A General Framework for Pairwise Unbiased Learning to Rank

Alexey Kurennoy
alexey.kurennoy@zalando.ie
Zalando
Dublin, Ireland

John Coleman
john.coleman@zalando.ie
Zalando
Dublin, Ireland

Ian Harris
ian.harris@zalando.ie
Zalando
Dublin, Ireland

Alice Lynch
alice.lynch@zalando.ie
Zalando
Dublin, Ireland

Oisin Mac Fhearai
oisin.mac.fhearai@zalando.ie
Zalando
Dublin, Ireland

Daphne Tsatsoulis
daphne.tsatsoulis@zalando.ie
Zalando
Dublin, Ireland

ABSTRACT

Pairwise debiasing is one of the most effective strategies in reducing position bias in learning-to-rank (LTR) models. However, limiting the scope of this strategy, are the underlying assumptions required by many pairwise debiasing approaches. In this paper, we develop an approach based on a minimalistic set of assumptions that can be applied to a much broader range of user browsing patterns and arbitrary presentation layouts. We implement the approach as a simplified version of the Unbiased LambdaMART and demonstrate that it retains the underlying unbiasedness property in a wider variety of settings than the original algorithm. Finally, using simulations with "golden" relevance labels, we will show that the simplified version compares favourably with the original Unbiased LambdaMART when the examination of different positions in a ranked list is not assumed to be independent.

CCS CONCEPTS

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

KEYWORDS

ranking, position bias, unbiased learning-to-rank, pairwise debiasing

ACM Reference Format:
Alexey Kurennoy, John Coleman, Ian Harris, Alice Lynch, Oisin Mac Fhearai, and Daphne Tsatsoulis. 2022. A General Framework for Pairwise Unbiased Learning to Rank. In *Proceedings of the 2022 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '22), July 11–12, 2022, Madrid, Spain.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539813.3545119

1 INTRODUCTION

Machine learning models for information retrieval and recommendation are typically trained on implicit user feedback (such as clicks, for instance). Implicit feedback has several attractive properties. For example, it is abundant and relatively cheap to obtain. However, it is prone to presentation biases. This means that the implicit feedback corresponding to a certain item depends on the way the item was presented to users during the feedback collection. One important type of presentation bias that is especially pronounced in ranking applications is position bias. This bias arises because user attention is not spread equally between different positions in a ranked list and some of the positions are seen or attract attention more frequently than others.

The position bias renders items that are ranked low by the existing production system as less relevant to users than they really are. Consequently, a machine learning method applied to the collected data tends to mimic the current system. If the current production model is not optimal, its sub-optimality is (at least, partially) passed onto the new model. As a result, making improvements to the existing ranking system becomes more difficult. This has several undesirable implications such as worse user experience, lower revenues, or fairness problems.

There has been a considerable amount of research on ways to eliminate or reduce the position bias in learning-to-rank and recommendation models. Both the offline [1, 13] and the online [18] environments have been considered and there exists work aiming to unify the two [4, 19]. We also refer the reader to a recent survey [7]. Note that the theme of position bias reduction is different from offline policy evaluation (see [16, 17, 23] and other references in [15]) even though the two domains have some similarities. While offline policy evaluation focuses on assessing the loss that would have been observed under a different policy the goal of debiasing is to estimate the value of the loss that we would have observed if the target signal based on the implicit feedback was not distorted due to the position bias. From this perspective, debiasing is relevant even when the only aim is to evaluate the existing ranking or recommender system (i. e. the logging policy).

A lot of papers on the topic of position bias removal, including the seminal work [13] and its subsequent generalisation [1], focused on modifying objective functions that involve summations over individual items in the training data.

Hu et. al. [11] proposed an alternative approach starting off of a pairwise loss function which is a sum of terms that depend on pairs of items rather than individual items. This approach, called *pairwise debiasing*, gave rise to Unbiased LambdaMART - a state-of-the-art method for unbiased learning-to-rank.

In this paper, we describe a general framework for pairwise unbiased learning-to-rank. In contrast to existing theories, it relies

ICTIR '22, July 11–12, 2022, Madrid, Spain

ACM ISBN 978-1-4503-9412-3/22/07…$15.00
https://doi.org/10.1145/3539813.3545119

on a smaller and more realistic set of assumptions. Importantly, we do not require that the examination of different positions happen independently. For example, if the user is presented with a number of choices, the independent examination assumption would mean that examining the bottom-most option does not increase in any way the chances that earlier positions have been observed too. However, when users tend to examine the choices in a top-down fashion this property is unlikely to hold.

Furthermore, we do not assume that the probability of irrelevance and click absence (conditional on item and context features) are proportional at each position. See [4, Section 4.1.4] for a discussion of why this assumption is undesirable.

In addition to the above, our framework allows for arbitrary presentation layouts and thus covers both web search where results are typically displayed in a list and e-commerce where users are usually presented with a grid of products.

We demonstrate how the framework can be used in several important contexts to produce unbiased learning-to-rank algorithms. We also utilise the framework to show that a simplified version of Unbiased LambdaMART maintains the underlying unbiasedness property in a wider range of settings than the original algorithm. We compare the simplified and the original Unbiased LambdaMART in a semi-synthetic experiment and find that the simplified version compares favourably to the original Unbiased LambdaMART when the examination of different positions is not independent.

The contributions of the paper can be summarised as follows.

- **Theory**
  We propose a general pairwise debiasing framework allowing for arbitrary presentation layouts and a broad range of user browsing patterns (including those in which there is dependence in the examination of different positions). To the best of our knowledge, this framework has the weakest set of assumptions to date.
- **Methods**
  - We show how the framework can be used to produce unbiased learning-to-rank methods for important types of user behaviour found in e-commerce and web search.
  - We demonstrate that a simplified version of Unbiased LambdaMART is robust in the sense that the underlying unbiasedness property holds in a broad range of settings.
- **Offline Experiments**
  We conduct an offline semi-synthetic experiment (based on public data with "golden" relevance labels) and find that the simplified version of Unbiased LambdaMART compares favourably with the original algorithm when the examination of different positions does not happen independently.

The paper is structured as follows. We discuss related work in Section 2. Section 3 presents the proposed pairwise debiasing framework. This includes stating its assumptions, formulating a novel unbiased version of the pairwise loss function, and proving its unbiasedness. In Section 4, we demonstrate how our framework can be used in several important settings to produce unbiased learning-to-rank algorithms. Section 5 demonstrates that Unbiased LambdaMART can be modified so that the underlying unbiasedness property holds in a wider range of situations. Finally, Section 6 contains the results of a semi-synthetic experiment in which we compare the performance of the original and the simplified Unbiased LambdaMART.

2 RELATED WORK

The idea of pairwise debiasing along with the Unbiased LambdaMART method were introduced in [11]. In addition to the standard examination hypothesis [8, Section 3.3] and the positivity of observation propensities, [11] assumes that clicks on different items happen independently of each other and that the probabilities of irrelevance and click absence are proportional for each position.

A recent work [20] proposes an unbiased pairwise loss function in the context of collaborative filtering with implicit feedback. It avoids the assumption about the proportionality between the irrelevance and click absence probabilities but still assumes independence between the examination of different positions. The eye-tracking experiments in [12] suggest that people generally view web search results from top to bottom which makes the reliance on the assumption about the examination independence undesirable. When the results are observed in a top-to-bottom fashion, the fact that an item down the list has been observed increases the chances that earlier items have been observed too and hence, the independence of examination indicators cannot hold. As will be seen in Section 4, our framework encompasses [20] as a special case.

Guo et. al. [10] build an unbiased learning-to-rank method focusing on the context of e-commerce. In this paper, we develop a unified approach that can tackle both the grid-based e-commerce domain and list-based web search scenarios (see examples in Section 4).

3 PROPOSED FRAMEWORK

In this section, we formulate a pairwise loss which is unbiased under only two assumptions: the examination hypothesis [8, Section 3.3] and the positivity of observation propensities. By the latter, we mean that each of the positions in the layout is observable by users (i.e. there are no positions that can never be examined) and that there are no pairs of positions that can never be examined jointly.

In learning-to-rank, the data is comprised of collections of items. Such collections can be, for example, lists of links returned by a web-search engine, grids of products in an online shop, or the set of elements of a recommendation carousel. We will adopt a common notation and use the letter q to denote a single collection of items from the data. This notation is likely to stem from the fact that each of the collections often has an associated query-string but this is not always the case and plays no role in the context of this research. The set of all item collections in the data will be denoted by Q. Furthermore, let n_q stand for the number of items in collection $q \in Q$ and let $x_{q,i}$ denote the feature vector associated with the i-th item of collection $q \in Q$. The feature vector can include both the item attributes and the properties of the context (such as user and query-string features). Finally, let $r_{q,i}$ be the relevance of the item. The item relevance is generally unobserved. Instead, the data contains a target label $c_{q,i}$ which is based on the user actions and can be, for example, a click indicator. For the ease of exposition, we will assume that both the relevance and the target label are binary but the results can be easily generalised to the case where the relevance and/or the label take more than two values.

3.1 Assumptions

We make the following common assumption about the relationship between the relevance and the observed target label (see, [8]).

Assumption 1 (the examination hypothesis). *The target label equals the true relevance if the user has examined the item and is zero otherwise, i.e.*

$$c_{q,i} = e_{q,i} \cdot r_{q,i}, \quad \forall\, q \in Q \quad \forall\, i = 1, \ldots, n_q,$$

where $e_{q,i}$ is the examination indicator.

Note that we do not assume that the examination indicator and the relevance indicator are independent of each other (i.e. we do not require that the probability of a click be the product of the examination and the relevance probabilities). In personal search, for example, the production system would adapt the ranking with each request and place items that are relevant to the user who submitted the request to more visible positions. In this scenario, being relevant increases the chances of being seen and the independence between relevance and examination does not hold.

Next, define the (conditional) examination probabilities

$$p_{q,i} = P\{e_{q,i} = 1 \mid \mathcal{I}_q\}, \quad q \in Q, \quad i = 1, \ldots, n_q, \tag{1}$$

and the (conditional) joint examination probabilities

$$p_{q,i,j} = P\{e_{q,i} e_{q,j} = 1 \mid \mathcal{I}_q\}, \quad q \in Q, \quad i, j = 1, \ldots, n_q, \tag{2}$$

where $\mathcal{I}_q = \{r_{q,1}, \ldots, r_{q,n_q}, x_{q,1}, \ldots, x_{q,n_q}\}$ is a set of variables we will condition upon in our analysis. See Section 3.3 below for an example of how the examination probabilities (1) and (2) can be estimated in practice.

Assumption 2. *The examination probabilities defined in* (1) *and* (2) *are non-zero.*

3.2 Unbiased Pairwise Loss

Let f be a (ranking) model. A pairwise loss function takes the following form

$$L_c \quad = \quad \sum_q \sum_{i,j=1}^{n_q} \ell(f(x_{q,i}),\, c_{q,i},\, f(x_{q,j}),\, c_{q,j}). \tag{3}$$

In other words, it is the sum of terms that correspond to item pairs in the data. Those terms can be typically decomposed as follows

$$\begin{aligned}
&\ell(f(x_{q,i}),\, c_{q,i},\, f(x_{q,j}),\, c_{q,j}) = \\
&\qquad = \ell_{1,1}(f(x_{q,i}),\, f(x_{q,j}))c_{q,i}c_{q,j} \\
&\qquad + \ell_{1,0}(f(x_{q,i}),\, f(x_{q,j}))c_{q,i}(1 - c_{q,j}) \\
&\qquad + \ell_{0,1}(f(x_{q,i}),\, f(x_{q,j}))(1 - c_{q,i})c_{q,j} \\
&\qquad + \ell_{0,0}(f(x_{q,i}),\, f(x_{q,j}))(1 - c_{q,i})(1 - c_{q,j}).
\end{aligned} \tag{4}$$

Note that the functions $\ell_{1,1}$, $\ell_{1,0}$, $\ell_{0,1}$, and $\ell_{0,0}$ depend only on the scores that the model assigns to the two items in the pair. In the case of RankNet [5], for example, the four functions are[1]

$$\begin{aligned}
&\ell_{1,1}(f(x_{q,i}),\, f(x_{q,j})) = 0, \\
&\ell_{1,0}(f(x_{q,i}),\, f(x_{q,j})) = \log\left(1 + e^{-\sigma\cdot(f(x_{q,i}) - f(x_{q,j}))}\right), \\
&\ell_{0,1}(f(x_{q,i}),\, f(x_{q,j})) = \log\left(1 + e^{-\sigma\cdot(f(x_{q,j}) - f(x_{q,i}))}\right), \\
&\ell_{0,0}(f(x_{q,i}),\, f(x_{q,j})) = 0.
\end{aligned} \tag{5}$$

[1] In the derivation of the RankNet algorithm, the functions $\ell_{1,1}$ and $\ell_{0,0}$ are non-zero but the final algorithm ignores pairs with equal target labels which is equivalent to setting the two functions to zero.

To make the notation more concise, we introduce the following two vector functions:

$$\begin{aligned}
\mathbf{z}(f(x_{q,i}),\, f(x_{q,j})) &= \begin{pmatrix} \ell_{1,1}(f(x_{q,i}),\, f(x_{q,j})) \\ \ell_{1,0}(f(x_{q,i}),\, f(x_{q,j})) \\ \ell_{0,1}(f(x_{q,i}),\, f(x_{q,j})) \\ \ell_{0,0}(f(x_{q,i}),\, f(x_{q,j})) \end{pmatrix}, \\
\mathbf{s}(b_1,\, b_2) &= \begin{pmatrix} b_1 b_2 \\ b_1(1 - b_2) \\ (1 - b_1)b_2 \\ (1 - b_1)(1 - b_2) \end{pmatrix}, \quad b_1,\, b_2 \in \{0,\, 1\}.
\end{aligned} \tag{6}$$

The symbols b_1 and b_2 in the definition of $\mathbf{s}$ stand for two binary indicators. Below we will use either click or relevance indicators in their place. Note that the function $\mathbf{s}$ one-hot encodes the type of the item pair. To be more specific, let us consider a given item pair $(i,\, j)$ from collection $q \in Q$. If $\mathbf{s}$ is computed from click indicators $c_{q,i}$ and $c_{q,j}$, it equals $(1,\, 0,\, 0,\, 0)^\mathsf{T}$ when both of the items were clicked, $(0,\, 1,\, 0,\, 0)^\mathsf{T}$ when only the first item from the pair was clicked, $(0,\, 0,\, 1,\, 0)^\mathsf{T}$ when only the second item was clicked, and $(0,\, 0,\, 0,\, 1)^\mathsf{T}$ when none of the two items was clicked. Similarly, if $\mathbf{s}$ is computed from relevance indicators it gives analogous one-hot encodings but with respect to the relevance of items i and j.

Using the notation introduced in (6), we can rewrite (3) and (4) as follows:

$$L = \sum_q \sum_{i,j=1}^{n_q} \left(\mathbf{z}(f(x_{q,i}),\, f(x_{q,j}))\right)^\mathsf{T} \mathbf{s}(c_{q,i}, c_{q,j}). \tag{7}$$

The loss in (7) (or, equivalently, in (3)) is computed from the implicit feedback (click indicators) as opposed to the true relevance indicators since the latter are unavailable (unobserved). Because of the position bias, the optimisation of (7) generally leads to sub-optimal algorithms and may reinforce undesirable properties of existing production systems (such as unfairness). Our goal now will be to construct a matrix (denoted by $\mathbf{A}_{q,i,j}$ below) that can convert the term computed from the click indicators into the same term but computed from the relevance indicators under the expectation sign. In other words, this matrix will enable us to utilise the available click data and "infer" how items i and j compare in terms of their true relevance in expectation. Then by injecting this matrix into the loss (7), we will obtain a new and unbiased loss function. This new loss will be still computed from the clicks but its expected value will equal that of the loss computed from the true (unobserved) relevance values.

We begin by defining a matrix (function) $\mathbf{B}$ depending on two examination indicators,

$$\mathbf{B}(e_1,\, e_2) = \begin{pmatrix} e_1 e_2 & 0 & 0 & 0 \\ e_1(1 - e_2) & e_1 & 0 & 0 \\ (1 - e_1)e_2 & 0, & e_2 & 0 \\ (1 - e_1)(1 - e_2) & (1 - e_1) & (1 - e_2) & 1 \end{pmatrix}. \tag{8}$$

For any $q \in Q$ and $i,\, j = 1, \ldots, n_q$, the matrix $\mathbf{B}(e_{q,i},\, e_{q,j})$ relates the terms $\mathbf{s}(c_{q,i},\, c_{q,j})$ and $\mathbf{s}(r_{q,i},\, r_{q,j})$. Specifically, (under Assumption 1) it holds that

$$\mathbf{s}(c_{q,i},\, c_{q,j}) = \mathbf{B}(e_{q,i},\, e_{q,j}) \cdot \mathbf{s}(r_{q,i},\, r_{q,j}).$$

For example, if both of the two items are relevant ($r_{q,i} = 1, r_{q,j} = 1$) we have that $\mathbf{s}(r_{q,i},\, r_{q,j}) = \mathbf{s}(1,\, 1) = (1,\, 0,\, 0,\, 0)^\mathsf{T}$. At the same time,

if only the first of the two items gets examined ($e_{q,i}=1,\ e_{q,j}=0$), according to the examination hypothesis, we will only observe a click on the first item and $\mathbf{s}(c_{q,i},\ c_{q,j}) = \mathbf{s}(1,\ 0) = (0,\ 1,\ 0,\ 0)^\mathsf{T}$. It holds that $\mathbf{B}(1,\ 0)\cdot\mathbf{s}(1,\ 1) = \mathbf{s}(1,\ 0)$, i. e. the matrix $\mathbf{B}(1,\ 0)$ produces what is observed ($\mathbf{s}(1,\ 0)$) from the underlying relevance-based term $\mathbf{s}(1,\ 1)$.

In practice, of course, we want to achieve the opposite: reconstruct the unobserved true preference $\mathbf{s}(r_{q,i},\ r_{q,j})$ from the click feedback $\mathbf{s}(c_{q,i},\ c_{q,j})$, at least, in expectation. To that end, we take the inverse of the expectation of $\mathbf{B}$ and set

$$\mathbf{A}_{q,i,j} = \left(\mathrm{E}[\mathbf{B}(e_{q,i},\ e_{q,j}) \mid \mathcal{I}_q]\right)^{-1}, \quad q\in Q, \quad i,\ j = 1,\ \ldots,\ n_q.$$

As will be seen in the proof of Theorem 3.1 below, this matrix has the desired property, that is it turns $\mathbf{s}(c_{q,i},\ c_{q,j})$ into $\mathbf{s}(r_{q,i},\ r_{q,j})$ under the expectation sign.

Before formulating our unbiased loss and proving its unbiasedness, we note that although the definition of $\mathbf{A}_{q,i,j}$ above involves matrix inversion, this matrix can be computed directly and efficiently from the examination probabilities (1) and (2). Specifically,

$$\mathbf{A}_{q,i,j} = \begin{pmatrix} a_{q,i,j} & 0 & 0 & 0 \\ a_{q,i} - a_{q,i,j} & a_{q,i} & 0 & 0 \\ a_{q,j} - a_{q,i,j} & 0, & a_{q,j} & 0 \\ 1 - a_{q,i} - a_{q,j} + a_{q,i,j} & 1 - a_{q,i} & 1 - a_{q,j} & 1 \end{pmatrix}, \tag{9}$$

where $a_{q,i}$ and $a_{q,j}$ are the inverses of the individual examination probabilities (1),

$$a_{q,i} = 1/p_{q,i}, \quad a_{q,j} = 1/p_{q,j}, \quad i,\ j = 1,\ \ldots,\ n_q,$$

and $a_{q,i,j}$ is the inverse of the joint examination probability (2),

$$a_{q,i,j} = 1/p_{q,i,j}, \quad i,\ j = 1,\ \ldots,\ n_q.$$

We state that the following loss is unbiased,

$$L_u = \sum_q \sum_{i,\,j=1}^{n_q} \left(\mathbf{z}(f(x_{q,i}),\ f(x_{q,j}))\right)^\mathsf{T} \mathbf{A}_{q,i,j}\mathbf{s}(c_{q,i},c_{q,j}). \tag{10}$$

Theorem 3.1. *Under assumptions 1–2, the loss defined by* (9)–(10), *is unbiased, i.e.*

$$E[L_u] = E\left[\sum_q \sum_{i,\,j=1}^{n_q} \left(\boldsymbol{z}(f(x_{q,i}),\ f(x_{q,j}))\right)^T \mathbf{s}(r_{q,i},r_{q,j})\right]. \tag{11}$$

Proof. First note that Assumption 1 implies that for all $q\in Q$ and all $i,\ j = 1,\ \ldots,\ n_q$, the terms $\mathbf{s}(c_{q,i},\ c_{q,j})$ and $\mathbf{s}(r_{q,i},\ r_{q,j})$ are related as follows,

$$\mathbf{s}(c_{q,i},\ c_{q,j}) = \mathbf{B}(e_{q,i},\ e_{q,j})\cdot\mathbf{s}(r_{q,i},\ r_{q,j}) \tag{12}$$

with the matrix function $\mathbf{B}$ defined in (8).

Next, for any q and any $i,\ j = 1,\ \ldots,\ n_q$ we have that

$$\mathrm{E}[\mathbf{B}(e_{q,i},\ e_{q,j}) \mid \mathcal{I}_q] = \\ = \begin{pmatrix} p_{q,i,j} & 0 & 0 & 0 \\ p_{q,i} - p_{q,i,j} & p_{q,i} & 0 & 0 \\ p_{q,j} - p_{q,i,j} & 0, & p_{q,j} & 0 \\ 1 - p_{q,i} - p_{q,j} + p_{q,i,j} & 1 - p_{q,i} & 1 - p_{q,j} & 1. \end{pmatrix} \tag{13}$$

Under Assumption 2, the matrix $\mathbf{A}_{q,i,j}$ given by (9) is well-defined. Combining its definition with (13), we compute that

$$\mathbf{A}_{q,i,j}\,\mathrm{E}[\mathbf{B}(e_{q,i},\ e_{q,j}) \mid \mathcal{I}_q] = \mathbf{I}_4 \quad \forall\, q\in Q \quad \forall\, i,\ j = 1,\ \ldots,\ n_q, \tag{14}$$

where $\mathbf{I}_4$ is the identity matrix of size 4. Therefore, for all $q\in Q$ and all $i,\ j = 1,\ \ldots,\ n_q$ we have that

$$\begin{aligned} & \mathrm{E}\left[\left(\mathbf{z}(f(x_{q,i}),\ f(x_{q,j}))\right)^\mathsf{T} \mathbf{A}_{q,i,j}\mathbf{s}(c_{q,i},c_{q,j})\right] = \\ = \quad & \mathrm{E}\left[\mathrm{E}\left[\left(\mathbf{z}(f(x_{q,i}),\ f(x_{q,j}))\right)^\mathsf{T} \mathbf{A}_{q,i,j}\mathbf{s}(c_{q,i},c_{q,j}) \mid \mathcal{I}_q\right]\right] \\ = \quad & \mathrm{E}\left[\left(\mathbf{z}(f(x_{q,i}),\ f(x_{q,j}))\right)^\mathsf{T} \mathbf{A}_{q,i,j}\,\mathrm{E}\left[\mathbf{s}(c_{q,i},c_{q,j}) \mid \mathcal{I}_q\right]\right] \\ = \quad & \mathrm{E}\Big[\left(\mathbf{z}(f(x_{q,i}),\ f(x_{q,j}))\right)^\mathsf{T} \\ & \times \mathbf{A}_{q,i,j}\,\mathrm{E}\left[\mathbf{B}(e_{q,i},\ e_{q,j}) \mid \mathcal{I}_q\right]\mathbf{s}(r_{q,i},\ r_{q,j})\Big] \\ \overset{eq.(14)}{=} \quad & \mathrm{E}\left[\left(\mathbf{z}(f(x_{q,i}),\ f(x_{q,j}))\right)^\mathsf{T} \mathbf{s}(r_{q,i},r_{q,j})\right], \end{aligned}$$

which, together with the definition of L_u (see formula (10)), implies the unbiasedness property (11). □

The theorem above says that in expectation the loss function that we introduced by (9)–(10) equals the loss computed from the true (unobserved) relevance indicators.

Interestingly, the matrix $\mathrm{E}[\mathbf{B}(e_{q,i},\ e_{q,j}) \mid \mathcal{I}_q]$ in (13) admits the following interpretation. If we categorise the pair of items $(i,\ j)$ as belonging to one of the following four types: type 1 ($r_{q,i}=1,\ r_{q,j}=1$), type 2 ($r_{q,i}=1,\ r_{q,j}=0$), type 3 ($r_{q,i}=0,\ r_{q,j}=1$), or type 4 ($r_{q,i}=0,\ r_{q,j}=0$) then the element of the matrix $\mathrm{E}[\mathbf{B}(e_{q,i},\ e_{q,j}) \mid \mathcal{I}_q]$ in row t_1 and column t_2 is the probability of a pair of type t_2 to appear as a pair of type t_1 in the click feedback. It means that this matrix consists of the probabilities of pair type distortion due to the presence of position bias. The matrix $\mathbf{A}_{q,i,j}$ used in the definition of the unbiased loss (10) is the inverse of $\mathrm{E}[\mathbf{B}(e_{q,i},\ e_{q,j}) \mid \mathcal{I}_q]$ but as we explained before, it can be computed directly (without matrix inversion).

There is another observation we would like to make. The terms $\ell_{1,1}(f(x_{q,i}),\ f(x_{q,j}))$ and $\ell_{0,0}(f(x_{q,i}),\ f(x_{q,j}))$ in (4) are usually set to zero (as in (5), for example). In the case of the conventional pairwise loss (3)–(4) (or, equivalently, (7)) it means that the loss is computed only from those pairs of items in which exactly one of the items was clicked. The complexity of computing the contribution of collection q to the loss is then $O(C_q\cdot N_q)$ where C_q and N_q are the numbers of clicked and non-clicked items of collection q, respectively. In contrast, in the case of the unbiased loss (10), not all of the remaining pairs will be eliminated because of the presence of the matrix $A_{q,i,j}$ in the formula. Therefore, it is legitimate to question whether the computation of the unbiased loss has a higher complexity. Luckily, this is not the case as we will explain now. Note that all the elements of the last column of $A_{q,i,j}$ except the bottom-most element are zero. It implies that as long as $\ell_{0,0}(f(x_{q,i}),\ f(x_{q,j})) = 0$, the pairs of items in which none of the items was clicked are still eliminated and the contribution of collection q to the unbiased loss can be computed in $O(C_q^2 + C_q\cdot N_q)$ operations, which is $O(C_q\cdot N_q)$ in the typical case when the number of clicked items C_q is lower than the number of non-clicked items N_q. Thus, the complexity of computing the unbiased loss (10) is

the same as the complexity of computing its conventional (biased) counterpart.

3.3 Estimation of Examination Probabilities

We consider the estimation of examination probabilities (1) and (2) as a separate big topic that is out of scope of this paper. However, in this section we briefly discuss how those probabilities can be estimated in practice.

When it comes to estimating the individual examination probabilities (1), multiple methods have been suggested in the literature, e.g. [2, 9, 13, 25, 26]. To the best of our knowledge, the estimation of joint examination probabilities (2) has not been considered but as will be seen from examples in Section 4 below, in many important cases the joint examination probability $p_{q,i,j}$ can be either computed from the individual examination probabilities $p_{q,i}$ and $p_{q,j}$ or expressed in terms of the same parameters. In such cases, the estimation of the joint examination probabilities does not pose an additional problem. That being said, estimating the joint examination probabilities directly is also possible. We will demonstrate that by describing a procedure for their estimation which is analogous to [13].

The method for the estimation of individual examination probabilities in [13] assumes that the examination probability depends only on the position where the respective item was placed, that is

$$p_{q,i} = \theta(rank_{q,i}) \quad \forall\, q \quad \forall\, i = 1, \ldots, n_q, \tag{15}$$

where θ is some function that maps ranks to the associated examination probabilities (observation propensities). Suppose that an analogous property holds for the joint examination probabilities, i. e.

$$p_{q,i,j} = \psi(rank_{q,i}, rank_{q,j}) \quad \forall\, q \quad \forall\, i, j = 1, \ldots, n_q,$$

with some function ψ depending on a pair of ranks (positions). To estimate the value of ψ for ranks k_1 and k_2 ($k_1 < k_2$) we can do the following. Whenever our existing ranker or recommender receives a user request (query) we randomly decide whether to swap the pair of items in positions k_1 and k_2 with the pair of items in positions 1 and 2. Let $i(k_1)$ and $i(k_2)$ be the items that the ranker assigns to positions k_1 and k_2, respectively. Then if we do not do the swap

$$P_{no-swap}\{c_{i(k_1)}c_{i(k_2)} = 1\} = \psi(k_1, k_2) P_{no-swap}\{r_{i(k_1)}r_{i(k_2)} = 1\} \tag{16}$$

and if we do the swap

$$P_{swap}\{c_{i(k_1)}c_{i(k_2)} = 1\} = \psi(1, 2) P_{swap}\{r_{i(k_1)}r_{i(k_2)} = 1\}. \tag{17}$$

At the same time, since swapping has no effect on the intrinsic relevance of the swapped items we have that

$$P_{no-swap}\{r_{i(k_1)}r_{i(k_2)} = 1\} = P_{swap}\{r_{i(k_1)}r_{i(k_2)} = 1\}. \tag{18}$$

From (16)–(18) we conclude that

$$\frac{P_{no-swap}\{c_{i(k_1)}c_{i(k_2)} = 1\}}{P_{swap}\{c_{i(k_1)}c_{i(k_2)} = 1\}} = \frac{\psi(k_1, k_2)}{\psi(1, 2)}. \tag{19}$$

Finally, if it is fair to assume that the first position is always examined the probability of examining both positions 1 and 2 equals the probability of examining position 2, that is

$$\psi(1, 2) = \theta(2). \tag{20}$$

Hence, having a consistent estimate of the individual examination probability at position 2, $\hat{\theta}(2)$, we can construct an estimate of the joint examination probability at positions k_1 and k_2 as follows,

$$\hat{\phi}(k_1, k_2) = \hat{\theta}(2) \cdot \frac{\sum_{q \in Q_{no-swap}} c_{i(k_1)}c_{i(k_2)} / |Q_{no-swap}|}{\sum_{q \in Q_{swap}} c_{i(k_1)}c_{i(k_2)} / |Q_{swap}|}$$

which is consistent due to (19), (20), and the consistency of $\hat{\theta}(2)$.

Note that when the joint examination probabilities are estimated directly the number of parameters is quadratic in the number of positions but this can be mitigated by doing the estimation only for some of the position pairs and extrapolating on the rest.

4 EXAMPLES

In this section, we will demonstrate how the proposed general framework can be used to derive unbiased pairwise loss functions in practice. We show by example that the framework allows us to focus on computing the examination probabilities and that once we compute them, an unbiased learning-to-rank method for the corresponding setting gets produced "automatically".

At this point, we will make the common simplifying assumption (15), i. e. suppose that the individual examination probability depends only on the position where the item is displayed. Contrary to the discussion at the end of Section 3, we will not need to maintain a similar assumption for the joint examination probabilities because depending on the considered model of user browsing behaviour, we will be able either to compute the joint examination probabilities from the individual ones or to express them through the same parameters. We consider three concrete examples of user browsing behaviour below.

Independent Examination. If the examination of different positions in the layout happens independently, it holds that

$$p_{q,i,j} = \theta(rank_{q,i})\theta(rank_{q,j}). \tag{21}$$

It can be easily checked that under this assumption the unbiased pairwise loss (10) recovers the one proposed in [20, Section 3.1] (and hence, the framework from [20] can be considered a special case of ours).

Continuous Examination. In this browsing model, users observe items continuously from positions with smaller ranks to positions with higher ranks without skipping. Eye-tracking experiments in [12] give some evidence suggesting that this may generally hold in web-search. In the case of such no-skipping behaviour,

$$p_{q,i,j} = \min\{\theta(rank_{q,i}), \theta(rank_{q,j})\}.$$

An application of our framework (9)–(10) gives an unbiased pairwise loss for this setting which, to the best of our knowledge, has not been proposed in the literature before. This loss can be optimised (e.g. by means of gradient descent) to obtain unbiased learning-to-rank models for the continuous examination setting.

Row skipping. Xie et al [27] analyse the behaviour of users presented with a grid layout. One of the browsing models they introduce is called "row skipping". As the name suggests, it captures the tendency of users to skip over rows when going through the grid layout. Specifically, it is assumed that at the start or after each row the user skips the next row with probability γ, otherwise they browse items in the row. After examining an item at rank v the user may stop browsing with probability $(1 - C_v)$. In this model,

the function θ that maps ranks to their associated examination probabilities takes the following form,

$$\theta(u) = \prod_{m=1}^{row(u)-1} \left((1-\gamma) \prod_{v'=S(m)+1}^{S(m)+N(m)} C_{v'} + \gamma \right) \times (1-\gamma) \prod_{v=S(row(u))+1}^{u-1} C_v, \quad (22)$$

where $row(u)$ is the row containing the rank u, $N(m)$ is the number of items in row m, and $S(m)$ is the total number of items before row m. In the above formula, the product before the $\times$ sign is the probability that the user does not quit before reaching the row containing the rank u. Then $(1-\gamma)$ accounts for the chance of skipping the respective row and the last product is the probability that the user does not stop before reaching the rank u when examining items in the row. The joint examination probability $p_{q,i,j}$ can be expressed in a similar fashion. Specifically, denoting $\min\{rank_{q,i}, rank_{q,j}\}$ by $h_{q,i,j}$ and $\max\{rank_{q,i}, rank_{q,j}\}$ by $w_{q,i,j}$, we can write

$$p_{q,i,j} = \theta(h_{q,i,j}) \prod_{v=h_{q,i,j}}^{S(row(h_{q,i,j}))+N(row(h_{q,i,j}))} C_v \times \prod_{m=row(h_{q,i,j})+1}^{row(w_{q,i,j})-1} \left((1-\gamma) \prod_{v'=S(m)+1}^{S(m)+N(m)} C_{v'} + \gamma \right) \times (1-\gamma) \prod_{v=S(row(w_{q,i,j}))+1}^{w_{q,i,j}-1} C_v, \quad (23)$$

if items i and j were displayed in different rows and

$$p_{q,i,j} = \theta(h_{q,i,j}) \prod_{v=h_{q,i,j}}^{w_{q,i,j}-1} C_v \quad (24)$$

otherwise. The function θ in (23) and (24) is the one from (22).

Assuming that the parameters γ and C_v are known or have been estimated, one can plug them in (22)–(24) and construct an unbiased pairwise loss according to (9) and (10). This loss can be then optimised to obtain an unbiased ranker for the case of row-skipping behaviour. This is another example of how the framework from Section 3 allows us to focus on computing the examination probabilities and an unbiased learning-to-rank method for the corresponding setting gets produced "automatically" - just by plugging them into (9) and (10).

5 ROBUST UNBIASED LAMBDAMART

The unbiased loss proposed in Section 3 can be combined with the so called "lambda-trick" [5] similarly to how it was done in [11]. The lambda-trick consists of re-weighting the item pairs in the gradient of the loss function (such as (10)) so that the optimisation process performs better at maximising an information retrieval (IR) metric (such as NDCG). The weights are set to the absolute difference $|\Delta Z_{i,j}|$ in the respective IR metric when the two items of the pair are swapped in the ranking induced by the current values of model parameters. The re-weighted gradient of the underlying pairwise loss function is called *lambda-gradient*.

As will be seen from the derivations below, the application of the lambda-trick to the unbiased pairwise loss (10) from Section 3 generates an interesting insight. Specifically, the resulting algorithm turns out to be the same regardless of the values of the joint examination probabilities $p_{q,i,j}$. It means that the algorithm arising this way is valid as long as the examination hypothesis holds and the observation propensities are positive. In particular, this is true regardless of the specific user behaviour patterns (such as the ones discussed in the previous section). This is especially interesting given that the version of Unbiased LambdaMART stemming from (10) is simpler than the original Unbiased LambdaMART in the sense that it does not have some of its parameters.

Similarly to [5] and [11], we will set the loss values as in formula (5). To write down an expression for the lambda-gradient based on our unbiased loss (10), we first compute the gradient of (10) (with $\mathbf{z}$ defined by (5)). We denote

$$\mu_{q,i,j} = \frac{1}{1+e^{f(x_{q,i})-f(x_{q,j})}}.$$

With this notation, the gradient equals

$$(L_u)'_{q,i} = \sum_{j=1}^{n_q} \Big[\big(-\sigma\mu_{q,i,j}(a_{q,i}-a_{q,i,j}) + \sigma\mu_{q,j,i}(a_{q,j}-a_{q,i,j})\big) c_{q,i} c_{q,j} - \sigma\mu_{q,i,j} a_{q,i} c_{q,i}(1-c_{q,j}) + \sigma\mu_{q,j,i} a_{q,j}(1-c_{q,i}) c_{q,j} \Big]. \quad (25)$$

It can be seen that the gradient depends not only on pairs with different target labels but also on pairs of items that were both clicked. However, when we proceed with the lambda-trick the contribution of such pairs gets eliminated since their respective $|\Delta Z_{i,j}|$ is zero. This gives the following formula for the lambda-gradient.

$$\lambda_{q,i} = \sum_{j=1}^{n_q} \left(\frac{\lambda_{q,i,j}}{p_{q,i}} c_{q,i}(1-c_{q,j}) - \frac{\lambda_{q,j,i}}{p_{q,j}} (1-c_{q,i}) c_{q,j} \right), \quad (26)$$

where $\lambda_{q,i,j} = -\sigma\mu_{q,i,j}$ and $\lambda_{q,j,i} = -\sigma\mu_{q,j,i}$. This version of Unbiased LabmdaMART is compared with the original LambdaMART and Unbiased LambdaMART in Table 1. It can be seen that the difference between formula (26) and the original Unbiased LambdaMART is the absence of t^- parameters[2].

The lambda-gradient formula (26) can be alternatively viewed as if it is obtained by omitting the terms corresponding to pairs of clicked items in the gradient (25) and then applying the lambda-trick heuristic in the "standard" way. From this perspective, one can expect (26) to perform better when the contribution of such pairs to the loss (10) is small - for example, when the clicks are sparse. See also the discussion in Section 6.4 below.

In the next section we compare the performance of the simplified Unbiased LambdaMART given by (26) with the original Unbiased LambdaMART in a semi-synthetic experiment.

[2] In Unbiased LambdaMART [11], the parameter t^-_k is defined as the ratio between the probability of click absence and the probability of irrelevance (at rank k).

Table 1: Pair Contribution to the Lambda-Gradient (index q is omitted)

Method	Pair Type	
	$c_i > c_j$	$c_i < c_j$
LambdaMART [5]	$\lambda_{i,j}$	$-\lambda_{j,i}$
Unbiased LambdaMART [11]	$\frac{\lambda_{i,j}}{\theta(rank_i)t^-_{rank_j}}$	$\frac{-\lambda_{j,i}}{\theta(rank_j)t^-_{rank_i}}$
Formula (26)	$\frac{\lambda_{i,j}}{\theta(rank_i)}$	$\frac{-\lambda_{j,i}}{\theta(rank_j)}$

6 EXPERIMENTAL SETUP AND RESULTS

We performed our simulation experiments using the Yahoo! C14 Learning to Rank Challenge[3] dataset. This dataset consists of 29921 queries divided into three parts (train, validation, and test). Each query has an associated list of documents (of varying length). Every document is described with a feature set containing 700 features and supplied with a relevance label set by human editors [6]. The relevance labels take values from 0 (*irrelevant*) to 4 (*highly relevant*). We used the train part for simulations and the test part for evaluation.

Our simulation setup is similar to that from [3] and [11].

6.1 Click Data Generation

The train part of the Yahoo! C14 dataset has 19944 queries. We used the same initial rankings of the associated document lists as in [11][4]. Given those initial rankings, the lists were truncated at a fixed position. We conducted several experiments with the truncation position set to 10, 20, and 30. A small number of queries that did not have any relevant documents were discarded[5].

For each document, we generated[6] a relevance indicator and an examination indicator. The click indicator was computed as the product of the two.

Relevance indicators were generated as independent binary (Bernoulli) random variables with the probability of success set to

$$P\{r_{q,i} = 1\} = (2^{y_{q,i}} - 1)/15,$$

where $y_{q,i}$ is the manual relevance label provided in the dataset.

When generating the examination indicators we considered two user browsing models mentioned in Section 4: continuous browsing and independent examination. In both cases, the probability of examining a given position was set as in [13], that is

$$p_{q,i} = \frac{1}{rank_{q,i}}, \quad i = 1, \ldots, n_q. \tag{27}$$

In the case of independent examination, the examination indicators were generated as independent Bernoulli random variables with the probability of success defined above.

In contrast, in the case of continuous browsing the examination indicators (corresponding to the same query) were dependent and had the property

$$rank_{q,i} \le rank_{q,j} \Rightarrow e_{q,i} \ge e_{q,j} \quad \forall\, q \in Q \quad \forall\, i,\ j = 1, \ldots, n_q.$$

To achieve that we first randomly drew the last examined position $d_q \in \{1, \ldots, n_{max}\}$ where n_{max} is the maximum document list length. The distribution of the last examined position d_q was set to

$$P\{d_q = k\} = \begin{cases} \frac{1}{k} - \frac{1}{k+1}, & k = 1, \ldots, n_{max} - 1, \\ \frac{1}{n_{max}}, & k = n_{max}. \end{cases}$$

to match (27). After drawing d_q for each query, we set the examination indicators as follows

$$e_{q,i} = \begin{cases} 1, & rank_{q,i} \le d_q, \\ 0, & rank_{q,i} > d_q \end{cases}, \quad i = 1, \ldots, n_q.$$

The generation process was repeated for each query 16 times, which formed our training data consisting of 306272 document lists with associated click indicators.

6.2 Methods under Comparison

In our experiments, we compared the following learning-to-rank algorithms.

LambdaMART trained on the click data. This is LambdaMART [5] fit to the click data. Similarly to [11], we consider the performance of this baseline as a lower bound. The reason is that it is trained on (biased) implicit feedback data using a machine learning algorithm that has no debiasing mechanism.

LambdaMART trained on the labeled data. This model is LambdaMART [5] trained on the "golden" (ground-truth) relevance labels provided in the dataset. The performance of this baseline is an upper bound since it is trained on manually labelled data, free from the position bias.

Unbiased LambdaMART. This is the original Unbiased LambdaMART from [11]. The examination propensities (t^+ parameters in the terminology of [11]) were fixed at their true values (27). The estimation of t^- parameters was carried out in the usual way, i.e. as part of the Unbiased LambdaMART training process. The algorithm applies additive regularisation to the t^+ and t^- parameters (which in our case affected only the t^- parameters since t^+ parameters were fixed at their true values). The regularisation is controlled by a hyper-parameter (denoted by p in [11]). We considered three values of it corresponding to no regularisation, L_1-regularisation, and L_2-regularisation, respectively.

Robust Unbiased LambdaMART. It is the simplified version of Unbiased LambdaMART that we constructed in Section 5. Note that we used the true values of the observation propensities both in the simplified and in the original versions of Unbiased LambdaMART to have a fair comparison.

We did not include Regression-EM since Unbiased LambdaMART showed a better performance compared to it in the experiments from [11].

All of the models were trained using LightGBM [14]. The hyper-parameters were set to the same values as in [11]. In particular, the number of trees was 300, the learning rate equalled 0.05, the maximum number of leaves in a tree was 31, the feature fraction was 0.9, and the bagging fraction was set to 0.9.

[3] https://webscope.sandbox.yahoo.com/catalog.php

[4] See https://github.com/acbull/Unbiased_LambdaMart.

[5] In our experiments we do not generate noisy clicks. Consequently, the click data generated for such queries would inevitably contain no clicks and would not be utilised by any of the methods we compare. The number of discarded queries varied between 784, 802, or 908 depending on the truncation position.

[6] We used `numpy.random.default_rng` generator with the seed set to 2022. Our source code for running the experiments is available at https://github.com/zalandoresearch/pairwise-debiasing.

6.3 Evaluation Protocol

We evaluated the algorithms on the test part of the dataset using the "golden" relevance judgements as target labels. The test part contains 6983 queries of which 248 do not have any associated documents with positive relevance labels and were excluded[7]. We did not truncate the document lists at the evaluation stage.

We evaluated the algorithms with the NDCG metric because it is the information retrieval metric we targeted when applying the lambda-trick. Specifically, we used NDCG at cutoff positions 1, 3, 5, and 10. We also report MAP for completeness.

6.4 Experimental Results

The experimental results are presented in Tables 2 and 3. In each of the two tables, we report the absolute performance of unregularised Unbiased LambdaMART and the relative performance of all of the other methods (in percentages). The relative changes typeset in bold are significant at a 5% significance level as assessed by a paired two-sided t-test with Bonferroni correction[8]. We additionally checked if the performance of Robust Unbiased LambdaMART was statistically different from that of the original Unbiased LambdaMART with $L2$-regularisation. Cases where the difference is statistically significant (according to a paired two-sided t-test with a 5% significance level) are highlighted with a frame.

Table 2 corresponds to the experiment with continuous examination. For that type of user behaviour, the robust version of Unbiased LambdaMART outperforms the unregularised Unbiased LambdaMART for all values of the truncation position with respect to NDCG. The uplift is especially pronounced for larger values of the truncation position (i. e. when the maximum training list length is bigger) and for smaller values of the cut-off position in the NDCG metric.

When the regularisation parameter in Unbiased LambdaMART gets increased the performance of it catches up with that of Robust Unbiased LambdaMART. This is expected because as the reguarisation parameter grows bigger the t^- parameters in the original Unbiased LambdaMART are regularised away and the method "converges" to Robust Unbiased LambdaMART. However, it can be seen from Table 2 that the sufficient level of regularisation needed for the original Unbiased LambdaMART to perform on par with the robust version depends on the maximum length of a training list. Note that tuning the regularisation parameter on the click data using conventional validation approaches can be misleading because the click data is affected by the position bias. Instead, one would need to use an unbiased version of the validation loss (such as (10)), similarly to [21, Section 6.1.4]. Although the latter is a valid and feasible approach, we still consider the absence of any debiasing-related hyper-parameters in Robust Unbiased LambdaMART an advantage since it is making the method simpler.

The evaluation results for independent examination can be found in Table 3. In this setting, Robust Unbiased LambdaMART performs better than the unregularised Unbiased LambdaMART for larger values of the truncation position (20 and 30). However, its performance is slightly worse than that of Unbaised LambdaMART when the maximum training list length equals 10. Our explanation is that when both the loss function from [11] and the loss given by (10) have similar unbiasedness properties (such as in the case of independent browsing) the application of the lambda-trick on top of the former can give a better result. This may be further explained by the fact that the loss from [11] does not contain terms corresponding to pairs of clicked items which contribution gets eliminated by the lambda-trick. This prompts to seek an adaptation of the lambda-trick that would propagate the contribution of such pairs into the lambda-gradient. We consider this a topic for future research. Note that even in the case of independent examination the comparison outcome between the robust and the original Unbiased LambdaMART still depends on the value of the regularisation parameter for larger values of the truncation position and Unbiased LambdaMART needs to be regularised appropriately to outperform the robust version (26).

[7]The information retrieval metrics we used for evaluation are not defined for such queries.

[8]The Bonferroni correction was applied globally, i. e. across all of the reported comparisons.

7 CONCLUSION

We advanced the theory of pairwise unbiased learning-to-rank by developing a general debiasing approach based on a minimalistic set of assumptions. We showed how our general framework can be used to construct unbiased pairwise loss functions and, consequently, unbiased learning-to-rank algorithms for different types of user behaviour. We further implemented our approach as a simplified but robust version of the Unbiased LambdaMART. Our experimental results show that this version performs better than the original algorithm when the examination of different items in the layout occurs in a dependent fashion.

One of the insights following from the theory developed in this paper is that in the presence of position bias, a learning-to-rank procedure can benefit from accounting not only for pairs with different target labels but also for pairs with the same (non-zero) target label. In the context of LambdaMART, this motivates future research aiming at adapting the lambda-trick so that it does not eliminate the contribution of such pairs to the trained model.

Another interesting direction for future research is to combine our approach with variance reduction techniques [22, 24].

ACKNOWLEDGMENTS

The authors would like to thank Dr. Christian Bracher from Zalando Research and Dr. Zeno Gantner for reading the draft of the paper and giving helpful feedback.

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 and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 5–14. https://doi.org/10.1145/3331184.3331202

[2] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating Position Bias without Intrusive Interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (Melbourne VIC, Australia) *(WSDM '19)*. Association for Computing Machinery, New York, NY, USA, 474–482. https://doi.org/10.1145/3289600.3291017

[3] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W. Bruce Croft. 2018. Unbiased Learning to Rank with Unbiased Propensity Estimation. In *The 41st International ACM SIGIR Conference on Research; Development in Information Retrieval*

Table 2: Evaluation Results for the Case of Continuous Examination (see Section 6.4 for details)

Trunc. pos.	Method	NDCG@1	NDCG@3	NDCG@5	NDCG@10	MAP
10	Unbiased LambdaMART (no regularisation)	0.684	0.680	0.702	0.752	0.880
	LambdaMART (click data)	**-6.71%**	**-5.11%**	**-4.40%**	**-3.28%**	**-0.72%**
	Unbiased LambdaMART (L_1-regularisation)	**+2.12%**	**+2.15%**	**+1.51%**	**+1.21%**	+0.16%
	Unbiased LambdaMART (L_2-regularisation)	+1.64%	**+1.86%**	**+1.54%**	**+1.12%**	-0.03%
	Robust Unbiased LambdaMART	+1.01%	**+1.75%**	**+1.38%**	**+0.96%**	-0.20%
	LambdaMART (labelled data)	**+3.93%**	**+4.41%**	**+3.65%**	**+2.82%**	+0.33%
20	Unbiased LambdaMART (no regularisation)	0.642	0.651	0.678	0.732	0.875
	LambdaMART (click data)	**-3.12%**	**-2.79%**	**-2.27%**	**-1.58%**	**-0.29%**
	Unbiased LambdaMART (L_1-regularisation)	**+6.45%**	**+4.74%**	**+3.87%**	**+2.88%**	**+0.65%**
	Unbiased LambdaMART (L_2-regularisation)	**+7.63%**	**+5.78%**	**+4.66%**	**+3.41%**	**+0.72%**
	Robust Unbiased LambdaMART	**+8.69%**	**+6.88%**	**+5.52%**	**+4.09%**	**+0.74%**
	LambdaMART (labelled data)	**+11.24%**	**+9.69%**	**+8.20%**	**+6.24%**	**+1.13%**
30	Unbiased LambdaMART (no regularisation)	0.613	0.629	0.660	0.718	0.870
	LambdaMART (click data)	+0.27%	-0.16%	-0.21%	-0.06%	+0.15%
	Unbiased LambdaMART (L_1-regularisation)	**+9.13%**	**+6.40%**	**+5.00%**	**+3.71%**	**+0.97%**
	Unbiased LambdaMART (L_2-regularisation)	**+11.46%**	**+8.18%**	**+6.50%**	**+4.67%**	**+1.23%**
	Robust Unbiased LambdaMART	**+13.02%**	**+10.11%**	**+7.94%**	**+5.98%**	**+1.26%**
	LambdaMART (labelled data)	**+17.04%**	**+14.03%**	**+11.61%**	**+8.66%**	**+1.84%**

Table 3: Evaluation Results for the Case of Independent Examination (see Section 6.4 for details)

Trunc. pos.	Method	NDCG@1	NDCG@3	NDCG@5	NDCG@10	MAP
10	Unbiased LambdaMART (no regularisation)	0.696	0.693	0.714	0.762	0.882
	LambdaMART (click data)	**-7.24%**	**-5.88%**	**-5.10%**	**-3.99%**	**-0.85%**
	Unbiased LambdaMART (L_1-regularisation)	+0.03%	+0.61%	+0.34%	+0.16%	**-0.28%**
	Unbiased LambdaMART (L_2-regularisation)	-0.17%	+0.44%	+0.08%	-0.08%	**-0.40%**
	Robust Unbiased LambdaMART	-1.37%	-0.57%	-0.59%	**-0.57%**	**-0.76%**
	LambdaMART (labelled data)	**+2.21%**	**+2.45%**	**+1.99%**	**+1.46%**	+0.12%
20	Unbiased LambdaMART (no regularisation)	0.654	0.663	0.688	0.740	0.877
	LambdaMART (click data)	**-2.97%**	**-3.01%**	**-2.44%**	**-1.75%**	**-0.30%**
	Unbiased LambdaMART (L_1-regularisation)	**+6.45%**	**+4.80%**	**+4.12%**	**+2.98%**	**+0.48%**
	Unbiased LambdaMART (L_2-regularisation)	**+6.65%**	**+5.17%**	**+4.40%**	**+3.21%**	**+0.43%**
	Robust Unbiased LambdaMART	**+5.64%**	**+4.93%**	**+3.96%**	**+2.90%**	+0.01%
	LambdaMART (labelled data)	**+9.09%**	**+7.57%**	**+6.54%**	**+4.98%**	**+0.89%**
30	Unbiased LambdaMART (no regularisation)	0.624	0.642	0.670	0.726	0.872
	LambdaMART (click data)	+1.05%	-0.30%	-0.30%	-0.14%	+0.02%
	Unbiased LambdaMART (L_1-regularisation)	**+10.60%**	**+7.43%**	**+6.18%**	**+4.65%**	**+1.03%**
	Unbiased LambdaMART (L_2-regularisation)	**+12.11%**	**+8.82%**	**+7.41%**	**+5.48%**	**+1.05%**
	Robust Unbiased LambdaMART	**+11.71%**	**+8.77%**	**+7.10%**	**+5.19%**	**+0.66%**
	LambdaMART (labelled data)	**+14.97%**	**+11.75%**	**+9.93%**	**+7.42%**	**+1.56%**

(Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 385–394. https://doi.org/10.1145/3209978.3209986

[4] Qingyao Ai, Tao Yang, Huazheng Wang, and Jiaxin Mao. 2021. Unbiased Learning to Rank: Online or Offline? *ACM Trans. Inf. Syst.* 39, 2, Article 21 (feb 2021), 29 pages. https://doi.org/10.1145/3439861

[5] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research. http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf

[6] Olivier Chapelle and Yi Chang. 2010. Yahoo! Learning to Rank Challenge Overview. In *Proceedings of the 2010 International Conference on Yahoo! Learning to Rank Challenge - Volume 14* (Haifa, Israel) *(YLRC'10)*. JMLR.org, 1–24.

[7] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and Debias in Recommender System: A Survey and Future Directions. *ArXiv* abs/2010.03240 (2020).

[8] Aleksandr Chuklin, Ilya Markov, and M. de Rijke. 2015. Click Models for Web Search. In *Click Models for Web Search*.

[9] Zhichong Fang, Aman Agarwal, and Thorsten Joachims. 2018. Intervention Harvesting for Context-Dependent Examination-Bias Estimation. *CoRR* abs/1811.01802 (2018). arXiv:1811.01802 http://arxiv.org/abs/1811.01802

[10] Ruocheng Guo, Xiaoting Zhao, Adam Henderson, Liangjie Hong, and Huan Liu. 2020. *Debiasing Grid-Based Product Search in E-Commerce*. Association for Computing Machinery, New York, NY, USA, 2852–2860. https://doi.org/10.1145/3394486.3403336

[11] 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*. 2830–2836.

[12] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately Interpreting Clickthrough Data as Implicit Feedback. *SIGIR Forum* 51, 1 (aug 2017), 4–11. https://doi.org/10.1145/3130332.3130334

[13] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 781–789. https://doi.org/10.1145/3018661.3018699

[14] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 3149–3157.

[15] Haruka Kiyohara and Yuta Saito. 2021. A Collection of Research and Review Papers on Offline Reinforcement Learning and Off-Policy Evaluation. https://github.com/hanjuku-kaso/awesome-offline-rl

[16] Haruka Kiyohara, Yuta Saito, Tatsuya Matsuhiro, Yusuke Narita, Nobuyuki Shimizu, and Yasuo Yamamoto. 2022. Doubly Robust Off-Policy Evaluation for Ranking Policies under the Cascade Behavior Model. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining* (Virtual Event, AZ, USA) *(WSDM '22)*. Association for Computing Machinery, New York, NY, USA, 487–497. https://doi.org/10.1145/3488560.3498380

[17] 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 and Data Mining* (London, United Kingdom) *(KDD '18)*. Association for Computing Machinery, New York, NY, USA, 1685–1694. https://doi.org/10.1145/3219819.3220028

[18] 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* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 1293–1302. https://doi.org/10.1145/3269206.3271686

[19] Harrie Oosterhuis and Maarten de Rijke. 2021. Unifying Online and Counterfactual Learning to Rank: A Novel Counterfactual Estimator That Effectively Utilizes Online Interventions. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (Virtual Event, Israel) *(WSDM '21)*. Association for Computing Machinery, New York, NY, USA, 463–471. https://doi.org/10.1145/3437963.3441794

[20] Yuta Saito. 2020. Unbiased Pairwise Learning from Biased Implicit Feedback. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval* (Virtual Event, Norway) *(ICTIR '20)*. Association for Computing Machinery, New York, NY, USA, 5–12. https://doi.org/10.1145/3409256.3409812

[21] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased Recommender Learning from Missing-Not-At-Random Implicit Feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 501–509.

[22] Adith Swaminathan and Thorsten Joachims. 2015. Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization. *Journal of Machine Learning Research* 16, 52 (2015), 1731–1755. http://jmlr.org/papers/v16/swaminathan15a.html

[23] Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual Risk Minimization: Learning from Logged Bandit Feedback. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37* (Lille, France) *(ICML'15)*. JMLR.org, 814–823.

[24] Adith Swaminathan and Thorsten Joachims. 2015. The Self-Normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2015/file/39027dfad5138c9ca0c474d71db915c3-Paper.pdf

[25] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Pisa, Italy) *(SIGIR '16)*. Association for Computing Machinery, New York, NY, USA, 115–124. https://doi.org/10.1145/2911451.2911537

[26] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 610–618. https://doi.org/10.1145/3159652.3159732

[27] Xiaohui Xie, Jiaxin Mao, Yiqun Liu, Maarten de Rijke, Yunqiu Shao, Zixin Ye, Min Zhang, and Shaoping Ma. 2019. Grid-Based Evaluation Metrics for Web Image Search. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 2103–2114. https://doi.org/10.1145/3308558.3313514