Policy-Aware Unbiased Learning to Rank for Top-k Rankings

Harrie Oosterhuis
University of Amsterdam
Amsterdam, The Netherlands
oosterhuis@uva.nl

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

ABSTRACT

使用存在bias的用户历史记录

Counterfactual Learning to Rank (LTR) methods optimize ranking systems using logged user interactions that contain interaction biases. Existing methods are only unbiased if users are presented with all relevant items in every ranking. There is currently no existing counterfactual unbiased LTR method for top-k rankings. We introduce a novel policy-aware counterfactual estimator for LTR metrics that can account for the effect of a stochastic logging policy. We prove that the policy-aware estimator is unbiased if every relevant item has a non-zero probability to appear in the top-k ranking. Our experimental results show that the performance of our estimator is not affected by the size of k: for any k, the policy-aware estimator reaches the same retrieval performance while learning from top-k feedback as when learning from feedback on the full ranking. Lastly, we introduce novel extensions of traditional LTR methods to perform counterfactual LTR and to optimize top-k metrics. Together, our contributions introduce the first policy-aware unbiased LTR approach that learns from top-k feedback and optimizes top-k metrics. As a result, counterfactual LTR is now applicable to the very prevalent top-k ranking setting in search and recommendation.

ACM Reference Format:
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 (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401102

1 INTRODUCTION

Learning to Rank (LTR) optimizes ranking systems to provide high quality rankings. Interest in LTR from user interactions has greatly increased in recent years with the introduction of unbiased LTR methods [18, 32]. The potential for learning from logged user interactions is great: user interactions provide valuable implicit feedback while also being cheap and relatively easy to acquire at scale [17]. However, interaction logs also contain large amounts of bias, which is the result of both user behavior and the ranker used during logging. For instance, users are more likely to examine items at the top of rankings, consequently the display position of an item heavily affects the number of interactions it receives [33]. This effect is called *position bias* and it is very dominant when learning from interactions with rankings. Naively ignoring it during learning can be detrimental to ranking performance, as the learning process is strongly impacted by what rankings were displayed during logging instead of *true* user preferences. The goal of unbiased LTR methods is to optimize a ranker w.r.t. the *true* user preferences, consequently, they have to account and correct for such forms of bias.

用户行为会受到推荐系统给出的排序顺序的影响

Previous work on unbiased LTR has mainly focussed on accounting for position bias through counterfactual learning [4, 18, 32]. The prevalent approach models the probability of a user examining an item in a displayed ranking. This probability can be inferred from user interactions [3, 4, 18, 32, 33] and corrected for using *inverse propensity scoring*. As a result, these methods optimize a loss that in expectation is unaffected by the examination probabilities during logging, hence it is unbiased w.r.t. position bias.

This approach has been applied effectively in various ranking settings, including search for scientific articles [18], email [32] or other personal documents [33]. However, a limitation of existing approaches is that in every logged ranking they require every relevant item to have a non-zero chance of being examined [8, 18]. In this paper, we focus on top-k rankings where the number of displayed items is systematically limited. These rankings can display at most k items, making it practically unavoidable that relevant items are missing. Consequently, existing counterfactual LTR methods are not unbiased in these settings. We recognize this problem as *item selection bias* introduced by the selection of (only) k items to display. This is especially concerning since top-k rankings are quite prevalent, e.g., in recommendation [10, 14], mobile search [5, 31], query autocompletion [7, 32, 33], and digital assistants [28].

在top-k推荐中肯定会有项目的缺失，使得传统的反偏差对每个item都有非0概率的假设不成立

Our main contribution is a novel policy-aware estimator for counterfactual LTR that accounts for both a stochastic logging policy and the users' examination behavior. Our policy-aware approach can be viewed as a generalization of the existing counterfactual LTR framework [1, 18]. We prove that our policy-aware approach performs unbiased LTR and evaluation while learning from top-k feedback. Our experimental results show that while our policy-aware estimator is unaffected by the choice of k, the existing policy-oblivious approach is strongly affected even under large values of k. For instance, optimization with the policy-aware estimator on top-5 feedback reaches the same performance as when receiving feedback on all results. Furthermore, because top-k metrics are the only relevant metrics in top-k rankings, we also propose extensions to traditional LTR approaches that are proven to optimize top-k metrics unbiasedly and introduce a pragmatic way to choose optimally between available loss functions.

Our work is based around two main contributions:

(1) A novel estimator for unbiased LTR from top-k feedback.
(2) Unbiased losses that optimize bounds on top-k LTR metrics.

To the best of our knowledge, our policy-aware estimator is the first estimator that is unbiased in top-k ranking settings.

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401102

2 BACKGROUND

In this section we discuss supervised LTR and the existing counterfactual LTR framework [18].

2.1 Supervised LTR

The goal of LTR is to optimize ranking systems w.r.t. specific ranking metrics. Ranking metrics generally involve items d, their relevance r w.r.t. a query q, and their position in the ranking R produced by the system. We will optimize the *Empirical Risk* [30] over the set of queries Q, with a loss $\Delta(R_i \mid q_i, r)$ for a single query q_i:

$$\mathcal{L} = \frac{1}{|Q|} \sum_{q_i \in Q} \Delta(R_i \mid q_i, r). \tag{1}$$

For simplicity we assume that relevance is binary: $r(q, d) \in \{0, 1\}$; for brevity we write: $r(q, d) = r(d)$. Then, ranking metrics commonly take the form of a sum over items:

$$\Delta(R \mid q, r) = \sum_{d \in R} \lambda(d \mid R) \cdot r(d), \tag{2}$$

where λ can be chosen for a specific metric, e.g., for Average Relevance Position (ARP) or Discounted Cumulative Gain (DCG):

$$\lambda^{ARP}(d \mid R) = \text{rank}(d \mid R), \tag{3}$$

$$\lambda^{DCG}(d \mid R) = -\log_2 \left(1 + \text{rank}(d \mid R)\right)^{-1}. \tag{4}$$

In a so-called *full-information* setting, where the relevance values r are known, optimization can be done through traditional LTR methods [6, 16, 23, 34].

2.2 Counterfactual LTR

Optimizing a ranking loss from the implicit feedback in interaction logs requires a different approach from supervised LTR. We will assume that clicks are gathered using a logging policy π with the probability of displaying ranking $\bar{R}$ for query q denoted as $\pi(\bar{R} \mid q)$. Let $o_i(d) \in \{0, 1\}$ indicate whether d was examined by a user at interaction i and $o_i(d) \sim P(o(d) \mid q_i, r, \bar{R}_i)$. Furthermore, we assume that users click on all relevant items they observe and nothing else: $c_i(d) = \mathbb{1}[r(d) \wedge o_i(d)]$. Our goal is to find an estimator $\hat{\Delta}$ that provides an unbiased estimate of the actual loss for N interactions this estimate is:

$$\hat{\mathcal{L}} = \frac{1}{N} \sum_{i=1}^{N} \hat{\Delta}(R_i \mid q_i, \bar{R}_i, \pi, c_i). \tag{5}$$

We write R_i for the ranking produced by the system for which the loss is being computed, while $\bar{R}_i$ is the ranking that was displayed when logging interaction i. For brevity we will drop i from our notation when only a single interaction is involved. A naive estimator could simply consider every click to indicate relevance:

$$\hat{\Delta}_{naive}(R \mid q, c) = \sum_{d : c(d)=1} \lambda(d \mid R). \tag{6}$$

Taking the expectation over the displayed ranking and observance variables results in the following expected loss:

$$\begin{aligned}
&\mathbb{E}_{o,\bar{R}}\left[\hat{\Delta}_{naive}(R \mid q, c)\right] \\
&= \mathbb{E}_{o,\bar{R}}\left[\sum_{d : c(d)=1} \lambda(d \mid R)\right] = \mathbb{E}_{o,\bar{R}}\left[\sum_{d \in R} \lambda(d \mid R) \cdot c(d)\right] \\
&= \mathbb{E}_{o,\bar{R}}\left[\sum_{d \in R} o(d) \cdot \lambda(d \mid R) \cdot r(d)\right] \\
&= \mathbb{E}_{\bar{R}}\left[\sum_{d \in R} P\left(o(d) = 1 \mid q, r, \bar{R}\right) \cdot \lambda(d \mid R) \cdot r(d)\right] \\
&= \sum_{\bar{R} \in \pi(\cdot \mid q)} \pi(\bar{R} \mid q) \cdot \sum_{d \in R} P\left(o(d) = 1 \mid q, r, \bar{R}\right) \cdot \lambda(d \mid R) \cdot r(d).
\end{aligned} \tag{7}$$

Here, the effect of position bias is very clear; in expectation, items are weighted according to their probability of being examined. Furthermore, it shows that examination probabilities are determined by both the logging policy π and user behavior $P(o(d) \mid q, r, \bar{R})$.

In order to avoid the effect of position bias, Joachims et al. [18] introduced an inverse-propensity-scoring estimator in the same vain as previous work by Wang et al. [32]. The main idea behind this estimator is that if the examination probabilities are known, then they can be corrected for per click:

$$\hat{\Delta}_{oblivious}\left(R \mid q, c, \bar{R}\right) = \sum_{d : c(d)=1} \frac{\lambda(d \mid R)}{P\left(o(d) = 1 \mid q, r, \bar{R}\right)}. \tag{8}$$

In contrast to the naive estimator (Eq. 6), this policy-oblivious estimator (Eq. 8) can provide an unbiased estimate of the loss:

$$\begin{aligned}
&\mathbb{E}_{o,\bar{R}}\left[\hat{\Delta}_{oblivious}\left(R \mid q, c, \bar{R}\right)\right] \\
&= \mathbb{E}_{o,\bar{R}}\left[\sum_{d : c(d)=1} \frac{\lambda(d \mid R)}{P\left(o(d) = 1 \mid q, r, \bar{R}\right)}\right] \\
&= \sum_{d \in R} \mathbb{E}_{o,\bar{R}}\left[\frac{o(d) \cdot \lambda(d \mid R) \cdot r(d)}{P\left(o(d) = 1 \mid q, r, \bar{R}\right)}\right] \\
&= \sum_{d \in R} \mathbb{E}_{\bar{R}}\left[\frac{P\left(o(d) = 1 \mid q, r, \bar{R}\right) \cdot \lambda(d \mid R) \cdot r(d)}{P\left(o(d) = 1 \mid q, r, \bar{R}\right)}\right] \\
&= \sum_{d \in R} \lambda(d \mid R) \cdot r(d) = \Delta(R \mid q, r).
\end{aligned} \tag{9}$$

We note that the last step assumes $P\left(o(d) = 1 \mid q, r, \bar{R}\right) > 0$, and that only relevant items $r(d) = 1$ contribute to the estimate [18]. Therefore, this estimator is unbiased as long as the examination probabilities are positive for every relevant item:

$$\forall d, \forall \bar{R} \in \pi(\cdot \mid q)\ \left[r(d) = 1 \rightarrow P\left(o(d) = 1 \mid q, r, \bar{R}\right) > 0\right]. \tag{10}$$

Intuitively, this condition exists because propensity weighting is applied to items clicked in the displayed ranking and items that cannot be observed can never receive clicks. Thus, there are no clicks that can be weighted more heavily to adjust for the zero observance probability of an item.

An advantageous property of the policy-oblivious estimator $\hat{\Delta}_{oblivious}$ is that the logging policy π does not have to be known. That is, as long as Condition 10 is met, it works regardless of how interactions were logged. Additionally, Joachims et al. [18] proved that it is still unbiased under click noise. Virtually all recent counterfactual LTR methods use the policy-oblivious estimator for LTR optimization [2–4, 18, 32, 33].

3 LEARNING FROM TOP-k FEEDBACK

In this section we explain why the existing policy-oblivious counterfactual LTR framework is not applicable to top-k rankings. Subsequently, we propose a novel solution through policy-aware propensity scoring that takes the logging policy into account.

3.1 The Problem with Top-k Feedback

An advantage of the existing policy-oblivious estimator for counterfactual LTR described in Section 2.2 is that the logging policy does not need to be known, making its application easier. However, the policy-oblivious estimator is only unbiased when Condition 10 is met: every relevant item has a non-zero probability of being observed in every ranking displayed during logging.

We recognize that in top-k rankings, where only k items can be displayed, relevant items may systematically *lack* non-zero examination probabilities. This happens because items outside the top-k cannot be examined by the user:

$$\forall d, \forall \bar{R} \left[\text{rank}\big(d \mid \bar{R}\big) > k \rightarrow P\big(o(d) = 1 \mid q, r, \bar{R}\big) = 0\right]. \tag{11}$$

In most top-k ranking settings it is very unlikely that Condition 10 is satisfied; If k is very small, the number of relevant items is large, or if the logging policy π is ineffective at retrieving relevant items, it is unlikely that all relevant items will be displayed in the top-k positions. Moreover, for a small value of k the performance of the logging policy π has to be near ideal for all relevant items to be displayed. We call this effect *item selection bias*, because in this setting the logging ranker makes a selection of which k items to display, in addition to the order in which to display them (position bias). The existing policy-oblivious estimator for counterfactual LTR (as described in Section 2.2) cannot correct for item selection bias when it occurs, and can thus be affected by this bias when applied to top-k rankings.

3.2 Policy-Aware Propensity Scoring

Item selection bias is inevitable in a single top-k ranking, due to the limited number of items that can be displayed. However, across multiple top-k rankings more than k items could be displayed if the displayed rankings differ enough. Thus, a stochastic logging-policy could provide every item with a non-zero probability to appear in the top-k ranking. Then, the probability of examination can be calculated as an expectation over the displayed ranking:

$$\begin{aligned} P\left(o(d) = 1 \mid q, r, \pi\right) &= \mathbb{E}_{\bar{R}}\left[P\big(o(d) = 1 \mid q, r, \bar{R}\big)\right] \\ &= \sum_{\bar{R} \in \pi(\cdot|q)} \pi\big(\bar{R} \mid q\big) \cdot P\big(o(d) = 1 \mid q, r, \bar{R}\big). \end{aligned} \tag{12}$$

This policy-dependent examination probability can be non-zero for all items, even if all items cannot be displayed in a single top-k ranking. Naturally, this leads to a *policy-aware* estimator:

$$\hat{\Delta}_{aware}\left(R \mid q, c, \pi\right) = \sum_{d:c(d)=1} \frac{\lambda(d \mid R)}{P\big(o(d) = 1 \mid q, r, \pi\big)}. \tag{13}$$

By basing the propensity on the policy instead of the individual rankings, the policy-aware estimator can correct for zero observance probabilities in some displayed rankings by more heavily weighting clicks on other displayed rankings with non-zero observance probabilities. Thus, if a click occurs on an item that the logging policy rarely displays in a top-k ranking, this click may be weighted more heavily than a click on an item that is displayed in the top-k very often. In contrast, the policy-oblivious approach only corrects for the observation probability for the displayed ranking in which the click occurred, thus it does not correct for the fact that an item may be missing from the top-k in other displayed rankings.

In expectation, the policy-aware estimator provides an unbiased estimate of the ranking loss:

$$\begin{aligned} &\mathbb{E}_{o,\bar{R}}\left[\hat{\Delta}_{aware}\left(R \mid q, c, \pi\right)\right] \\ &= \mathbb{E}_{o,\bar{R}}\left[\sum_{d:c(d)=1} \frac{\lambda\big(d \mid R\big)}{P\left(o(d) = 1 \mid q, r, \pi\right)}\right] \\ &= \sum_{d \in R} \mathbb{E}_{o,\bar{R}}\left[\frac{o(d) \cdot \lambda\big(d \mid R\big) \cdot r(d)}{\sum_{\bar{R}' \in \pi(\cdot|q)} \pi\big(\bar{R}' \mid q\big) \cdot P\big(o(d) = 1 \mid q, r, \bar{R}'\big)}\right] \\ &= \sum_{d \in R} \mathbb{E}_{\bar{R}}\left[\frac{P\big(o(d) = 1 \mid q, r, \bar{R}\big) \cdot \lambda\big(d \mid R\big) \cdot r(d)}{\sum_{\bar{R}' \in \pi(\cdot|q)} \pi\big(\bar{R}' \mid q\big) \cdot P\big(o(d) = 1 \mid q, r, \bar{R}'\big)}\right] \\ &= \sum_{d \in R} \frac{\sum_{\bar{R} \in \pi(\cdot|q)} \pi\big(\bar{R} \mid q\big) \cdot P\big(o(d) = 1 \mid q, r, \bar{R}\big) \cdot \lambda\big(d \mid R\big) \cdot r(d)}{\sum_{\bar{R}' \in \pi(\cdot|q)} \pi\big(\bar{R}' \mid q\big) \cdot P\big(o(d) = 1 \mid q, r, \bar{R}'\big)} \\ &= \sum_{d \in R} \lambda\big(d \mid R\big) \cdot r(d) = \Delta\big(R \mid q, r\big). \end{aligned} \tag{14}$$

In contrast to the policy-oblivious approach (Section 2.2), this proof is sound as long as every relevant item has a non-zero probability of being examined under the logging policy π:

$$\forall d \left[r(d) = 1 \rightarrow \sum_{\bar{R} \in \pi(\cdot|q)} \pi\big(\bar{R} \mid q\big) \cdot P\big(o(d) = 1 \mid q, r, \bar{R}\big) > 0\right]. \tag{15}$$

It is easy to see that Condition 10 implies Condition 15, in other words, for all settings where the policy-oblivious estimator (Eq. 8) is unbiased, the policy-aware estimator (Eq. 13) is also unbiased. Conversely, Condition 15 does not imply Condition 10, thus there are cases where the policy-aware estimator is unbiased but the policy-oblivious estimator is not guaranteed to be.

To better understand for which policies Condition 15 is satisfied, we introduce a substitute Condition 16:

$$\forall d \Big[r(d) = 1 \rightarrow \exists \bar{R} \left[\pi\big(\bar{R} \mid q\big) > 0 \wedge P\big(o(d) = 1 \mid q, r, \bar{R}\big) > 0\right]\Big]. \tag{16}$$

Since Condition 16 is equivalent to Condition 15, we see that the policy-aware estimator is unbiased for any logging-policy that provides a non-zero probability for every relevant item to appear in a position with a non-zero examination probability. Thus to satisfy Condition 16 in a top-k ranking setting, every relevant item requires a non-zero probability of being displayed in the top-k.

As long as Condition 16 is met, a wide variety of policies can be chosen according to different criteria. Moreover, the policy can be deterministic if k is large enough to display every relevant item. Similarly, the policy-oblivious estimator can be seen as a special case of the policy-aware estimator where the policy is deterministic (or assumed to be). The big advantage of our policy-aware estimator is that it is applicable to a much larger number of settings than the existing policy-oblivious estimator, including those were feedback is only received on the top-k ranked items.

3.3 Illustrative Example

To better understand the difference between the policy-oblivious and policy-aware estimators, we introduce an illustrative example that contrasts the two. We consider a single query q and a logging policy π that chooses between two rankings to display: $\bar{R}_1$ and $\bar{R}_2$, with: $\pi(\bar{R}_1 \mid q) > 0$; $\pi(\bar{R}_2 \mid q) > 0$; and $\pi(\bar{R}_1 \mid q) + \pi(\bar{R}_2 \mid q) = 1$. Then for a generic estimator we consider how it treats a single relevant item d_n with $r(d_n) \neq 0$ using the expectation:

$$\mathbb{E}_{o,\bar{R}}\Big[\frac{c(d_n)\cdot\lambda\big(d_n \mid R\big)}{\rho\big(o(d_n)=1 \mid q,d_n,\bar{R},\pi\big)}\Big] = \lambda\big(d_n \mid R\big)\cdot r(d_n)\cdot \tag{17}$$
$$\Bigg(\frac{\pi(\bar{R}_1|q)\cdot P\big(o(d_n)=1|q,r,\bar{R}_1\big)}{\rho\big(o(d_n)=1 \mid q,d_n,\bar{R}_1,\pi\big)} + \frac{\pi(\bar{R}_2|q)\cdot P\big(o(d_n)=1|q,r,\bar{R}_2\big)}{\rho\big(o(d_n)=1 \mid q,d_n,\bar{R}_2,\pi\big)}\Bigg),$$

where the propensity function ρ can be chosen to match either the policy-oblivious (Eq. 8) or policy-aware (Eq. 13) estimator.

First, we examine the situation where d_n appears in the top-k of both rankings $\bar{R}_1$ and $\bar{R}_2$, thus it has a positive observance probability in both cases: $P\big(o(d_n)=1 \mid q,r,\bar{R}_1\big) > 0$ and $P\big(o(d_n)=1 \mid q,r,\bar{R}_2\big) > 0$. Here, the policy-oblivious estimator $\hat{\Delta}_{oblivious}$ (Eq. 8) removes the effect of observation bias by adjusting for the observance probability per displayed ranking:

$$\Bigg(\frac{\pi(\bar{R}_1|q)\cdot P\big(o(d_n)=1|q,r,\bar{R}_1\big)}{P\big(o(d_n)=1 \mid q,r,\bar{R}_1\big)} + \frac{\pi(\bar{R}_2|q)\cdot P\big(o(d_n)=1|q,r,\bar{R}_2\big)}{P\big(o(d_n)=1 \mid q,r,\bar{R}_2\big)}\Bigg)$$
$$\cdot\,\lambda\big(d_n \mid R\big)\cdot r(d_n) = \lambda\big(d_n \mid R\big)\cdot r(d_n). \tag{18}$$

The policy-aware estimator $\hat{\Delta}_{aware}$ (Eq. 13) also corrects for the examination bias, but because its propensity scores are based on the policy instead of the individual rankings (Eq. 12), it uses the same score for both rankings:

$$\frac{\pi(\bar{R}_1 \mid q)\cdot P\big(o(d_n)=1|\,q,r,\bar{R}_1\big) + \pi(\bar{R}_2 \mid q)\cdot P\big(o(d_n)=1|\,q,r,\bar{R}_2\big)}{\pi(\bar{R}_1 \mid q)\cdot P\big(o(d_n)=1|\,q,r,\bar{R}_1\big) + \pi(\bar{R}_2 \mid q)\cdot P\big(o(d_n)=1|\,q,r,\bar{R}_2\big)}$$
$$\cdot\,\lambda\big(d_n \mid R\big)\cdot r(d_n) = \lambda\big(d_n \mid R\big)\cdot r(d_n). \tag{19}$$

Then, we consider a different relevant item d_m with $r(d_m) = r(d_n)$ that unlike the previous situation only appears in the top-k of $\bar{R}_1$. Thus it only has a positive observance probability in $\bar{R}_1$: $P\big(o(d_m) = 1 \mid q,r,\bar{R}_1\big) > 0$ and $P\big(o(d_m)=1 \mid q,r,\bar{R}_2\big) = 0$. Consequently, no clicks will ever be received in $\bar{R}_2$, i.e., $\bar{R} = \bar{R}_2 \rightarrow c(d_m) = 0$, thus the expectation for d_m only has to consider $\bar{R}_1$:

$$\mathbb{E}_{o,\bar{R}}\Big[\frac{c(d_m)\cdot\lambda\big(d_m \mid R\big)}{\rho\big(o(d_m)=1 \mid q,d_m,\bar{R},\pi\big)}\Big]$$
$$= \frac{\pi(\bar{R}_1 \mid q)\cdot P\big(o(d_m)=1 \mid q,r,\bar{R}_1\big)}{\rho\big(o(d_m)=1 \mid q,d_m,\bar{R}_1,\pi\big)}\cdot\lambda\big(d_m \mid R\big)\cdot r(d_m). \tag{20}$$

In this situation, Condition 10 is not satisfied, and correspondingly, the policy-oblivious estimator (Eq. 8) does not give an unbiased estimate:

$$\frac{\pi(\bar{R}_1 \mid q)\cdot P\big(o(d_m)=1 \mid q,r,\bar{R}_1\big)}{P\big(o(d_m)=1 \mid q,r,\bar{R}_1\big)}\cdot\lambda\big(d_m \mid R\big)\cdot r(d_m)$$
$$< \lambda\big(d_m \mid R\big)\cdot r(d_m). \tag{21}$$

Since the policy-oblivious estimator $\hat{\Delta}_{oblivious}$ only corrects for the observance probability per displayed ranking, it is unable to correct for the zero probability in R_2 as no clicks on d_m can occur here. As a result, the estimate is affected by the logging policy π: the more item selection bias π introduces (determined by $\pi(\bar{R}_1 \mid q)$) the further the estimate will deviate. Consequently, in expectation $\hat{\Delta}_{oblivious}$ will biasedly estimate that d_n should be ranked higher than d_m, which is incorrect since both items are actually equally relevant.

In contrast, the policy-aware estimator $\hat{\Delta}_{aware}$ (Eq. 13) avoids this issue because its propensities are based on the logging policy π. When calculating the probability of observance conditioned on π, $P\big(o(d_m)=1 \mid q,r,\pi\big)$ (Eq. 12), it takes into account that there is a $\pi(\bar{R}_2 \mid q)$ chance that d_m is not displayed to the user:

$$\frac{\pi(\bar{R}_1|\,q)\cdot P\big(o(d_m)=1|\,q,r,\bar{R}_1\big)}{\pi(\bar{R}_1|\,q)\cdot P\big(o(d_m)=1|\,q,r,\bar{R}_1\big)}\cdot\lambda\big(d_m|\,R\big)\cdot r(d_m)$$
$$= \lambda\big(d_m|\,R\big)\cdot r(d_m). \tag{22}$$

Since in this situation Condition 16 is true (and therefore also Condition 15), we know beforehand that in expectation the policy-aware estimator is unaffected by position and item-selection bias.

This concludes our illustrative example. It was meant to contrast the behavior of the policy-aware and policy-oblivious estimators in two different situations. When there is no item selection bias, i.e., an item is displayed in the top-k of all rankings the logging policy may display, both estimators provide unbiased estimates albeit using different propensity scores. However, when there is item selection bias. i.e., an item is not always present in the top-k, the policy-oblivious estimator $\hat{\Delta}_{oblivious}$ no longer provides an unbiased estimate, while the policy-aware estimator $\hat{\Delta}_{aware}$ is still unbiased w.r.t. both position bias and item selection bias.

4 LEARNING FOR TOP-k METRICS

This section details how counterfactual LTR can be used to optimize top-k metrics, since these are the relevant metrics in top-k rankings.

4.1 Top-k Metrics

Since top-k rankings only display the k highest ranked items to the user, the performance of a ranker in this setting is only determined by those items. Correspondingly, only top-k metrics matter here, where items beyond rank k have no effect:

$$\lambda^{metric@k}\big(d \mid R\big) = \begin{cases}\lambda^{metric}\big(d \mid R\big), & \text{if } \text{rank}(d \mid R) \leq k,\\ 0, & \text{if } \text{rank}\big(d \mid R\big) > k.\end{cases} \tag{23}$$

These metrics are commonly used in LTR since, usually, performance gains in the top of a ranking are the most important for the user experience. For instance, NDCG@k, which is the normalized version of DCG@k, is often used:

$$\lambda^{DCG@k}\big(d \mid R\big) = \begin{cases}-\log_2\big(1+\text{rank}(d \mid R)\big)^{-1}, & \text{if } \text{rank}(d \mid R) \leq k,\\ 0, & \text{if } \text{rank}\big(d \mid R\big) > k.\end{cases} \tag{24}$$

Generally in LTR, DCG is optimized in order to maximize NDCG [6, 34]. In unbiased LTR it is not trivial to estimate the normalization factor for NDCG, further motivating the optimization of DCG instead of NDCG [1, 8].

Importantly, top-k metrics bring two main challenges for LTR. First, the rank function is not differentiable, a problem for almost every LTR metric [23, 34]. Second, changes in a ranking beyond position k do not affect the metric's value thus resulting in zero-gradients. The first problem has been addressed in existing LTR

methods, we will now propose adaptations of these methods that address the second issue as well.

4.2 Monotonic Upper Bounding

A common approach for enabling optimization of ranking methods, is by finding lower or upper bounds that can be minimized or maximized, respectively. For instance, similar to a hinge loss, the rank function can be upper bounded by a maximum over score differences [16, 18]. Let s be the scoring function used to rank (in descending order), then:

$$\text{rank}(d \mid R) \leq \sum_{d' \in R} \max\Big(1 - \big(s(d) - s(d')\big), 0\Big). \tag{25}$$

Alternatively, the logistic function is also a popular choice [34]:

$$\text{rank}(d \mid R) \leq \sum_{d' \in R} \log_2\Big(1 + e^{s(d') - s(d)}\Big). \tag{26}$$

Minimizing one of these differentiable upper bounds will directly minimize an upper bound on the ARP metric (Eq. 3).

Furthermore, Agarwal et al. [1] showed that this approach can be extended to any metric based on a monotonically decreasing function. For instance, if $\overline{\text{rank}}(d \mid R)$ is an upper bound on the $\text{rank}(d \mid R)$ function, then the following is an upper bound on the DCG loss (Eq. 4):

$$\begin{aligned}\lambda^{DCG}(d \mid R) &\leq -\log_2\big(1 + \overline{\text{rank}}(d \mid R)\big)^{-1} \\ &= \hat{\lambda}^{DCG}(d \mid R).\end{aligned} \tag{27}$$

More generally, let α be a monotonically decreasing function. A loss based on α is always upper bounded by:

$$\begin{aligned}\lambda^{\alpha}(d \mid R) &= -\alpha\big(\text{rank}(d \mid R)\big) \\ &\leq -\alpha\big(\overline{\text{rank}}(d \mid R)\big) = \hat{\lambda}^{\alpha}(d \mid R).\end{aligned} \tag{28}$$

Though appropriate for many standard ranking metrics, $\hat{\lambda}^{\alpha}$ is not an upper bound for top-k metric losses. To understand this, consider that an item beyond rank k may still receive a negative score from $\hat{\lambda}^{\alpha}$, for instance, for the DCG upper bound: $\text{rank}(d \mid R) > k \rightarrow \hat{\lambda}^{DCG}(d \mid R) < 0$. As a result, this is not an upper bound for a DCG@k based loss.

We propose a modification of the $\hat{\lambda}^{\alpha}$ function to provide an upper bound for top-k metric losses, by simply giving a positive penalty to items beyond rank k:

$$\hat{\lambda}^{\alpha@k}(d \mid R) = -\alpha\big(\overline{\text{rank}}(d \mid R)\big) + \mathbb{1}\big[\text{rank}(d \mid R) > k\big] \cdot \alpha(k). \tag{29}$$

The resulting function is an upper bound on top-k metric losses based on a monotonic function: $\lambda^{\alpha@k}(d \mid R) \leq \hat{\lambda}^{\alpha@k}(d \mid R)$. The main difference with $\hat{\lambda}^{\alpha}$ is that items beyond rank k acquire a positive score from $\lambda^{\alpha@k}$, thus providing an upper bound on the actual metric loss. Interestingly, the gradient of $\hat{\lambda}^{\alpha@k}$ w.r.t. the scoring function s is the same as that of $\hat{\lambda}^{\alpha}$.[1] Therefore, the gradient of either function optimizes an upper bound on $\lambda^{\alpha@k}$ top-k metric losses, while only $\hat{\lambda}^{\alpha@k}$ provides an actual upper bound.

While this monotonic function-based approach is simple, it is unclear how coarse these upper bounds are. In particular, some upper bounds on the rank function (e.g., Eq. 25) can provide gross overestimations. As a result, these upper bounds on ranking metric losses may be very far removed from their actual values.

[1] We consider the indicator function to never have a non-zero gradient.

4.3 Lambda-Based Losses for Counterfactual top-k LTR

Many supervised LTR approaches, such as the well-known LambdaRank and subsequent LambdaMART methods [6], are based on Expectation Maximization (EM) procedures [11]. Recently, Wang et al. [34] introduced the LambdaLoss framework, which provides a theoretical way to prove that a method optimizes a lower bound on a ranking metric. Subsequently, it was used to prove that LambdaMART optimizes such a bound on DCG, similarly it was also used to introduce the novel LambdaLoss method which provides an even tighter bound on DCG. In this section, we will show that the LambdaLoss framework can be used to find proven bounds on counterfactual LTR losses and top-k metrics. Since LambdaLoss is considered state-of-the-art in supervised LTR, making its framework applicable to counterfactual LTR could potentially provide competitive performance. Additionally, adapting the LambdaLoss framework to top-k metrics further expands its applicability.

The LambdaLoss framework and its EM-optimization approach work for metrics that can be expressed in item-based gains, $G(d_n \mid q, r)$, and discounts based on position, $D\big(\text{rank}(d_n \mid R)\big)$; for brevity we use the shorter G_n and D_n, respectively, resulting in:

$$\Delta\big(R \mid q, r\big) = \sum_{d_n \in R} G(d_n \mid q, r) \cdot D\big(\text{rank}(d_n \mid R)\big) = \sum_{n=1}^{|R|} G_n \cdot D_n. \tag{30}$$

For simplicity of notation, we choose indexes so that: $n = \text{rank}(d_n \mid R)$, thus D_n is always the discount for the rank n. Then, we differ from the existing LambdaLoss framework by allowing the discounts to be zero ($\forall n\, D_n \geq 0$), thus also accounting for top-k metrics. Furthermore, items at the first rank are not discounted or the metric can be scaled so that $D_1 = 1$. Additionally, higher ranked items should be discounted less or equally: $n > m \rightarrow D_n \leq D_m$. Most ranking metrics meet these criteria; for instance, G_n and D_n can be chosen to match ARP or DCG. Importantly, our adaption also allows Δ to match top-k metrics such as DCG@k or Precision@k.

In order to apply the LambdaLoss framework to counterfactual LTR, we consider a general inverse-propensity-scored estimator:

$$\hat{\Delta}_{IPS}(R \mid q, c, \cdot) = \sum_{d_n : c(d_n) = 1} \frac{\lambda(d_n \mid R)}{\rho\big(o(d_n) = 1 \mid q, r, \bar{R}, \pi\big)}, \tag{31}$$

where the propensity function ρ can match either the policy-oblivious (Eq. 8) or the policy-aware (Eq. 13) estimator. By choosing

$$G_n = \frac{1}{\rho\big(o(d_n) = 1 \mid q, r, \bar{R}, \pi\big)} \text{ and } D_n = \lambda(d_n \mid R), \tag{32}$$

the estimator can be described in terms of gains and discounts. In contrast, in the existing LambdaLoss framework [34] gains are based on item relevance. For counterfactual top-k LTR, we have designed Eq. 32 so that gains are based on the propensity scores of observed clicks, and the discounts can have zero values.

The EM-optimization procedure alternates between an expectation step and a maximization step. In our case, the expectation step sets the discount values D_n according to the current ranking R of

the scoring function s. Then the maximization step updates s to optimize the ranking model. Following the LambdaLoss framework [34], we derive a slightly different loss. With the delta function:

$$\delta_{nm} = D_{|n-m|} - D_{|n-m|+1}, \tag{33}$$

our differentiable *counterfactual loss* becomes:

$$\sum_{G_n > G_m} -\log_2 \left(\left(\frac{1}{1 + e^{s(d_m) - s(d_n)}} \right)^{\delta_{nm} \cdot |G_n - G_m|} \right). \tag{34}$$

The changes we made do not change the validity of the proof provided in the original LambdaLoss paper [34]. Therefore, the counterfactual loss (Eq. 34) can be proven to optimize a lower bound on counterfactual estimates of top-k metrics.

Finally, in the same way the LambdaLoss framework can also be used to derive counterfactual variants of other supervised LTR losses/methods such as LambdaRank or LamdbaMART. Unlike previous work that also attempted to find a counterfactual lambda-based method by introducing a pairwise-based estimator [13], our approach is compatible with the prevalent counterfactual approach since it uses the same estimator based on single-document propensities [2–4, 18, 32, 33]. Our approach suggests that the division between supervised and counterfactual LTR methods may disappear in the future, as a state-of-the-art supervised LTR method can now be applied to the state-of-the-art counterfactual LTR estimators.

4.4 Unbiased Loss Selection

So far we have introduced two counterfactual LTR approaches that are proven to optimize lower bounds on top-k metrics: with monotonic functions (Section 4.2) and through the LambdaLoss framework (Section 4.3). To the best of our knowledge, we are the first to introduce theoretically proven lower bounds for top-k LTR metrics. Nevertheless, previous work has also attempted to optimize top-k metrics, albeit through heuristic methods. Notably, Wang et al. [34] used a truncated version of the LambdaLoss loss to optimize DCG@k. Their loss uses the discounts D_n based on full-ranking DCG but ignores item pairs outside of the top-k:

$$\sum_{G_n > G_m} -\mathbb{1}\left[n \leq k \vee m \leq k\right] \cdot \log_2 \left(\left(\frac{1}{1 + e^{s(d_m) - s(d_n)}} \right)^{\delta_{nm} \cdot |G_n - G_m|} \right). \tag{35}$$

While empirical results motivate its usage, there is no known theoretical justification for this loss, and thus it is considered a heuristic.

This leaves us with a choice between two theoretically motivated counterfactual LTR approaches for optimizing top-k metrics (Eq. 29 and 34) and an empirically motivated heuristic (Eq. 35). We propose a pragmatic solution by recognizing that counterfactual estimators can unbiasedly evaluate top-k metrics. Therefore, in practice one can optimize several ranking models using various approaches, and subsequently, estimate which resulting model provides the best performance. Thus, using counterfactual evaluation to select from resulting models is an unbiased method to choose between the available counterfactual LTR approaches.

5 EXPERIMENTAL SETUP

We follow the standard setup in unbiased LTR [4, 8, 15, 18] and perform semi-synthetic experiments: queries and items are based on datasets of commercial search engines and interactions are simulated using probabilistic click models.

5.1 Datasets

We use the queries and documents from two of the largest publicly available LTR datasets: MLSR-WEB30K [26] and Yahoo! Webscope [9]. Each was created by a commercial search engine and contains a set of queries with corresponding preselected document sets. Query-document pairs are represented by feature vectors and five-grade relevance annotations ranging from not relevant (0) to perfectly relevant (4). In order to binarize the relevancy, we only consider the two highest relevance grades as relevant. The MSLR dataset contains 30 000 queries with on average 125 preselected documents per query, and encodes query-document pairs in 136 features. The Yahoo dataset has 29 921 queries and on average 24 documents per query encoded in 700 features. Presumably, learning from top-k feedback is harder as k becomes a smaller percentage of the number of items. Thus, we expect the MSLR dataset with more documents per query to pose a more difficult problem.

5.2 Simulating Top-k Settings

The setting we simulate is one where interactions are gathered using a non-optimal but decent production ranker. We follow existing work [4, 15, 18] and use supervised optimization for the ARP metric on 1% of the training data. The resulting model simulates a real-world production ranker since it is much better than a random initialization but leaves enough room for improvement [18].

We then simulate user-issued queries by uniformly sampling from the training partition of the dataset. Subsequently, for each query the production ranker ranks the documents preselected by the dataset. Depending on the experimental run that we consider, randomization is performed on the resulting rankings. In order for the policy-aware estimator to be unbiased, every relevant document needs a chance of appearing in the top-k (Condition 16). Since in a realistic setting relevancy is unknown, we choose to give every document a non-zero probability of appearing in the top-k. Our randomization policy takes the ranking of the production ranker and leaves the first $k-1$ documents unchanged but the document at position k is selected by sampling uniformly from the remaining documents. The result is a minimally invasive randomized top-k ranking since most of the ranking is unchanged and the placement of the sampled documents is limited to the least important position.

We note that many other logging policies could be applied (see Condition 16), e.g., an alternative policy could insert sampled documents at random ranks for less obvious randomization. Unfortunately, a full exploration of the effect of using different logging policies is beyond the scope of this work.

Clicks are simulated on the resulting ranking $\bar{R}$ according to position bias and document relevance. Top-k position bias is modelled through the probability of observance, as follows:

$$P\big(o(d) = 1 \mid q, r, \bar{R}\big) = \begin{cases} \text{rank}(d \mid \bar{R})^{-1}, & \text{if rank}(d \mid \bar{R}) \leq k, \\ 0, & \text{if rank}(d \mid \bar{R}) > k. \end{cases} \tag{36}$$

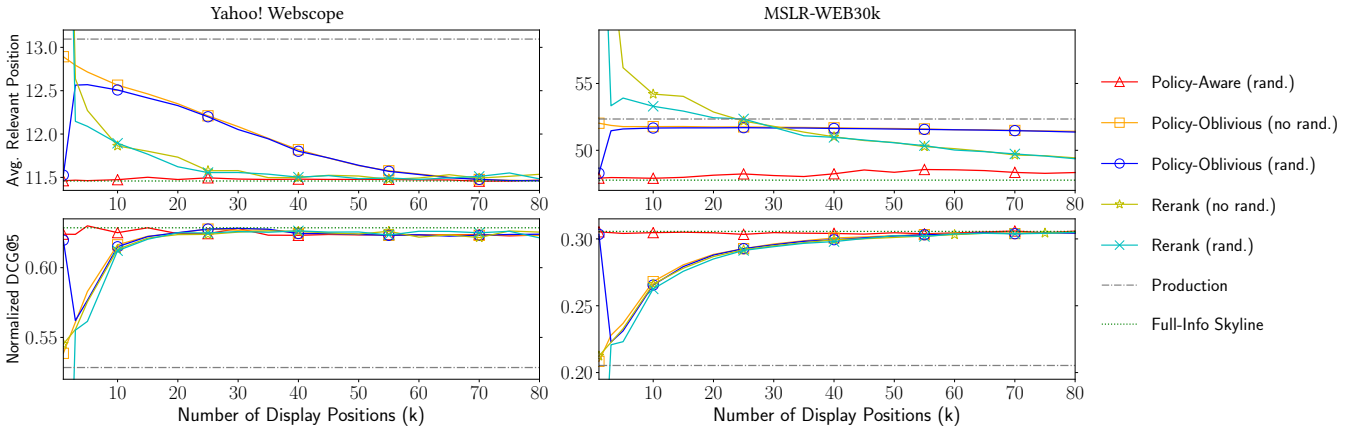

Figure 1: The effect of item selection bias on different estimators. Optimization on 10^8 clicks simulated on top-k rankings with varying number of display positions (k), with and without randomization (for each datapoint 10^8 clicks were simulated independently). The top row optimizes the average relevance position through the linear upper bound (Eq. 25); the bottom row optimizes DCG@5 using the truncated LambdaLoss (Eq. 35). Left: results on the *Yahoo* dataset; right: on the *MSLR* dataset.

The randomization policy results in the following examination probabilities w.r.t. the logging policy (cf. Eq. 12):

$$P\big(o(d)=1 \mid q,r,\pi\big) = \begin{cases} \text{rank}(d \mid \bar{R})^{-1}, & \text{if rank}(d \mid \bar{R}) < k, \\ \big(\text{rank}(d \mid \bar{R}) \cdot (|\bar{R}| - k + 1)\big)^{-1}, & \text{if rank}(d \mid \bar{R}) \geq k. \end{cases} \tag{37}$$

The probability of a click is conditioned on the relevance of the document according to the dataset:

$$P\big(c(d)=1 \mid q,r,\bar{R},o\big) = \begin{cases} 1, & \text{if } r(d)=1 \wedge o(d)=1, \\ 0.1, & \text{if } r(d)=0 \wedge o(d)=1, \\ 0, & \text{if } o(d)=0. \end{cases} \tag{38}$$

Note that our previous assumption that clicks only take place on relevant items (Section 2.2) is not true in our experiments.

Optimization is performed on training clicks simulated on the training partition of the dataset. Hyperparameter tuning is done by estimating performance on (unclipped) validation clicks simulated on the validation partition; the number of validation clicks is always 15% of the number of training clicks. Lastly, evaluation metrics are calculated on the test partition using the dataset labels.

5.3 Experimental Runs

In order to evaluate the performance of the policy-aware estimator (Eq. 13) and the effect of item selection bias, we compare with the following baselines: (i) The policy-oblivious estimator (Eq. 8). In our setting, where the examination probabilities are known beforehand, the policy-oblivious estimator also represents methods that jointly estimate these probabilities while performing LTR, i.e., the following methods reduce to this estimator if the examination probabilities are given: [2, 4, 18, 32]. (ii) A rerank estimator, an adaption of the policy-oblivious estimator. During optimization the rerank estimator applies the policy-oblivious estimator but limits the document set of an interaction i to the k displayed items $R_i = \{d \mid \text{rank}(d|\bar{R}_i) \leq k\}$ (cf. Eq. 8). Thus, it is optimized to rerank the top-k of the production ranker only, but during inference it is applied to the entire document set. (iii) Additionally, we evaluate performance without any cutoff k or randomization; in these circumstances all three estimators (Policy-Aware, Policy-Oblivious, Rerank) are equivalent. (iv) Lastly, we use supervised LTR on the dataset labels to get a *full-information skyline*, which shows the hypothetical optimal performance.

To evaluate the effectiveness of our proposed loss functions for optimizing top-k metrics, we apply the monotonic lower bound (Eq. 29) with a linear (Eq. 25) and a logistic upper bound (Eq. 26). Additionally, we apply several versions of the LamdbaLoss loss function (Eq. 34): one that optimizes full DCG, another that optimizes DCG@5, and the heuristic truncated loss also optimizing DCG@5 (Eq. 35). Lastly, we apply unbiased loss selection where we select the best-performing model based on the estimated performance on the (unclipped) validation clicks.

Optimization is done with stochastic gradient descent; to maximize computational efficiency we rewrite the loss (Eq. 5) for a propensity scoring function ρ in the following manner:

$$\begin{aligned} \hat{\mathcal{L}} &= \frac{1}{N}\sum_{i=1}^{N} \hat{\Delta}\big(R_i|q_i,\bar{R}_i,\pi,c_i\big) = \frac{1}{N}\sum_{i=1}^{N} \sum_{d:c_i(d)=1} \frac{\lambda\big(d \mid R_i\big)}{\rho\big(o_i(d)=1|q_i,r,\cdot\big)} \\ &= \frac{1}{N}\sum_{q\in Q}\sum_{d\in R_q} \left(\sum_{i=1}^{N} \frac{\mathbb{1}[q_i=q]\cdot c_i(d)}{\rho\big(o_i(d)=1 \mid q,r,\cdot\big)}\right) \cdot \lambda\big(d \mid R_q\big) \\ &= \frac{1}{N}\sum_{q\in Q}\sum_{d\in R_q} \omega_d \cdot \lambda\big(d \mid R_q\big). \end{aligned} \tag{39}$$

After precomputing the document weights ω_d, the complexity of computing the loss is only determined by the dataset size. This allows us to optimize over very large numbers of clicks with very limited increases in computational costs.

We optimize linear models, but our approach can be applied to any differentiable model [1]. Propensity clipping [18] is applied to training clicks and never applied to the validation clicks; we also use self-normalization [29].

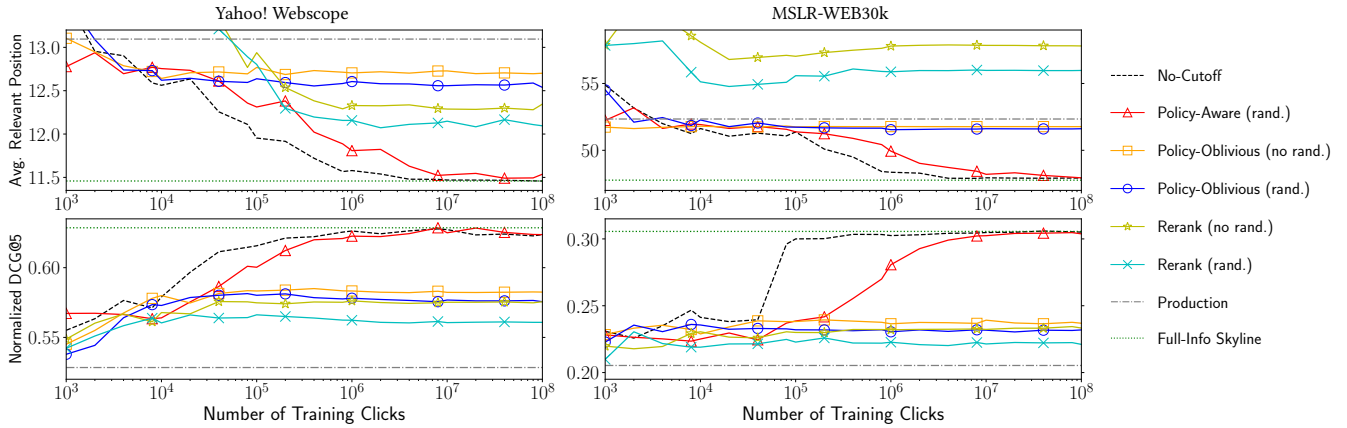

Figure 2: Performance of different estimators learning from different numbers of clicks simulated on top-5 rankings, with and without randomization. The top row optimizes the average relevance position through the linear upper bound (Eq. 25); the bottom row optimizes DCG@5 using the truncated LambdaLoss (Eq. 35).

6 RESULTS AND DISCUSSION

In this section we discuss the results of our experiments and evaluate our policy-aware estimator and the methods for top-k LTR metric optimization empirically.

6.1 Learning under Item Selection Bias

First we consider the question: *Is the policy-aware estimator effective for unbiased counterfactual LTR from top-k feedback?* Figure 1 displays the performance of different approaches after optimization on 10^8 clicks under varying values for k. Both the policy-oblivious and rerank estimators are greatly affected by the item selection bias introduced by the cutoff at k. On the MSLR dataset neither approach is able to get close to optimal ARP performance, optimal DCG@5 is only reached when $k > 50$. On the Yahoo dataset, the policy-oblivous approach can only approximate optimal ARP when $k > 60$; for DCG@5 it requires $k > 25$. The rerank approach reaches optimal ARP when $k > 50$ and optimal DCG@5 when $k > 20$. Considering that on average a query in the Yahoo dataset only has 24 preselected documents, it appears that even a little item selection bias has a substantial effect on both estimators. Furthermore, randomization appears to have a very limited positive effect on the policy-oblivious and rerank approaches. The one exception is the policy-oblivious approach when $k = 1$ where it reaches optimal performance under randomization. Here, the randomization policy gives every item an equal probability of being presented, thus trivially removing item selection bias; additionally, there is no position bias as there is only a single position. However, besides this trivial exception, the baseline estimators are strongly affected by item selection bias and simply logging with randomization is unable to remove the effect of item selection bias.

In contrast, the policy-aware approach is hardly affected by the choice of k. It consistently approximates optimal performance in terms of ARP and DCG@5 on both datasets. On the MSLR dataset, the policy-aware approach provides near optimal ARP performance; however, for $k > 15$ there is a small but noticeable gap. We suspect that this is a result of variance from click-noise and can be closed by gathering more clicks. Across all settings, the policy-aware approach appears unaffected by the choice of k and thus the effect of item selection bias. Moreover, it consistently provides performance at least as good as the baselines; and on the Yahoo dataset it outperforms them for $k < 20$ and on the MSLR dataset outperforms them for all tested values of k. We note that the randomization policy is the same for all methods; in other words, under randomization the clicks for the policy-oblivious, policy-aware and rerank approaches are acquired in the exact same way. Thus, our results show that in order to benefit from randomization, a counterfactual LTR method has to take its effect into account, hence only the policy-aware approach has improved performance.

Figure 2 displays the performance when learning from top-5 feedback while varying the number of clicks. Here we see that the policy-oblivious approach performance is stable after 10^5 clicks have been gathered. The rerank approach has stable performance after 10^6 clicks when optimized for ARP and 10^5 for DCG@5. Both baseline approaches show biased behavior where adding additional data does not lead to improved performance. This confirms that their estimators are unable to deal with item selection bias. In contrast, the policy-aware approach reaches optimal performance in all settings. However, it appears that the policy-aware approach requires more clicks than the no-cutoff baseline; we suspect that this difference is due to variance added by the randomization and smaller propensity scores.

In conclusion, we answer our first question positively: our results show that the policy-aware approach is unbiased w.r.t. item selection bias and position bias. Where all baseline approaches are affected by item selection bias even in small amounts, the policy-aware approach approximates optimal performance regardless of the cutoff value k.

6.2 Optimizing Top-k Metrics

Next, we consider the question: *Are our novel counterfactual LTR loss functions effective for top-k LTR metric optimization?* Figure 3 shows the performance of the policy-aware approach after optimizing different loss functions under top-5 feedback. While on the Yahoo dataset little differences are observed, on the MSLR dataset substantial differences are found. Interestingly, there seems to be no advantage in optimizing for DCG@5 instead of full DCG with the LambdaLoss. Furthermore, the monotonic loss function works

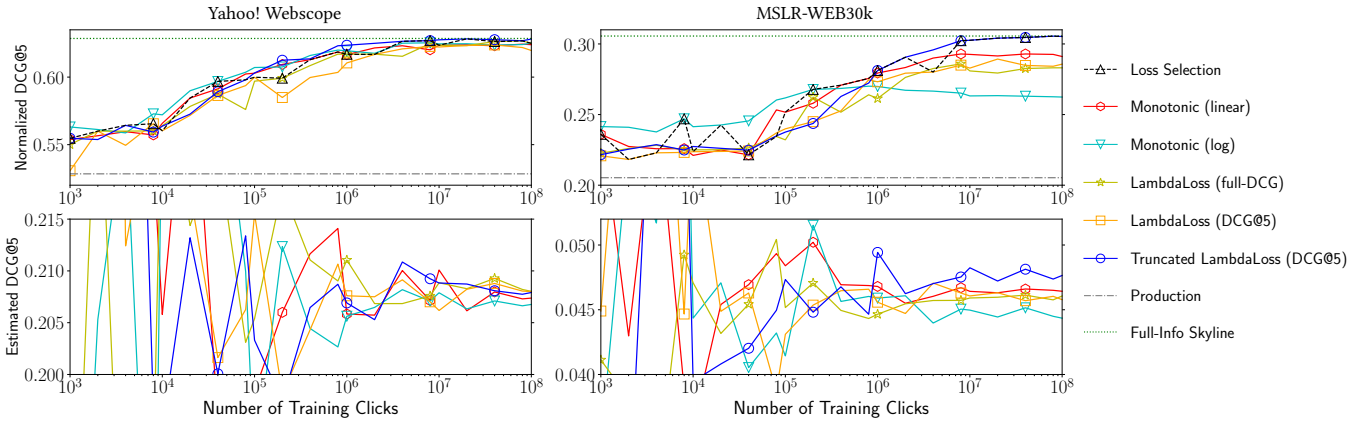

Figure 3: Performance of the policy-aware estimator (Eq. 13) optimizing DCG@5 using different loss functions. The loss selection method selects the estimated optimal model based on clicks gathered on separate validation queries. Varying numbers of clicks on top-5 rankings with randomization, the number of validation clicks is 15% of the number of training clicks.

very well with a linear upper bound, yet poorly when using the log upper bound. On both datasets the heuristic truncated LambdaLoss loss function provides the best performance, despite being the only method without a theoretical basis. When few clicks are available, the differences change; e.g., the monotonic loss function with a log upper bound outperforms the other losses on the MSLR dataset when fewer than 10^5 clicks are available.

Finally, we consider unbiased loss selection; Figure 3 displays both the performance of the selected models and the estimated performance on which the selections are based. For the most part the optimal models are selected, but variance does cause mistakes in selection when few clicks are available. Thus, unbiased optimal loss selection seems effective as long as enough clicks are available.

In conclusion, we answer our second question positively: our results indicate that the truncated counterfactual LambdaLoss loss function is most effective at optimizing DCG@5. Using this loss, our counterfactual LTR method reaches state-of-the-art performance comparable to supervised LTR on both datasets. Alternatively, our proposed unbiased loss selection method can choose optimally between models that are optimized by different loss functions.

7 RELATED WORK

Section 2.1 has discussed supervised LTR and Section 2.2 has described the existing counterfactual LTR framework; this section contrasts additional related work with our policy-aware approach.

Interestingly, some existing work in unbiased LTR was performed in top-k rankings settings [2, 3, 32, 33]. Our findings suggest that the results of that work are affected by item selection bias and that there is the potential for considerable improvements by applying the policy-aware method.

Carterette and Chandar [8] recognized that counterfactual evaluation cannot evaluate rankers that retrieve items that are unseen in the interaction logs, essentially due to a form of item selection bias. Their proposed solution is to gather new interactions on rankings where previously unseen items are randomly injected. Accordingly, they adapt propensity scoring to account for the random injection strategy. In retrospect, this approach can be seen as a specific instance of our policy-aware approach. In contrast, we have focused on settings where item selection bias takes place systematically and propose that logs should be gathered by any policy that meets Condition 16. Instead of expanding the logs to correct for missing items, our approach avoids systematic item selection bias altogether.

Other previous work has also used propensity scores based on a logging policy and examination probabilities. Komiyama et al. [19] and subsequently Lagrée et al. [20] use such propensities to find the optimal ranking for a single query by casting the ranking problem as a multiple-play bandit. Li et al. [21] use similar propensities to counterfactually evaluate ranking policies where they estimate the number of clicks a ranking policy will receive. Our policy-aware approach contrasts with these existing methods by providing an unbiased estimate of LTR-metric-based losses, and thus it can be used to optimize LTR models similar to supervised LTR.

Lastly, online LTR methods where interactive processes learn from the user [35] also make use of stochastic ranking policies. They correct for biases through randomization in rankings but do not use an explicit model of examination probabilities. To contrast with counterfactual LTR, while online LTR methods appear to provide robust performance [15], they are not proven to unbiasedly optimize LTR metrics [24, 25]. Unlike counterfactual LTR, they are not effective when applied to historical interaction logs [12].

8 CONCLUSION

In this work, we have proposed a policy-aware estimator for LTR, the first counterfactual method that is unbiased w.r.t. both position bias and item selection bias. Our experimental results show that existing policy-oblivious approaches are greatly affected by item selection bias, even when only small amounts are present. In contrast, the proposed policy-aware LTR method can learn from top-k feedback without being affected by the choice of k. Furthermore, we proposed three counterfactual LTR approaches for optimizing top-k metrics: two theoretically proven lower bounds on DCG@k based on monotonic functions and the LambdaLoss framework, respectively, and another heuristic truncated loss. Additionally, we introduced unbiased loss selection that can choose optimally between models optimized with different loss functions. Together, our contributions provide a method for learning from top-k feedback and for top-k metrics. To the best of our knowledge, this is the first counterfactual LTR method that is unbiased in top-k ranking

settings. Arguably, this work also serves to further bridge the gap between supervised and counterfactual LTR methods, as we have shown that state-of-the-art lambda-based supervised LTR methods can be applied to the state-of-the-art counterfactual LTR estimators.

Future work in supervised LTR could verify whether potential novel supervised methods can be applied to counterfactual losses. A limitation of the policy-aware LTR approach is that the logging policy needs to be known; future work could investigate whether a policy estimated from logs also suffices [21, 22]. Finally, existing work on bias in recommendation [27] has not considered position bias, thus we anticipate further opportunities for counterfactual LTR methods for top-k recommendations.

ACKNOWLEDGMENTS

We want to thank the anonymous reviewers for their feedback and Rolf Jagerman and Jin Huang for their helpful comments. This research was partially supported by the Netherlands Organisation for Scientific Research (NWO) under project nr 612.001.551 and by the Innovation Center for AI (ICAI). 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/2020topkunbiasedltr.

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] Wolf-Tilo Balke, Ulrich Güntzer, and Werner Kießling. 2002. On Real-Time Top k Querying for Mobile Services. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 125–143.

[6] Christopher J.C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report MSR-TR-2010-82. Microsoft.

[7] Fei Cai and Maarten de Rijke. 2016. A Survey of Query Auto Completion in Information Retrieval. *Foundations and Trends in Information Retrieval* 10, 4 (2016), 273–363.

[8] 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*. ACM, 705–714.

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

[10] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of Recommender Algorithms on Top-n Recommendation Tasks. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 39–46.

[11] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* 39, 1 (1977), 1–22.

[12] 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.

[13] 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*. ACM, 2830–2836.

[14] Neil Hurley and Mi Zhang. 2011. Novelty and Diversity in Top-n Recommendation–Analysis and Evaluation. *ACM Transactions on Internet Technology (TOIT)* 10, 4 (2011), 14.

[15] 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.

[16] 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.

[17] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *SIGIR*. ACM, 154–161.

[18] 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.

[19] Junpei Komiyama, Junya Honda, and Hiroshi Nakagawa. 2015. Optimal Regret Analysis of Thompson Sampling in Stochastic Multi-armed Bandit Problem with Multiple Plays. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15)*. JMLR.org, 1152–1161.

[20] Paul Lagrée, Claire Vernade, and Olivier Cappé. 2016. Multiple-play Bandits in the Position-based Model. In *Advances in Neural Information Processing Systems*. 1597–1605.

[21] Shuai Li, Yasin Abbasi-Yadkori, Branislav Kveton, S. Muthukrishnan, Vishwa Vinay, and Zheng Wen. 2018. Offline Evaluation of Ranking Policies with Click Models. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1685–1694.

[22] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. 2018. Breaking the Curse of Horizon: Infinite-Horizon Off-Policy Estimation. In *Advances in Neural Information Processing Systems*. 5356–5366.

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

[24] 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.

[25] 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.

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

[27] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations As Treatments: Debiasing Learning and Evaluation. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 1670–1679.

[28] Igor Shalyminov, Ondřej Dušek, and Oliver Lemon. 2018. Neural Response Ranking for Social Conversation: A Data-Efficient Approach. In *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd Int'l Workshop on Search-Oriented Conversational AI*. 1–8.

[29] Adith Swaminathan and Thorsten Joachims. 2015. The Self-Normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*. 3231–3239.

[30] Vladimir Vapnik. 2013. *The Nature of Statistical Learning Theory*. Springer Science & Business Media.

[31] Akrivi Vlachou, Christos Doulkeridis, and Kjetil Nørvåg. 2011. Monitoring Reverse Top-k Queries over Mobile Devices. In *Proceedings of the 10th ACM International Workshop on Data Engineering for Wireless and Mobile Access*. ACM, 17–24.

[32] 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.

[33] 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.

[34] 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.

[35] 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.