Unifying Online and Counterfactual Learning to Rank

A Novel Counterfactual Estimator that Effectively Utilizes Online Interventions

Harrie Oosterhuis
Radboud University
Nijmegen, The Netherlands
harrie.oosterhuis@ru.nl

Maarten de Rijke
University of Amsterdam & Ahold Delhaize Research
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

Optimizing ranking systems based on user interactions is a well-studied problem. State-of-the-art methods for optimizing ranking systems based on user interactions are divided into online approaches – that learn by directly interacting with users – and counterfactual approaches – that learn from historical interactions. Existing online methods are hindered without online interventions and thus should not be applied counterfactually. Conversely, counterfactual methods cannot directly benefit from online interventions.

We propose a novel intervention-aware estimator for both counterfactual and online Learning to Rank (LTR). With the introduction of the intervention-aware estimator, we aim to bridge the online/-counterfactual LTR division as it is shown to be highly effective in both online and counterfactual scenarios. The estimator corrects for the effect of position bias, trust bias, and item-selection bias by using corrections based on the behavior of the logging policy and on online interventions: changes to the logging policy made during the gathering of click data. Our experimental results, conducted in a semi-synthetic experimental setup, show that, unlike existing counterfactual LTR methods, the intervention-aware estimator can greatly benefit from online interventions.

ACM Reference Format:
Harrie Oosterhuis and Maarten de Rijke. 2021. Unifying Online and Counterfactual Learning to Rank: A Novel Counterfactual Estimator that Effectively Utilizes Online Interventions. In *The 14th ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Jerusalem, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441794

1 INTRODUCTION

Ranking systems form the basis for most search and recommendation applications [16]. As a result, the quality of such systems can greatly impact the user experience, thus it is important that the underlying ranking models perform well. The Learning to Rank (LTR) field considers methods to optimize ranking models. Traditionally this was based on expert annotations. Over the years the limitations of expert annotations have become apparent; some of the most important ones are: (i) they are expensive and time-consuming to acquire [6, 22]; (ii) in privacy-sensitive settings expert annotation is unethical, e.g., in email or private document search [28]; and (iii) often expert annotations appear to disagree with actual user preferences [23].

User interaction data solves some of the problems with expert annotations: (i) interaction data is virtually free for systems with active users; (ii) it does not require experts to look at potentially privacy-sensitive content; (iii) interaction data is indicative of users' preferences. For these reasons, interest in LTR methods that learn from user interactions has increased in recent years. However, user interactions are a form of implicit feedback and generally also affected by other factors than user preference [14]. Therefore, to be able to reliably learn from interaction data, the effect of factors other than preference has to be corrected for. In clicks on rankings three prevalent factors are well known: (i) *position bias*: users are less likely to examine, and thus click, lower ranked items [7]; (ii) *item-selection bias*: users cannot click on items that are not displayed [19, 21]; and (iii) *trust bias*: because users trust the ranking system, they are more likely to click on highly ranked items that they do not actually prefer [2, 14]. As a result of these biases, which ranking system was used to gather clicks can have a substantial impact on the clicks that will be observed. Current LTR methods that learn from clicks can be divided into two families: *counterfactual approaches* [15] – that learn from historical data, i.e., clicks that have been logged in the past – and *online approaches* [30] – that can perform interventions, i.e., they can decide what rankings will be shown to users. Recent work has noticed that some counterfactual methods can be applied as an online method [11], or vice versa [5, 31]. Nonetheless, every existing method was designed for either the online or counterfactual setting, never both.

In this work, we propose a novel estimator for both counterfactual and online LTR from clicks: the *intervention-aware estimator*. The intervention-aware estimator builds on ideas that underlie the latest existing counterfactual methods: the policy-aware estimator [19] and the affine estimator [25]; and expands them to consider the effect of online interventions. It does so by considering how the effect of bias is changed by an intervention, and utilizes these differences in its unbiased estimation. As a result, the intervention-aware estimator is both effective when applied as a counterfactual method, i.e., when learning from historical data, and as an online method where online interventions lead to enormous increases in efficiency. In our experimental results the intervention-aware estimator is shown to reach state-of-the-art LTR performance in both online and counterfactual settings, and it is the only method that reaches top-performance in both settings.

The main contributions of this work are:

(1) A novel intervention-aware estimator that corrects for position bias, trust bias, item-selection bias, and the effect of online interventions.

WSDM '21, March 8–12, 2021, Jerusalem, Israel

ACM ISBN 978-1-4503-8297-7/21/03.
https://doi.org/10.1145/3437963.3441794

(2) An investigation into the effect of online interventions on state-of-the-art counterfactual and online LTR methods.

2 INTERACTIONS WITH RANKINGS

The theory in this paper assumes that three forms of interaction bias occur: position bias, item-selection bias, and trust bias.

Position bias occurs because users only click an item after examining it, and users are more likely to examine items displayed at higher ranks [7]. Thus the rank (a.k.a. position) at which an item is displayed heavily affects the probability of it being clicked. We model this bias using $P(E = 1 \mid k)$: the probability that an item d displayed at rank k is examined by a user E [28].

Item-selection bias occurs when some items have a zero probability of being examined in some displayed rankings [21]. This can happen because not all items are displayed to the user, or if the ranked list is so long that no user ever considers the entire list. We model this bias by stating: $\exists k, \forall k',\ (k' > k \rightarrow P(E = 1 \mid k') = 0)$, i.e., there exists a rank k such that items ranked lower than k have no chance of being examined. The distinction between position bias and item-selection bias is important because some methods can only correct for the former if the latter is not present [19].

Finally, *trust bias* occurs because users trust the ranking system and, consequently, are more likely to perceive top ranked items as relevant even when they are not [14]. We model this bias using: $P(C = 1 \mid k, R, E)$: the probability of a click conditioned on the displayed rank k, the relevance of the item R, and examination E.

To combine these three forms of bias into a single click model, we follow Agarwal et al. [2] and write:

$$\begin{aligned} &P(C = 1 \mid d, k, q) \\ &\quad = P(E = 1 \mid k)\big(P(C = 1 \mid k, R = 0, E = 1)P(R = 0 \mid d, q) \\ &\qquad\qquad + P(C = 1 \mid k, R = 1, E = 1)P(R = 1 \mid d, q)\big), \end{aligned} \tag{1}$$

where $P(R = 1 \mid d, q)$ is the probability that an item d is deemed relevant w.r.t. query q by the user. An analysis on real-world interaction data performed by Agarwal et al. [2], showed that this model better captures click behavior than models that only capture position bias [28] on search services for retrieving cloud-stored files and emails.

To simplify the notation, we follow Vardasbi et al. [25] and adopt:

$$\begin{aligned} \alpha_k &= P(E = 1 \mid k)\big(P(C = 1 \mid k, R = 1, E = 1) \\ &\qquad\qquad - P(C = 1 \mid k, R = 0, E = 1)\big), \\ \beta_k &= P(E = 1 \mid k)P(C = 1 \mid k, R = 0, E = 1). \end{aligned} \tag{2}$$

This results in a compact notation for the click probability (1):

$$P(C = 1 \mid d, k, q) = \alpha_k P(R = 1 \mid d, q) + \beta_k. \tag{3}$$

For a single ranking y, let k be the rank at which item d is displayed in y; we denote $\alpha_k = \alpha_{d,y}$ and $\beta_k = \beta_{d,y}$. This allows us to specify the click probability conditioned on a ranking y:

$$P(C = 1 \mid d, y, q) = \alpha_{d,y} P(R = 1 \mid d, q) + \beta_{d,y}. \tag{4}$$

Finally, let π be a ranking policy used for logging clicks, where $\pi(y \mid q)$ is the probability of π displaying ranking y for query q, then the click probability conditioned on π is:

$$P(C = 1 \mid d, \pi, q) = \sum_y \pi(y \mid q)\Big(\alpha_{d,y} P(R = 1 \mid d, q) + \beta_{d,y}\Big). \tag{5}$$

The proofs in the remainder of this paper will assume this model of click behavior.

3 BACKGROUND

In this section we cover the basics on LTR and counterfactual LTR.

3.1 Learning to Rank

The field of LTR considers methods for optimizing ranking systems w.r.t. ranking metrics. Most ranking metrics are additive w.r.t. documents; let $P(q)$ be the probability that a user-issued query is query q, then the metric reward $\mathcal{R}$ commonly has the form:

$$\mathcal{R}(\pi) = \sum_q P(q) \sum_{d \in D_q} \lambda(d \mid D_q, \pi, q) P(R = 1 \mid d, q). \tag{6}$$

Here, the λ function scores each item d depending on how π ranks d when given the preselected item set D_q; λ can be chosen to match a desired metric, for instance, the common Discounted Cumulative Gain (DCG) metric [12]:

$$\lambda_{\text{DCG}}(d \mid D_q, \pi, q) = \sum_y \pi(y \mid q)\big(\log_2(\text{rank}(d \mid y) + 1)\big)^{-1}. \tag{7}$$

Supervised LTR methods can optimize π to maximize $\mathcal{R}$ if relevances $P(R = 1 \mid d, q)$ are known [16, 29]. However in practice, finding these relevance values is not straightforward.

3.2 Counterfactual Learning to Rank

Over time, limitations of the supervised LTR approach have become apparent. Most importantly, finding accurate relevance values $P(R = 1 \mid d, q)$ has proved to be impossible or infeasible in many practical situations [27]. As a solution, LTR methods have been developed that learn from user interactions instead of relevance annotations. Counterfactual LTR concerns approaches that learn from historical interactions. Let $\mathcal{D}$ be a set of collected interaction data over T timesteps; for each timestep t it contains the user issued query q_t, the logging policy $\bar{\pi}_t$ used to generate the displayed ranking $\bar{y}_t$, and the clicks c_t received on the ranking:

$$\mathcal{D} = \{(\bar{\pi}_t, q_t, \bar{y}_t, c_t)\}_{t=1}^{T}, \tag{8}$$

where $c_t(d) \in \{0, 1\}$ indicates whether item d was clicked at timestep t. While clicks are indicative of relevance they are also affected by several forms of bias, as discussed in Section 2.

Counterfactual LTR methods utilize estimators that correct for bias to unbiasedly estimate the reward of a policy π. The prevalent methods introduce a function $\hat{\Delta}$ that transforms a single click signal to correct for bias. The general estimate of the reward is:

$$\hat{\mathcal{R}}(\pi \mid \mathcal{D}) = \frac{1}{T} \sum_{t=1}^{T} \sum_{d \in D_{q_t}} \lambda(d \mid D_{q_t}, \pi, q) \hat{\Delta}(d \mid \bar{\pi}_t, q_t, \bar{y}_t, c_t). \tag{9}$$

We note the important distinction between the policy π for which we estimate the reward, and the policy $\bar{\pi}_t$ that was used to gather interactions. During optimization only π is changed in order to maximize the estimated reward.

The original Inverse-Propensity-Scoring (IPS) based estimator introduced by Wang et al. [27] and Joachims et al. [15] weights

clicks according to examination probabilities:

$$\hat{\Delta}_{\text{IPS}}(d \mid \bar{y}_t, c_t) = \frac{c_t(d)}{P(E = 1 \mid \bar{y}_t, d)}. \tag{10}$$

This estimator results in unbiased optimization under two requirements. First, every relevant item must have a non-zero examination probability in all displayed rankings:

$$\forall t, \forall d \in D_{q_t} \ \left(P(R = 1 \mid d, q_t) > 0 \rightarrow P(E = 1 \mid \bar{y}_t, d) > 0\right). \tag{11}$$

Second, the click probability conditioned on relevance on examined items should be the same on every rank:

$$\forall k, k' \ \left(P(C \mid k, R, E = 1) = P(C \mid k', R, E = 1)\right), \tag{12}$$

i.e., no trust bias is present. These requirements illustrate that this estimator can only correct for position bias, and is biased when item-selection bias or trust bias is present. For a proof we refer to previous work by Joachims et al. [15] and Vardasbi et al. [25].

Oosterhuis and de Rijke [19] adapt the IPS approach to correct for item-selection bias as well. They weight clicks according to examination probabilities conditioned on the logging policy, instead of the single displayed ranking on which a click took place. This results in the *policy-aware* estimator:

$$\begin{aligned}\hat{\Delta}_{\text{aware}}(d \mid \bar{\pi}_t, q_t, c_t) &= \frac{c_t(d)}{P(E = 1 \mid \bar{\pi}_t, q_t, d)} \\ &= \frac{c_t(d)}{\sum_y \bar{\pi}(y \mid q_t) P(E = 1 \mid y, d, q_t)}.\end{aligned} \tag{13}$$

This estimator can be used for unbiased optimization under two assumptions. First, every relevant item must have a non-zero examination probability under the logging policy:

$$\forall t, \forall d \in D_{q_t} \ (P(R = 1 \mid, d, q_t) > 0 \rightarrow P(E = 1 \mid \bar{\pi}_t, d, q_t) > 0). \tag{14}$$

Second, no trust bias is present as described in Eq 12. Importantly, this first requirement can be met under item-selection bias, since a stochastic ranking policy can always provide every item a non-zero probability of appearing in a top-k ranking. Thus, even when not all items can be displayed at once, a stochastic policy can provide non-zero examination probabilities to all items. For a proof of this claim we refer to previous work by Oosterhuis and de Rijke [19].

Lastly, Vardasbi et al. [25] prove that IPS cannot correct for trust bias. As an alternative, they introduce an estimator based on affine corrections. This *affine* estimator penalizes an item displayed at rank k by β_k while also reweighting inversely w.r.t. α_k:

$$\hat{\Delta}_{\text{affine}}(d \mid \bar{y}_t, c_t) = \frac{c_t(d) - \beta_{d,\bar{y}_t}}{\alpha_{d,\bar{y}_t}}. \tag{15}$$

The β penalties correct for the number of clicks an item is expected to receive due to its displayed rank, instead of its relevance. The affine estimator is unbiased under a single assumption, namely that the click probability of every item must be correlated with its relevance in every displayed ranking:

$$\forall t, \forall d \in D_{q_t}, \ \alpha_{d,\bar{y}_t} \neq 0. \tag{16}$$

Thus, while this estimator can correct for position bias and trust bias, it cannot correct for item-selection bias. For a proof of these claims we refer to previous work by Vardasbi et al. [25].

We note that all of these estimators require knowledge of the position bias ($P(E = 1 \mid k)$) or trust bias (α and β). A lot of existing work has considered how these values can be inferred accurately [2, 8, 28]. The theory in this paper assumes these values are known.

This concludes our description of existing counterfactual estimators on which our method expands. To summarize, each of these estimators corrects for position bias, one also corrects for item-selection bias, and another also for trust bias. Currently, there is no estimator that corrects for all three forms of bias together.

4 RELATED WORK

One of the earliest approaches to LTR from clicks was introduced by Joachims [13]. It infers pairwise preferences between items from click logs and uses pairwise LTR to update an SVM ranking model. While this approach had some success, in later work Joachims et al. [15] notes that position bias often incorrectly pushes the pairwise loss to flip the ranking displayed during logging. To avoid this biased behavior, Joachims et al. [15] proposed the idea of counterfactual LTR, in the spirit of earlier work by Wang et al. [27]. This led to estimators that correct for position bias using IPS weighting (see Section 3.2). This work sparked the field of counterfactual LTR which has focused on both capturing interaction biases and optimization methods that can correct for them. Methods for measuring position bias are based on EM optimization [28], a dual learning objective [4], or randomization [3, 8]; for trust bias only an EM-based approach is currently known [2]. Agarwal et al. [1] showed how counterfactual LTR can optimize neural networks and DCG-like methods through upper-bounding. Oosterhuis and de Rijke [19] introduced an IPS estimator that can correct for item-selection bias (see Section 3.2), while also showing that the LambdaLoss framework [29] can be applied to counterfactual LTR. Lastly, Vardasbi et al. [25] proved that IPS estimators cannot correct for trust bias and introduced an affine estimator that is capable of doing so (see Section 3.2). There is currently no known estimator that can correct for position bias, item selection bias, and trust bias simultaneously.

The other paradigm for LTR from clicks is online LTR [30]. The earliest method, Dueling Bandit Gradient Descent (DBGD), samples variations of a ranking model and compares them using online evaluation [10]; if an improvement is recognized the model is updated accordingly. Most online LTR methods have increased the data-efficiency of DBGD [9, 24, 26]; later work found that DBGD is not effective at optimizing neural models [17] and often fails to find the optimal linear-model even in ideal scenarios [18]. To these limitations, alternative approaches for online LTR have been proposed. Pairwise Differentiable Gradient Descent (PDGD) takes a pairwise approach but weights pairs to correct for position bias [17]. While PDGD was found to be very effective and robust to noise [11], it can be proven that its gradient estimation is affected by position bias, thus we do not consider it to be unbiased. In contrast, Zhuang and Zuccon [31] introduced Counterfactual Online Learning to Rank (COLTR), which takes the DBGD approach but uses a form of counterfactual evaluation to compare candidate models. Despite making use of counterfactual estimation, Zhuang and Zuccon [31] propose the method solely for online LTR.

Interestingly, with COLTR the line between online and counterfactual LTR methods starts to blur. Recent work by Jagerman et al. [11] applied the original counterfactual approach [15] as an online method and found that it lead to improvements. Furthermore, Ai

et al. [5] noted that with a small adaptation PDGD can be applied to historical data. Although this means that some existing methods can already be applied both online and counterfactually, no method has been found that is the most reliable choice in both scenarios.

5 AN ESTIMATOR OBLIVIOUS TO ONLINE INTERVENTIONS

Before we propose our main contribution, the intervention-aware estimator, we will first introduce an estimator that simultaneously corrects for position bias, item-selection bias, and trust bias, without considering the effects of interventions. Subsequently, the resulting intervention-oblivious estimator will serve as a method to contrast the intervention-aware estimator with.

Section 3.2 described how the policy-aware estimator corrects for item-selection bias by taking into account the behavior of the logging policy used to gather clicks [19]. Furthermore, Section 3.2 also detailed how the affine estimator corrects for trust bias by applying an affine transformation to individual clicks [25]. We will now show that a single estimator can correct for both item-selection bias and trust bias simultaneously, by combining the approaches of both these existing estimators.

First we note the probability of a click conditioned on a single logging policy π_t can be expressed as:

$$P(C=1|\,d,\pi_t,q) = \sum_{\bar{y}} \pi_t(\bar{y} \mid q)\Big(\alpha_{d,\bar{y}}P(R=1 \mid d,q) + \beta_{d,\bar{y}}\Big) \quad (17)$$
$$= \mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q]P(R=1 \mid d,q) + \mathbb{E}_{\bar{y}}[\beta_d \mid \pi_t, q].$$

where the expected values of α and β conditioned on π_t are:

$$\begin{aligned}\mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q] &= \textstyle\sum_{\bar{y}} \pi_t(\bar{y} \mid q)\alpha_{d,\bar{y}},\\ \mathbb{E}_{\bar{y}}[\beta_d \mid \pi_t, q] &= \textstyle\sum_{\bar{y}} \pi_t(\bar{y} \mid q)\beta_{d,\bar{y}}.\end{aligned} \quad (18)$$

By reversing Eq. 17 the relevance probability can be obtained from the click probability. We introduce our *intervention-oblivious estimator*, which applies this transformation to correct for bias:

$$\hat{\Delta}_{\text{IO}}(d \mid q_t, c_t) = \frac{c_t(d) - \mathbb{E}_{\bar{y}}[\beta_d \mid \pi_t, q_t]}{\mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q_t]}. \quad (19)$$

The intervention-oblivious estimator brings together the policy-aware and affine estimators: on every click it applies an affine transformation based on the logging policy behavior. Unlike existing estimators, we can prove that the intervention-oblivious estimator is unbiased w.r.t. our assumed click model (Section 2).

Theorem 5.1. *The estimated reward $\hat{\mathcal{R}}$ (Eq. 9) using the intervention-oblivious estimator (Eq. 19) is unbiased w.r.t. the true reward $\mathcal{R}$ (Eq. 6) under two assumptions: (1) our click model (Eq. 4), and (2) the click probability on every item, conditioned on the logging policies per timestep π_t, is correlated with relevance:*

$$\forall t, \forall d \in D_{q_t}\ \mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q_t] \neq 0. \quad (20)$$

Proof. Using Eq. 17 and Eq. 27 the relevance probability can be derived from the click probability by:

$$P(R=1 \mid d,q) = \frac{P(C=1 \mid d,\pi_t,q) - \mathbb{E}_{\bar{y}}[\beta_d \mid \pi_t, q]}{\mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q]}. \quad (21)$$

Eq. 21 can be used to show that $\hat{\Delta}_{\text{IO}}$ is an unbiased indicator of relevance:

$$\mathbb{E}_{\bar{y},c}\big[\hat{\Delta}_{\text{IO}}(d \mid q_t, c_t) \mid \pi_t\big]$$

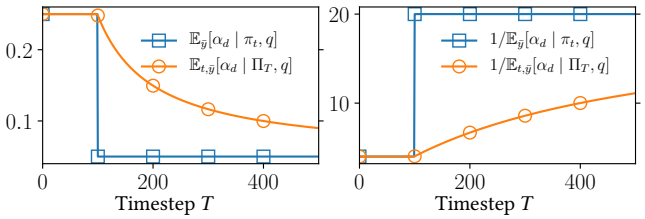

Figure 1: Example of an online intervention and the weights used by the intervention-oblivious and intervention-aware estimators for a single item as more data is gathered.

$$= \mathbb{E}_{\bar{y},c}\left[\frac{c_t(d) - \mathbb{E}_{t,\bar{y}}[\beta_d \mid \pi_t, q_t]}{\mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q_t]} \mid \pi_t, q_t\right] \quad (22)$$
$$= \frac{\mathbb{E}_{\bar{y},c}[c_t(d) \mid \pi_t, q_t] - \mathbb{E}_{\bar{y}}[\beta_d \mid \pi_t, q_t]}{\mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q_t]}$$
$$= \frac{P(C=1 \mid d,\pi_t,q_t) - \mathbb{E}_{\bar{y}}[\beta_d \mid \pi_t, q_t]}{\mathbb{E}_{\bar{y}}[\alpha_d \mid \pi_t, q_t]}$$
$$= P(R=1 \mid d, q_t).$$

Finally, combining Eq. 6 with Eq. 9 and Eq. 23 reveals that $\hat{\mathcal{R}}$ based on the intervention-oblivious estimator $\hat{\Delta}_{\text{IO}}$ is unbiased w.r.t. $\mathcal{R}$:

$$\mathbb{E}_{t,q,\bar{y},c}\big[\hat{\mathcal{R}}(\pi \mid \mathcal{D})\big] \quad (23)$$
$$= \sum_q P(q) \sum_{d \in D_q} \lambda(d \mid D_q, \pi, q)\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\bar{y},c}\big[\hat{\Delta}_{\text{IO}}(d \mid c, q) \mid \pi_t, q\big]$$
$$= \sum_q P(q) \sum_{d \in D_q} \lambda(d \mid D_q, \pi, q)P(R=1 \mid d, q) = \mathcal{R}(\pi). \quad \square$$

5.1 Example with an Online Intervention

Existing estimators for counterfactual LTR are designed for a scenario where the logging policy is static: $\forall(\pi_t, \pi_{t'}) \in \mathcal{D}$, $\pi_t = \pi_{t'}$. However, we note that if an online intervention takes place [11], meaning that the logging policy was updated during the gathering of data: $\exists(\pi_t, \pi_{t'}) \in \mathcal{D}$, $\pi_t \neq \pi_{t'}$, the intervention-oblivious estimator is still unbiased. This was already proven in Theorem 5.1 because its assumptions cover both scenarios where online interventions do and do not take place.

However, the individual corrections of the intervention-oblivious estimator are only based on the single logging policy that was deployed at the timestep of each specific click. It is completely oblivious to the logging policies applied at different timesteps. Although this does not lead to bias in its estimation, it does result in unintuitive behavior. We illustrate this behavior in Figure 1, here a logging policy that results in $\mathbb{E}[\alpha_d \mid \pi_t, q] = 0.25$ for an item d is deployed during the first $t \leq 100$ timesteps. Then an online intervention takes place and the logging policy is updated so that for $t > 100$, $\mathbb{E}[\alpha_d \mid \pi_t, q] = 0.05$. The intervention-oblivous estimator weights clicks inversely to $\mathbb{E}[\alpha_d \mid \pi_t]$; so clicks for $t \leq 100$ will be weighted by $1/0.25 = 4$ and for $t > 100$ by $1/0.05 = 20$. Thus, there is a sharp and sudden difference in how clicks are treated before and after $t = 100$. What is unintuitive about this example is that the way clicks are treated after $t = 100$ is completely independent of what the situation was before $t = 100$. For instance, consider another item d' where $\forall t, \mathbb{E}[\alpha_{d'} \mid \pi_t, q] = 0.05$. If both d and d' are clicked on timestep $t = 101$, these clicks would both be weighted by 20,

despite the fact that d has so far been treated completely different than d'. One would expect that in such a case the click on d should be weighted less, to compensate for the high $\mathbb{E}[\alpha_d \mid \pi_t, q]$ it had in the first 100 timesteps. The question is whether such behavior can be incorporated in an estimator without introducing bias.

6 THE INTERVENTION-AWARE ESTIMATOR

Our goal for the intervention-aware estimator is to find an estimator whose individual corrections are not only based on single logging policies, but instead consider the entire collection of logging policies used to gather the data $\mathcal{D}$. Importantly, this estimator should also be unbiased w.r.t. position bias, item-selection bias and trust bias.

For ease of notation, we use Π_T for the set of policies that gathered the data in $\mathcal{D}$: $\Pi_T = \{\pi_1, \pi_2, \ldots, \pi_T\}$. The probability of a click can be conditioned on this set:

$$\begin{aligned} &P(C = 1 \mid d, \Pi_T, q) \\ &\quad= \frac{1}{T}\sum_{t=1}^{T}\sum_{\bar{y}} \pi_t(\bar{y} \mid q)\Big(\alpha_{d,\bar{y}} P(R = 1 \mid d, q) + \beta_{d,\bar{y}}\Big) \qquad (24) \\ &\quad= \mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q] P(R = 1 \mid d, q) + \mathbb{E}_{t,\bar{y}}[\beta_d \mid \Pi_T, q], \end{aligned}$$

where the expected values of α and β conditioned on Π_T are:

$$\begin{aligned} \mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q] &= \frac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{\bar{y}} \pi_t(\bar{y} \mid q)\alpha_{d,\bar{y}}, \\ \mathbb{E}_{t,\bar{y}}[\beta_d \mid \Pi_T, q] &= \frac{1}{T}\textstyle\sum_{t=1}^{T}\sum_{\bar{y}} \pi_t(\bar{y} \mid q)\beta_{d,\bar{y}}. \end{aligned} \qquad (25)$$

Thus $P(C = 1 \mid d, \Pi_T, q)$ gives us the probability of a click given that any policy from Π_T could be deployed. We propose our *intervention-aware estimator* that corrects for bias using the expectations conditioned on Π_T:

$$\hat{\Delta}_{\text{IA}}(d \mid q_t, c_t) = \frac{c_t(d) - \mathbb{E}_{t,\bar{y}}[\beta_d \mid \Pi_T, q_t]}{\mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q_t]}. \qquad (26)$$

The salient difference with the intervention-oblivious estimator is that the expectations are conditioned on Π_T, all logging policies in $\mathcal{D}$, instead of an individual logging policy π_t. While the difference with the intervention-oblivious estimator seems small, our experimental results show that the differences in performance are actually quite sizeable. Lastly, we note that when no interventions take place the intervention-oblivious estimator and intervention-aware estimators are equivalent. Because the intervention-aware estimator is the only existing counterfactual LTR estimator whose corrections are influenced by online interventions, we consider it to be a step that helps to bridge the gap between counterfactual and online LTR.

Before we revisit our online intervention example with our novel intervention-aware estimator, we prove that it is unbiased w.r.t. our assumed click model (Section 2).

Theorem 6.1. *The estimated reward $\hat{\mathcal{R}}$ (Eq. 9) using the intervention-aware estimator (Eq. 26) is unbiased w.r.t. the true reward $\mathcal{R}$ (Eq. 6) under two assumptions: (1) our click model (Eq. 4), and (2) the click probability on every item, conditioned on the set of logging policies Π_T, is correlated with relevance:*

$$\forall q, \forall d \in D_q, \quad \mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q] \neq 0. \qquad (27)$$

Proof. Using Eq. 24 and Eq. 27 the relevance probability can be derived from the click probability by:

$$P(R = 1 \mid d, q) = \frac{P(C = 1 \mid d, \Pi_T, q) - \mathbb{E}_{t,\bar{y}}[\beta_d \mid \Pi_T, q]}{\mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q]}. \qquad (28)$$

Eq. 28 can be used to show that $\hat{\Delta}_{\text{IA}}$ is an unbiased indicator of relevance:

$$\begin{aligned} &\mathbb{E}_{t,\bar{y},c}\big[\hat{\Delta}_{\text{IA}}(d \mid q_t, c_t) \mid \Pi_T\big] \\ &\qquad= \mathbb{E}_{t,\bar{y},c}\left[\frac{c_t(d) - \mathbb{E}_{t,\bar{y}}[\beta_d \mid \Pi_T, q_t]}{\mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q_t]} \mid \Pi_T, q_t\right] \qquad (29) \\ &\qquad= \frac{\mathbb{E}_{t,\bar{y},c}[c_t(d) \mid \Pi_T, q_t] - \mathbb{E}_{t,\bar{y}}[\beta_d \mid \Pi_T, q_t]}{\mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q_t]} \\ &\qquad= \frac{P(C = 1 \mid d, \Pi_T, q_t) - \mathbb{E}_{t,\bar{y}}[\beta_d \mid \Pi_T, q_t]}{\mathbb{E}_{t,\bar{y}}[\alpha_d \mid \Pi_T, q_t]} \\ &\qquad= P(R = 1 \mid d, q_t). \end{aligned}$$

Finally, combining Eq. 30 with Eq. 9 and Eq. 6 reveals that $\hat{\mathcal{R}}$ based on the intervention-aware estimator $\hat{\Delta}_{\text{IA}}$ is unbiased w.r.t. $\mathcal{R}$:

$$\begin{aligned} &\mathbb{E}_{t,q,\bar{y},c}\big[\hat{\mathcal{R}}(\pi \mid \mathcal{D})\big] \\ &\quad= \sum_q P(q) \sum_{d \in D_q} \lambda(d \mid D_q, \pi, q)\mathbb{E}_{t,\bar{y},c}\big[\hat{\Delta}_{\text{IA}}(d \mid c, q) \mid \Pi_T, q\big] \\ &\quad= \sum_q P(q) \sum_{d \in D_q} \lambda(d \mid D_q, \pi, q) P(R = 1 \mid d, q) \qquad (30) \\ &\quad= \mathcal{R}(\pi). \end{aligned}$$

□

6.1 Online Intervention Example Revisited

We will now revisit the example in Figure 1, but this time consider how the intervention-aware estimator treats item d. Unlike the intervention-oblivious estimator, clicks are weighted by $\mathbb{E}[\alpha_d \mid \Pi_T]$ which means that the exact timestep t of a click does not matter, as long as $t < T$. Furthermore, the weight of a click can change as the total number of timesteps T increases. In other words, as more data is gathered, the intervention-aware estimator retroactively updates the weights of all clicks previously gathered.

We see that this behavior avoids the sharp difference in weights of clicks occurring before the intervention $t \leq 100$ and after $t > 100$. For instance, for a click on d occuring at $t = 101$ while $T = 400$, results in $\mathbb{E}[\alpha_d \mid \Pi_T] = 0.1$ and thus a weight of $1/0.1 = 10$. This is much lower than the intervention-oblivious weight of $1/0.05 = 20$, because the intervention-aware estimator is also considering the initial period where $\mathbb{E}[\alpha_d \mid \pi_t, q]$ was high. Thus we see that the intervention-aware estimator has the behavior we intuitively expected: it weights clicks based on how the item was treated throughout all timesteps. In this example, it leads weights considerably smaller than those used by the intervention-oblivious estimator. In IPS estimators, low propensity weights are known to lead to high variance [15], thus we may expect that the intervention-aware estimator reduces variance in this example.

6.2 An Online and Counterfactual Approach

While the intervention-aware estimator takes into account the effect of interventions, it does not prescribe what interventions should take place. In fact, it will work with any interventions that result

in Eq. 27 being true, including the situation where no intervention takes place at all. For clarity, we will describe the intervention approach we applied during our experiments here. Algorithm 1 displays our online/counterfactual approach. As input it requires a starting policy (π_0), a choice for λ, the α and β parameters, a set of intervention timesteps (Φ), and the final timestep T.

The algorithm starts by initializing an empty set to store the gathered interaction data (Line 2) and initializes the logging policy with the provided starting policy π_0. Then for each timestep i in Φ the dataset is expanded using the current logging policy so that $|\mathcal{D}| = i$ (Line 5). In other words, for $i - |\mathcal{D}|$ timesteps π is used to display rankings to user-issued queries, and the resulting interactions are added to $\mathcal{D}$. Then a policy is optimized using the available data in $\mathcal{D}$ which becomes the new logging policy. For this optimization, we split the available data in training and validation partitions in order to do early stopping to prevent overfitting. We use stochastic gradient descent where we use π_0 as the initial model; this practice is based on the assumption that π_0 has a better performance than a randomly initialized model. Thus, during optimization, gradient calculation uses the intervention-aware estimator on the training partition of $\mathcal{D}$, and after each epoch, optimization is stopped if the intervention-aware estimator using the validation partition of $\mathcal{D}$ suspects overfitting. Each iteration results in an intervention as the resulting policy replaces the logging policy, and thus changes the way future data is logged. After iterating over Φ is completed, more data is gathered so that $|\mathcal{D}| = T$ and optimization is performed once more. The final policy is the end result of the procedure.

We note that, depending on Φ, our approach can be either online, counterfactual, or somewhere in between. If $\Phi = \emptyset$ the approach is fully counterfactual since all data is gathered using the static π_0. Conversely, if $\Phi = \{1, 2, 3, \ldots, T\}$ it is fully online since at every timestep the logging policy is updated. In practice, we expect a fully online procedure to be infeasible as it is computationally expensive and user queries may be issued faster than optimization can be performed. In our experiments we will investigate the effect of the number of interventions on the approach's performance.

Algorithm 1 Our Online/Counterfactual LTR Approach

1: **Input**: Starting policy: π_0; Metric weight function: λ;
Inferred bias parameters: α and β;
Interventions steps: Φ; End-time: T.
2: $\mathcal{D} \leftarrow \{\}$ *// initialize data container*
3: $\pi \leftarrow \pi_0$ *// initialize logging policy*
4: **for** $i \in \Phi$ **do**
5: $\mathcal{D} \leftarrow \mathcal{D} \cup \text{gather}(\pi, i - |\mathcal{D}|)$ *// observe $i - |\mathcal{D}|$ timesteps*
6: $\pi \leftarrow \text{optimize}(\mathcal{D}, \alpha, \beta, \pi_0)$ *// optimize based on available data*
7: $\mathcal{D} \leftarrow \mathcal{D} \cup \text{gather}(\pi, T - |\mathcal{D}|)$ *// expand data to T*
8: $\pi \leftarrow \text{optimize}(\mathcal{D}, \alpha, \beta, \pi_0)$ *// optimize based on final data*
9: **return** π

7 EXPERIMENTAL SETUP

Our experiments aim to answer the following research questions:

RQ1 Does the intervention-aware estimator lead to higher performance than existing counterfactual LTR estimators when online interventions take place?

RQ2 Does the intervention-aware estimator lead to performance comparable with existing online LTR methods?

We use the semi-synthetic experimental setup that is common in existing work on both online LTR [9, 17, 18, 31] and counterfactual LTR [15, 21, 25]. In this setup, queries and documents are sampled from a dataset based on commercial search logs, while user interactions and rankings are simulated using probabilistic click models. The advantage of this setup is that it allows us to investigate the effects of online interventions on a large scale while also being easy to reproduce by researchers without access to live ranking systems.

We use the publicly-available Yahoo Webscope dataset [6], which consists of 29 921 queries with, on average, 24 documents preselected per query. Query-document pairs are represented by 700 features and five-grade relevance annotations ranging from not relevant (0) to perfectly relevant (4). The queries are divided into training, validation and test partitions.

At each timestep, we simulate a user-issued query by uniformly sampling from the training and validation partitions. Subsequently, the preselected documents are ranked according to the logging policy, and user interactions are simulated on the top-5 of the ranking using a probabilistic click model. We apply Eq. 3 with $\alpha = [0.35, 0.53, 0.55, 0.54, 0.52]$ and $\beta = [0.65, 0.26, 0.15, 0.11, 0.08]$; the relevance probabilities are based on the annotations from the dataset: $P(R = 1 \mid d, q) = 0.25 \cdot \text{relevance\_label}(d, q)$. The values of α and β were chosen based on those reported by Agarwal et al. [2] who inferred them from real-world user behavior. In doing so, we aim to emulate a setting where realistic levels of position bias, item-selection bias, and trust bias are present.

All counterfactual methods use the approach described in Section 6.2. To simulate a production ranker policy, we use supervised LTR to train a ranking model on 1% of the training partition [15]. The resulting production ranker has much better performance than a randomly initialized model, yet still leaves room for improvement. We use the production ranker as the initial logging policy. The size of Φ (the intervention timesteps) varies per run, and the timesteps in Φ are evenly spread on an exponential scale. All ranking models are neural networks with two hidden layers, each containing 32 hidden units with sigmoid activations. Gradients are calculated using a Monte-Carlo method following Oosterhuis and de Rijke [20]. All policies apply a softmax to the document scores produced by the ranking models to obtain a probability distribution over documents. Clipping is only applied on the training clicks, denominators of any estimator are clipped by $10/\sqrt{|\mathcal{D}|}$ to reduce variance. Early stopping is applied based on counterfactual estimates of the loss using (unclipped) validation clicks.

The following methods are compared: (i) The intervention-aware estimator. (ii) The intervention-oblivious estimator. (iii) The policy-aware estimator [19]. (iv) The affine estimator [25]. (v) PDGD [17], we apply PDGD both online and as a counterfactual method. As noted by Ai et al. [5], this can be done by separating the logging models from the learned model and, basing the debiasing weights on the logging function. (vi) Biased PDGD, identical to PDGD except that we do not apply the debiasing weights. (vii) COLTR [31]. We compute the Normalized DCG (NDCG) of both the logging policy and of a policy trained on all available data. Every reported result is the average of 20 independent runs, figures plot the mean, shaded

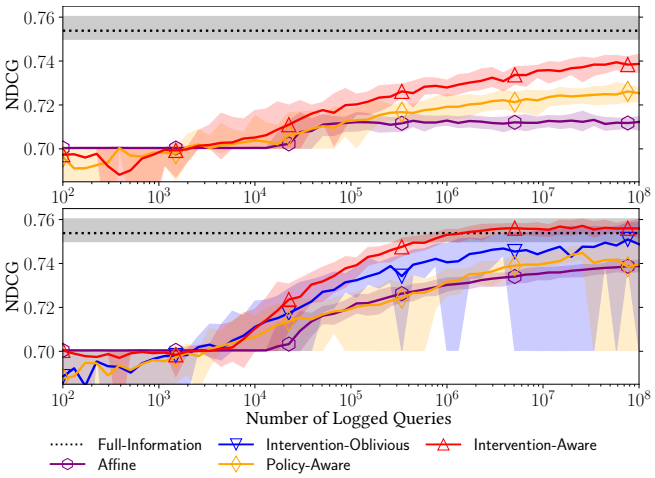

Figure 2: Comparison of counterfactual LTR estimators. Top: Counterfactual runs (no interventions); Bottom: Online runs (50 interventions). Results based on an average of 20 runs, shaded area indicates the 90% confidence bounds.

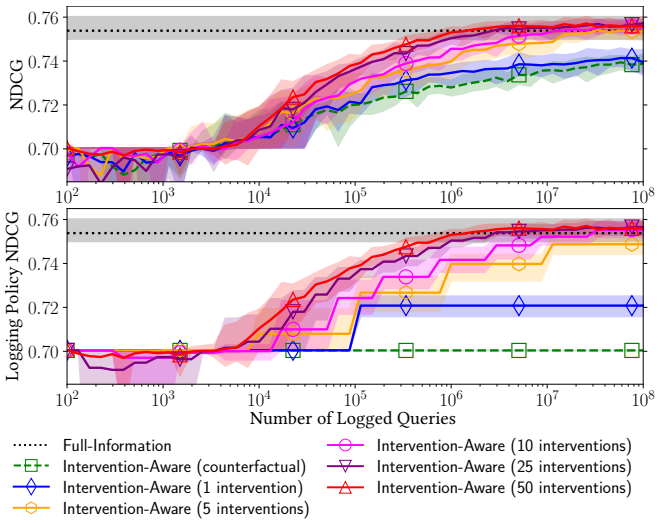

Figure 3: Effect of online interventions on LTR with the intervention-aware estimator. Results based on an average of 20 runs, shaded area indicates the 90% confidence bounds.

areas indicate 90% confidence bounds. To facilitate reproducibility, our implementation will be made publicly available.

8 RESULTS AND DISCUSSION

8.1 Comparison with Counterfactual LTR

To answer the first research question: *whether the intervention-aware estimator leads to higher performance than existing counterfactual LTR estimators when online interventions take place*, we consider Figure 2 which displays the performance of LTR using different counterfactual estimators.

First we consider the top of Figure 2 which displays performance in the counterfactual setting where the logging policy is static.[1] We clearly see that the affine estimator converges at a suboptimal point of convergence, a strong indication of bias. The most probable cause is that the affine estimator is heavily affected by the presence of item-selection bias. In contrast, neither the policy-aware estimator nor the intervention-aware estimator have converged after 10^8 queries. However, very clearly the intervention-aware estimator quickly reaches a higher performance. While the theory guarantees that it will converge at the optimal performance, we were unable to observe the number of queries it requires to do so. From the result in the counterfactual setting, we conclude that by correcting for position-bias, trust-bias, and item-selection bias the intervention-aware estimator already performs better without online interventions.

Second, we turn to the bottom of Figure 2 which considers the online setting where the estimators perform 50 online interventions during logging. We see that online interventions have a positive effect on all estimators; leading to a higher performance for the affine and policy-aware estimators as well. However, interventions also introduce an enormous amount of variance for the policy-aware and intervention-oblivious estimators. In stark contrast, the amount of variance of the intervention-aware estimator hardly increases while it learns much faster than the other estimators.

Thus we answer the first research question positively: the intervention-aware estimator leads to higher performance than existing estimators, moreover, its data-efficiency becomes even greater when online interventions take place.

8.2 Effect of Interventions

To better understand how much the intervention-aware estimator benefits from online interventions, we compared its performance under varying numbers of interventions in Figure 3. It shows both the performance of the resulting model when training from the logged data (top), as the performance of the logging policy which reveals when interventions take place (bottom). When comparing both graphs, we see that interventions lead to noticeable immediate improvements in data-efficiency. For instance, when only 5 interventions take place the intervention-aware estimator needs more than 20 times the amount of data to reach optimal performance as with 50 interventions. Despite these speedups there are no large increases in variance. From these observations, we conclude that the intervention-aware estimator can effectively and reliably utilize the effect of online interventions for optimization, leading to enormous increases in data-efficiency.

8.3 Comparison with Online LTR

In order to answer the second research question: *whether the intervention-aware estimator leads to performance comparable with existing online LTR methods*, we consider Figure 4 which displays the performance of two online LTR methods: PDGD and COLTR and the intervention-aware estimator with 100 online interventions.

First, we notice that COLTR is unable to outperform its initial policy, moreover, we see its performance drop as the number of iterations increase. We were unable to find hyper-parameters for COLTR where this did not occur. It seems likely that COLTR is unable to deal with trust-bias, thus causing this poor performance.

[1]Since under a static logging policy the intervention-aware and the intervention-oblivious estimators are equivalent, our conclusions apply to both in this setting.

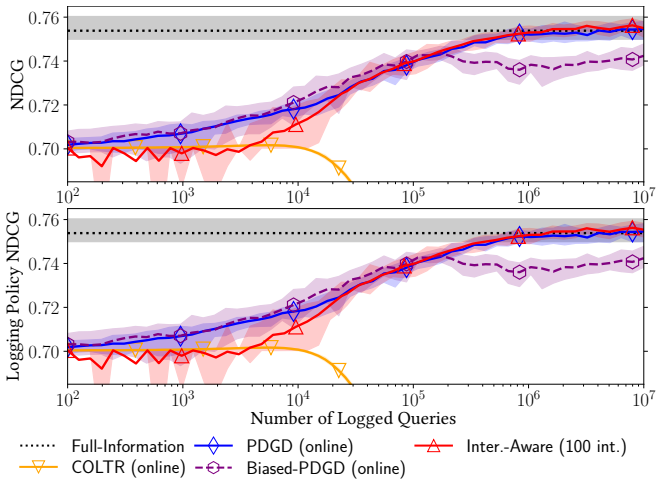

Figure 4: Comparison with online LTR methods. Results based on an average of 20 runs, shaded area indicates the 90% confidence bounds.

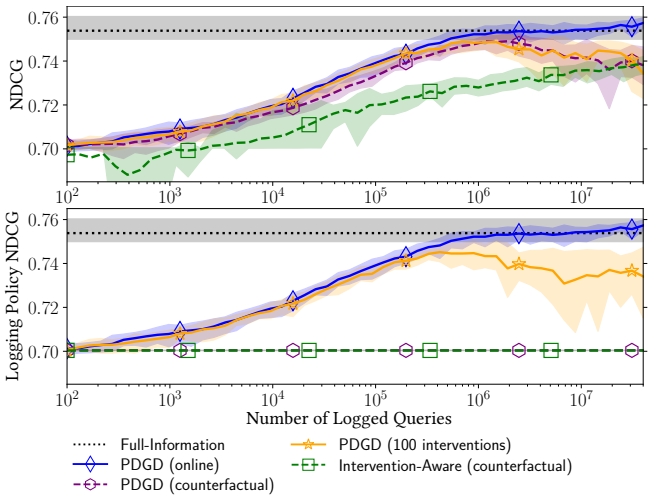

Figure 5: Effect of online interventions on PDGD. Results based on an average of 20 runs, shaded area indicates the 90% confidence bounds.

However, we note that Zhuang and Zuccon [31] already show COLTR performs poorly when no bias or noise is present, suggesting that it is perhaps an unstable method overall.

Second, we see that the difference between PDGD and the intervention-aware estimator becomes negligible after $2 \cdot 10^4$ queries. Despite PDGD running fully online, and the intervention-aware estimator only performing 100 interventions in total. We do note that PDGD initially outperforms the intervention-aware estimator, thus it appears that PDGD works better with low numbers of interactions. Additionally, we should also consider the difference in overhead: while PDGD requires an infrastructure that allows for fully online learning, the intervention-aware estimator only requires 100 moments of intervention, yet has comparable performance after a short initial period. By comparing Figure 4 to Figure 2, we see that the intervention-aware estimator is the first counterfactual LTR estimator that leads to stable performance while being comparably efficient with online LTR methods.

Thus we answer the second research question positively: besides an initial period of lower performance, the intervention-aware estimator has comparable performance to online LTR and only requires 100 online interventions to do so. To the best of our knowledge, it is the first counterfactual LTR method that can achieve this feat.

8.4 Understanding the Effectiveness of PDGD

Now that we concluded that the intervention-aware estimator reaches performance comparable to PDGD when enough online interventions take place, the opposite question seems equally interesting: *Does PDGD applied counterfactually provide performance comparable to existing counterfactual LTR methods?*

To answer this question, we ran PDGD in a counterfactual way following Ai et al. [5], both fully counterfactual and with only 100 interventions. The results of these runs are displayed in Figure 5. Quite surprisingly, the performance of PDGD ran counterfactually and with 100 interventions, reaches much higher performance than the intervention-aware estimator without interventions. However, after a peak in performance around 10^6 queries, the PDGD performance starts to drop. This drop cannot be attributed to overfitting, since online PDGD does not show the same behavior. Therefore, we must conclude that PDGD is biased when not ran fully online. This conclusion does not contradict the existing theory, since Oosterhuis and de Rijke [17] only proved it is unbiased w.r.t. *pairwise* preferences. In other words, PDGD is not proven to unbiasedly optimize a ranking metric, thus also not proven to converge on the optimal model. This drop is particularly unsettling because PDGD is a continuous learning algorithm: there is no known early stopping method for PDGD. Yet these results show there is a great risk in running PDGD for too many iterations if it is not applied fully online. To answer our PDGD question: although PDGD reaches high performance when run counterfactually and appears to have great data-efficiency initially, it appears to converge at a suboptimal biased model. Thus we cannot conclude PDGD is a reliable method for counterfactual LTR.

To better understand PDGD, we removed its debiasing weights resulting in the performance shown in Figure 4 (Biased-PDGD). Clearly, PDGD needs these weights to reach optimal performance. Similarly, from Figure 5 we see it also needs to be run fully online. This makes the choice between the intervention-aware estimator and PDGD complicated: on the one hand, PDGD does not require us to know the α and β parameters, unlike the intervention-aware estimator; furthermore, PDGD has better initial data-efficiency even when not run fully online. On the other hand, there are no theoretical guarantees for the convergence of PDGD, and we have observed that not running it fully online can lead to large drops in performance. It seems the choice ultimately depends on what guarantees a practitioner prefers.

9 CONCLUSION

In this paper, we have introduced an intervention-aware estimator: an extension of existing counterfactual approaches that corrects

for position-bias, trust-bias, and item-selection bias, while also considering the effect of online interventions. Our results show that the intervention-aware estimator outperforms existing counterfactual LTR estimators, and greatly benefits from online interventions in terms of data-efficiency. With only 100 interventions it is able to reach performance comparable to state-of-the-art online LTR methods.

With the introduction of the intervention-aware estimator, we hope to further unify the fields of online LTR and counterfactual LTR as it appears to be the most reliable method for both settings. Future work could investigate what kind of interventions work best for the intervention-aware estimator.

ACKNOWLEDGMENTS

This research was partially supported by the Netherlands Organisation for Scientific Research (NWO) under project nr 612.001.551 and by the Hybrid Intelligence Center, a 10-year programme funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research, https://hybrid-intelligence-centre.nl All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

CODE AND DATA

To facilitate the reproducibility of the reported results, this work only made use of publicly available data and our experimental implementation is publicly available at https://github.com/HarrieO/2021wsdm-unifying-LTR.

REFERENCES

[1] 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 & Development in Information Retrieval*. ACM, 5–14.

[2] 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*. ACM, 4–14.

[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, 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 *Proceedings of the 41st International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 385–394.

[5] Qingyao Ai, Tao Yang, Huazheng Wang, and Jiaxin Mao. 2020. Unbiased Learning to Rank: Online or Offline? *arXiv preprint arXiv:2004.13574* (2020).

[6] Olivier Chapelle and Yi Chang. 2011. Yahoo! Learning to Rank Challenge Overview. *Journal of Machine Learning Research* 14 (2011), 1–24.

[7] 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*. 87–94.

[8] Zhichong Fang, Aman Agarwal, and Thorsten Joachims. 2019. Intervention Harvesting for Context-dependent Examination-bias Estimation. In *SIGIR*. 825–834.

[9] Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten de Rijke. 2013. Reusing Historical Interaction Data for Faster Online Learning to Rank for IR. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 183–192.

[10] Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2011. A Probabilistic Method for Inferring Preferences from Clicks. In *CIKM*. ACM, 249–258.

[11] 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 & Development in Information Retrieval*. ACM, 15–24.

[12] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[13] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 133–142.

[14] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum*, Vol. 51. Acm New York, NY, USA, 4–11.

[15] 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, 781–789.

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

[17] 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*. ACM, 1293–1302.

[18] Harrie Oosterhuis and Maarten de Rijke. 2019. Optimizing Ranking Models in an Online Setting. In *Advances in Information Retrieval*. Springer International Publishing, Cham, 382–396.

[19] Harrie Oosterhuis and Maarten de Rijke. 2020. Policy-Aware Unbiased Learning to Rank for Top-k Rankings. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 489–498.

[20] Harrie Oosterhuis and Maarten de Rijke. 2020. Taking the Counterfactual Online: Efficient and Unbiased Online Evaluation for Ranking. arXiv:2007.12719 [cs.IR]

[21] Zohreh Ovaisi, Ragib Ahsan, Yifan Zhang, Kathryn Vasilaky, and Elena Zheleva. 2020. Correcting for Selection Bias in Learning-to-rank Systems. In *Proceedings of The Web Conference 2020*. 1863–1873.

[22] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 datasets. *arXiv preprint arXiv:1306.2597* (2013).

[23] Mark Sanderson. 2010. Test Collection Based Evaluation of Information Retrieval Systems. *Foundations and Trends in Information Retrieval* 4, 4 (2010), 247–375.

[24] Anne Schuth, Harrie Oosterhuis, Shimon Whiteson, and Maarten de Rijke. 2016. Multileave Gradient Descent for Fast Online Learning to Rank. In *WSDM*. ACM, 457–466.

[25] Ali Vardasbi, Harrie Oosterhuis, and Maarten de Rijke. 2020. When Inverse Propensity Scoring does not Work: Affine Corrections for Unbiased Learning to Rank. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*.

[26] Huazheng Wang, Sonwoo Kim, Eric McCord-Snook, Qingyun Wu, and Hongning Wang. 2019. Variance reduction in gradient exploration for online learning to rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 835–844.

[27] 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, 115–124.

[28] 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, 610–618.

[29] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The LambdaLoss Framework for Ranking Metric Optimization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1313–1322.

[30] 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*. ACM, 1201–1208.

[31] Shengyao Zhuang and Guido Zuccon. 2020. Counterfactual Online Learning to Rank. In *European Conference on Information Retrieval*. Springer, 415–430.