External Evaluation of Ranking Models under Extreme Position-Bias

Yaron Fairstein, Elad Haramaty, Arnon Lazerson, Liane Lewin-Eytan
{yaronf,eladh,arnonl,lliane}@amazon.com
Amazon
Haifa, Israel

ABSTRACT

Implicit feedback from users behavior is a natural and scalable source for training and evaluating ranking models in human-interactive systems. However, inherent biases such as the position bias are key obstacles to its effective usage. This is further accentuated in cases of extreme bias, where behavioral feedback can be collected exclusively on the top ranked result. In fact, in such cases, state-of-art debiasing methods cannot be applied. A prominent use case of extreme position bias is the voice shopping medium, where only a small amount of information can be presented to the user during a single interaction, resulting in user behavioral signals that are almost exclusively limited to the top offer. There is no way to know how the user would have reacted to a different offer than the top one he was actually exposed to. Thus, any new ranker we wish to evaluate with respect to a behavioral metric, requires online experimentation. We propose a novel approach, based on an *external estimator* model, for accurately predicting offline the performance of a new ranker. The accuracy of our solution is proven theoretically, as well as demonstrated by a line of experiments. In these experiments, we focus on the use case of purchase prediction, and show that our estimator can accurately predict offline the purchase rate of different rankers over a segment of voice shopping traffic. Our prediction is validated online, as being compared to the actual performance obtained by each ranker when being exposed to users.

CCS CONCEPTS

• **Computing methodologies** → **Learning from implicit feedback**; • **Information systems** → **Retrieval models and ranking**.

KEYWORDS

voice search, position bias, off-policy evaluation

ACM Reference Format:
Yaron Fairstein, Elad Haramaty, Arnon Lazerson, Liane Lewin-Eytan. 2022. External Evaluation of Ranking Models under Extreme Position-Bias. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3488560.3498420

WSDM '22, February 21–25, 2022, Tempe, AZ, USA

ACM ISBN 978-1-4503-9132-0/22/02…$15.00
https://doi.org/10.1145/3488560.3498420

1 INTRODUCTION

Evaluating the performance of ranking models at scale in human-interactive system is commonly based on users' implicit feedback such as clicks, dwell time, or other engagement actions (e.g. purchases in e-commerce search engines). While this source of data has many advantages (scalable, timely, user centric), it suffers from inherent biases such as the position bias, which strongly influences the feedback that a result receives. A long line of studies have addressed this problem [1, 2, 9, 19, 24, 29–31, 35], however, they do not focus on the case of severe position bias, where user feedback is available almost exclusively on the top ranked result.

A prominent use case of this severe bias can be found in voice shopping, which evolved in the last years as an emerging usage of intelligent voice assistants. In the traditional voice shopping scenario, users issue a product search query by voice, and get back products upon which they can take some action such as purchase or add-to-cart. As both input and output are spoken in voice, users are exposed to only few offers out of the results list which introduces a significant position bias. This phenomenon is even more pronounced in devices without a screen, which are limited to presenting one offer at a time. The sequential nature of this process, together with the limited attention span of users to a synthesized voice output, cause users to interact almost exclusively with the *top offer*[1]. We model this almost absolute position bias as *extreme position bias* where user feedback is available only on the top result.

The quality of the top offer presented in the voice medium is thus a leading factor in user satisfaction. In order to improve quality, a ranker is trained to optimize a predefined metric. Given the extreme position bias, we could directly optimize offline an annotations-based metric such as objective relevance. This is done using a labeled dataset, where each query–product pair is manually labeled according to the relevance of the product to the query. However, objective relevance fails to capture specific user preferences. In fact is has been shown that users purchase or engage with search results that are objectively judged irrelevant [4].

Optimizing for behavioral metrics is crucial in order to capture the subjective notion of relevance. Behavioral metrics infer users satisfaction from their actions, and take into account also personal preferences (e.g. taste and price-quality preference). Unfortunately, it is impossible to obtain a fully labeled dataset based on users actions in our setting. As behavioral signals are limited to the top offer only, we have no way of knowing how the user would have reacted to a different offer without actually trying it online. It is possible to apply online debiasing techniques such as result randomization or model perturbation approaches. However, they require looking directly at users behavior using online experiments,

[1] *Top offer/result* refers to *top ranked offer/result*

which is costly, time consuming, and can have a negative impact on the users [16].

Offline debiasing techniques used in Web search cannot be applied in our case either, as they rely on biased user feedback collected over the list of results, whereas in our case such feedback exists on the top offer only. The extreme position bias was studied under the framework of Off-policy Evaluation in Contextual Bandits [10, 20, 32]. However, in a setting similar to ours, these techniques will result in a bias error, as explained in Section 2.

It is tempting to use the ranker itself offline for self-evaluation, that is, apply the common approach which uses as estimation the ranker's output over an evaluation dataset. This would be a mistake and would not lead to an accurate evaluation, as self-evaluation amplifies the inherent noise that is part of any model estimation output. We thus propose a novel approach, based on an external *estimator* model, for evaluating the performance of a ranker offline.

We consider purchases as a primary example of user behavioral feedback, as purchase is one of the strongest engagement action a user can take in the shopping scenario. We note that in the case of extreme position bias, standard ranking metrics such as $precision@k$, MRR or DCG are all equivalent to $precision@1$. We thus focus on estimating the *purchase rate*, which is the percentage of purchased top offers out of all top offers. Although the external estimator is based on historical data collected over top offers only, it can accurately predict the performance of a given ranker offline, according to the estimated purchase rate. Using a line of experiments, we demonstrate the accuracy of our external estimator, as opposed to different baselines, including the self-evaluation approach.

Our approach allows for faster iteration of ranking improvements, as it enables to effectively evaluate offline each change involving model features, model architecture, hyper-parameter tuning, and more. Our contribution can be summarized as follows:

- We present a new offline approach for evaluating the performance of a ranker with respect to a behavioral metric, under an extreme position bias. Our approach is based on an external estimator, and a list of requirements that must be satisfied, in order to guarantee its accuracy. We provide a theoretical proof for this guarantee.
- We present several experiments, focusing on purchase rate prediction of different rankers over a segment of voice shopping traffic. In each of these experiments we demonstrate the accuracy of the external estimator.

The rest of the paper is structured as follows. In section 2, we cover related work focusing on the evaluation of ranking models, emphasizing their difference from the new approach we introduce. In Section 3 we present the external estimator framework and the requirements it should satisfy. Then, in Section 4, we establish a theoretical guarantee for the accuracy achieved by the external estimator, given that the aforementioned requirements are met. To ease the reading and provide first a complete overview of our work, the theoretical analysis is completed towards the end of the paper in Section 6, while in Section 5 we present our experiments. We conclude our work in Section 7.

2 RELATED WORK

Traditional approaches rely on expert-annotated relevance-labels for training and evaluating ranking models. While annotator based labels are effective, they are expensive to collect and cannot capture unique user preferences. Modern retrieval systems rely on utilizing behavioral or implicit labels, such as clickthrough data [17] to overcome these limitations. While behavioral feedback is an indispensable resource, it suffers from multiple biases [18, 19, 22, 25, 27], most notably the position bias [9] – as higher ranked documents are more likely to be clicked, even if not more relevant. Debiasing implicit user feedback and training unbiased models has attracted considerable attention.

Click models [5, 9, 11, 27, 28] were originally suggested as a way of taking bias into account in order to accurately estimate query-doc relevance in web search [8]. These models make some assumptions on user browsing behavior and use it to optimize the likelihood of the observed clicks. Click models often depend on multiple observations of the same query–doc pair, thus while they work well on head queries in web search they are invalid for tail queries or personalized search [21].

Another approach for handling presentation bias is result randomization [19, 26, 31]. According to this approach a small fraction of the production traffic is used to estimate the bias by randomly re-ordering or interleaving the (top) results of the retrieval system. This approach, however, requires intervention in the production system and can negatively affect user experience [31].

Recently, an extensive line of work was dedicated to unbiased learning to rank (ULTR) [1, 2, 19, 24, 29–31, 33], where an unbiased ranking model is directly trained with biased user feedback. These works can broadly be divided into two families, online and offline learning. Lately there has been an effort to compare and bridge the gap between offline and online ULTR approaches [2, 16].

Online ULTR algorithms interactively optimize and update a ranking model. They are based on estimating an unbiased model by intervening with the displayed results. For example, the Dueling Bandit Gradient Descent algorithm [34] iteratively optimizes the ranking model by comparing the current ranking model with a perturbed variation at each step. The Pairwise Differentiable Gradient Descent algorithm [23] infers preferences between document pairs from user interactions and constructs an unbiased gradient after each step. The risk of online learning algorithms is that they may harm user experience [16], much like the randomization approach. In addition, experimenting with a new model architecture or a new set of feature requires expensive changes to the production system, which may be discarded if found unhelpful.

Offline ULTR algorithms [1, 29, 33] are trained on historical data and use statistical tools to debias the data. They obtain an unbiased model, usually by estimating the propensity score. The work of [33] presents an LTR approach that addresses both position bias and merit-based fairness. Counterfactual LTR [6, 19] is an offline approach that addresses bias by considering the interactions between observations and clicks, and models the probability of a user observing an item in a displayed ranking.

The offline ULTR approach is close to ours, however, there are several important differences. First, most ULTR works deal with web-search while we focus on voice-based product search which

is inherently different [15]. Second, ULTR algorithms focus on training unbiased ranking models while we focus on evaluation. As such, they cannot provide a clear signal whereas a new proposed model is worth being productized. Third, ULTR does not handle extreme position bias well. While the re-weighting schemes of ULTR methods can theoretically be applied to debias traffic data for offline evaluation, they require some signals collected over the list of result, which are not obtainable.

While most ULTR papers focus on web search, similar approaches have also been applied to eCommerce and product search. For example, Ruocheng et al. [13] address a product-search use-case where results are displayed in 2-dimensional grids rather than 1-dimensional lists. Aslanyan and Porwal [3] exploit the fact that the same query–doc pair changes rank over time in eCommerce settings to derive a simple likelihood function that depends on propensities only. These methods are not applicable to our setting as well, as they also rely on signals collected for different ranks, which do not exist in our setting due to the extreme position bias.

Off-Policy Evaluation in Contextual bandits [10, 20, 32] is analogous to ranker estimation under extreme position bias. In this setting the learner observes a context (e.g features describing the query and a list of offers), takes an action (e.g. selects the top offer) according to some policy (ranker), and observes a reward for the chosen action (e.g. offer purchased or not). The Off-Policy Evaluation problem estimates the value of a target policy using data defined by another policy. For example, the *Direct Method* estimates the reward for a given pair of context and action. This method suffers from large bias due to the assumption that the overall reward can be modeled solely by rewards available for actions selected by the original policy. Several methods were suggested in order to lower the bias error [20, 32], these include Doubly Robust [10] and Inverse Propensity Score (IPS) [14]. While these methods provide better results in many cases, they still suffer from an inherent bias. In fact, in this work we suggest an extension to the Direct Method, and present a set of conditions that if met, guarantee zero position bias error when using the Direct Method.

3 EXTERNAL ESTIMATOR FRAMEWORK

We describe the external estimator framework. We consider the setting in which given a query, a set of results is collected by some retrieval process. Then, a ranker is applied to rank the retrieved results, and present the first one as top offer. The retrieval process and the ranker training process are out of the scope of this work.

As stated before, the goal of our work is offline estimation of the performance of a new ranker with respect to a behavioral metric, in environments with extreme position bias. To simplify presentation, we focus on purchase rate as our metric, but a similar framework holds for any other behavioral score, binary or non-binary.

The core of our framework is a new dedicated external model, different from the ranker, called the *estimator*. The estimator is a probabilistic classifier optimized to predict the purchase probability of an offer[2]. The estimator is applied to estimate the purchase probability of each offer in the set of top ranked offers returned by the ranker (that is, the first ranked offer for each query). Averaging over the estimated purchase probability of these offers we get the estimated purchase rate of the ranker over the respective traffic.

The framework consists of the following steps along with a set of requirements that must be satisfied to guarantee its accuracy:

(1) Train an estimator, i.e. a probabilistic classifier optimized to predict the purchase probability of an offer.
(2) Given a ranker and a sample of the traffic, apply the ranker to obtain the set of top ranked offers – the first ranked offer for each query.
(3) Apply the estimator over the top ranked offers, in order to get their estimated purchase probability.
(4) The estimated purchase rate of the ranker over the traffic is the average of the estimated purchase probabilities of the top ranked offers.

Note that two rankers play a role in the estimation process of the framework. The first is the *original ranker*, used for ranking the traffic that is the source of the implicit user feedback (users purchases) we currently have in the system. The second is the *alternative ranker*, whose purchase rate we wish to estimate. The framework is guaranteed to be accurate (see Section 4) if the following requirements are satisfied:

R1:Top – The estimator's training set should include only top ranked offers. As mentioned before, the main difficulty with behavioral signals in our setting is the extreme presentation bias. By considering only signals that are evaluated on the top position presented to the user, presentation bias is eliminated completely. In addition, we want to predict user behavior on the top ranked results only.

R2:Disjoint – The estimator's training set should be different from the ranker's training set. Failing to do so, may cause some over-fitting between the estimator and the ranker. This would lead to an accumulation of their prediction errors, instead of mutual cancellation (on average), and result in an overestimation of the ranker's performance. In order to eliminate this phenomena, one should decouple both models by avoiding any information leak from the estimator training set to the ranker. This is similar to the way a test set is established in the standard ML framework. In particular, the estimator cannot be used for ranking.

R3:Superset – The estimator's features should include all the features used by the ranker. The estimator can estimate accurately the performance of a ranker, only if it is at least as strong as the original and the alternative rankers. Thus, it should have access to all the features of the rankers. As a result, the estimator is not universal, and fits only a specific set of rankers – the rankers based on part of its features. Concretely, when adding a feature to the ranker, a new estimator should be trained with this feature.

R4:Generalize – The estimator's features should generalize across all offer positions. While most features behave similarly across different positions, some are biased by the rank itself. A natural example are statistic-based features (that is, stats reflecting aggregated user actions over offers), as users are exposed almost solely to the top ranked offer. Such features are not just a function of the offer and the query, but also of the original ranker. Hence, we cannot predict their value if we change the ranker, which makes them unusable, or even harmful, if given as input to the estimator.

[2]In the non-binary case, the estimator is a regressor trained to predict the label.

R5:Context – The estimator's features should include indicative contextual features. Contextual features are features that are independent of the offer itself, and depend only on the context of the query (e.g. query utterance, user, or time). The contextual features should be indicative enough to describe the user behavior, i.e., the purchase probability of each offer is the same for any query with identical contextual feature values. For example, a set of contextual features that provide a good description of the user purchasing behavior may be appropriate. Alternatively, the contextual feature can describe the retrieval process. This can be accomplished using suitable features or by concatenating the feature vectors of all retrieved offers. This vector represents the context, as it is not associated with any specific offer, but rather with the whole set, which is specific to the query. In this case, it is possible to reduce the number of features using dimension reduction techniques. The motivation for this requirement is formally discussed in Section 6.

In Section 4, we provide a theoretical guarantee for the accuracy of the external estimation framework, given that the requirements presented above are met. We complement our analysis in Section 6, where we show that those requirements are also necessary.

4 ESTIMATOR ACCURACY GUARANTEE

We establish a theoretical guarantee for the accuracy achieved by the external estimator framework. We claim that if the five requirements for the estimator, as presented in Section 3, are met, then it estimates the performance of the new ranker accurately (e.g. purchase rate).

In order to precisely define the notion of an accurate estimation, we define the following notations. Let R be the original ranker, and R' be the alternative ranker. Given a traffic distribution $\mathcal{T}$, $R(\mathcal{T})$ and $R'(\mathcal{T})$ represent the distribution of the top offers as ranked by the original and the alternative ranker respectively. Each offer is described by a feature vector X available to the ranker, as well as label L, indicating the user action assuming the offer was top ranked .[3] Note that we only have access to label L for offers that were ranked top by the original ranker R.

Using the above notations, the objective is to estimate the performance of the ranker R', defined by:

$$\mathrm{Perf}(R') \triangleq \mathbb{E}_{X,L\sim R'(\mathcal{T})}[L] \ . \tag{1}$$

Note that when L is a binary label representing purchase, then $\mathrm{Perf}(R')$ is exactly the purchase rate of R'. Let E_S be an estimator trained on a labeled dataset S sampled from $R(\mathcal{T})$. Similarly, S' is a test dataset sampled from $R'(\mathcal{T})$. We denote by $E_S(X)$ the estimation of the ranker given an offer represented by feature vector X. In case the label is binary, E_S is a probabilistic classifier and $E_S(X)$ is the estimated probability of the label being positive. Using the above definitions, the overall estimation is defined as:

$$\mathrm{Est}_S(S') \triangleq \mathbb{E}_{X,L\sim S'}[E_S(X)] \ . \tag{2}$$

Note that the estimation is non-deterministic, since it is influenced by the statistical noise resulting by the sampling process of S and S'. Nevertheless, with high probability it lies within a known confidence interval that diminishes with the sizes of S and S'. The center of this confidence interval can be expressed as the expectation of the ranker's estimation, i.e. $\mathbb{E}_{S\subseteq R(\mathcal{T}),S'\subseteq R'(\mathcal{T})}[\mathrm{Est}_S(S')]$.

Ideally, we want the center of the confidence interval of the estimation to be around the ranker's true performance. The following theorem states that indeed the expected estimation is equal to the actual ranker's performance, if the five requirements hold:

Theorem 1 (informal)**.** *Assuming all 5 requirements of the framework are satisfied, then,*

$$\mathbb{E}_{S\subseteq R(\mathcal{T}),S'\subseteq R'(\mathcal{T})}[\mathrm{Est}_S(S')] = \mathrm{Perf}(R') \ .$$

In Section 6 we formally prove that the requirements are both necessary and sufficient for achieving an accurate estimation.

5 EXPERIMENTS

We validate our external estimator framework by conducting several experiments for estimating the purchase rate of various rankers. The evaluation is performed over a segment of voice shopping traffic, exhibiting extreme presentation bias.

Given a ranker and a traffic segment, the external estimator predicts the purchase rate of the ranker over the traffic, that is, the expected purchase rate given the offers that are top-ranked according to the ranker. The only way to validate the accuracy of the estimation is by an online experiment in which the ranker is applied to the same traffic segment.

5.1 Experiment Setup

We experiment with several types of rankers – a random ranker that randomly shuffles the offers; purchase-optimized rankers, and relevance-optimized rankers that are optimized for *objective* relevance, as judged by external annotators.

The rankers and the external estimator are gradient boosting decision trees (GBDT) models [12], implemented using the XGboost library [7]. They exploit features based on textual similarity, semantic similarity and behavioral-based stats. In addition, the external estimator was augmented with contextual features that represent the users preferences based on their shopping history.[4]

We ran five online A/B testing experiments on segments of voice-shopping traffic both during typical shopping days and after shopping events like Black Friday (which majorly influences shopping habits). In each experiment, we partitioned the data randomly into treatment and control groups. Each group consists of at least several thousands of queries [5]. In the control group the offers were ranked according to the production ranker, while in the treatment group the offers were ranked according to an alternative ranker, whose performance we wish to estimate. We refer to the ranker used in production as the *original ranker*. In each experiment, the estimator's training set was sampled from the control. Thus, the estimator was exposed only to behavioral signals on top ranked offers as ranked by the original ranker.

Table 1 provide information for each experiment: the date, the types of the original and the proposed rankers (random, or metric

[3] The label is close to the notion referred as *relevance* in some of the debiasing literature – it is the property of the offer that makes the user perform the action in case he is exposed to it.

[4] We validated that the contextual features are sufficient for accurate prediction, using past online experiments. We emphasize that alternatively, we could concatenate the query-offers feature vectors as our contextual features (see R5:Context), but this would inflate significantly the number of features.

[5] The groups are disjoint with respect to the users to avoid information leakage.

they are optimized for), and whether it was conducted during typical shopping days or post shopping event.

#	Month	Proposed ranker	Original ranker	Traffic type
1	Nov'20	random	relevance	typical
2	Dec'20	purchase	relevance	shopping-event
3	Mar'21	purchase	relevance	typical
4	Mar'21	relevance	purchase	typical
5	Jun'21	relevance	relevance	typical

Table 1: Experiment settings

We view the first experiment, where we estimate the performance of a random ranker, as the most challenging setting. The reason is, that the characteristics of the top offers presented by the random ranker are very different from the characteristics of top offers presented by the original ranker which is optimized for relevance, as the random ranker may suggest irrelevant offers. The last experiment is the least challenging, as the two rankers target relevance, and are thus similar in nature. Nevertheless, it represents a common use case where incremental changes are made to the ranking algorithm.

During each experiment, we gathered the traffic data of the control group, and used it to train our estimator for predicting purchase probability, by minimizing the log-loss function with respect to the binary purchase label. We remind the reader that the estimator is trained on the traffic we have from the original ranker in production. We used the external estimator, as well as the baseline methods described in the next section, to estimate the purchase rate of the treatment group, and compared it to the actual purchase rate obtained online for the treatment. Demonstrating an accurate prediction would mean, that it can indeed be used to effectively replace online experiments.

5.2 Baselines

We explore several offline approaches for estimating ranker performance under extreme position bias, in which only the top result can get a meaningful signal.

We start with the "biased" approach in which we ignore the presentation bias. In this case we treat any offer that does not have a positive signal as having a negative signal, even if the offer was not presented to the user. Note that Inverse Propensity Score (IPS) [14], treats missing signals as negative and weighs positive signals by the inverse of their probability. Assuming the existence of extreme bias and a deterministic (original) ranker, the "biased" approach described above and IPS are equivalent.

Our second baseline considers only the queries on which the proposed ranker agrees with the original ranker. We note that those are the only queries for which we have a reliable signal with respect to purchase probability. We denote by "agreement estimation" the purchase rate of this traffic.

Finally, in the cases in which the ranker is a classifier estimating purchase probability (experiments #2 and #3 in Table 1), we can use the ranker itself to estimate its own performance. We consider the average ranker estimation over the traffic (that is, over all returned offers) as the "self estimation".

We note that the above baselines are not expected to perform well, as discussed in Section 5.1. However, they provide natural candidates, and are used here to demonstrate the existing gap resolved by our framework. To the best of our knowledge, there are no state-of-art offline baselines for the near-absolute position bias setting considered in this work.

5.3 Results

In Figure 1 we summarize the different experimental results. For each experiment, we present the relative estimation error of the external estimator and the baselines, namely the ratio between the estimation error and the actual purchase rate. In addition, we add the confidence interval at 95% confidence level, which results from the stochastic process of choosing the control and the treatment groups. We can see that the external estimator provides an accurate estimation for each of the experiments. Moreover, it is significantly superior compared to the baselines.

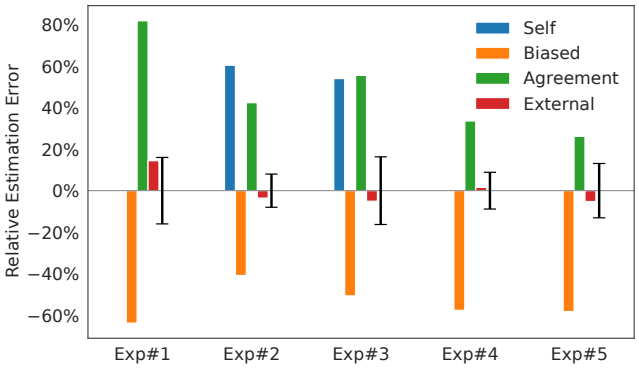

Figure 1: Summary of purchase rate estimation experiments

The self-estimation baseline, where available (experiments #2 and #3), overestimates the purchase rate. This phenomena is due to the inherent noise in ranker estimation (as for any model), in which some of the offers are underestimated and some are overestimated. Naturally, the overestimated offers are more likely to be picked by the ranker. Thus, on average, the ranker's estimation with respect to its top-ranked offers is exaggerated. For a more detailed discussion of this phenomena, see paragraph R2:Disjoint in Section 6.1.

The biased estimation always underestimates the purchase rate. This is not surprising, since under our extreme presentation bias, an offer that was top-ranked by the proposed ranker is evaluated as purchased only if it was top-ranked by the original ranker and actually purchased. This occurs as the biased estimation assigns a negative signal to any offer that has either not been presented to the user (not top-ranked by the original ranker), or not purchased.

The agreement estimation always overestimates the purchase rate. This is expected, since it returns the purchase rate of the traffic where the two rankers agree on the best offer. Those offers tend to be more attractive than offers ranked top by a single ranker, hence we expect that in average their purchase rate is higher.

In all experiments, the external estimator provides the most accurate estimation of the actual purchase rate. In fact, the estimation error is even smaller than the confidence interval imposed

by the sampling process. That is, the external estimator predicts the treatment purchase probability as accurately as measuring the performance of the alternative ranker on a different randomly assigned treatment group. Our offline estimator thus allows to infer the purchase rate of the treatment without presenting it to the user. We conclude that in those experiments, the online experiment could have indeed been replaced with an offline estimation as provided by the external estimator.

6 FORMAL ANALYSIS

In this section we formally analyze the external estimation framework. Specifically, we show that the five requirements presented in Section 3 are both necessary and sufficient to provide an accurate estimation as defined in Section 4.

6.1 Requirements Necessity

We describe five scenarios, one for each requirement. In each scenario the estimator fails to meet one requirement but satisfies all others. We show that in each scenario the estimator fails to produce an accurate estimation, thus proving the necessity of the requirements. The following scenarios are over-simplified and sometimes extreme, but they capture the reasons for which the framework may yield inaccurate estimation if the requirements are not met.

R1:Top. Consider the extreme case in which users purchase any of the offers they are exposed to. Moreover, due to the presentation bias, users are exposed only to the top ranked offer. Assume that each query has 40 offers, and each offer has an ID. The ID is a unique identifier which is drawn uniformly between 0 to 1. Assume furthermore, that the original ranker ranks the offers in ascending order with respect to their ID, and that we want to estimate the purchase rate of ranking by descending order. An accurately generalizing estimator that is not trained on top offers only (contradicting R1:Top), may be trained on offers that do not have the minimal ID. It will learn that such offers are not purchased. Thus, the estimated purchase probability of an offer is exactly the probability that its ID is smaller than the other 39 random IDs. For offers with "large" ID (ID larger than half), the estimated purchase probability will be almost zero (The probability that 39 offers are smaller than half is $0.5^{39} \approx 0$). When applied to offers ranked in descending order, almost 100% of the top ranked offers will have a large ID (the probability that an offer will not satisfy the condition and the maximal ID will be smaller than half is $0.5^{40} \approx 0$), and thus the estimated purchase rate will be 0. This contradicts the fact that according to our assumption, the purchase rate of the alternative ranker is 1.

R2:Disjoint. We assume for simplicity that all offers have the same purchase probability of $p = 30\%$ and each query has $k = 40$ offers. Consider the case in which the training set is extremely large, including many instances of each possible query-offer pair. In this case one can use a statistic-based model that estimates the purchase probability of a query-offer pair as the average purchase labels of the last $m = 10$ times it was offered as top. Note that this model is an unbiased generalizing model, as well as fairly accurate. We want to estimate the purchase probability when using this statistics-based model as a ranker. In this simple case the purchase probability of the alternative ranker is also 30%. In contradiction to R2:Disjoint, in this example we don't require different training sets for the ranker and estimator, and the same model can also be used for estimation. As apposed to the true purchase probability of an offer, the estimated purchase probability is a random variable which depends on the offer's history. The estimation is based on m offers, each purchased with probability p, hence, it is distributed according to the binomial distribution $Bin(p, m)$. Since the ranker is similar to the estimator, for each query with k offers the ranker will rank first the offer with the maximal estimation out of those k. Thus, the estimation of the top offers is a random variable, distributed as the maximum of k binomial random variables. The overall estimation is expected to be around the expectation of this maximum, which is greater than 60%, as can be verified empirically. This is very far from the actual purchase rate of the ranker (30%).

R3:Superset. Assume that for each query there are 40 offers, each offer is chosen to be sold with a significant discount with probability 10%. It can be shown that in this case 98.5% of the queries have at least one discounted offer (the probability of each offer not having a discount is 90% independently, thus the probability of all offers not having discount is $0.9^{40} \approx 1.5\%$). Moreover, assume that none of the estimator's features indicates whether the offer gets a discount. We further assume that the user purchases only every discounted offer and no other). Note that in this case an accurate generalizing model that was trained on offered ranked independently of their discount, will just learn that each offer has a purchase probability of 10%. Hence, 10% will be its overall estimation for every ranker. Now consider a ranker that ranks first the discounted offers. Note that this violates R3:Superset, since the discount information is a ranker's feature that is inaccessible to the estimator. Thus, for each query that has a discounted offer, the ranker will offer it as top, and the offer will be purchased. This results in a purchase probability of 98.5% for the ranker, while the estimation is only 10%.

R4:Generalize. We demonstrate this requirement by considering statistics-based features. We consider an extreme case in which each query has a constant list of offers, and all queries appear in the historical logs. We assume that some of the offers are attractive and are always purchased if presented as top. Specifically, assume each offer is originally ranked according to a unique ID. Assume that 50% of the offers (independently of the IDs) are cheap, where 99% of the cheap offers are attractive. We assume for simplicity that there are 40 offers per query. We want to estimate the purchase rate of a ranker that ranks offers according to their price. It can be shown that almost all of the queries have a cheap offer that will be ranked top by the new ranker. Since 99% of the cheap offers are attractive and as such are purchased if presented as top, the overall purchase rate of this ranker is $\approx 99\%$.

Now, consider an estimator that has access to the ID, price, and purchase statistics of the query-offer pair based on historical data (purchase stats is a biased feature, contradicting R4:Generalize) . Since all queries and offers appeared before, all the top ranked attractive offers have positive purchase statistics and were indeed purchased, while the unattractive offers have zero purchase statistics, and indeed were not purchased. Thus, an estimator trained on the top offers learns that the purchase-statics feature accurately predicts whether the offer is purchased, and chooses to consider this feature only. Predicting according to the price is less accurate (99%) than using purchase-statistics (100%), and thus the estimator

learns to ignore the price signal. Note that for each query only one offer out of 40 was originally presented as top, and can have a positive purchase statistic. Thus, the estimator will estimate that less than 1/40 of the offers provided by the new ranker are purchased, resulting in an estimation of at most 2.5%.

R5:Context. We consider the case where the purchase decision depends solely on the context, however the estimator has no contextual features – contradicting R5:Context. Assume there are two user populations – explorers and buyers. The explorers population generates 99% of the traffic, and the buyers generate the remaining 1%. The explorers initiate "exploration queries" that are characterized by offers with a varied range of prices (uniformly distributed between 0 to 100$). The explorers never purchase anything. On the other hand, the buyers initiate "buy queries" associated with low priced offers (lower than 50$) and always purchase the top ranked offer (regardless of its features). We assume that each query has 40 offers. Note that the purchase probability of any ranker is 1% (the ratio of the buyers' generated traffic).

Now, assume that the original ranker ranked the offers in a descending order according to their price, and we want to estimate the purchase probability of an alternative ranker that ranks the offers in an ascending price order. Note that for an exploration query it is very unlikely that all offers have a low price (since an offer's price for the exploration queries is uniform between 0 to 100$, its probability to be below 50$ is 0.5 - thus the probability that all 40 offers will have price lower than 10$ is $0.5^{40} \approx 0$). However, for a buy query, this is always the case. Thus, the original ranker almost always suggests an offer with a high price (>50$) to the explorers and a low-priced offer to the buyers. Any accurate generalizing estimator will learn that offers are purchased if and only if they have a low price. Consider the alternative ranker that ranks low priced offers at the top. It can be verified that for almost all of the exploration queries and for all of the buy queries there will be a low-priced offer ranked top (since an offer's price for the exploration queries is uniform between 0-100, its probability to be above 50$ is 0.5 – thus, the probability that all 40 offers will have price greater than 50$ is $0.5^{40} \approx 0$). The estimator will estimate that these low-priced offers are purchased, yielding an overall estimation of 100%, while the real purchase rate remains 1%.

6.2 Requirements Sufficiency

We prove that the five requirements stated in Section 3 are sufficient to guarantee the accuracy of the external estimator.

We formally define our setting. In order to define a model that can generalize on new unseen examples, we view the queries and the offers as part of a stochastic process. As common in this case, capital letters denote random variables, while lowercase represents their values. Let $\mathcal{T}$ represents the traffic distribution, i.e., a distribution over query instances. Each query instance Q drawn from $\mathcal{T}$ defines a vector, denoted by $C(Q) \in \mathbb{R}^m$, describing the distribution over offers (following the query) and the value of each feature in the contextual feature set C. We consider the offers as the output of a stochastic process, as their availability and respective label depends on unpredictable factors that are out of our control. Each offer is a pair $(z, \ell) \in \mathbb{R}^n \times \mathbb{R}$, defined by z, a vector of offer features, and label ℓ representing the action performed by the user given that this offer was ranked as top. The full feature vector is defined as $x = (c, z)$, where c is a vector of contextual features and z is a vector of offer features. We abuse the notation and represent an offer sampled for query Q by $(Z, L) \sim Q$. To ease the reading, we shorten the terms "traffic distribution" and "query instance" into *traffic* and *query*.

A ranker R is an algorithm which, given a list of offers represented by the feature vector z, selects a single offer as the top offer. Given traffic $\mathcal{T}$ we denote the distribution of offers selected by R with respect to queries sampled from $\mathcal{T}$ by $(X, L) \sim R(\mathcal{T})$. Note, for any traffic $\mathcal{T}$ we fix the set of features (both contextual and offer features), as defined above. Thus, from now on we assume all rankers and learning algorithms are restricted to this features set (this is essential to guarantee R3:Superset defined in Sec. 3). Moreover, we denote by $S \subseteq R(\mathcal{T})$ a dataset consisting of top ranked offers sampled from $R(\mathcal{T})$.

Consider two rankers, the original ranker R and an alternative ranker R'. Given a learning algorithm E, let E_S be the external estimator that E outputs when trained on the labeled training set $S \subseteq R(\mathcal{T})$. The objective of the external estimator is to accurately estimate the performance of R'.

Definition 2. Let $\mathcal{T}$ be some traffic, R a ranker and E a learning algorithm. We say that a learning algorithm E on $R(\mathcal{T})$ estimates distribution $R'(\mathcal{T})$ *accurately* if

$$\mathbb{E}_{S \subseteq R(\mathcal{T}), S' \subseteq R'(\mathcal{T})}[\mathbb{E}_{X,L \sim S'}[E_S(X)]] = \mathbb{E}_{X,L \sim R'(\mathcal{T})}[L]. \quad (3)$$

Note that $\mathbb{E}_{X,L \sim S'}[E_S(X)]$ is the overall estimation of E_S, the estimator returned by E when trained on a subset $S \subseteq R(\mathcal{T})$. Also, $\mathbb{E}_{X,L \sim R'(\mathcal{T})}[L]$ is the performance of ranker R'. Thus, Equation (3) requires the estimator to accurately estimate (in expectation) the performance of R' (equivalent to equation of Theorem 1 in Sec. 4).

Our theoretical analysis proves that a learning algorithm E estimates $R'(\mathcal{T})$ accurately if the five requirements stated in Section 3 are met. The first three requirements follow from the definition of the estimator and its objective. (i) The training data $S \subseteq R(\mathcal{T})$ contains only top ranked offers (R1:Top). (ii) The fact that in Equation (3) R' is fixed before S was sampled implies it did not have access to S (R2:Disjoint). (iii) E_S has access to the feature vector of x, the same set of features utilized by R and R' (R3:Superset).

Next, we consider requirement R4:Generalize, which is translated to a property of a learning algorithm E. The learning algorithm generates an estimator which is expected to model the expected label of an offer sampled from $R(\mathcal{T})$ by learning from a training set $S \subseteq R(\mathcal{T})$. This property is given formally in the next definition.

Definition 3. A learning algorithm E is *generalizing* distribution $R(\mathcal{T})$, if for each $x \in \mathbb{R}^n$ such that $\Pr_{X,L \sim R(\mathcal{T})}[X = x] > 0$, it holds

$$\mathbb{E}_{S \subseteq R(\mathcal{T})}[E_S(x)] = \mathbb{E}_{X,L \sim R(\mathcal{T})}[L \mid X = x] .$$

Observe that noise in the sampling process of S may affect the learning process. In practice the prediction of E_S falls within some confidence interval which diminishes as $|S|$ grows.

Lastly, we consider requirement R5:Context. The set of contextual features C are independent of any specific offer. They may include for example information about the query utterance, the user, and the time. While the generalizing property allows the estimator to estimate user behavior over the entire traffic according to the original ranker, the contextual features are needed to extend

this ability for estimating accurately also the performance of an alternative ranker.

We achieve this goal by requiring the set of contextual features C to obey at least one of two conditions. The first condition states that each pair of queries with identical contextual features have an identical offer distribution as well. As the ranker cannot differentiate between the two queries, it will select the same top offer for these queries, allowing the learning algorithm E to treat these queries as similar and learn the respective user behavior. The second condition states that the expected label of an offer can be described as a function of the offer features for each pair of queries with the same contextual features. This clearly means that the label can be described as a function of all features, contextual and offer features, so a learning algorithm utilizing these features can generate a good estimator. The following definition describes this notion formally.

Definition 4. Given a set of contextual features C, an offer feature vector $z \in \mathbb{R}^n$, and two queries $Q, Q' \sim \mathcal{T}$ such that $C(Q) = C(Q')$, we say that C is *sufficient* if at least one of the two conditions holds:

(1) $\Pr_{Z,L\sim Q}[Z = z] = \Pr_{Z,L\sim Q'}[Z = z]$
(2) $\mathbb{E}_{Z,L\sim Q}[L \mid Z = z] = \mathbb{E}_{Z,L\sim Q'}[L \mid Z = z]$.

The sufficient context property implies that the contextual features suffice to identify the difference between queries. We observe that sufficient context can always be guaranteed by selecting all offer features for the whole matching set of offers. This means that for any two queries Q and Q' such that $C(Q) = C(Q')$ the matching sets are identical. Thus, Condition (1) in the above definition holds as the probability that an offer with feature vector z is in the matching set of Q and Q' is identical and deterministic. Alternatively, sufficient context can be achieved by having indicative features that capture user behavioral preferences, according to Condition (2).

Given the above definitions we can now provide a formal phrasing of Theorem 1 (see Section 4).

Theorem 1. *Let $\mathcal{T}$ be a traffic distribution with a set of contextual features C and let R and R' be the original and the alternative ranker respectively. If C is sufficient, and learning algorithm E generalizes $R(\mathcal{T})$, then E on $R(\mathcal{T})$ estimates $R'(\mathcal{T})$ accurately.*

The proof of Theorem 1 follows from the two next constructive lemmas. We first define for any offer distribution $\mathcal{D}$ the conditional distribution $\mathcal{D}_x$ of L given $X = x$, where X, L are drawn from $\mathcal{D}$. I.e., $\mathbb{E}_{L\sim\mathcal{D}_x}[L] = \mathbb{E}_{X,L\sim\mathcal{D}}[L \mid X = x]$. The following definition is necessary for presenting the lemmas.

Definition 5. We say that two offer distributions $\mathcal{D}$ and $\mathcal{D}'$ are *label consistent* if for any feature vector $x \in \mathbb{R}^n$ such that $\Pr_{X,L\sim\mathcal{D}}[X = x], \Pr_{X,L\sim\mathcal{D}'}[X = x] \neq 0$, it holds that

$$\mathbb{E}_{L\sim\mathcal{D}_x}[L] = \mathbb{E}_{L\sim\mathcal{D}'_x}[L] \ .$$

Lemma 6. *Let $\mathcal{D}$ and $\mathcal{D}'$ be label consistent distributions. If a learning algorithm E is generalizing $\mathcal{D}$, then it estimates $\mathcal{D}'$ accurately.*

Proof.

$$\begin{aligned}
\mathbb{E}_{X,L\sim\mathcal{D}'}[L] &= \mathbb{E}_{X,L\sim\mathcal{D}'}\mathbb{E}_{L'\sim\mathcal{D}'_X}[L'] \\
&= \mathbb{E}_{X,L\sim\mathcal{D}'}\mathbb{E}_{L'\sim\mathcal{D}_X}[L'] \\
&= \mathbb{E}_{X,L\sim\mathcal{D}'}\mathbb{E}_{S\subseteq\mathcal{D}}E_S(X) \\
&= \mathbb{E}_{S'\subseteq\mathcal{D}'}\mathbb{E}_{X,L\sim S'}\mathbb{E}_{S\subseteq\mathcal{D}}E_S(X) \\
&= \mathbb{E}_{S\subseteq\mathcal{D},S'\subseteq\mathcal{D}'}\left[\mathbb{E}_{X,L\sim S'}[E_S(X)]\right].
\end{aligned}$$

The first and forth equalities follow from the law of total expectation. The second equality follows from label-consistency of the distributions. The third equality follows from the generalizing property of the learning algorithm. Lastly, the fifth equality follows linearity of expectation. □

Theorem 1 follows from Lemma 6 and the following lemma.

Lemma 7. *Let $\mathcal{T}$ be a traffic distribution with contextual feature set C. If C is sufficient, then for any two rankers R and R', it holds that $R(\mathcal{T})$ and $R'(\mathcal{T})$ are label consistent.*

Proof. See Appendix A □

7 DISCUSSION

In this work, we presented a novel approach for predicting offline the performance of a new ranker, with respect to a behavioral metric, in a setting of extreme position bias. The core of our approach is an external estimator framework, which incorporates several requirements that have to be met. We presented these requirements, proving both their necessity, as well as the guaranteed accuracy of the estimator given that the requirements hold. Using a line of experiments, we demonstrated that we can accurately predict offline the purchase rate of various rankers. In those experiments, the estimator was validated online, by comparing its prediction to the actual purchase rate achieved by each ranker when being exposed to users.

We note that while our solution can replace the need for continuous online experiments when developing a new ranker, it cannot, and is not intended to eliminate online experiments completely. For example, production constraints like latency, are not considered by our framework. However, considering the natural continuous work invested in developing and improving a new ranker, our solution can be used to eliminate a series of online experiments, that were needed until today, for any change of features, model architecture, or hyper-parameter tuning.

We view our work as the first and most significant step of devising and fully characterizing this new proposed framework. However, our solution still suffers from several drawbacks, left for future work. One remaining challenge is validating the external estimator completely offline, as at least one online experiment is recommended to validate that the overall external estimation system is tuned appropriately. Another challenge is being able to estimate a ranker that uses position-biased features, as according to our requirements, they cannot be used as is by the external estimator.

Finally, it would also be interesting to adapt our solution to cases that suffer from a position bias, but not an extreme one. Exploring if it has some benefit for those cases and comparing it to state-of-art debiasing baselines, could yield a new offline debiasing technique.

REFERENCES

[1] 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*. 385–394.

[2] Qingyao Ai, Tao Yang, Huazheng Wang, and Jiaxin Mao. 2021. Unbiased Learning to Rank: Online or Offline? *ACM Transactions on Information Systems (TOIS)* 39, 2 (2021), 1–29.

[3] Grigor Aslanyan and Utkarsh Porwal. 2019. Position bias estimation for unbiased learning-to-rank in ecommerce search. In *International Symposium on String Processing and Information Retrieval*. Springer, 47–64.

[4] David Carmel, Elad Haramaty, Arnon Lazerson, Liane Lewin-Eytan, and Yoelle Maarek. 2020. Why do people buy seemingly irrelevant items in voice product search? On the relation between product relevance and customer satisfaction in ecommerce. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 79–87.

[5] 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*. 1–10.

[6] Mouxiang Chen, Chenghao Liu, Jianling Sun, and Steven C.H. Hoi. 2021. Adapting Interactional Observation Embedding for Counterfactual Learning to Rank. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 285–294.

[7] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. ACM, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

[8] 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.

[9] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*. 87–94.

[10] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly robust policy evaluation and learning. *arXiv preprint arXiv:1103.4601* (2011).

[11] Georges E Dupret and Benjamin Piwowarski. 2008. A user browsing model to predict search engine click data from past observations.. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. 331–338.

[12] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[13] Ruocheng Guo, Xiaoting Zhao, Adam Henderson, Liangjie Hong, and Huan Liu. 2020. Debiasing grid-based product search in e-commerce. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2852–2860.

[14] Daniel G Horvitz and Donovan J Thompson. 1952. A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association* 47, 260 (1952), 663–685.

[15] Amir Ingber, Arnon Lazerson, Liane Lewin-Eytan, Alexander Libov, and Eliyahu Osherovich. 2018. The Challenges of Moving from Web to Voice in Product Search. In *Proc. 1st International Workshop on Generalization in Information Retrieval (GLARE 2018). http://glare2018. dei. unipd. it/paper/glare2018-paper5. pdf.*

[16] Rolf Jagerman, Harrie Oosterhuis, and Maarten de Rijke. 2019. To model or to intervene: A comparison of counterfactual and online learning to rank from user interactions. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 15–24.

[17] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. 133–142.

[18] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)* 25, 2 (2007), 7–es.

[19] 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*. 781–789.

[20] Lihong Li, Rémi Munos, and Csaba Szepesvári. 2015. Toward minimax off-policy value estimation. In *Artificial Intelligence and Statistics*. PMLR, 608–616.

[21] Jiaxin Mao, Zhumin Chu, Yiqun Liu, Min Zhang, and Shaoping Ma. 2019. Investigating the Reliability of Click Models. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*. 125–128.

[22] Jiaxin Mao, Cheng Luo, Min Zhang, and Shaoping Ma. 2018. Constructing click models for mobile search. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 775–784.

[23] 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*. 1293–1302.

[24] Harrie Oosterhuis and Maarten de Rijke. 2020. Taking the Counterfactual Online: Efficient and Unbiased Online Evaluation for Ranking. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*. 137–144.

[25] Maeve O'Brien and Mark T Keane. 2006. Modeling result-list searching in the World Wide Web: The role of relevance topologies and trust bias. In *Proceedings of the 28th annual conference of the cognitive science society*, Vol. 28. Citeseer, 1881–1886.

[26] Filip Radlinski and Thorsten Joachims. 2006. Minimally invasive randomization for collecting unbiased preferences from clickthrough logs. In *Proceedings of the national conference on artificial intelligence*, Vol. 21. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 1406.

[27] Chao Wang, Yiqun Liu, Min Zhang, Shaoping Ma, Meihong Zheng, Jing Qian, and Kuo Zhang. 2013. Incorporating vertical results into search click models. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. 503–512.

[28] Hongning Wang, ChengXiang Zhai, Anlei Dong, and Yi Chang. 2013. Content-aware click modeling. In *Proceedings of the 22nd international conference on World Wide Web*. 1365–1376.

[29] Nan Wang, Zhen Qin, Xuanhui Wang, and Hongning Wang. 2021. Non-Clicks Mean Irrelevant? Propensity Ratio Scoring As a Correction. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 481–489.

[30] 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*. 115–124.

[31] 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*. 610–618.

[32] Yu-Xiang Wang, Alekh Agarwal, and Miroslav Dudık. 2017. Optimal and adaptive off-policy evaluation in contextual bandits. In *International Conference on Machine Learning*. PMLR, 3589–3597.

[33] Himank Yadav, Zhengxiao Du, and Thorsten Joachims. 2021. Policy-Gradient Training of Fair and Unbiased Ranking Functions. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 1044–1053.

[34] Yisong Yue and Thorsten Joachims. 2009. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 1201–1208.

[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 *Proceedings of the 19th International Conference on World Wide Web (WWW '10)*. 1011–1018.

A PROOF OF LEMMA 7

Proof. We will show that for any feature vector $x = (c, z)$ the expression $\mathbb{E}_{L\sim R(\mathcal{T})_x}[L]$ is independent of the ranker R. This implies that $\mathbb{E}_{L\sim R(\mathcal{T})_x}[L] = \mathbb{E}_{L\sim R'(\mathcal{T})_x}[L]$, as required.

Fix some $x = (c, z)$ and query Q such that $C(Q) = c$ and $\Pr_{Z,L\sim Q}[Z = z] > 0$. We consider ranker R as a function that receives k feature vectors as well as contextual features, and chooses an index of one offer.

$$\begin{aligned}\mathbb{E}_{L\sim R(Q)_z}[L] &= \mathbb{E}_{Z,L\sim R(Q)}[L \mid Z = z] \\ &= \mathbb{E}_{S\subseteq Q}[L_{R(S,c)} \mid Z_{R(S,c)} = z] \\ &= \mathbb{E}_{Z,L\sim Q}[L \mid Z = z] \\ &= \mathbb{E}_{L\sim Q_z}[L]\,.\end{aligned} \tag{4}$$

The first,second and last equalities follow from the definitions. The third equality follows from the fact that for index $i = R(S, c)$, the expectation is calculated on L_i under the conditions that (i) $R(Z_1, \ldots, Z_{i-1}, z, Z_{i+1}, \ldots, Z_k, c) = i$, and that (ii) $Z_i = z$. Since L_i is independent of $\{Z_j\}_{j\neq i}$, condition (i) can be discarded.

There are two conditions under which C is sufficient, as defined in Definition 4. If Condition (1) holds, we get that $\mathbb{E}_{L\sim R(\mathcal{T})_x}[L]$ is independent of R, as follows:

$$\begin{aligned}\mathbb{E}_{L\sim R(\mathcal{T})_x}[L] &= \mathbb{E}_{X,L\sim R(\mathcal{T})}[L \mid X = x] \\ &= \mathbb{E}_{Q\sim\mathcal{T},(Z,L)\sim R(Q)}[L \mid Z = z, C(Q) = c] \\ &= \mathbb{E}_{Q\sim\mathcal{T},L\sim R(Q)_z}[L \mid C(Q) = c] \\ &= \mathbb{E}_{Q\sim\mathcal{T},L\sim Q_z}[L \mid C(Q) = c]\,,\end{aligned}$$

where the first two equalities follow from the definitions. The third equality follows from Condition (1), as all queries with the same context are associated with the same distribution. Thus, the event $Z = z$ poses no additional constraint on the choice of Q. The last equality follows from Equation (4).

Finally, we show that $\mathbb{E}_{L\sim R(\mathcal{T})_x}[L]$ is independent of R, assuming Condition (2) holds (see Definition 4).

$$\begin{aligned}&\mathbb{E}_{L\sim R(\mathcal{T})_x}[L] \\ &= \mathbb{E}_{X,L\sim R(\mathcal{T})}[L \mid X = x] \\ &= \mathbb{E}_{Q\sim\mathcal{T},Z,L\sim R(Q)}[L \mid C(Q) = c, Z = z] \\ &= \mathbb{E}_{Q\sim\mathcal{T},(Z,L)\sim R(Q)}\left[\mathbb{E}_{(Z',L')\sim R(Q)}[L' \mid Z' = z] \mid C(Q) = c, Z = z\right] \\ &= \mathbb{E}_{Q\sim\mathcal{T},(Z,L)\sim R(Q)}\left[\mathbb{E}_{L'\sim R(Q)_z}[L'] \mid C(Q) = c, Z = z\right] \\ &= \mathbb{E}_{Q\sim\mathcal{T},(Z,L)\sim R(Q)}\left[\mathbb{E}_{L'\sim Q_z}[L'] \mid C(Q) = c, Z = z\right] \\ &= \mathbb{E}_{Q\sim\mathcal{T},(Z,L)\sim R(Q)}[\ell \mid C(Q) = c, Z = z] = \ell.\end{aligned}$$

The first, second and fourth equations follow from the definitions. The third equality follows from the law of total expectation and the last equality follows from Equality (4). □