effects. arXiv:stat.ML/1712.04912
[25] Rama Kumar Pasumarthi, Sebastian Bruch, Xuanhui Wang, Cheng Li, Michael Bendersky, Marc Najork, Jan Pfeifer, Nadav Golbandi, Rohan Anil, and Stephan Wolf. 2019. TF-Ranking: Scalable TensorFlow Library for Learning-to-Rank. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2970–2978. https://doi.org/10.1145/3292500.3330677
[26] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013). http://arxiv.org/abs/1306.2597
[27] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting Clicks: Estimating the Click-through Rate for New Ads. In *Proceedings of the 16th International Conference on World Wide Web* (Banff, Alberta, Canada) *(WWW '07)*. Association for Computing Machinery, New York, NY, USA, 521–530. https://doi.org/10.1145/1242572.1242643
[28] P. M. Robinson. 1988. Root-N-Consistent Semiparametric Regression. *Econometrica* 56, 4 (1988), 931–954. http://www.jstor.org/stable/1912705
[29] PAUL R. ROSENBAUM and DONALD B. RUBIN. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (04 1983), 41–55. https://doi.org/10.1093/biomet/70.1.41 arXiv:https://academic.oup.com/biomet/article-pdf/70/1/41/662954/70-1-41.pdf
[30] Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology* 66, 5 (1974), 688.
[31] Anne Schuth, Harrie Oosterhuis, Shimon Whiteson, and Maarten de Rijke. 2016. Multileave Gradient Descent for Fast Online Learning to Rank. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining* (San Francisco, California, USA) *(WSDM '16)*. Association for Computing Machinery, New York, NY, USA, 457–466. https://doi.org/10.1145/2835776.2835804
[32] Yi Su, Lequn Wang, Michele Santacatterina, and Thorsten Joachims. 2019. CAB: Continuous Adaptive Blending for Policy Evaluation and Learning. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.), Vol. 97. PMLR, Long Beach, California, USA, 6005–6014. http://proceedings.mlr.press/v97/su19a.html
[33] Stefan Wager and Susan Athey. 2018. Estimation and Inference of Heterogeneous Treatment Effects using Random Forests. *J. Amer. Statist. Assoc.* 113, 523 (2018), 1228–1242. https://doi.org/10.1080/01621459.2017.1319839 arXiv:https://doi.org/10.1080/01621459.2017.1319839
[34] 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* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 115–124. https://doi.org/10.1145/2911451.2911537
[35] 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* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 610–618. https://doi.org/10.1145/3159652.3159732
[36] 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* (Montreal, Quebec, Canada) *(ICML '09)*. Association for Computing Machinery, New York, NY, USA, 1201–1208. https://doi.org/10.1145/1553374.1553527