Addressing Trust Bias for Unbiased Learning-to-Rank

Aman Agarwal
Cornell University
Ithaca, NY
aa2398@cornell.edu

Xuanhui Wang
Google Inc.
Mountain View, CA
xuanhui@google.com

Cheng Li
Google Inc.
Mountain View, CA
chgli@google.com

Mike Bendersky
Google Inc.
Mountain View, CA
bemike@google.com

Marc Najork
Google Inc.
Mountain View, CA
najork@google.com

ABSTRACT

Existing unbiased learning-to-rank models use counterfactual inference, notably Inverse Propensity Scoring (IPS), to learn a ranking function from biased click data. They handle the click incompleteness bias, but usually assume that the clicks are noise-free, i.e., a clicked document is always assumed to be relevant. In this paper, we relax this unrealistic assumption and study click noise explicitly in the unbiased learning-to-rank setting. Specifically, we model the noise as the position-dependent trust bias and propose a noise-aware Position-Based Model, named TrustPBM, to better capture user click behavior. We propose an Expectation-Maximization algorithm to estimate both examination and trust bias from click data in TrustPBM. Furthermore, we show that it is difficult to use a pure IPS method to incorporate click noise and thus propose a novel method that combines a Bayes rule application with IPS for unbiased learning-to-rank. We evaluate our proposed methods on three personal search data sets and demonstrate that our proposed model can significantly outperform the existing unbiased learning-to-rank methods.

KEYWORDS

Unbiased learning-to-rank; trust bias; inverse propensity scoring; click noise

ACM Reference Format:
Aman Agarwal, Xuanhui Wang, Cheng Li, Mike Bendersky, and Marc Najork. 2019. Addressing Trust Bias for Unbiased Learning-to-Rank. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, San Francisco, CA, USA, 11 pages. https://doi.org/10.1145/3308558.3313707

1 INTRODUCTION

Implicit user feedback (such as clicks, dwelling time etc.) is routinely logged in many applications such as web search and personal email search. Using this data for learning-to-rank (LTR) is attractive not only because it is abundantly available, but also because it is the only viable large-scale labeled data for many applications like personal search [32, 33]. Moreover, unlike explicit relevance judgments obtained from experts or crowd-sourcing which can become obsolete quickly, implicit feedback reflects the time-varying preferences of the actual user population and is easy to maintain.

Despite these clear benefits, LTR from implicit feedback, in particular clicks on search results, is challenging due to inherent biases in user behavior. For example, due to position bias, a result displayed at a higher position is more likely to receive clicks than a result at a lower position. Thus directly using clicks as relevance judgments can lead to biased results because (1) clicks are incomplete as relevant documents may be missed non-uniformly and (2) clicks are noisy since a click does not necessarily imply the document is relevant.

Recently, unbiased LTR has been actively studied as a promising approach to learn from biased click logs. It is based on counterfactual inference [21] and has been shown to outperform the traditional click modeling approaches [3]. The commonly used technique in unbiased LTR is Inverse Propensity Scoring (IPS). Different from the traditional IPS approaches used in counterfactual inference where the propensity is known, a key question in unbiased LTR is how to estimate the propensity – the probability that the relevance of a result is observed. Examination-based click models, and the Position-Based Model (PBM) in particular, have been leveraged for this. A nice property of PBM is that the examination bias in it is the propensity needed. Different techniques have been proposed to estimate the examination bias using result randomization [21, 32], random pairs harvesting [2], a regression-based Expectation-Maximization (EM) technique [33], and a dual learning algorithm [3].

Existing PBM methods mainly focus on addressing the click incompleteness bias, by assuming that clicks are noise-free. Such an assumption is unrealistic. How to address click noise for unbiased LTR is still an open question. Joachims et al. [21] showed the robustness of the IPS-based approach when *position-independent constant* noise is present in click data. However, as the user study in [19] showed qualitatively, there is a *trust bias* after examining results, capturing the intuition that users may overestimate (or underestimate) the relevance of a result given its high (or low) position, due to their trust in the effectiveness of the search application to rank relevant documents higher. Thus a non-relevant result

This paper is published under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC BY-NC-ND 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution. In case of republication, reuse, etc., the following attribution should be used: "Published in WWW2019 Proceedings © 2019International World Wide Web Conference Committee, published under Creative Commons CC BY-NC-ND 4.0 License."
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC BY-NC-ND 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313707

may be clicked and a relevant result may not be clicked, and the probabilities are not constant but depend on the position of the result.

Trust bias can be thought of as another form of position bias, but to the best of our knowledge, it has not been quantitatively formulated in the existing literature. In this paper, we introduce a noise-aware PBM, named TrustPBM, to model trust bias in user clicks. The standard PBM model assumes that an examined and relevant document is always clicked. We extend it to capture the following uncertainty: after examination, a relevant document can be missed, and meanwhile a non-relevant document can be clicked. TrustPBM is a click model that gives a rigorous mathematical formulation of the trust bias. However, its bias parameters cannot be estimated based on intervention like the result randomization used for PBM. Even if we can estimate bias parameters in TrustPBM, it is still unclear how to use them for unbiased LTR.

In this paper, we first show that TrustPBM can be effectively estimated using an EM algorithm, inspired by the previous work [33]. Since TrustPBM is no longer a factorization model, a purely IPS method for unbiased LTR is not feasible. We thus derive a novel and mathematically principled Bayes-IPS correction that uses a Bayesian approach for trust bias and IPS for examination bias. Conceptually, the Bayes-IPS method addresses both click incompleteness bias (using IPS) and click noise (using Bayes rule) for unbiased LTR.

We empirically evaluate our method on click logs from search services for Gmail, Google Drive, and Gmail Expert users. Our TrustPBM estimates obtained via the EM algorithm reveal novel insights into the nature of relevance noise present in real settings. Most interestingly, we observe that most of the noise comes from highly ranked irrelevant documents being judged relevant, while relevant documents tend to be judged accurately even at lower positions. Moreover, our results demonstrate that the TrustPBM fits the click data better than the plain PBM. Furthermore, ranking models trained with the Bayes-IPS correction have superior live experiment metrics than those trained with only the IPS correction based on PBM, estimated either via EM or result randomization.

In summary, the contributions of this paper are as follows:

- We propose TrustPBM, a novel click model that captures both position-dependent click incompleteness and click noise.
- We extend the EM algorithm designed for PBM to TrustPBM and show that it can effectively estimate the model parameters for TrustPBM.
- We present a novel Bayes-IPS correction method for unbiased LTR based on the parameters in TrustPBM estimated by the EM algorithm.
- We validate our approach on large-scale click logs. The experimental results show interesting patterns for user behavior and our proposed Bayes-IPS method achieves superior results compared to baselines.

The rest of the paper is organized as follows. In Section 2, we review previous related work. We describe our TrustPBM model in Section 3 and the EM algorithm for its parameter estimation in Section 4. Our Bayes-IPS approach for unbiased LTR is presented in Section 5. We report our evaluation on click logs in Section 6. Finally, we conclude this paper and discuss future work in Section 7.

2 RELATED WORK

Leveraging user interaction data such as clicks has been shown to be quite promising in improving search quality. A simple approach is to use click and non-click information as relevance judgment of results. This can be used for evaluation or to train a new ranking function in the LTR setting. However, such an approach can lead to misleading evaluation results or sub-optimal ranking functions due to various types of bias in click data, e.g., position bias [19], presentation bias [35], and trust bias [19, 25]. In the past, a large amount of research has been devoted to extracting more accurate signals from click data. For example, some heuristic methods have been proposed to address the position bias by utilizing the pairwise preferences between clicked and skipped documents (e.g., SkipAbove) [18–20]. Though these methods have been found to provide more accurate relevance assessments, their data is fundamentally biased. For example, a ranking function trained based on the SkipAbove heuristic tends to reverse the presented order due to its sampling bias [18].

Recently, unbiased LTR has been actively studied to learn a ranking function from click data based on the counterfactual inference framework [21]. With the assumption that feedback is noise-free, it provides an IPS-based correction method which is a provably unbiased and consistent approach to LTR even with biased feedback data. The IPS method requires knowledge of the propensity of obtaining a particular feedback signal. Click modeling, especially PBM, has been used for estimating propensity. Under this model, Wang et al. [32] proposed a result randomization method that randomly shuffles the top n results and uses the average CTR at each position as the propensity. Joachims et al. [21] proposed a pivot-based randomization where position 1 and k are randomly shuffled. The relative propensity between these two positions p_k/p_1 can be derived and this is sufficient for LTR. Wang et al. [33] proposed a RandPair approach where adjacent positions are randomly flipped to obtain p_{k+1}/p_k and a chain rule is used to estimate the ratio p_k/p_1. Wang et al. [32] extended the PBM model to let the propensity depend on query or user contexts, in addition to positions. All of these approaches require interventions.

The more recent development in unbiased LTR is on how to estimate propensity without interventions. Wang et al. [33] proposed a regression-based EM algorithm to fit the PBM model to the regular click logs without interventions. Similarly, Ai et al. [3] presented a framework to jointly learn an unbiased ranker and an unbiased propensity model from regular click data. Agarwal et al. [2] proposed a method for harvesting naturally randomized pairs from logs of multiple rankers in operational systems. All of these methods are based on the standard PBM model and do not address position-dependent click noise.

With the propensity estimated, the IPS-based approach is also used for search quality evaluations, in addition to learning a ranking function. For example, Carterette and Chandar [6] extend the counterfactual inference framework [21] by considering the case of evaluating new rankers that can retrieve previously unseen documents. Chandar and Carterette [7] studied the click-through bias in the cascade model for evaluation.

IPS was developed in causal inference [28] and is a widely accepted technique for handling sampling bias. It has been employed in unbiased evaluation and learning [1, 11, 22, 23, 29, 30]. The common assumption in most of these studies is that the propensities are under the system's control and are thus known. In the unbiased LTR setting, however, propensities arise from user behavior and thus need to be estimated. While the IPS can be used to address the sample bias or click incompleteness, it is not straightforward for click noise.

Besides PBM, there are many other click models for extracting relevance signals from click data. Click models are parameterized generative models [9] and their parameters are typically estimated via generative maximum likelihood under specific modeling assumptions about user behavior. There are two classic click models: the PBM model [27] and the Cascade model [10]. Based on these two models, more advanced models have been developed. For example, the user browsing model (UBM) [12] extends PBM to condition the examination on a previously clicked position, in addition to the position of the current result. The dynamic Bayesian network model (DBN) [8] and the click chain model (CCM) [15] extend the Cascade model to handle multiple clicks. Both models assume a sequential user behavior over the result list. They focus on modeling the probability that document gets examined depending on the previous documents and differ in their modeling formulation. DBN and CCM are different from PBM and UBM in that the examination of a later document depends on the relevance of previous documents. However, most of these models assume clicks are noise-free. The focus of this paper is on click noise, specifically trust bias. We propose the TrustPBM model by extending the standard PBM model.

Though, in theory, the estimated examination probability in any click model can be used in unbiased LTR, the PBM model is more robust given that the examination does not depend on the relevance of previous documents since estimating relevance is as hard as learning a ranking function and thus leads to high variance of the estimated examination probability. Our TrustPBM model enjoys the same advantage where the trust bias depends on positions only.

The alternate approach of directly using position as a feature is generally not effective for learning a ranking function. This is because the position of a result is highly predictive of its click rate, however, the position feature cannot be used during inference time when a collection of documents must be scored to determine their positions in the presented ranking. Using the position feature in training and then unplugging it during inference leads to worse results [33].

3 TRUST ENHANCED POSITION-BASED MODEL

3.1 Position-Based Model

The Position-Based Model (PBM) is a simple generative model. It has been extensively used in unbiased LTR due to its simplicity. Most existing click models have the *examination hypothesis.* Suppose we have N positions, and for a query q, document d is displayed at position $k \in [1, N]$. A click can be generated only after the user examines the result. In the PBM model, the user clicks the document if and only if they examine the document and the document is relevant. The examination only depends on the position k, but not on q or d. Such a model factors out q and d, and makes the estimation of the probability of examination simpler and more robust, compared to other click models.

Formally, let C be a binary random variable signifying whether the user clicks document d, E whether the user examines the document, R for the true relevance and $\tilde{R}$ to represent the perceived or judged relevance. The PBM makes the following assumptions

$$C = 1 \Leftrightarrow E = 1, \tilde{R} = 1$$
$$R \Leftrightarrow \tilde{R}$$
$$\Pr(E = 1|q, d, k) = \Pr(E = 1|k)$$
$$\Pr(R|q, d, k) = \Pr(R|q, d)$$

Thus, the probability of examining a document only depends on the position k and the perceived relevance is the same as the true relevance. Based on these assumptions, the PBM has the following probability for a click.

$$\begin{aligned}\Pr(C = 1|q, d, k) &= \Pr(E = 1, \tilde{R} = 1|q, d, k)\\ &= \Pr(E = 1, R = 1|q, d, k)\\ &= \Pr(E = 1|k) \Pr(R = 1|q, d)\\ &= \theta_k \gamma_{q,d}\end{aligned}$$

where we use θ_k and $\gamma_{q,d}$ as short-hands for the probabilities.

3.2 Trust Bias

The PBM model and its extensions [32] have a noise-free assumption that perceived relevance is the same as the true relevance. This is not the case in general. How to model the relation between perceived relevance and true relevance is relatively less studied, even in the click models. DBN [8] is one of the few models that incorporates this relation. In DBN, a set of document-dependent parameters α_d are introduced to model the relation between perceived relevance and true relevance:

$$\Pr(R = 1|q, d) = \alpha_d \Pr(\tilde{R} = 1|q, d)$$

where a parameter α_d depend on the document d. The intuition is that the probability of true relevance is a discounted version of perceived relevance and that the discount is document-dependent. As a result, the model has a large number of additional parameters that equals to the number of unique documents in the data.

In this paper, we model the relation between perceived relevance and true relevance based on trust bias introduced in [19]. In this user study, eye tracking was used to monitor examination. The study showed that, after examination, users tend to trust the results presented higher on the result page more, demonstrated by higher click ratio despite the fact that the documents are both examined and their expected relevance is the same. This suggests a position-dependent trust bias in addition to examination bias, due to which higher ranked non-relevant documents may be clicked and lower ranked relevant documents not clicked. However, this has not been mathematically formulated in the past. In this paper, we model the trust bias as follows,

$$\Pr(\tilde{R} = 1|R = 1, E = 1, k) = \epsilon_k^+$$
$$\Pr(\tilde{R} = 1|R = 0, E = 1, k) = \epsilon_k^-$$

Since $C = 1 \Leftrightarrow E = 1, \tilde{R} = 1$, we have $\Pr(C = 1|R = 1, E = 1, k) = \epsilon_k^+$ and $\Pr(C = 1|R = 0, E = 1, k) = \epsilon_k^-$. Intuitively, after examination, a relevant document at position k can be missed with probability $1 - \epsilon_k^+$, but a non-relevant document can be clicked mistakenly with probability ϵ_k^-. Notice that the noise is position-dependent. A similar formulation has been studied in [21] where the noise is constant across positions.

3.3 TrustPBM

We enhance the PBM model by explicitly modeling the trust bias introduced above. We have

$$\begin{aligned}
&\Pr(\tilde{R} = 1 | E = 1, q, d, k) \\
=&\Pr(\tilde{R} = 1 | R = 1, E = 1, k) \Pr(R = 1 | q, d) \\
+&\Pr(\tilde{R} = 1 | R = 0, E = 1, k) \Pr(R = 0 | q, d) \\
=&\epsilon_k^+ \gamma_{q,d} + \epsilon_k^- (1 - \gamma_{q,d})
\end{aligned}$$

Thus, in TrustPBM, the probability of a click is given as

$$\begin{aligned}
\Pr(C = 1 | q, d, k) &= \Pr(E = 1 | k) \Pr(\tilde{R} = 1 | E = 1, q, d, k) \\
&= \theta_k (\epsilon_k^+ \gamma_{q,d} + \epsilon_k^- (1 - \gamma_{q,d}))
\end{aligned}$$

PBM can be viewed as a special case of TrustPBM, where $\epsilon_k^+ = 1$ and $\epsilon_k^- = 0$ to reflect the noise-free assumption. Both PBM and TrustPBM model examination bias in the same way. On the other hand, TrustPBM is different from DBN. The noise depends on position only, not on the document. Compared to PBM, we introduce $2N$ more parameters into the model where N is usually small (e.g., 10), so the model complexity of TrustPBM is much smaller than DBN.

The TrustPBM parameters can be estimated using logged click data. Let $\mathcal{L}$ be the logged data consisting of tuples (q, d, k, c) for each received user query q, each document d displayed at any position k and a binary indicator c for whether the document was clicked or not (i.e. the realized value of the random variable C). Then, the log-likelihood of generating $\mathcal{L}$ (over the randomness in user click behavior) is,

$$\begin{aligned}
\log \Pr(\mathcal{L}) = &\sum_{(q,d,k,c) \in \mathcal{L}} c \log \theta_k (\epsilon_k^+ \gamma_{q,d} + \epsilon_k^- (1 - \gamma_{q,d})) \\
&+ (1 - c) \log(1 - \theta_k (\epsilon_k^+ \gamma_{q,d} + \epsilon_k^- (1 - \gamma_{q,d}))).
\end{aligned}$$

In order to find the parameters that maximize the above log-likelihood objective, we use Expectation-Maximization, extending the Regression-based EM method for the plain PBM in [33].

4 ESTIMATION VIA EXPECTATION-MAXIMIZATION

As we will show in Section 5, we only need the parameters θ_k, ϵ_k^+, and ϵ_k^- for unbiased LTR. However, unlike the standard PBM model whose parameters can be estimated based on result randomization, TrustPBM does not allow estimation of these parameters using a randomized data set. In this paper, we extend the previously proposed regression-based EM algorithm [33] to estimate the parameters.

In the standard EM procedure, parameters are estimated by alternating Expectation and Maximization steps until convergence. Suppose $\{\theta_k^{(t)}, \epsilon_k^{+(t)}, \epsilon_k^{-(t)} : k \in [N]\}$ and $\{\gamma_{q,d}^{(t)} : q, d \in \mathcal{L}\}$ are the parameter estimates at iteration t (initialized appropriately at the first iteration). We will get a new set of parameter estimates after iteration $t + 1$.

4.1 Expectation Step

In the Expectation step at iteration $t + 1$, the posterior distribution of the hidden variables of examination E and true relevance R given the observed click data $\mathcal{L}$ is expressed in terms of the parameter estimates at iteration t, as shown in Figure 1 (we omit superscript (t) for readability). All the formulas follow directly from Bayes rules. For instance, in the third from last equation, we have

$$\begin{aligned}
&\Pr(E = 0, R = 1 | q, d, k, c = 0) \\
&= \frac{\Pr(C = 0 | E = 0, R = 1, q, d, k) \Pr(E = 0, R = 1 | q, d, k)}{\Pr(C = 0 | q, d, k)} \\
&= \frac{\Pr(E = 0, R = 1 | q, d, k)}{\Pr(C = 0 | q, d, k)} \\
&= \frac{\Pr(E = 0 | k) \Pr(R = 1 | q, d)}{\Pr(C = 0 | q, d, k)}
\end{aligned}$$

since $\Pr(C = 0 | E = 0, R = 1, q, d, k) = 1$, and examination and relevance are independent events. Also, a click means that the document must have been examined and thus $\Pr(E = 0, R = 1 | q, d, k, c = 1) = 0$ in the second equation.

4.2 Maximization Step

In the Maximization step at iteration $t + 1$, the parameters are updated to their maximum likelihood values given the click data and the posterior probabilities from the Expectation step as shown in Figure 2. Once again, we drop the superscript $(t + 1)$ for clarity in these formulas. Different from PBM model, we have additional ϵ_k^+ and ϵ_k^- parameters updated in this step.

Regression-based EM. For the M-step to work with query-document pairs (q, d), a requirement is that (q, d) should repeat and also be shown in different positions. Since the query-document pairs (q, d) vary arbitrarily, using free parameters $\gamma_{q,d}$ makes the objective very sparse and difficult to learn. Moreover, exposing individual (q, d) identifiers can also pose a privacy or security risk. Therefore, as in [33], we propose to estimate the $\gamma_{q,d}$ parameters in each Maximization step via regression, i.e. fit a function (such as a Gradient Boosted Decision Tree) from features of (q, d) to the derived target value for $P(R = 1 | c, q, d, k)$ at iteration $t + 1$.

More specifically, we assume that there is a feature vector $\mathbf{x}_{q,d}$ representing them and use a function to compute the relevance $\gamma_{q,d} = f(\mathbf{x}_{q,d})$. The Maximization step is then to find a regression function $f(\mathbf{x})$ to maximize the likelihood given the estimation from the Expectation step. Specifically, for each $(q, d, k, c) \in \mathcal{L}$, the expectation step gives a probability $P(R = 1 | q, d, k, c)$. Intuitively, we can regress the feature vector $\mathbf{x}_{q,d}$ to the probability $P(R = 1 | c, q, d, k)$. Similar to [33], we convert such a regression problem to a classification problem based on sampling: we sample a binary relevance label $r \in \{0, 1\}$ according to $P(R = 1 | c, q, d, k)$. This conversion allows us to use widely available classification tools to solve our problem. After sampling, we have a training set $\{(\mathbf{x}, r)\}$ for $f(\mathbf{x})$.

$$Pr(E=1,R=1|q,d,k,c=1)=\frac{\epsilon_k^+\gamma_{q,d}}{\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d})},\qquad \Pr(E=0,R=1|q,d,k,c=1)=0$$

$$Pr(E=1,R=0|q,d,k,c=1)=\frac{\epsilon_k^-(1-\gamma_{q,d})}{\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d})},\qquad \Pr(E=0,R=0|q,d,k,c=1)=0$$

$$Pr(E=1,R=1|q,d,k,c=0)=\frac{\theta_k(1-\epsilon_k^+)\gamma_{q,d}}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))},\qquad \Pr(E=0,R=1|q,d,k,c=0)=\frac{(1-\theta_k)\gamma_{q,d}}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))}$$

$$\Pr(E=1,R=0|q,d,k,c=0)=\frac{\theta_k(1-\epsilon_k^-)(1-\gamma_{q,d})}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))},\qquad \Pr(E=0,R=0|q,d,k,c=0)=\frac{(1-\theta_k)(1-\gamma_{q,d})}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))}$$

Figure 1: Formulas for computing probabilities of hidden variables in the E-step of the EM algorithm for TrustPBM.

$$\theta_k=\frac{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}(c+(1-c)P(E=1|q,d,k,C=0))}{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}}$$

$$\gamma_{q,d}=\frac{\sum_{(q',d',k,c)\in\mathcal{L}}1\{q'=q,d'=d\}(c\cdot P(R=1|q,d,k,C=1)+(1-c)P(R=1|q,d,k,C=0))}{\sum_{(q',d',k,c)\in\mathcal{L}}1\{q'=q,d'=d\}}$$

$$\epsilon_k^+=\frac{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}\cdot c\cdot P(E=1,R=1|,q,d,k,C=1)}{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}(c\cdot P(E=1,R=1|q,d,k,C=1)+(1-c)P(E=1,R=1|q,d,k,C=0))}$$

$$\epsilon_k^-=\frac{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}\cdot c\cdot P(E=1,R=0|,q,d,k,C=1)}{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}(c\cdot P(E=1,R=0|q,d,k,C=1)+(1-c)P(E=1,R=0|q,d,k,C=0))}$$

Figure 2: Formulas for updating parameters in the M-step of the EM algorithm for TrustPBM, where 1 is the indicator function.

The objective function is the log likelihood:

$$\sum_{\{(\mathbf{x},r)\}} r\log(f(\mathbf{x}))+(1-r)\log(1-f(\mathbf{x})),$$

where we use the sigmoid in the objective function

$$f(\mathbf{x})=\frac{1}{1+e^{-F(\mathbf{x})}}$$

$F(x)$ is the log odd of function $f(x)$ and is learned by Gradient Boosted Decision Tree (GBDT) method [13] in this paper.

5 UNBIASED LTR VIA BAYES-IPS CORRECTION

Now we turn to the question of using the parameter estimates of the TrustPBM to learn a new ranking function from the click data in an unbiased way. Specifically, we need to adjust each click signal in the logged data such that it reflects the actual underlying relevance signal in expectation, unconfounded by the inherent examination and trust biases captured by TrustPBM.

In existing work [21, 33] which deals with the standard PBM, this is achieved via Inverse Propensity Scoring (IPS) which corrects for the examination bias factor. Since TrustPBM does not factorize into purely bias (from examination and trust) and relevance terms, a direct IPS correction is not feasible. In the following, we first review the existing unbiased LTR and then present our two-step approach to deal with noise: correct for examination bias via IPS, and then de-noise for trust bias via Bayes law.

5.1 LTR

We begin with a brief overview of the LTR setting. In the standard LTR, a set of labeled data for query and documents are given $\mathcal{L}_U=\{(q,d,r)\}$ where r represents the relevance labels for the (q,d) pair. We also use $\Omega_q=\{d|(q,d)\in\mathcal{L}_U\}$ to represent all documents that are associated with query q. Collection $\mathcal{L}_U$ is *unbiased* and a uniform sample from the space of queries, documents, and relevance. In general, fully supervised LTR algorithms such as LambdaMART and RankingSVM are used to directly or indirectly optimize a performance metric which decomposes as a sum over the (q,d) samples in the training set, as such

$$M=\sum_{(q,d,r)\in\mathcal{L}_U} r\cdot f(q,d,\Omega_q)$$

where f is usually a rank-related function. For instance, Average Relevance Position (ARP) [21, 33, 34] and Discounted Cumulative Gain (DCG) like metrics [17, 33] are shown as below,

$$ARP=\sum_{(q,d,r)\in\mathcal{L}_U} r\,\text{rank}(q,d,\Omega_q)$$

$$DCG=\sum_{(q,d,r)\in\mathcal{L}_U} r\,\frac{1}{\text{rank}(q,d,\Omega_q)}.$$

Note that lower *ACP* and higher *DCG* indicate better ranking performance with more relevant documents at higher ranks. We use a simple DCG definition that uses a binary relevance and a linear function for position discount to be consistent with the rest of paper. But the methodology is applicable to any DCG definitions.

In pairwise learning-to-rank algorithms, scores s are learned for (q, d) pairs by expressing rank as a pairwise objective

$$\begin{aligned} ARP &= \sum_{(q,d,r)\in\mathcal{L}_U} r \operatorname{rank}(q, d, \Omega_q) \\ &= \sum_{(q,d,r)\in\mathcal{L}_U} r\,(\sum_{d'\in\Omega_q} \mathbb{I}_{s_{(q,d)}<s_{(q,d')}} + 1) \\ &= \sum_{(q,d,r)\in\mathcal{L}_U} \sum_{d'\in\Omega_q} r\,\mathbb{I}_{s_{(q,d)}<s_{(q,d')}} + const \qquad (1) \end{aligned}$$

The loss above can then be upper-bounded by common classification loss functions (e.g., the hinge loss or the logistic loss [16]) as in the RankingSVM method. Moreover, rank-based metrics such as *ARP* and *DCG* can also be optimized in the LambdaLoss framework [34] where LambdaRank and LambdaMART [5] are special cases in the framework.

5.2 Unbiased LTR

The challenge in learning from implicit feedback such as click data is that only the click information c is observed, not the true relevance r for each (q, d) pair. The relation between r and c is captured by click models. The Inverse Propensity Scoring (IPS) technique is used to tackle this issue by de-biasing click data because the examination bias in PBM is equivalent to the propensity of observing relevance labels in the counterfactual framework [21].

Recall that we have logged data $\mathcal{L}$ consisting of tuples (q, d, k, c),

$$\mathbb{E}[C|q, d, k] = \Pr(C = 1|q, d, k) = \theta_k \Pr(\tilde{R} = 1|E = 1, q, d, k)$$

where the expectation is over the randomness in the user click behavior. As in previous work, we can employ IPS to correct for the examination bias in the following way,

$$\begin{aligned} M_{IPS} &= \sum_{(q,d,k,c)\in\mathcal{L}} \frac{c}{\theta_k} f(q, d, \Omega_q) \\ &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} f(q, d, \Omega_q) \end{aligned}$$

M_{IPS} is an unbiased estimation of metric M because

$$\begin{aligned} \mathbb{E}[M_{IPS}] &= \sum_{(q,d,k,c)\in\mathcal{L}} \frac{\mathbb{E}[C]}{\theta_k} f(q, d, \Omega_q) \\ &= \sum_{(q,d,k,c)\in\mathcal{L}} \Pr(\tilde{R} = 1|E = 1, q, d, k) f(q, d, \Omega_q) \end{aligned}$$

In the plain PBM which assumes that $\Pr(\tilde{R} = 1|E = 1, q, d, k) = \Pr(R = 1|q, d)$, i.e. judgment upon examination reflects actual relevance free from noise, interpreting r as $\Pr(R = 1|q, d)$ (the degree of true relevance), M_{IPS} equals M in expectation and thus can be used as a direct unbiased substitute in the learning-to-rank algorithm since it is simply a linear weight modification.

$$ARP_{IPS} = \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \operatorname{rank}(q, d, \Omega_q)$$

$$DCG_{IPS} = \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \frac{1}{\operatorname{rank}(q, d, \Omega_q)}.$$

For instance, the upper-bounding technique shown in Eq 1 still applies with ARP_{IPS} and an IPS-based RankingSVM was derived in [21] and the LambdaMART algorithm can be used to optimize DCG_{IPS}.

5.3 Bayes-IPS Correction

Now, for the TrustPBM, we do not have the equivalence between perceived relevance $\tilde{R}$ and R. The standard IPS approach gives an unbiased estimation of $\tilde{R}$. Note that we only care about the clicked documents in M_{IPS}. To get the estimate of R, we only need to estimate it for $C = 1$, i.e., $\Pr(R = 1|C = 1, q, d, k)$. Such an estimation can involve many parameters due to the dependency on (q, d) pairs. To make it tractable, we thus approximate it by the average relevance at position k: $\Pr(R = 1|C = 1, k)$, i.e., $\Pr(R = 1|\tilde{R} = 1, E = 1, k)$. By Bayes law, we have

$$\begin{aligned} &\Pr(R = 1|\tilde{R} = 1, E = 1, k) \\ &= \frac{\Pr(\tilde{R} = 1|R = 1, E = 1, k)\Pr(R = 1|E = 1, k)}{\Pr(\tilde{R} = 1|E = 1, k)} \\ &= \frac{\epsilon_k^+ \Pr(R = 1|E = 1, k)}{\epsilon_k^+ \Pr(R = 1|E = 1, k) + \epsilon_k^- \Pr(R = 0|E = 1, k)} \\ &= \frac{\epsilon_k^+ \Pr(R = 1)}{\epsilon_k^+ \Pr(R = 1) + \epsilon_k^- \Pr(R = 0)} \\ &= \frac{\epsilon_k^+}{\epsilon_k^+ + \epsilon_k^-} \end{aligned}$$

The first step is based on Bayes law. The second is the definition of ϵ_k^+ and ϵ_k^-. The third step is because R is independent of E and k. The last step is based on a weak uninformative *prior* on actual relevance that $\Pr(R = 1) = \Pr(R = 0) = 0.5$. Putting it all together, we have

$$\begin{aligned} M_{Bayes-IPS} &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \Pr(R = 1|C = 1, k) f(q, d, \Omega_q) \\ &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \frac{\epsilon_k^+}{\epsilon_k^+ + \epsilon_k^-} f(q, d, \Omega_q). \end{aligned}$$

Informally, the Bayes-IPS estimate corrects for the examination bias and the likelihood that the result was actually relevant given that it was judged so upon examination. As a sanity check, note that in the noise-free case of the plain PBM, i.e. when $\epsilon_k^+ = 1$ and $\epsilon_k^- = 0$, $M_{Bayes-IPS}$ reduces to M_{IPS}. Moreover, since the proposed correction is simply a change of weights, a learning-to-rank framework with IPS can be easily adapted to use Bayes-IPS weights to replace IPS weights.

6 EMPIRICAL EVALUATION

In this section, we evaluate our proposed TrustPBM click model and the Bayes-IPS correction for unbiased LTR. We first describe our experiment setup and then present our experimental results.

6.1 Experiment Setup

Our experiments are based on both offline evaluation and online experiments. For offline evaluation, we use the standard framework for supervised LTR evaluation [24] by splitting the data into training and test sets. For online experiments, we evaluate our proposed methods through online A/B testing. In this section, we describe

Table 1: Statistics of the three data sets used in offline evaluation.

	Email	File Storage	Email Expert
#Queries per user	> 3	> 3	< 4
#Docs in total	59.69M	50.77M	13.80M
#Docs per query	5.99	5.00	6.00

the data sets and metrics used in offline evaluation as well as the metrics used in online experiments.

6.1.1 Data Sets. For offline evaluation, we use search logs collected from three search services: Gmail for general users (denoted as **Email**), Google Drive for general users (denoted as **File Storage**), and Gmail for expert users (denoted as **Email Expert**). The Email and Email Expert services have the same type of contents, but the users in the Email Expert dataset are more active, as measured by their search activity. The File Storage service has a different type of content compared with the other two. In all services, each query can result in at most a single click of a document because an overlay showing results is displayed as a query is typed and it disappears as a result is clicked. The clicked document is then shown on the screen.

In our data sets, we discard all the queries that did not result in any clicks. No further pre-processing is done. In all these services, ranking features are routinely computed and logged. We collected a sample from the search logs of each service in a two-week period in May 2018. Data from the first week are used for training, and those from the second week are used for testing. Basic statistics for the data sets are shown in Table 1. Each data set contains millions of queries, and between 5 and 6 documents per query. The logs also contain existing ranking features for each query-document pair, which we use in our regression-based EM algorithms and unbiased LTR algorithms.

6.1.2 Offline evaluation metrics. Our offline evaluation consists of two parts. The first part is to compare the PBM and TrustPBM models based on their fitness to the data. For this part, we use the standard log-likelihood on how the models predict clicks in our test data.

$$\text{LogLikelihood} = \frac{1}{|\mathcal{L}|}\sum_{\mathcal{L}} c\log(p) + (1-c)\log(1-p) \qquad (2)$$

where $p = \theta_k \gamma_{q,d}$ for PBM and $p = \theta_k(\epsilon_k^+ \gamma_{q,d} + \epsilon_k^-(1-\gamma_{q,d}))$ for TrustPBM. Log-likelihood is commonly used to evaluate click models [9]. It is a negative number and the larger the better.

For the second part, we compared different methods on the effectiveness of learned ranking functions for unbiased LTR. Following [32, 33], we use the weighted version of Mean Reciprocal Rank (MRR) as the evaluation metric for offline experiment. It is an unbiased version of MRR and more suitable than standard MRR for offline evaluation given click bias. Specifically, given a test data set with n queries, the standard MRR is defined as:

$$\text{MRR} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\text{rank}_i} \qquad (3)$$

where rank_i denotes the rank of the first clicked document of the i-th query in our data sets. Note that since each query can have at most one clicked document, we do not consider more complicated metrics like normalized discounted cumulative gain. We denote weighted MRR [33] by MRR_w, which is defined as:

$$\text{MRR}_w = \frac{1}{\sum_{i=1}^{n} w_i}\sum_{i=1}^{n} w_i \frac{1}{\text{rank}_i} \qquad (4)$$

where w_i is the IPS correction weight estimated from our algorithm. Given different propensity estimation methods, the weights are used for both training and offline evaluation.

6.1.3 Online experiment metrics. For online experiments, we run A/B testing using ranking functions learned based on different IPS weights. This is the ultimate test of effectiveness of ranking functions. A better ranking function can attract more clicks from users and push the user clicks to higher positions. We thus use the click-through rate (CTR) that is defined as the ratio of queries with clicks and the total number of queries. We also use the standard MRR in Eq 3 for online evaluation to take into account the effect of positions. For any queries without clicks, their MRR values are set to 0 for online experiments.

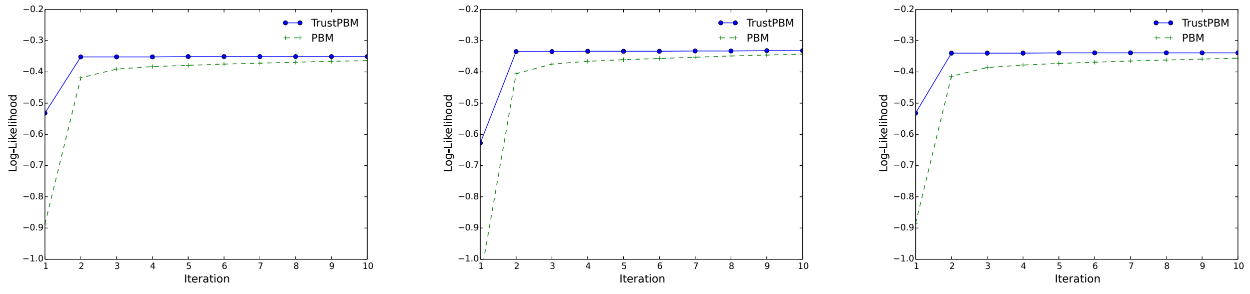
Table 2: Comparison of PBM and TrustPBM click models based on log-likelihood. The best number per row is in bold face and + means statistically significance based on the student t-test.

	PBM	TrustPBM	Improvement
Email	-0.364	**-0.351**	$3.57\%^+$
File Storage	-0.343	**-0.332**	$3.21\%^+$
Email Expert	-0.356	**-0.339**	$4.78\%^+$

6.2 Experimental Results

In this section, we report both offline and online experimental results. Our comparisons are conducted for both click modeling and unbiased LTR.

6.2.1 Does the extended model fit the data better? Both PBM and TrustPBM are click models. Our first result is to compare them on how well they can fit our click data. We run our EM algorithms for both PBM and TrustPBM respectively and compare them in terms of log-likelihood in Eq 2. The results on our test data sets are shown in Table 2. From this table, we can see that the TrustPBM model has higher log-likelihood values than the PBM model on all the three data sets. We also report the relative improvement of the log-likelihood in this table. All these improvements are statistically significant based on the student t-test. This means that our proposed method of incorporating click noise in TrustPBM is reasonable since TrustPBM can indeed fit the data significantly better than PBM. In addition, we also study the convergence of the EM algorithms for PBM and TrustPBM in Figure 3, where we plot log-likelihood along with the EM iterations. It can be observed that TrustPBM not only converges to a higher log-likelihood, but also converges faster than PBM at early stages. This confirms that our proposed EM algorithm can estimate parameters for TrustPBM effectively. Overall, this result shows that TrustPBM is better that PBM.



(a) Email (b) File Storage (c) Email Expert

Figure 3: Convergence of EM based on log-likelihood.

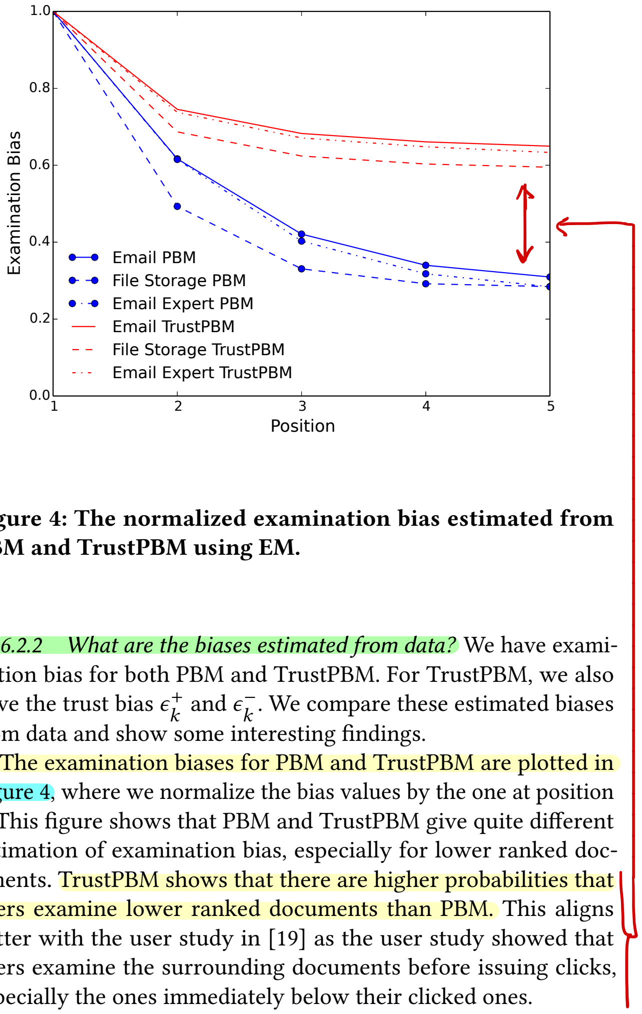


Figure 4: The normalized examination bias estimated from PBM and TrustPBM using EM.

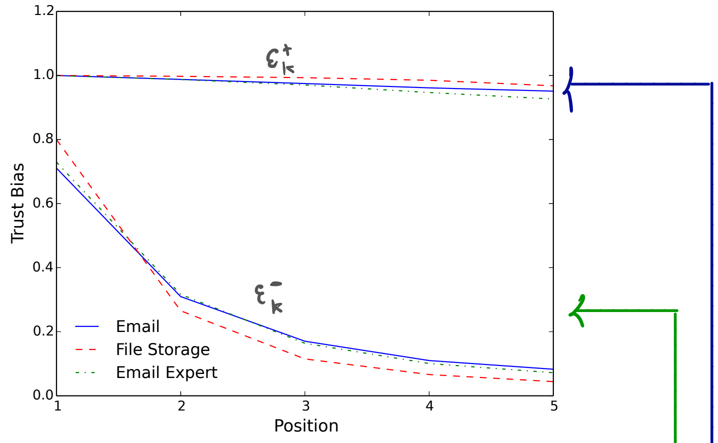


Figure 5: The normalized trust bias estimated for TrustPBM using EM. The upper lines are ϵ_k^+ and the lower lines are ϵ_k^-.

6.2.2 What are the biases estimated from data? We have examination bias for both PBM and TrustPBM. For TrustPBM, we also have the trust bias ϵ_k^+ and ϵ_k^-. We compare these estimated biases from data and show some interesting findings.

The examination biases for PBM and TrustPBM are plotted in Figure 4, where we normalize the bias values by the one at position 1. This figure shows that PBM and TrustPBM give quite different estimation of examination bias, especially for lower ranked documents. TrustPBM shows that there are higher probabilities that users examine lower ranked documents than PBM. This aligns better with the user study in [19] as the user study showed that users examine the surrounding documents before issuing clicks, especially the ones immediately below their clicked ones.

The trust bias estimated by TrustPBM is shown in Figure 5 where we normalize all of them by the ϵ_k^+ for $k = 1$. There are a few interesting observations. (1) All ϵ_k^+ are higher than ϵ_k^-. This means that the probability of clicks given relevant documents is higher than the probability given non-relevant documents after examination. Thus clicks represent reasonable relevance judgments even though they are noisy. (2) ϵ_k^+ are very close to 1 over all positions k, and the difference over positions is very small, showing that relevant documents are not missed often once examined. (3) In contrast, ϵ_k^- varies a lot over positions k. It is much higher at the higher positions (lower values of k) than lower positions, meaning that users are more inclined to click non-relevant documents shown at high positions. This is congruent with the trust bias user study [19], which found that users may place more trust in highly-ranked results. (4) Across different data sets, the trust bias is different. For example, the File Storage data set has a steeper curve for ϵ_k^-, meaning that there is higher trust bias for this service. Overall, this result shows that TrustPBM is better that PBM qualitatively.

6.2.3 Is the Bayes-IPS correction different? We now turn to the unbiased LTR evaluation. The inverse propensity weights estimated by different methods in the Email data set are shown in Figure 7. For result randomization, we use the FairPair approach in [33], that

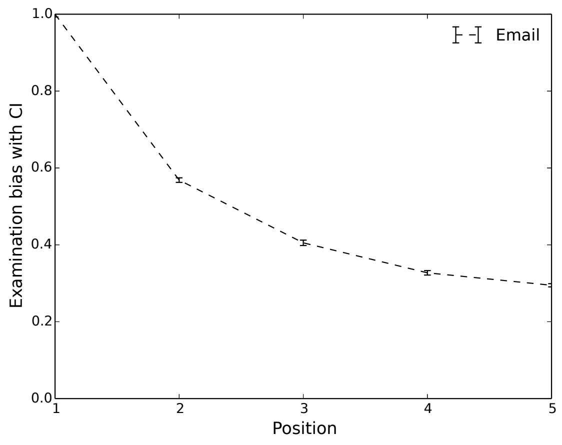


Figure 6: The examination bias and the confidence interval (CI) based on randomization data.

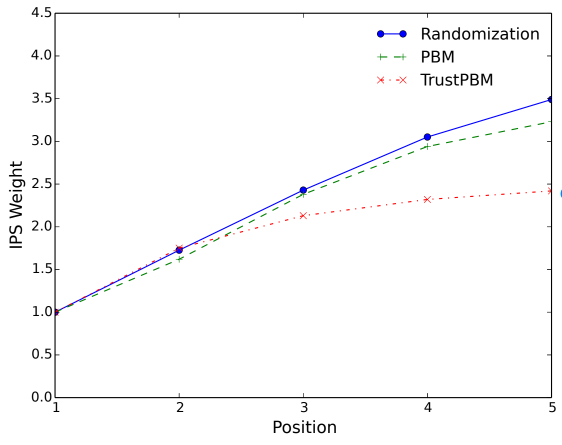


Figure 7: The IPS or Bayes-IPS weights based on Randomization (IPS), PBM (IPS) and TrustPBM (Bayes-IPS).

is, two adjacent documents are randomly swapped before presenting results to users. This method has been shown to be aligned well with other randomization methods for the PBM model. We use Randomization to denote the IPS weights estimated from this method. We use PBM to denote the IPS weights estimated by EM in PBM, and we TrustPBM to denote the Bayes-IPS weights estimated by EM in TrustPBM. Compared with TrustPBM, the EM-based PBM is much closer to Randomization. This result is expected since Randomization is also based on PBM. However, TrustPBM gives much smaller estimation of weights for lower positions. This is because TrustPBM uses a different model that disentangles examination and trust bias, leading to rather different estimation of inverse propensity weights.

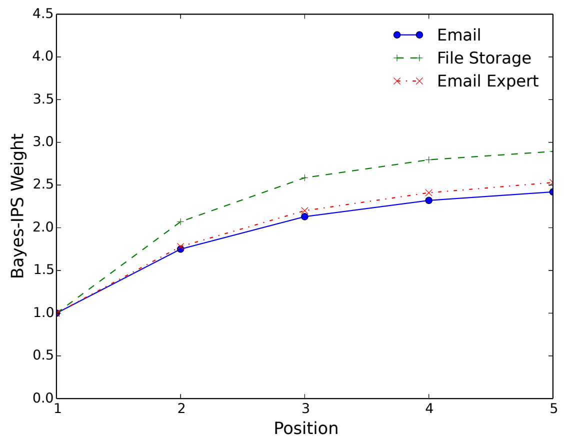


Figure 8: The Bayes-IPS weights estimated for different data sets.

Table 3: Cross offline evaluation to compare IPS weights for training and evaluation.

Training	Evaluation: Weighted MRR in Eq 4			
	NoIPS	Randomization	PBM	TrustPBM
NoIPS	**0.00%**	-0.50%	-0.36%	-0.20%
Randomization	-1.11%	**0.00%**	0.00%	-0.11%
PBM	-0.89%	-0.02%	**0.00%**	-0.06%
TrustPBM	-0.48%	-0.11%	-0.05%	**0.00%**

The lower IPS weights at lower positions in TrustPBM is more desired. As commonly known, IPS-based approaches usually suffer from the problem of the unbounded variance since the variance for a smaller propensity (corresponding to the lower positions in unbiased LTR) can lead to a large difference in IPS weights. Several techniques such as capping methods are developed to address this problem [4, 11, 14, 26, 31]. For Randomization, we compute the variance of our estimated propensity θ_k as follows. We split our randomized data set into 10 folds and use them to estimate the confidence interval (CI). The results are show in Figure 6. From this figure, we see that there is a rather negligible variance based on CI. This means that we have sufficient randomized data for IPS weights estimation. As we will show later that TrustPBM leads to better online experiment results, the result here indicates that PBM is a less accurate model than TrustPBM for extracting relevance from user click behaviors.

In addition, Figure 8 shows the Bayes-IPS weights estimated for different data sets. The weights being different for different data sets means that our method can indeed take click noise into account that varies across data sets.

6.2.4 *Can the learning algorithms leverage different IPS weights?*

We have shown that different methods give different IPS weights. These IPS weights can be used for both evaluation and learning for unbiased LTR. In this sense, we do not have ground-truth IPS

Table 4: Online experiment results based on LambdaMART models learned using different weights. Relative improvement over Randomization is reported, together with the confidence interval (CI). The best number per column is bold and + means the improvement is significant based on student t-test.

	MRR	MRR CI	CTR	CTR CI
Randomization	0.00%	-	0.00%	-
PBM	0.25%	±0.39%	0.26%	±0.40%
TrustPBM	**0.33%**$^+$	±0.29%	**0.33%**$^+$	±0.28%

weights to compare different methods. We rely on the online experiments to do the comparison. Before doing so, we need to make sure that our learning algorithms can effectively leverage the IPS weights during training.

We use an in-house implementation of LambdaMART as our learning algorithm. Table 3 shows the cross comparison using different IPS weights for both training and evaluation. This comparison serves as a sanity check as to whether our training methods can take into account different settings of weights. All the values in Table 3 are relative numbers comparing to the diagonal lines per column. The diagonal values are expected to be the largest numbers per column since both training and evaluation use the same weights. The results in this table confirm that. For example, using the Bayes-IPS weights in TrustPBM, all models trained with different weights show negative relative evaluation numbers. Thus our learning algorithms are able to differentiate different weights to optimize their intended metrics.

6.2.5 Results of online experiments. We compare Randomization, PBM and TrustPBM using the live traffic of the Email service, conducting an online experiment for each method on a portion of the search traffic. These experiments were run for several weeks in August 2018 and our comparisons are based on approximately 100M user queries per experiment. For clarity, we normalize the online evaluation metrics using the Randomization method as the baseline.

We compute online metrics MRR and CTR; the results are shown in Table 4. In this table, we can see that TrustPBM improves over the Randomization method significantly, with MRR and CTR both increased by +0.33%, and these results are statistically significant. In contrast, PBM achieves neutral results compared with the Randomization baseline. Moreover, TrustPBM is more robust than PBM because it has a much smaller confidence interval (CI). This is probably due to the fact that having higher weights on lower positions can lead to higher variance when learning a ranking function for the PBM model. By addressing click noise, TrustPBM is more robust.

7 CONCLUSION

In this paper, we proposed to handle click noise for unbiased LTR. We have introduced an enhanced position bias model called TrustPBM which explicitly models position-dependent click noise, conceptualized as a result of the trust users have in the quality of the search ranking. The parameters of the TrustPBM are estimated from offline click logs via regression-based Expectation Maximization, and the novel Bayes-IPS correction provides a simple weighting scheme for unbiased, de-noised LTR using the estimated parameters. Experiments on three widely used commercial search services revealed new insights about the asymmetry in positive and negative trust bias. Furthermore, the Bayes-IPS correction yielded superior performance metrics over the IPS correction based on the vanilla PBM estimated via EM or randomization.

Future work can include alternate methods for TrustPBM estimation, such as via intervention harvesting [2] which requires click logs from multiple rankers but avoids relevance modeling, and alternate schemes for unbiased LTR that weaken the prior assumption in our Bayes-IPS correction.

REFERENCES

[1] Aman Agarwal, Soumya Basu, Tobias Schnabel, and Thorsten Joachims. 2017. Effective Evaluation using Logged Bandit Feedback from Multiple Loggers. In *Proc. of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).* 687–696.

[2] Aman Agarwal, Ivan Zaitsev, and Thorsten Joachims. 2018. Consistent Position Bias Estimation without Online Interventions for Learning-to-Rank. (2018). arXiv:cs.LG/1806.03555

[3] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W. Bruce Croft. 2018. Unbiased Learning to Rank with Unbiased Propensity Estimation. In *Proc. of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR).* 385–394.

[4] Léon Bottou, Jonas Peters, Joaquin Qui nonero Candela, Denis X. Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research* 14 (2013), 3207–3260.

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

[6] Ben Carterette and Praveen Chandar. 2018. Offline Comparative Evaluation with Incremental, Minimally-Invasive Online Feedback. In *Proc. of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR).* 705–714.

[7] Praveen Chandar and Ben Carterette. 2018. Estimating Clickthrough Bias in the Cascade Model. In *Proc. of the 27th ACM International Conference on Information and Knowledge Management (CIKM).* 1587–1590.

[8] Olivier Chapelle and Ya Zhang. 2009. A dynamic Bayesian network click model for web search ranking. In *Proceedings of the 18th International Conference on World Wide Web (WWW).* 1–10.

[9] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click Models for Web Search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 7, 3 (2015), 1–115.

[10] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An Experimental Comparison of Click Position-bias Models. In *Proc. of the 1st International Conference on Web Search and Data Mining (WSDM).* 87–94.

[11] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly Robust Policy Evaluation and Learning. In *Proc. of the 28th International Conference on Machine Learning (ICML).* 1097–1104.

[12] Georges E. Dupret and Benjamin Piwowarski. 2008. A User Browsing Model to Predict Search Engine Click Data from Past Observations. In *Proc. of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).* 331–338.

[13] Jerome H. Friedman. 2000. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29 (2000), 1189–1232.

[14] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B Testing for Recommender Systems. In *Proc. of the 11th ACM International Conference on Web Search and Data Mining (WSDM).* 198–206.

[15] Fan Guo, Chao Liu, Anitha Kannan, Tom Minka, Michael Taylor, Yi-Min Wang, and Christos Faloutsos. 2009. Click Chain Model in Web Search. In *Proc. of the 18th International Conference on World Wide Web (WWW).* 11–20.

[16] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning.* Springer.

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

[18] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).* 133–142.

[19] Thorsten Joachims, Laura A. Granka, Bing Pan, Helene Hembrooke, and Geri

Gay. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *Proc. of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 154–161.

[20] Thorsten Joachims, Laura A. Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the Accuracy of Implicit Feedback from Clicks and Query Reformulations in Web Search. *ACM Transactions on Information Systems (TOIS)* 25, 2 (April 2007).

[21] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proc. of the 10th ACM International Conference on Web Search and Data Mining (WSDM)*. 781–789.

[22] Lihong Li, Shunbao Chen, Jim Kleban, and Ankur Gupta. 2014. Counterfactual Estimation and Optimization of Click Metrics for Search Engines. (2014). arXiv:cs.LG/1403.1891

[23] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased Offline Evaluation of Contextual-bandit-based News Article Recommendation Algorithms. In *Proc. of the 4th International Conference on Web Search and Web Data Mining (WSDM)*. 297–306.

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

[25] Maeve O'Brien and Mark T. Keane. 2007. Modeling User Behavior Using a Search-Engine. In *Proc. of the 12th International Conference on Intelligent User Interfaces (IUI)*. 357–360.

[26] M. J. D. Powell and J. Swann. 1966. Weighted Uniform Sampling — a Monte Carlo Technique for Reducing Variance. *IMA Journal of Applied Mathematics* 2, 3 (1966), 228–236.

[27] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting Clicks: Estimating the Click-through Rate for New Ads. In *Proc. of the 16th International Conference on World Wide Web (WWW)*. 521–530.

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

[29] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as Treatments: Debiasing Learning and Evaluation. In *Proceedings of the 33nd International Conference on Machine Learning (ICML)*. 1670–1679.

[30] Adith Swaminathan and Thorsten Joachims. 2015. Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization. *Journal of Machine Learning Research (JMLR)* 16 (Sep 2015), 1731–1755.

[31] Adith Swaminathan and Thorsten Joachims. 2015. The Self-Normalized Estimator for Counterfactual Learning. In *Proc. of the 29th Conference on Neural Information Processing Systems (NIPS)*. 3231–3239.

[32] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proc. of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 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 11th ACM International Conference on Web Search and Data Mining (WSDM)*. 610–618.

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

[35] Yisong Yue, Rajan Patel, and Hein Roehrig. 2010. Beyond Position Bias: Examining Result Attractiveness as a Source of Presentation Bias in Clickthrough Data. In *Proc. of the 19th International Conference on World Wide Web (WWW)*. 1011–1018.