Toward Predicting the Outcome of an A/B Experiment for Search Relevance

Lihong Li [*]
Microsoft Corp
One Microsoft Way
Redmond, WA 98052
lihongli@microsoft.com

Jin Young Kim [†]
Microsoft Corp
One Microsoft Way
Redmond, WA 98052
jink@microsoft.com

Imed Zitouni
Microsoft Corp
One Microsoft Way
Redmond, WA 98052
izitouni@microsoft.com

ABSTRACT

A standard approach to estimating online click-based metrics of a ranking function is to run it in a controlled experiment on live users. While reliable and popular in practice, configuring and running an online experiment is cumbersome and time-intensive. In this work, inspired by recent successes of offline evaluation techniques for recommender systems, we study an alternative that uses historical search log to reliably predict online click-based metrics of a *new* ranking function, without actually running it on live users.

To tackle novel challenges encountered in Web search, variations of the basic techniques are proposed. The first is to take advantage of diversified behavior of a search engine over a long period of time to simulate randomized data collection, so that our approach can be used at very low cost. The second is to replace exact matching (of recommended items in previous work) by *fuzzy* matching (of search result pages) to increase data efficiency, via a better trade-off of bias and variance. Extensive experimental results based on large-scale real search data from a major commercial search engine in the US market demonstrate our approach is promising and has potential for wide use in Web search.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Experimentation; Performance

Keywords

Web search; information retrieval; experimentation; evaluation; contextual bandits

[*]Joint first authorship
[†]Joint first authorship

1. INTRODUCTION

Many Web-based services are by nature systems that interact with users. A search engine, for example, starts with a query issued by a user, returns a ranked list of documents, and observes user feedback often in the form of clicks and document consumption. Other examples are advertising and recommender systems that suggest one or multiple items to a user and in return receive feedback such as whether the user adopts/purchases the item(s) or not. Very often, such user feedback strongly indicates quality of the system. In advertising and recommender systems, for instance, it is natural to prefer a system with high adoption/conversion rate and possibly the revenue associated with it (*e.g.*, [7, 20]).

In Web search ranking, the problem we focus on in this work, useful signals can be extracted from user (click) feedback to infer whether a search event is successful or not. Commonly used examples include click-through rate (CTR), time to click on the search result page (SERP), and mean reciprocal of click positions, *etc.* In this paper, we consider the problem of estimating the average value of such pre-defined *online* signals across user visits.

Although human judgment labels are commonly used in practice to optimize such Web-based systems, they are inherently approximations to the quality of end user experience. This limitation suggests the need and importance to measure and optimize online metrics, even though label-based optimization can be a very useful starting point.

Often the metric of interest depends on user feedback, and hence on system output as well.[1] Therefore, one usually does not know what the user feedback *would have been* if the system output changes, unless a highly accurate user model is available. Such an observation leads to a fundamental challenge when estimating a metric of a search engine: based on search log collected by running a version of the engine on users in the past, it is often difficult or even impossible to reliably estimate a metric of a *modified* engine.

The standard approach to this problem is to run an A/B experiment(*e.g.*, [16]), also known as a randomized controlled experiment: incoming users are randomly split into two groups, where the *control* group is served by a baseline system and the *treatment* group by a variant. After running the experiment for a period of time (often ranging from one week to a few weeks), online metrics of the two systems are compared, and the one with higher metric values wins. While A/B experiments are highly valuable for making data-

[1]In this paper, we focus on metrics that depend on user feedback. Metrics that do *not* depend on user feedback are typically easy to estimate from historical data.

driven decisions to improve Web-based services, they are expensive for a few reasons. First, the turn-around time is long: it usually takes days or even weeks before comparisons of the control and treatment engines start to be statistically significant. Second, the engineering efforts are nontrivial to run a flight: as the new ranker will serve live users, substantial care has to be taken to ensure the ranker does not lead to catastrophic results. Third, search traffic is a limited resource. Quite often, multiple experiments compete for search traffic, and it is challenging to run several experiments at the same time.

Offline evaluation is therefore of great, practical interest. It aims to predict online metrics of a search engine *without* actually running it on live users, thus addressing all three difficulties above and substantially improving experimentation agility and quality. Offline evaluation has been studied and successfully applied to recommendation and advertising (*e.g.*, [3, 7, 17, 21, 27]), using statistical tools from causal inference. The key of these techniques is to first collect randomized data and then properly correcting sampling bias when evaluating a new system.

In the context of Web search ranking, however, the problem is even more challenging due to the exponentially large number of possible SERPs for any given query. First, data collection is very expensive since all possible SERPs will need to be shown with nonzero probability, a condition often required by reliable offline evaluation methods [3, 21]. Second, offline evaluation results' variance roughly depends reciprocally on the number of possible SEPRs (*c.f.*, Section 5) and can be too large in practice to be useful.

Our contributions are three-fold. First, to the best of our knowledge, our work is the first to apply causal inference techniques to offline-evaluate metrics of a Web search ranker. Second, in order to address novel challenges encountered in Web search, two variations of the basic offline evaluation technique are proposed and studied, making the method realistic for this application domain. Third, we demonstrate the proposed solution is promising through extensive experiments using large-scale real search log from a commercial search engine.

2. PRELIMINARIES

This section provides notation, setup, and related work.

2.1 Notation

Here, we formulate the ranking problem as a contextual bandit, a popular model in machine learning and statistics, for capturing interaction between user and the search ranker. In a contextual bandit [18], the system repeatedly observes contextual information that is independent and identically distributed (IID). For each observation, it takes an *action* and receives a possibly randomized *reward*. The reward depends on contextual information and the selected action, and rewards for other (not selected) actions are not observed.

In the context of Web search ranking, we denote the set of contextual information by $\mathcal{Q}$—the set of distinct queries, and for any $q \in \mathcal{Q}$, let $\mathcal{A}_q$ be the set of actions—the set of possible SERPs for query q. A ranker, denoted π, selects a SERP $a \in \mathcal{A}_q$. For convenience in this paper, we consider the general case where π is stochastic. That is, for each q, it computes a multinomial distribution $\pi(\cdot|q)$ over $\mathcal{A}_q$, so that $a \in \mathcal{A}_q$ is chosen with probability $\pi(a|q)$. In practice, π is often a deterministic function [4, 29]. For this important special case, we abuse notation a bit using $\pi(q)$ to denote the unique action π selects for query q. In the contextual bandit literature, π is often considered as a *policy*. In the rest of the paper, the pairs policy/ranker, context/query, and action/SERP are used interchangeably.

The interaction between ranker and user proceeds in a round-by-round fashion as follows: at round t,

1. A user visits the search engine and submits a query $q \in \mathcal{Q}$, drawn IID from some unknown distribution D.
2. The ranker selects the action $a = \pi(q) \in \mathcal{A}_q$; that is, it shows the SERP a to the user.
3. The user consumes the SERP and based on her feedback a numerical reward signal $r \in \mathbb{R}$ is computed.

A few notes are in place. First, we assume $r \in [0, R]$ for some known constant $R > 0$, since any bounded reward signal can be easily shifted and rescaled to lie within this range. Second, the distribution that q is drawn from can be time-dependent, to reflect daily and weekly population changes of queries encountered by a search engine. Our approach (and its associated guarantees) is not affected when the distribution becomes time-dependent.

The possibly stochastic reward signal measures how successful the system meets the user's need in that round. Its distribution depends on both the query and the action, and is not known (and usually difficult to model accurately). Examples of rewards are found in the next subsection. Our goal in this work is to estimate the average value of a *pre-defined* reward function when serving users with a *new* ranker π:

$$v(\pi) := \mathbb{E}_{q\sim D, a\sim\pi(\cdot|q)}[r] = \mathbb{E}_q\left[\sum_{a\in\mathcal{A}_q} \pi(a|q)\mathbb{E}[r|q,a]\right] . \quad (1)$$

In words, the metric is the average reward obtained by running the (possibly stochastic) policy on queries encountered by a search engine over a long period of time. If π is deterministic, the definition may be simplified as

$$v(\pi) := \mathbb{E}_{q\sim D}[r|q, \pi(q)] . \quad (2)$$

2.2 Example Metrics

Many important metrics used in practice can be obtained by defining appropriate rewards. For example, if we define r to be 1 if there is a click on the SERP and 0 otherwise, then $v(\pi)$ is the per-impression click-through rate (CTR) of the ranker. One can also easily define similar metrics for clicks that satisfy certain desired conditions, or incorporate monetary information to measure revenue.

Another popular type of metrics attempts to quantify how soon a user finds the information she needs. An example is the amount of time it takes for a user to click (or take any other pre-defined user action) after she submits a query. A related one is the reciprocal rank of the document that the user clicks on the SERP.

In our experiment, we focus on estimating the fraction of successful search activities of a ranking function. This requires defining what it means by a success in a Web search scenario, which typically depends on user clicks on the documents, amount of time spent on clicked documents (e.g., [5]), positions of clicked documents, and perhaps other information. For sensitivity reasons, the precise definition of success cannot be revealed, but it suffices to note that this metric is nothing but a concrete way to define the reward signal in the contextual bandit framework. A reader can simply interpret it as CTR of the whole SERP (not individual documents on the SERP), without sacrificing understanding of the technical details of this work.

2.3 Related Work

A lot of work has been done to evaluate rankers in information retrieval and Web search [25]. The dominant approach is to collect relevance labels from human judges for a collection of query-document pairs, which are then used to compute *offline* metrics like mean average precision [1] and normalized discounted cumulative gains [14]. For these metrics, the label information used to compute them is independent of user feedback, unlike the online metrics we focus on here. Such a non-interactive nature makes the judgment-based approach a relatively inexpensive and easy-to-use solution for ranking function evaluation.

However, the non-interactive nature also causes limitations, as pointed out by several authors [2, 28], as they are approximations of the quality of end user experience. Consequently, in industry, people have also measured various online metrics that indicate whether a search activity is successful or not. These metrics are the ones we try to estimate in this paper, since they depend on user feedback. The standard way is to run an A/B experiment (*e.g.*, [1, 16]) to directly measure online metrics, or to run an interleaving experiment that gives directional evaluation results [6, 24]. In contrast to existing methodology, we aim to rely on historical data to perform *offline* estimates of them.

Recent, some authors applied offline evaluation techniques to estimate online metrics for a specific type of experiments, namely interleaving [12, 13]. Such interesting work assumes availability of properly randomized, interleaved SERPs, and uses such data to predict outcomes of interleaving experiments for a new pair of rankers. In contrast, our focus here is for more general online experiments and can be directly applied to a wider range of online metrics (such as those involving revenue), with minimal need for data collection.

An alternative to A/B testing is to build a user click model and then use it to simulate user clicks on results produced by a new ranker. Much progress has been made in the past that led to refined models that deepen our understanding of user engagement (*e.g.*, [8, 10, 11]). Furthermore, some of these click models can be used to define useful offline metrics that have positive correlations with certain online metrics [9]. Here, we aim to estimate the online metrics themselves, instead of finding an proxy metric.

As explained in Section 3, the approach we take here is closely related to previous offline evaluation work for recommender and advertising systems based on contextual bandits [18, 21], and to causal inference in statistics [17] as well. Our work is along the same line, although we make two critical adjustments to enable it to work for Web search. The first is the idea of using "natural exploration" to replace active exploration [26], taking advantage of inherent diversity of a search engine. The second is to use approximate action matching for a better bias/variance trade-off.

3. BASIC TECHNIQUES WHEN RANDOMIZATION DATA ARE AVAILABLE

If one is able to run the policy on live users (that is, *online* evaluation), as in an A/B experiment, it is straightforward to estimate the policy value: we can simply run the policy to serve users, observe user feedback, compute the reward for each impression, and then average the reward. By the law of large numbers, this empirical average will converge to the true metric value with probability one.

However, the fundamental challenge in offline evaluation lies in the following *counterfactual* nature: for a particular query, if the ranker produces a different ranking than the one by the data-collection ranker, it is difficult, if not impossible at all, to reliably predict user feedback in response to the new ranking result. One therefore simply does not have the reward information for evaluating a new ranker.

As studied in previous work [21, 3], the key to reliable offline evaluation is to use randomization during data collection. Specifically, we may run an experiment on a small fraction of users that is dedicated to randomized data collection: when a user issues a query q, instead of always following production ranker π_0 to return a SERP $\pi_0(q)$, we add some pre-defined noise to π_0 so that a random SERP is shown to the user, with reward observed and recorded. Note that such randomization occurs on the impression level, so if the same user issues the same query a second time, she may see a different SERP. The process can be adapted if one is interested in user-level or search-session-level metrics.

When the data-collection experiment finishes, we will have a set of "exploration" data $\mathcal{D} = \{\langle q_i, a_i, p_i, r_i \rangle\}_{1\le i\le n}$, where i indexes a distinct impression, q_i is the query issued by user in that impression, a_i is a ranking list (say, top 10 document list), $p_i = p(a_i|q_i)$ is the probability of showing a_i for q_i in the data-collection flight, and r_i is the resulting click-based metric (say, search-success-or-not, time to click). In the literature, p_i is also known as the *propensity score*.

Now, given *any* new, possibly randomized ranker π, we can estimate its value by the following estimator [3, 26]:

$$\widehat{v}_1(\pi) := \frac{1}{n}\sum_{i=1}^{n}\frac{\pi(a_i|q_i)}{p_i}r_i\,.$$

The intuition behind the estimator is that, for any given query q, some actions have higher selection probability during data collection, so they are over-represented in the data. The division by propensity scores is to correct such a sampling bias. In fact, it can be easily shown that the estimator is unbiased in the following sense:

$$\mathbb{E}[\widehat{v}_1(\pi)] = v(\pi)\,.$$

In other words, on average $\widehat{v}_1$ is identical to the true policy value that we try to estimate. Therefore, as long as variance is controlled, one can estimate the policy value to within a certain confidence level; further details and discussions can be found in [3, 19, 26].

4. SIMULATE RANDOMIZATION WITH PRODUCTION RANKER DIVERSITY

The approach above assume availability of randomized exploration data, a large amount of which may not be easy to collect, due to various practical restrictions like user dissatisfaction concerns. In contrast, the production ranker already produces a large amount of search log at almost no cost. It would be desirable to extract the most use of such data.

One idea that we pursue here is to use inherent diversity in ranker behavior (for the same query, in different flights and/or over different time periods) to *simulate* randomization, even if the individual rankings were computed deterministically in each impression. Such diversity is usually caused by factors such as continuous updates of the ranking function, changing features of query–document pairs, constant updates of the engine's indexes, etc. Such diversity leads to "natural exploration" behavior of the system that we will try to take advantage of. Since such data will play the same role as the (randomized) exploration data described previously, we use the same notation $\mathcal{D} = \{\langle q_i, a_i, p_i, r_i \rangle\}_{1 \le i \le n}$ for it and also refer it to as the exploration data. No confusion should be raised, as it is the only data we use for the following discussions and experiments. Furthermore, the ranker we try to evaluate is in general different from past rankers that were used to generate exploration data.

To be precise, let us define by

$$\begin{aligned} n(q) &:= \sum_{i=1}^{n} \mathbb{I}\{q_i = q\} \\ n(q,a) &:= \sum_{i=1}^{n} \mathbb{I}\{q_i = q, a_i = a\} \end{aligned}$$

the counts of query q and of the query-SERP pair (q,a), respectively, observed in the exploration data, where $\mathbb{I}\{C\}$ is 1 if condition C holds true and 0 otherwise. The probability of the simulated randomization becomes

$$p_i = p(a_i|q_i) := \frac{n(q_i,a_i)}{n(q_i)},$$

and the previous unbiased estimator becomes

$$\widehat{v}_1(\pi) = \frac{1}{n} \sum_i \frac{\pi(a_i|q)n(q_i)}{n(q_i,a_i)} r_i \,.$$

By grouping the terms in the summation by distinct query-action pairs, we obtain the following offline metric estimator for a new ranker:

$$\begin{aligned} \widehat{v}_1(\pi) &= \frac{1}{n} \sum_q n(q) \sum_{a \in \mathcal{A}_q} \pi(a|q) \widehat{r}(q,a) \\ &= \sum_q \widehat{\mu}(q) \sum_{a \in \mathcal{A}_q} \pi(a|q) \widehat{r}(q,a) \,, \qquad (3) \end{aligned}$$

where $\widehat{\mu}(q) := n(q)/n$ is the relative frequency of q in data $\mathcal{D}$, and

$$\widehat{r}(q,a) := \frac{1}{n(q,a)} \sum_i \mathbb{I}\{q_i = q, a_i = a\} r_i$$

is the average of reward for (q,a) in $\mathcal{D}$. We adopt the convention that $\widehat{r}(q,a) = 0$ when $n(q,a) = 0$. In other words, the estimator $\widehat{v}_1$ is like first building a regressor $\widehat{r}$ to predict $\mathbb{E}[r(q,a)]$ for every observed (q,a) pairs, and then plugging it in the definition of $v(\pi)$. For this reason, it is also called a *regression estimator* and is near-optimal in a strong statistical sense: for reasonable sample size n, the regression estimator's mean square error is always within constant factor of the optimal estimator's mean square error, and is asymptotically optimal as the sample size goes to infinity. [22]

4.1 A Variant with Target Query Distribution

In our experiment, we will use data collected in the past (the exploration data) to estimate online metrics for flights that occurred later (the test data). It is a reasonable assumption that query distributions in exploration and test data are similar. But for certain segments of queries, this assumption is likely to be violated. For example, time-sensitive queries may have a highly varying distribution over a short period of time, so queries that are common six months ago may not be popular any more now, and vice versa. Such a gap makes it hard to *validate* the evaluation approach we propose here.

To remove this gap, we can replace $\widehat{v}_1$ above by summing over queries in test data, although the reward estimates $\widehat{r}$ are still estimated using exploration data. This yields a slightly different estimator, denoted $\widehat{v}_2$:

$$\begin{aligned} \widehat{v}_2(\pi) &:= \frac{1}{n_*} \sum_q n_*(q) \sum_{a \in \mathcal{A}_q} \pi(q,a) \widehat{r}(q,a) \\ &= \frac{1}{n_*} \sum_q \sum_{a \in \mathcal{A}_q} n_*(q,a) \widehat{r}(q,a) \,, \qquad (4) \end{aligned}$$

where n_*, $n_*(q)$, and $n_*(q,a)$ are the corresponding counts in test data, and the second equality is due to the observation that the ranker to be evaluated is defined through test data by $\pi(a|q) = n_*(q,a)/n_*(q)$.

It should be emphasized that $\widehat{v}_2$ is given to remove query distribution mismatch between exploration and test data for the purpose of our experiments in the paper. What it tries to validate empirically is whether metrics are similar between exploration and treatment data, when query distributions match perfectly. But when building an experiment success predictor in reality, test data are not available, since frequent online experiments with new rankers is what we try to avoid in the first place. On the other hand, there is a reason to use as many search log as possible to get better estimate of $\widehat{r}$ and to increase coverage of (q,a), implying pretty outdated data have to be used. So, query frequencies $n(q)$ observed *in the past* (used in $\widehat{v}_1$) may be quite different from what would be observed if π is flighted *now*. An easy fix is to use recent search log to obtain query frequencies $n(q)$ and plug them in $\widehat{v}_1$.

4.2 Special Case for Deterministic π

In most practical situations, we are interested in evaluating a deterministic ranker, in which case the estimator can be simplified from the more general form presented earlier. We provide the formula here due to the importance of such a special case. Recall from Section 2 that, when π is deterministic, $\pi(q)$ refers to the unique action it selects for query q. The two estimators can be expressed as:

$$\begin{aligned} \widehat{v}_1(\pi) &= \frac{1}{n} \sum_q n(q) \widehat{r}(q, \pi(q)) \,, \\ \widehat{v}_2(\pi) &= \frac{1}{n_*} \sum_q n_*(q) \widehat{r}(q, \pi(q)) \,. \end{aligned}$$

4.3 Variance Estimation

In addition to the *point* estimates given above, variance information is also needed to evaluate whether a difference between estimates of two policies' values is statistically significant. Here, we give a simple-to-compute *approximation* for variance.

First of all, recall that $r_i \in [0, R]$ for some constant R. If r_i corresponds to binary events like click-or-not or success-or-not, then $R = 1$. Under the boundedness condition, the variance of r_i can be bounded by $\mathbb{V}(r_i) \le R^2/4$, which is tight when r_i is a Bernoulli random variable (taking values in $\{0, R\}$) with parameter $1/2$. Since $\widehat{r}(q,a)$ is the average of $n(q,a)$ such IID random variables, we immediately have

$$\mathbb{V}(\widehat{r}(q,a)) \leq \frac{R^2}{4n(q,a)}\,. \tag{5}$$

Consequently, the variance of the two estimators follows immediately:

$$\begin{aligned}
\mathbb{V}(\widehat{v}_1(\pi)) &= \frac{1}{n^2}\sum_q \sum_{a\in\mathcal{A}_q} \frac{\pi(a|q)^2 n(q,a)^2}{p(a|q)^2}\mathbb{V}(\widehat{r}(q,a)) \\
&\leq \frac{R^2}{4n^2}\sum_q \frac{n(q)^2}{n_*(q)^2}\sum_{a\in\mathcal{A}_q}\frac{n_*(q,a)^2}{n(q,a)}\,, \\
\mathbb{V}(\widehat{v}_2(\pi)) &= \frac{1}{n_*^2}\sum_q \sum_{a\in\mathcal{A}_q} n_*(q,a)^2\mathbb{V}(\widehat{r}(q,a)) \\
&\leq \frac{R^2}{4n_*^2}\sum_q \sum_{a\in\mathcal{A}_q}\frac{n_*(q,a)^2}{n(q,a)}\,.
\end{aligned}$$

For the important special case when π is deterministic, the above formula may be simplified as

$$\begin{aligned}
\mathbb{V}(\widehat{v}_1(\pi)) &= \frac{1}{n^2}\sum_q n(q)^2\mathbb{V}(\widehat{r}(q,\pi(q))) \\
&\leq \frac{R^2}{4n^2}\sum_q \frac{n(q)^2}{n(q,\pi(q))} \\
\mathbb{V}(\widehat{v}_2(\pi)) &= \frac{1}{n_*^2}\sum_q n_*(q)^2\mathbb{V}(\widehat{r}(q,\pi(q))) \\
&\leq \frac{R^2}{4n^2}\sum_q \frac{n_*(q)^2}{n(q,\pi(q))}\,.
\end{aligned}$$

Given a variance estimate, confidence intervals can be estimated. For example, 95% confidence interval is $1.96 \times \sqrt{\mathbb{V}(\widehat{v}_i(\pi)}$.

Note that the above variance estimation formula works for both continuous and discrete-valued metrics, as long as the metrics have a bounded range inside $[0, R]$. While being very general, the variance upper bound given in Equation 5 may not be most tight. An empirically better alternative, as discussed in previous work [3, 19], is to estimate $\mathbb{V}(\widehat{r}(q,a))$ from data, using classic normal approximation techniques or recent statistics results [23].

4.4 Limitations

Although the proposal here is inspired by the unbiased estimator reviewed in Section 3 and shares many similarities, the estimator obtained from *simulated* exploration data loses the nice unbiasedness guarantee in general. One key problem is that, the search log is not exploratory enough to cover *all* possible rankings for any query. Consequently, if for some query q the target policy π selects a ranking that has no occurrence in data (meaning $n(q,\pi(q)) = 0$), there will be no information to predict what the online metric would be if ranking $\pi(q)$ is displayed to the user.

Another source of bias of our approach is introduced by potential confounding in the data [3] — for example, when selection of SERPs in exploration data depends on both the query and additional information (such as user location).

Randomized data collection can remove these two biases, hence should be attempted whenever possible.

5. FUZZY MATCHING

A major difficulty common to the approaches in Sections 3 and 4 is that $\mathcal{A}_q$ can be large for many queries q, leading to two consequences:

- The larger $\mathcal{A}_q$ is, the smaller $n(q,a)$ tends to be for every a, and a larger variance is expected from our estimators (*c.f.*, Equation 5). To understand the influence, consider the special situation when π is deterministic, all $n(q)$ are identical and equals n/Q (Q is the number of distinct queries found in D), and $|\mathcal{A}_q| = K$. Then, the variance estimation for $\widehat{v}_1$, as given in the previous section, becomes $R^2K/(4n)$. So the variance roughly grows linearly with the size of average size of $\mathcal{A}_q$.
- The larger $\mathcal{A}_q$ is, the less likely metric information can be extracted from data to estimate $v(\pi)$. In fact, as observed empirically, queries that share common SERPs between exploration and test data tend to be more head-ish ones. It follows that $\mathcal{A}_q$ tends to be larger when q is tail, and so have lower chances to be included in the offline estimation formula.

The idea of *fuzzy matching* is to relax the condition of *exact* matching between two rankings, so that two rankings a and a' are considered identical when they are "close" enough. By doing so, we can effectively reduce the size of $\mathcal{A}_q$, since many rankings collapse into one, and can alleviate the difficulty described above.

Here, we consider a special case of fuzzy matching that focus on top-ranked URLs in a ranking. For example, if we only focus on the top $L = 4$ URLs on the SERP, two rankings that only differ in positions below L will be treated as the same ranking. Such fuzzy matching will inevitably introduce bias in the evaluation results (since URLs below top L do have influence on metrics), but hopefully we will be able to find a good trade-off by choosing L properly.

Some notation is in place. For any query q, let $\mathcal{A}_q$ be all rankings observed in data. We say $a \sim a'$ if a and a' are considered the same. With the binary equivalence relation $\sim$, $\mathcal{A}_q$ can be partitioned into a collection of distinct subsets, $\mathcal{A}_{q,1}, \mathcal{A}_{q,2}, \ldots$, so that each $\mathcal{A}_{q,j}$ contains rankings that are considered identical. Now, we just need to treat $\mathcal{A}_{q,i}$ as new actions and apply the same techniques as before. Similar to the exact-match case, define the following quantities:

$$\begin{aligned}
n(q,\mathcal{A}_{q,j}) &:= \sum_{a\in\mathcal{A}_{q,j}} n(q,a) \\
\tilde{r}(q,\mathcal{A}_{q,j}) &:= \frac{\sum_i \mathbb{I}\{q_i = q, a_i \in \mathcal{A}_{q,j}\}r_i}{\sum_i \mathbb{I}\{q_i = q, a_i \in \mathcal{A}_{q,j}\}} \\
&= \frac{1}{n(q,\mathcal{A}_{q,j})}\sum_i \mathbb{I}\{q_i = q, a_i \in \mathcal{A}_{q,j}\}r_i
\end{aligned}$$

Finally, given q and a, let $j(q,a)$ be the (inverted) index such that $a \in \mathcal{A}_{q,j(q,a)}$.

For fuzzy matching with binary relation $\sim$, the previous estimators can be adapted using the quantities above:

PROPOSITION 1. *The estimator in Equation 3 becomes the following with fuzzy matching:*

$$\tilde{v}_1(\pi) = \sum_q \widehat{\mu}(q)\sum_{a\in\mathcal{A}_q}\frac{\sum_{a'\sim a}\pi(a'|q)n(q,a)\widehat{r}(q,a)}{\sum_{a'\sim a}n(q,a')}$$

The extension to $\widehat{v}_2$ is similar.

6. EXPERIMENTS

In this section, we describe the experiments to evaluate the performance of the prediction method. Our experiment has several objectives. First, we would like to know how accurate the predictions for individual flights are. Also, since the goal of online experiment is to determine the relative performance of each flight, we would like to know how accurately the relative performance between two ranking methods are predicted.

For this paper, our method is to use the existing experimentation data to evaluate the prediction technique. In other words, we collected experimentation data for a year, and split the data into two periods to generate the exploration and test data, respectively. Finally, we can see to what extent the metric value estimates based on exploration data approximate the metric values in test data.

6.1 Data Collection

For our experiment, we used large-scale log data from the random sample of ranking-related online experiments for a major search engine in US market. Each record in the data takes the form of (Query, Top-8 results, Metric Values, Impression Count). Our data contains the log data of one year period, where the first 6 months are used as exploration-set and the following months are used as test set. Our data set contains 190 ranking algorithms (rankers) within 44 ranking experiments.

We created two types of data sets for our experiments which are then used to estimate and evaluate the model, respectively. First, we used the data in Period I to generate the exploration data set described in Section 4. That is, we aggregated the data by query, action, the (simulated) probability of action and reward values. Second, we used the data in Period II to generate the test data set. That is, we aggregated the data by *ranker*, query, action, the (simulated) probability of action and reward values.

Given the two data sets, our experiment is to estimate the metric value for each ranker in the test data using reward signals from the exploration data. We achieve this by joining the exploration and test data using (query, action) as key. And the fuzzy matching introduced in Section 5 is used to control the definition of action used for the experiments using both estimators $\tilde{v}_1$ and $\tilde{v}_2$.

6.2 Results

6.2.1 Metric Value Prediction

We present the results for predicting absolute metric values, where we plotted the predicted metric values against actual metric values. The results in Figure 1 and Figure 2 shows that the correlation is strongest when defining action based on Top 3 web results, followed by Top 5 web results. In fact, there seems to be virtually no correlation when using Top 8 web results (i.e., exact match) to define action.

Table 1 shows the same trends, where the level of correlation (Cor(.)) gets lower as we define action in a more fine-grained manner. As a possible explanation, you can see that the number of (Query,Action) matches decreases sharply when we use the exact match. Between the 1st and the 2nd estimator, we find that the 2nd estimator provides slightly better correlation, which implies that the mismatch in query count does have some impact.

Table 1: Results for predicting absolute metric values.

TopK	Cor($\widehat{v}_1$)	Cor($\widehat{v}_2$)	#(Query,Action)
3	0.549	0.596	645,749,791
5	0.431	0.438	244,046,777
8	-0.271	-0.254	63,646,334

Table 2: Results for predicting the delta in metric values between two rankers.

TopK	Accuracy	Correlation
3	58.5%	0.450
5	56.1%	0.396
8	50.7%	0.370

6.2.2 Metric Delta Prediction

We present the results for predicting the delta in metric values between two rankers. Each experiment in our test data contains two rankers: one is a baseline ranker ("control") and the other is a proposed new ranker ("treatment"). The experiment was used to discover whether the proposed ranker is superior to the baseline ranker. Such pairs of rankers are therefore ideal pairs for our purpose to predict metric deltas.

In particular, for each experiment, we calculated the difference in the metric value between the two rankers. The results in Figure 3 shows the distribution of predicted delta between two rankers within each experiment for varying level of approximate match, against the true metric differences based solely on test data. It can be seen that the correlation is quite strong for Top-3 (fuzzy) match, and worst for Top-8 (exact) match.

While the scatterplot shows the general trend, what really matters are the decisions based on the values. In other words, each comparison between two rankers can be determined to be WIN (proposed ranker significantly better than the baseline), LOSS (proposed ranker significantly worse than the baseline), or TIE (otherwise). We want to see whether our offline evaluation method gives good predictions of online flight successes or not.

In order to make decisions based on the metric values, we used the two-sample t-test with p-value threshold of 0.05 on both predicted and actual delta values. Then we calculated the accuracy as the sum of diagonal divided by the total number of pairs being compared. Table 2 presents the accuracy values as well as the correlation between the predicted and actual delta. Note that if we make random decisions among WIN/LOSS/TIE, the accuracy is 33.3%, and our improvement is statistically significant. If we predict the majority outcome, the prediction is TIE which is not useful at all.

Table 3 presents the confusion table between predicted outcome and the actual outcome of experiments, as well as the accuracy/recall of predictions for each outcome. Again, there seems to be a clear bias in the prediction results. That is, there are many cases where predicted outcome was positive (a WIN case) while actual outcome was TIE (or even LOSS), but very few cases in the opposite direction. We can call it an "optimistic" bias in that the prediction results are more positive than actual experimental outcome. We delve

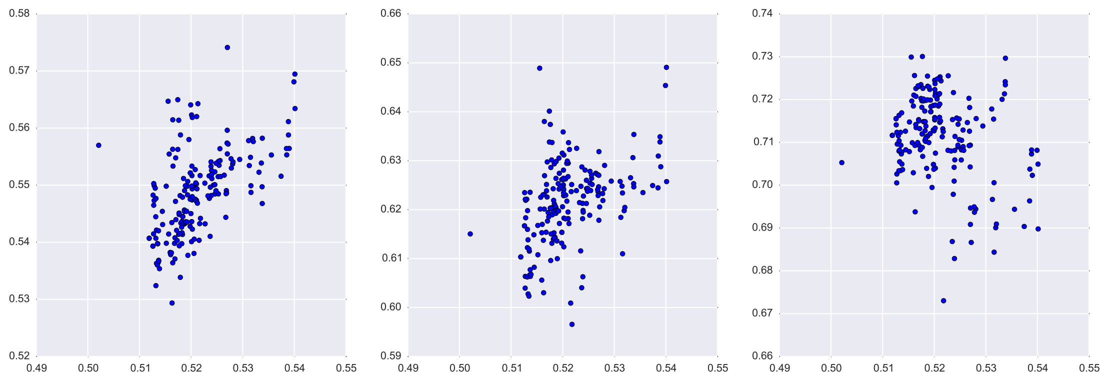

Figure 1: Scatterplot for actual (X) vs. predicted (Y) click ratio based on the 1st estimator ($\widehat{v}_1$), where the action is defined by Top 3, 5 and 8 web results, respectively.

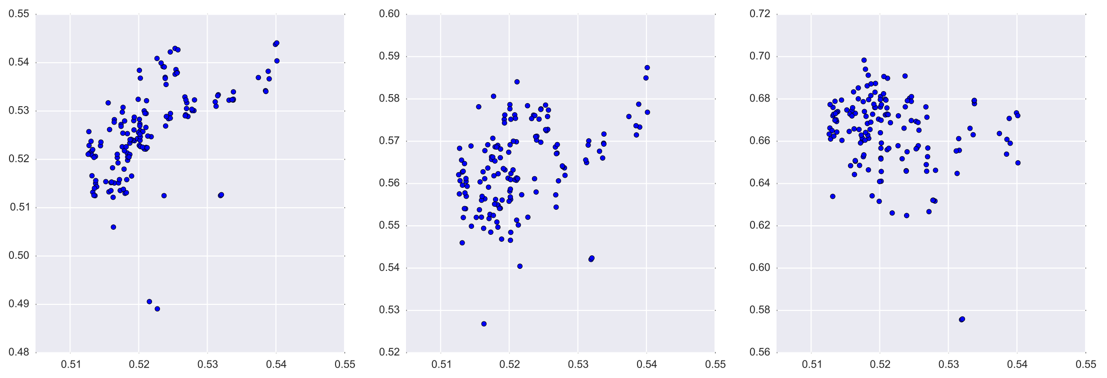

Figure 2: Scatterplot for actual (X) vs. predicted (Y) click ratio based on the 2nd estimator ($\widehat{v}_2$), where the action is defined by Top 3, 5 and 8 web results, respectively.

into why that is the case in the following sections. Such a bias is also reflected by the relatively high accuracy of LOSS predictions

It should be noted that, for our purpose, an optimistic bias is relatively easy to handle: if a proposed ranker is predicted to be a WIN, one can run it in a real experiment on users to find out whether it is really the case. On the other hand, if a LOSS prediction is made, it is a strong indication that the ranker is not worth running a real experiment. Therefore, our technique can be used to weed out unpromising techniques early on without going through with an online experiment.

Finally, we examine how often the offline prediction gives results opposite to the actual outcomes. A *disagreement* happens when a WIN is predicted for a LOSS flight or when a LOSS is predicted for a WIN flight. The *disagreement ratio* is the ratio between number of disagreements and number of WIN/LOSS flights. As shown in Table 4, the disagreement ratio is reasonably low across various K values.

Table 3: WIN/TIE/LOSS confusion table between actual outcomes (columns) and predicted outcomes (rows) for different K values of topK fuzzy matching. Accuracy/recall are also included for each outcome.

K=3	LOSS	TIE	WIN	Accuracy	Recall
WIN	3.4%	19.0%	13.7%	37.8%	65.1%
TIE	8.8%	38.0%	6.8%	70.9%	63.4%
LOSS	6.8%	2.9%	0.5%	66.7%	35.9%
K=5	LOSS	TIE	WIN	Accuracy	Recall
WIN	3.9%	11.7%	5.9%	27.3%	27.9%
TIE	12.2%	47.3%	15.1%	63.4%	78.9%
LOSS	2.9%	1.0%	0.0%	75.0%	15.4%
K=8	LOSS	TIE	WIN	Accuracy	Recall
WIN	4.4%	18.5%	7.3%	24.2%	34.9%
TIE	12.7%	41.5%	13.7%	61.2%	69.1%
LOSS	2.0%	0.0%	0.0%	100%	10.3%

6.3 Analysis of Bias

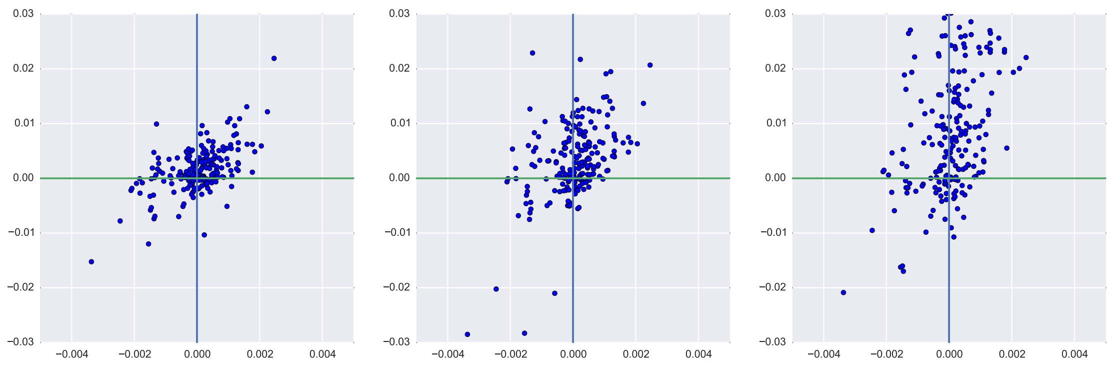

Figure 3: Scatterplot for actual (X) vs. predicted (Y) delta in click ratio between two rankers in the same experiment where the action is defined by Top 3, 5 and 8 web results, respectively.

K	3	5	8
Disagreement ratio	9.8%	9.8%	11.0%

Table 4: Disagreement ratio for different K values.

In Section 6.2.2, we observed that the metric delta prediction results are more optimistic than actual experimental outcome. This means that the predicted metric value for experimental ranker tends to be higher than the control ranker. Among many possible explanations, one clear difference between two rankers is the coverage of exploration data. That is, experimental rankers should produce more results that are not found in the exploration data set than the control rankers, which is mostly the rankers already in production.

In order to verify the hypothesis, we investigated the relationship between the degree of optimistic bias measured by the ratio of predicted metric value against actual metric value, and the coverage of exploration data in terms of the percentage of matching records. Figure 4 shows the results, where the negative correlation is clear in the rightmost plot, which employs the exact matching strategy.

The result indicates that the increase in coverage means reduced bias, and the trend is most conspicuous for the case of exact match. This trend is consistent with what we observed in Figure 3 and Table 3, where the results were most concentrated in the upper left corner for exact match case.

6.4 Segment-level Results

Our prediction method relies on the assumptions that there is no confounding in the exploration data (*c.f.*, Section 4.4), that there are sufficient amount of exploration data (i.e., metric values for different set of results) for each query, and that we have enough observations for each (Query, Action) pair. The other requirement is that the users' click behavoir would be consistent between exploration and exploration data for the same query.

Since these assumptions may not hold for certain type of queries, we can hypothesize that the performance of prediction method should vary across different query segments. We classified queries into different segments, and then calculated the predicted and actual metric values as before, to see how the correlation between predicted and actual metric values changes for different segments.

Table 5: Correlation between predicted and actual metric values for different segments with different K values in fuzzy matching.

TopK	Segment	Cor	Count
3	Head	0.916	364,757
	Body	0.962	553,892,081
	Tail	0.722	90,867,404
	Local	0.342	60,448,865
	Navigational	0.642	120,343,701
	Overall	0.934	645,124,406
5	Head	0.875	345,640
	Body	0.894	229,862,589
	Tail	0.860	13,636,733
	Local	0.413	24,049,649
	Navigational	0.648	58,680,950
	Overall	0.885	243,845,041
8	Head	0.563	304,142
	Body	0.962	61,433,473
	Tail	0.993	1,809,956
	Local	0.453	6,369,383
	Navigational	0.575	23,107,195
	Overall	0.872	63,547,643

The results in Figure 5 shows the level of correlation for different segments. It is clear that the correlation for the body segment is strongest in overall in Top 3 and 5 matching, and that the Local segment has lowest correlation, followed by Navigational. Table 5 presents the same results with the number of matching records. Note that the number of tail queries in Top 8 match is much smaller than Top 3 and 5 matches. We hypothesize that this have led to exceptionally higher correlation on that segment.

Between Head/Body/Tail, the Body segment shows the highest correlation in overall, which is reasonable in that there would be sufficient variety of per-query results shown to the user, with enough sample size for each result. In Head segment where there are many signals for ranking, we are unlikely to observe lots of variation in Top-k results. For

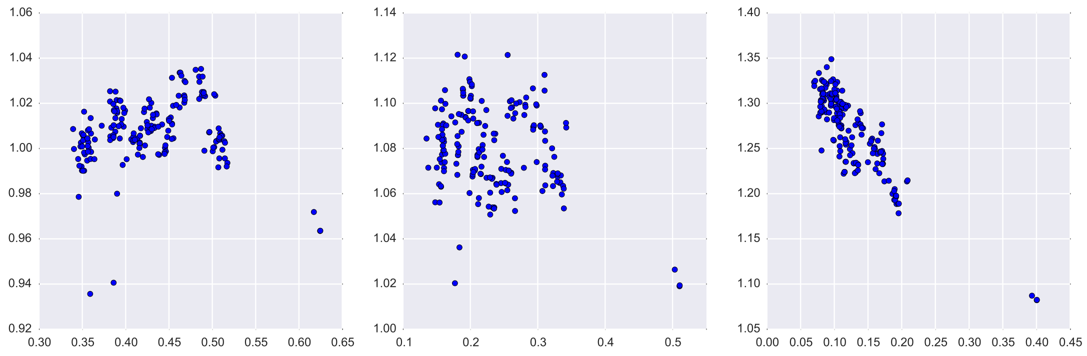

Figure 4: Scatterplot for the ratio of predicted metric value against actual metric value, plotted against % of matching records, where the action is defined by Top 3, 5 and 8 web results, respectively.

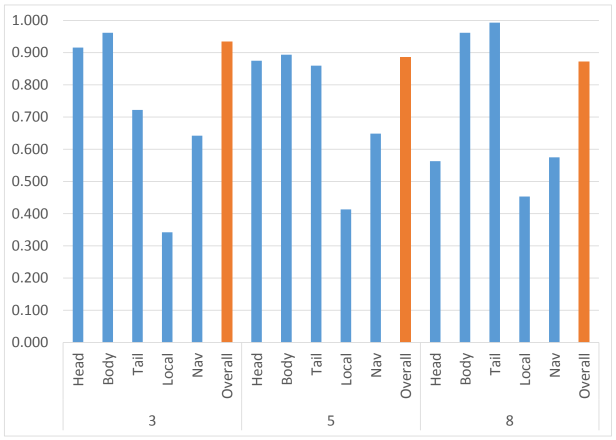

Figure 5: Correlation between predicted and actual metric values for different segments with varying K values in fuzzy matching.

tail, while the variety might exist, there should be very few data points for each alternative ranking.

Low correlation for local segment can be explained if we note that the click pattern is likely to vary across different locations even for the same query, creating confounding that leads to a bias. For navigational queries, where the clicks would be heavily skewed toward the top few results, we hypothesized that there may not be enough exploration data that can allow effective prediction.

7. CONCLUSIONS

In this work, we proposed an offline approach to estimating online metrics of a new ranker that typically depend on user feedback. This approach does not require running every such new ranker on live users and is therefore less expensive than standard A/B experiments. Compared to previous work with the same motivation in recommender and advertising systems, we proposed two variations to address novel challenges encountered in Web search.

Our approach shows promise based on extensive experiments on large-scale data collected from a major search engine. The fuzzy matching technique based on Top-3 results can predict actual metric values closely (Pearson correlation 0.6), and the prediction of experimental results between control and treatment ranking techniques shows reasonably high accuracy upto 58.5% – significantly higher than chance results (33.3%).

The promising results lead to a few directions for future work. First, while there is no conceptual difficulty, we would like to apply the same technique to estimate session-level and user-level metrics. Second, we conjecture that our evaluation method can benefit from more flexible fuzzy matching criterion, in which the similarity between two SERPs of the same query can be measured by a real number (*e.g.*, [15]), as opposed to the binary same-or-different SERP matching outcomes considered in this paper.

Acknowledgement

We thank Dongwoo Kim for help in our experiments, and the anonymous reviewers for numerous suggestions that improved the presentation of the paper.

8. REFERENCES

[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Technology behind Search*. ACM Press Books. Addison-Wesley Professional, 2nd edition, 2011.

[2] Nicholas J. Belkin. Some(what) grand challenges for information retrieval. *ACM SIGIR Forum*, 42(1):47–54, 2008.

[3] Léon Bottou, Jonas Peters, Joaquin Quiñonero Candela, Denis Xavier Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14:3207–3260, 2013.

[4] Christopher J. C. Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Gregory N. Hullender. Learning to rank using gradient

descent. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 89–96, 2005.

[5] Georg Buscher, Ludger van Elst, and Andreas Dengel. Segment-level display time as implicit feedback: A comparison to eye tracking. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 67–74, 2009.

[6] Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. Large scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Science*, 30(1), 2012.

[7] Olivier Chapelle, Eren Manavoglu, and Romer Rosales. Simple and scalable response prediction for display advertising. *ACM Transactions on Intelligent Systems and Technology.* To appear.

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

[9] Aleksandr Chuklin, Pavel Serdyukov, and Maarten de Rijke. Click model-based information retrieval metrics. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 493–502, 2013.

[10] Georges Dupret and Benjamin Piwowarski. 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*, pages 331–338, 2008.

[11] Fan Guo, Chao Liu, Anitha Kannan, Tom Minka, Michael J. Taylor, Yi-Min Wang, and Christos Faloutsos. Click chain model in Web search. In *Proceedings of the 18th International Conference on World Wide Web*, pages 11–20, 2009.

[12] Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten de Rijke. Reusing historical interaction data for faster online learning to rank for IR. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, pages 183–192, 2013.

[13] Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. Estimating interleaved comparison outcomes from historical click data. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 1779–1783, 2012.

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

[15] Gabriella Kazai and Homer Sung. Dissimilarity based query selection for efficient preference based IR evaluation. In *Proceedings of the European Conference on Information Retrieval*, pages 172–183, 2014.

[16] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. Controlled experiments on the web: Survey and practical guide. *Data Minining and Knowledge Discovery*, 18:140–181, 2009.

[17] Diane Lambert and Daryl Pregibon. More bang for their bucks: Assessing new features for online advertisers. *SIGKDD Explorations*, 9(2):100–107, 2007.

[18] John Langford, Alexander L. Strehl, and Jennifer Wortman. Exploration scavenging. In *Proceedings of the 25th International Conference on Machine Learning*, pages 528–535, 2008.

[19] Lihong Li, Shunbao Chen, Ankur Gupta, and Jim Kleban. Counterfactual analysis of click metrics for search engine optimization. Technical Report MSR-TR-2014-32, Microsoft Research, 2014.

[20] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pages 661–670, 2010.

[21] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the 4th International Conference on Web Search and Data Mining*, pages 297–306, 2011.

[22] Lihong Li, Remi Munos, and Csaba Szepesvári. On minimax optimal off-policy policy evaluation. Technical report, Microsoft Research, 2014.

[23] Andreas Maurer and Massimiliano Pontil. Empirical Bernstein bounds and sample-variance penalization. In *Proceedings of the Twenty-Second Conference on Learning Theory*, pages 247–254, 2009.

[24] Filip Radlinski and Nick Craswell. Optimized interleaving for online retrieval evaluation. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, pages 245–254, 2013.

[25] Stephen Robertson. On the history of evaluation in IR. *Journal of Information Science*, 34(4):439–456, 2008.

[26] Alexander L. Strehl, John Langford, Lihong Li, and Sham M. Kakade. Learning from logged implicit exploration data. In *Advances in Neural Information Processing Systems 23*, pages 2217–2225, 2011.

[27] Liang Tang, Romer Rosales, Ajit Singh, and Deepak Agarwal. Automatic ad format selection via contextual bandits. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pages 1587–1594, 2013.

[28] Andrew H. Turpin and William Hersh. Why batch and user evaluations do not give the same results. In *Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 225–231, 2001.

[29] Zhaohui Zheng, Hongyuan Zha, Tong Zhang, Olivier Chapelle, Keke Chen, and Gordon Sun. A general boosting method and its application to learning ranking functions for web search. In *Advances in Neural Information Processing Systems 20*, pages 1000–1007, 2008.