Bayesian Ranker Comparison Based on Historical User Interactions

Artem Grotov
a.grotov@uva.nl

Shimon Whiteson
s.a.whiteson@uva.nl

Maarten de Rijke
derijke@uva.nl

University of Amsterdam, Amsterdam, The Netherlands

ABSTRACT

We address the problem of how to safely compare rankers for information retrieval. In particular, we consider how to control the risks associated with switching from an existing *production ranker* to a new *candidate ranker*. Whereas existing online comparison methods require showing potentially suboptimal result lists to users during the comparison process, which can lead to user frustration and abandonment, our approach only requires user interaction data generated through the natural use of the production ranker. Specifically, we propose a Bayesian approach for (1) comparing the production ranker to candidate rankers and (2) estimating the confidence of this comparison. The comparison of rankers is performed using click model-based information retrieval metrics, while the confidence of the comparison is derived from Bayesian estimates of uncertainty in the underlying click model. These confidence estimates are then used to determine whether a risk-averse decision criterion for switching to the candidate ranker has been satisfied. Experimental results on several learning to rank datasets and on a click log show that the proposed approach outperforms an existing ranker comparison method that does not take uncertainty into account.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Ranker evaluation; Learning to rank; Click models

1. INTRODUCTION

Comparing rankers is an essential problem in information retrieval. In an industrial setting, an existing *production ranker* must often be compared to a new *candidate ranker* to determine whether to replace the former with the latter. Practical constraints make it difficult to make such decisions well. Because the performance of the candidate ranker is unknown, trying it out to gather data about its performance is often too risky: if it proves to be substantially inferior to the production ranker, the user experience may degrade, leading to abandonment. However, existing historical data (typically collected using the production ranker) may not be sufficiently informative about the candidate ranker. Hence, decisions about when to switch to the candidate ranker that are based on such data can be erroneous, also leading to a degraded user experience.

Controlling the risks associated with switching to a candidate ranker requires ranker comparison methods that can (1) estimate the performance of the production and candidate rankers using only historical data, and (2) quantify their uncertainty about such estimates. Quantification of uncertainty is essential for controlling risk because it enables system designers to switch to a candidate ranker only when they are highly confident that its performance will not be substantially worse than that of the production ranker.

Existing ranker evaluation methods do not fully meet these requirements. Traditional methods rely on explicit relevance labels provided by human assessors to measure metrics such as Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP) [20]. However, such labels are expensive to obtain and may not reflect the preferences of real users. Click-based approaches overcome these limitations by estimating performance from the implicit signals in users' click behavior, using A/B testing or interleaving experiments [5, 6, 12, 13, 15, 16, 18, 19, 21]. Nonetheless, these methods typically require trying out the candidate ranker to gain data about it, which, as mentioned above, may be too risky. While interleaving methods have been developed that use importance sampling to estimate ranker performance given only historical data [14], they have limited generalization properties and require historical data to be gathered stochastically.

Click models [3, 10], i.e., probabilistic models of user behavior, can also estimate ranker performance given only historical data [7] and do not share these limitations. The uncertainty in the resulting estimates depends on the relationship between the historical data and the rankers to be evaluated. E.g., if a candidate ranker differs from the production ranker only in that it swaps a lowly ranked but relevant document with a highly ranked but irrelevant one, then the click model can confidently conclude that the candidate ranker is better so long as the historical data shows which of the two documents is relevant. It can do so even if the result lists containing such a swap do not appear in the historical data. By contrast, if the candidate ranker gives a high rank to a document that does not appear in the historical data, then its relevance cannot be estimated with confidence. A key limitation of existing click models is that they do not distinguish between cases such as these. Because they do not quantify their uncertainty about the comparisons they perform, they cannot judge whether the information in the logs is sufficient to compare rankers. Only methods that can measure the confidence of the performed comparisons can be used to safely decide whether to switch from the production ranker to a candidate one.

SIGIR'15, August 09 - 13, 2015, Santiago, Chile.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3621-5/15/08 ...$15.00.
DOI: http://dx.doi.org/10.1145/2766462.2767730.

We present *Bayesian Ranker Comparison* (BARACO), a click model-based approach to ranker evaluation that compares the performance of ranker pairs using only historical data and quantifies the uncertainty in such comparisons. The key novelty lies in maintaining a full posterior distribution over the relevance of documents for queries. This posterior is used to estimate the probability that the candidate ranker's performance is not substantially worse than that of the production ranker. By switching to the candidate ranker only when this probability is sufficiently high, the risks associated with switching can be controlled in a principled way.

BARACO is able to estimate not only the probability of a candidate ranker being as good as the production ranker, but also the expected difference in performance between two rankers. Measuring the expected difference can be useful in practice, e.g., to select a single ranker from a set of candidates, all of which have a high probability of beating the production ranker.

We present the results of an empirical evaluation on several learning to rank datasets and on a real click log published by Yandex that compares BARACO to an existing non-Bayesian click model-based ranker evaluation method that estimates only the expected difference in performance between ranker pairs, without quantifying uncertainty. We addresses the following research questions:

RQ1 Can BARACO determine whether the production ranker should be replaced with a candidate ranker better than the non-Bayesian baseline?

RQ2 Does BARACO produce better estimates of the difference in performance of the rankers than the non-Bayesian baseline?

Our results show that BARACO has better performance than the baseline method and can identify more good rankers in a set of candidates more reliably. In addition, the full Bayesian approach enables the production of more precise estimates of the expected differences between rankers than the baseline method.

2. RELATED WORK

Ranker evaluation has long been a central topic in information retrieval. The classical approach is to perform offline evaluation based on the Cranfield paradigm [8, 20], where a collection of documents is manually annotated by human experts. A representative set of queries together with associated user intents are crafted by a group of experts. Then, for each query, documents in the collection are assigned relevance labels. The documents and their relevance labels can then be used in metrics such as NDCG. This approach is widely used, well understood, and benefits from controlled laboratory settings. But it is expensive and assessors' relevance judgements may not adequately reflect real users' opinions.

Online evaluation is a family of techniques that addresses these difficulties by letting users be the judges. In A/B testing [16], the user population is split into two groups, and pairs of rankers are compared by presenting one group with one ranker and the other group with another ranker. The ranker with the best performance on a selected metric (such as CTR) is typically considered to be the winner [16]. Another approach is interleaved comparison: search engine result pages (SERPs) presented to users are obtained by interleaving SERPs of two competing rankers under consideration. The user feedback in the form of clicks is then interpreted as user preference for one ranker over the other [19]. The key problem of online methods is that they require user feedback to evaluate each pair of rankers, which often requires exposing suboptimal SERPs. Consequently, they may be too risky for many real-world settings.

Interleaving methods that exploit importance sampling [14] provide a way to compare rankers using only historical click data. However, importance sampling requires that the source distribution is non-zero everywhere where the target distribution is non-zero (i.e., every SERP that can result from interleaving has a non-zero probability of occurring in the historical data), a requirement that is often not met in practice. Importance sampling guarantees only that the resulting estimator is unbiased, not that it has low variance: the addition of more historical data is not guaranteed to improve estimates, as new data can actually increase the variance [1]. Additionally, importance sampling has poor generalization properties compared to click-based approaches that infer relevance labels of documents. In particular, the latter can generalize across SERPs that differ in the order of documents, while the former cannot.

In recent years, a number of probabilistic models have been developed to describe, understand, and predict user behavior while interacting with an IR system. In particular, click models such as DBN [3], DCM [10] and UBM [9] infer the relevance of documents for queries and model user clicks on documents in the SERP by analysing search engine log files. These inferred relevance labels can be further used for learning to rank [3] and for ranker comparisons using click model-based metrics [7]. However, to our knowledge, none of these approaches provide a way to quantify the uncertainty in the resulting comparison, which is critical for making informed decisions about when to switch from a production ranker to a candidate ranker. For metrics such as EBU [23], ERR [4] and the utility and effort based metrics in [2, 7], the effect on the comparison of a document seen just once and that of one seen a hundred of times is the same, given that the inferred relevance labels have the same value. Furthermore, these metrics do not take into account the number of previously unseen documents that appear in the SERPs produced by the candidate rankers. Hence, these approaches cannot properly control the risks associated with switching to a candidate ranker, as the uncertainty in the comparison is not measured.

We present a new approach to ranker evaluation that is designed to overcome these limitations. It compares rankers using only the log files collected through natural user interactions with the production ranker. In contrast to importance sampling based interleaving methods [14], our method does not require the data to be obtained stochastically. In contrast to the metrics in [2, 4, 7, 23], our method takes into account the uncertainty associated with the inferred relevance labels and the presence of unseen documents and measures the confidence of the resulting comparison.

3. PROBLEM SETTING

We consider two related problems. The first is the *Switching Problem*: deciding whether or not to switch from a production ranker R_p to a candidate ranker R_c. The second is the *Difference Estimation Problem*: estimating the expected difference in performance between the production ranker R_p and candidate ranker R_c.

The Switching Problem is as follows. System designers must decide whether or not to switch from a production ranker R_p to a candidate ranker R_c. They are willing to do so only if they are highly confident that the candidate ranker is at least almost as good as the production ranker, i.e., if

$$p(M_c + \epsilon \geq M_p) \geq 1 - \delta, \tag{1}$$

where M_p and M_c are the expected performance of the production and candidate ranker, respectively, according to some metric; ϵ is the degree to which the candidate ranker is allowed to be worse than the production ranker; and δ is the probability with which the candidate ranker is allowed to fall outside this threshold. The goal of a ranker comparison method in the Switching Problem is to determine whether (1) holds. Note that this requires explicitly reasoning about the uncertainty of comparisons between R_p and R_c.

By contrast, the goal of a ranker comparison method in the Difference Estimation Problem is to accurately estimate the expected difference of the metric values for the two rankers:

$$\mathbb{E}[M_c - M_p]. \tag{2}$$

Unlike the Switching Problem, the Difference Estimation Problem does not require explicitly reasoning about uncertainty. Nonetheless, it can be useful for, e.g., selecting a single ranker from a set of candidates, all of which satisfy (1).

For both problems, the ranker comparison method is given only a log $L = [l_1, \ldots, l_{|L|}]$ of user interaction sessions gathered using R_p. Each session l_j consists of the following: q, the query the user has submitted; $[d_0, \ldots, d_N]$, the list of N documents returned by R_p that make up the SERP; and $[c_0, \ldots, c_N]$, the clicks produced by the user on those documents.

4. CLICK MODEL-BASED METRIC

In this section, we describe the metric used to define M_p and M_c within BARACO, the Bayesian ranker comparison method we introduce in §5. While BARACO can work with any click model that provides relevance estimates, our implementation uses DBN [3] because it has shown high performance in predicting user clicks [7].

DBN, the Dynamic Bayesian Network click model [3], represents the relevance of a document for a query as a combination of two variables: *attractiveness* and *satisfactoriness*. In DBN, each user interaction session l_j begins with a query q from which the SERP, consisting of titles and snippets of the documents $[d_{q,0},\ldots, d_{q,N}]$, is generated. By assumption, the user starts by examining the first document's title and snippet. If they find the document *attractive*, they click on and examine the document, or otherwise proceed to examine the following documents' titles and snippets in order. If a clicked document *satisfies* the user, they terminate the session. Otherwise, they may or may not continue to examine the SERP. DBN assumes the user will not click a document prior to examining its snippet and cannot be satisfied by a document before clicking on it.

Specifically, for a query q the document at position i in the SERP is modelled with four binary variables: whether it is attractive ($A_{q,i}$), satisfactory ($S_{q,i}$), examined ($E_{q,i}$), and clicked ($C_{q,i}$). The following relationships hold between variables for a given query:

- $E_{q,0} = 1$: the first document is examined,
- $A_{q,i} = 1, E_{q,i} = 1 \Leftrightarrow C_{q,i} = 1$: a document is clicked if, and only if, it is attractive and examined,
- $C_{q,i} = 0 \implies S_{q,i} = 0$: only clicked documents can be satisfactory,
- $S_{q,i} = 1 \implies E_{q,i+1} = 0$: satisfied users abandon the search, and
- $E_{q,i} = 0 \implies E_{q,i+1} = 0$: users do not skip documents.

In addition, the following stochastic relationships hold:

- $p(E_{q,i+1} = 1 | E_{q,i} = 1, S_{q,i} = 0) = \gamma$: users continue to examine the list after not being satisfied with probability γ,
- $p(A_{q,i} = 1) = a_{q,i}$, and
- $p(S_{q,i} = 1 | C_{q,i} = 1) = s_{q,i}$.

Thus, attractiveness and satisfactoriness are governed by stationary Bernoulli distributions with unknown parameters $a_{q,i}$ and $s_{q,i}$, which must be inferred from clicks, the only observed variables. As in [3, 7], we assume γ is fixed and known. Fig. 1 depicts the relationships between all variables in a given session.

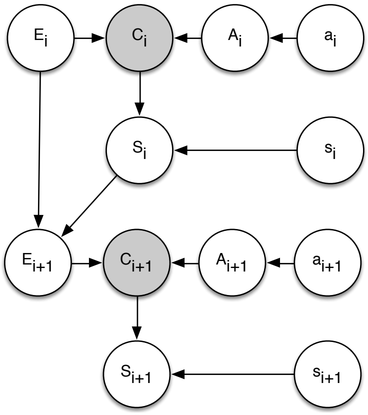


Figure 1: Graphical representation of the DBN click model. The grey circles represent the observed variables.

Once $a_{q,i}$ and $s_{q,i}$ have been inferred, they can be used to compute a metric such as EBU [23], ERR [4], and the utility and effort-based metrics described in [2, 7]. In this paper, we use the expected effort metric defined in [7]. For a query q that yields a document list with N documents, the metric is defined as:

$$rrMetric(q) = \sum_{i=1}^{N} \left(\frac{s_{q,i} a_{q,i}}{i} \prod_{j=1}^{i-1} (1 - s_{q,j} a_{q,j}) \right). \tag{3}$$

For a given ranker R_x, the metric M_x required by (1) can then be defined as the expected value of $rrMetric(q)$ across queries:

$$M_x = \sum_{q \in Q} rrMetric(q) p(q). \tag{4}$$

Comparing the production ranker to a candidate ranker is complicated by the presence of unseen documents in the candidate rankings. In order to compute the values of a metric for rankings with unseen documents, the unseen documents can be either assigned some a priori values of attractiveness and satisfactoriness or excluded from the ranking. The latter strategy is taken in [7] and in the baseline method used in our experiments.

Using an approach based on expectation maximisation (EM), a maximum a posteriori estimate of M_x can be computed [3]. Hence, the Difference Estimation Problem can be addressed by simply estimating M_c and M_p using this EM-based method and then computing the difference between them. However, this approach suffers from a key limitation. Because it is based only on maximum a posteriori estimates, the difference computed by such an EM-based approach is not the true expected difference of the metric values for the two rankers. Instead, it is only the first mode of the posterior distribution of the difference of the metric values, which may not be equal to the expectation if the distribution is multimodal or asymmetric. Because BARACO is a fully Bayesian approach, it can estimate the true expected value of the difference between M_c and M_p, which leads to better quality estimates, as we will see.

Furthermore, the Switching Problem cannot be directly addressed using an EM-based approach because such an approach gives no information about the uncertainty in the resulting estimate. That is, it does not estimate the *distribution* over the metric values of the two rankers. Consequently, it provides no way to estimate the probability that $M_c + \epsilon$ is greater than or equal to M_p. Instead, addressing the Switching Problem using an EM-based method requires resorting to a heuristic approach, e.g., switching only when the estimated difference exceeds some manually tuned threshold. This forms the core of the baseline method we compare against in §7.

Below, we present a method that addresses the shortcomings of an EM-based approach and enables better solutions to both the Switching Problem and the Difference Estimation Problem.

5. BARACO

In this section, we present BARACO. First, §5.1 describes how the posterior distributions over $a_{q,i}$ and $s_{q,i}$ can be inferred from L, the set of user interaction sessions; §5.2 describes how to solve the Switching Problem by evaluating (1) when M_p and M_c are defined according to (4), given posterior distributions over $a_{q,i}$ and $s_{q,i}$; §5.3 describes how to solve the Difference Estimation Problem.

5.1 Inferring click model posteriors

Evaluating (1) and (2) requires knowing the posterior probabilities $p(a_{q,i}|L)$ and $p(s_{q,i}|L)$ for each ranker, query, and document. In this subsection, we describe how to estimate these posteriors.

The algorithm works by iterating over the sessions. To process the first session, the posteriors $p(a_{q,i}|l_1)$ and $p(s_{q,i}|l_1)$ are computed given uniform priors $p(a_{q,i})$ and $p(s_{q,i})$. Then, for each subsequent session l_j, $p(a_{q,i}|L_j)$ and $p(s_{q,i}|L_j)$ are computed given $p(a_{q,i}|L_{j-1})$ and $p(s_{q,i}|L_{j-1})$, where $L_j = [l_1, \ldots, l_j]$. The algorithm terminates when $l = |L|$, yielding $p(a_{q,i}|L)$ and $p(s_{q,i}|L)$.

We now describe how to compute $p(a_{q,i}|L_j)$ and $p(s_{q,i}|L_j)$ given $p(a_{q,i}|L_{j-1})$ and $p(s_{q,i}|L_{j-1})$ and the next session l_j. Because $A_{q,i}$ and $S_{q,i}$ are Bernoulli variables, we can model the distribution over their parameters $a_{q,i}$ and $s_{q,i}$ using a Beta distribution. Focusing on $a_{q,i}$, this yields:

$$p(a_{q,i}|L_j) = Beta(\alpha, \beta) = \frac{{a_{q,i}}^{\alpha-1}(1-a_{q,i})^{\beta-1}}{B(\alpha,\beta)}, \tag{5}$$

where α is the number of observations of $A_{q,i} = 1$ and β is the number of observations of $A_{q,i} = 0$ that have occurred up to and including session j, and $B(\alpha, \beta)$ is a Beta function. $Beta(1, 1)$ corresponds to a uniform distribution, i.e., no prior knowledge about $a_{q,i}$. If $A_{q,i}$ is observed in session l_j, then $p(a_{q,i}|L_j)$ can be updated using Bayes' rule:

$$\begin{aligned} &p(a_{q,i}|L_j) = \\ &\qquad p(a_{q,i}|A_{q,i}, L_{j-1}) = \frac{p(A_{q,i}|a_{q,i})p(a_{q,i}|L_{j-1})}{p(A_{q,i}|L_{j-1})}, \end{aligned} \tag{6}$$

In this case, the update reduces to simply incrementing α or β. Since the Beta distribution is the conjugate prior for the Bernoulli variable, the posterior remains a Beta distribution.

In our setting, $A_{q,i}$ is not directly observed. However, when we know $E_{q,i}$, we can directly infer $A_{q,i}$ from $C_{q,i}$ because the user always clicks on an attractive examined document. Thus, the difficult case is when we do not know $E_{q,i}$, which occurs whenever $i > c$, where c is the index of the last clicked document. The remainder of this subsection describes how to address this case.

There are two steps. Since we do not know $E_{q,i}$ when $i > c$, in the first step we must instead compute a posterior over it: $p(E_{q,i}|L_j)$. Then, in the second step, we use $p(E_{q,i}|L_j)$ to estimate $p(a_{q,i}|L_j)$ and $p(s_{q,i}|L_j)$.

To perform the first step, we use the *sum-product* message passing algorithm [1]. In particular, we extract the subgraph of the graphical model that represents the documents below the last clicked one and remove the nodes representing the satisfactoriness ($S_{q,i}$ and $s_{q,i}$) for documents without clicks. This is because it is not possible to say anything about $s_{q,i}$ for $i > c$ as it is not observed and has no effect on what is observed. Since the resulting graph, shown in Fig. 2, is a polytree, the sum-product algorithm enables us to perform exact inference, which yields $p(E_{q,i}|L_j)$.

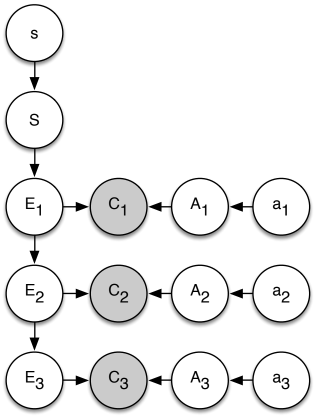


Figure 2: Graphical representation of the DBN click model for the part of the SERP below and including the last clicked document in a session. Grey circles represent observed variables.

For the second step, we compute $p(a_{q,i}|L_j)$ and $p(s_{q,i}|L_j)$ given $p(E_{q,i}|L_j)$. Focusing now on $p(a_{q,i}|L_j)$ and starting from Bayes' rule, we have:

$$\begin{aligned} &p(a_{q,i}|L_j) = \\ &\qquad p(a_{q,i}|C_{q,i}, L_{j-1}) = \frac{p(C_{q,i}|a_{q,i})p(a_{q,i}|L_{j-1})}{p(C_{q,i}|L_{j-1})}. \end{aligned} \tag{7}$$

The likelihood term $p(C_{q,i}|a_{q,i})$ can be computed by marginalizing across A and E:

$$\begin{aligned} &p(C_{q,i}|a_{q,i}) = \\ &\quad \sum_{A\in\{0,1\}} \sum_{E\in\{0,1\}} \Big(p(C_{q,i}|E_{q,i} = E, A_{q,i} = A) \\ &\qquad\qquad p(E_{q,i} = E) p(A_{q,i} = A|a_{q,i}) \Big). \end{aligned} \tag{8}$$

Sticking (8) into (7) and ignoring normalization gives:

$$\begin{aligned} &p(a_{q,i}|C_{q,i}, L_{j-1}) \propto \\ &\quad \sum_{A\in\{0,1\}} \sum_{E\in\{0,1\}} \Big(p(C_{q,i}|E_{q,i} = E, A_{q,i} = A) \\ &\qquad\qquad p(E_{q,i} = E) p(A_{q,i} = A|a_{q,i}) p(a_{q,i}|L_{j-1}) \Big) \\ &\propto \;\; p(a_{q,i}|L_{j-1}) \sum_{A\in\{0,1\}} \Big(p(A_{q,i} = A|a_{q,i}) \\ &\qquad \big(p(C_{q,i}|E_{q,i} = 0, A_{q,i} = A) p(E = 0) + \\ &\qquad + p(C_{q,i}|E_{q,i} = 1, A_{q,i} = A) p(E_{q,i}) \big) \Big). \end{aligned}$$

Since we are focusing on the case where $i > c$, we know that $C_{q,i} = 0$, i.e., $d_{q,i}$ was not clicked. Furthermore, we know that $p(C_{q,i} = 0|E_{q,i} = 0, A_{q,i} = A) = 1$ and $p(C_{q,i} = 0|E_{q,i} = 1, A_{q,i} = A) = 1 - A$, yielding:

$$\begin{aligned} &p(a_{q,i}|C_{q,i} = 0, L_{j-1}) \propto \\ &\quad p(a_{q,i}|L_{j-1}) \sum_{A\in\{0,1\}} \Big(p(A_{q,i} = A|a_{q,i}) \\ &\qquad (1 \cdot p(E_{q,i} = 0) + (1 - A_{q,i}) \cdot p(E_{q,i} = 1)) \Big) \\ &\propto \;\; p(a_{q,i}|L_{j-1}) \sum_{A\in\{0,1\}} \Big(p(A_{q,i} = A|a_{q,i}) \end{aligned}$$

Algorithm 1 ComputePosteriors(L)

```
InitializePriors()
for $l_j$ in $L$ do
    for $i$ in $l_j$ do
        $p(E_{q,i}) \leftarrow sumProduct(l_j)$
        if $C_{q,i} = 1$ then
            $p(a_{q,i}|C_{q,i} = 1, L_{j-1}) \leftarrow p(a_{q,i}|L_{j-1})a_{q,i}$
            $p(s_{q,i}|C_{q,i+1} = 0, L_{j-1}) \leftarrow p(s_{q,i}|L_{j-1})$
                $(p(E_{q,i+1} = 1) + s_{q,i}p(E_{q,i+1} = 0))$
        else
            $p(a_{q,i}|C_{q,i} = 0, L_{j-1}) \leftarrow p(a_{q,i}|L_{j-1})$
                $(1 - a_{q,i} \cdot p(E_{q,i} = 1))$
return $p(a|L), p(s|L)$
```

$$
\begin{aligned}
&\quad (1 - p(E_{q,i} = 1) + p(E_{q,i} = 1) - A \cdot p(E_{q,i} = 1))\Big) \\
\propto \;\; & p(a_{q,i}|L_{j-1}) \sum_{A \in \{0,1\}} \Big(p(A_{q,i} = A|a_{q,i}) \\
&\quad (1 - A \cdot p(E_{q,i} = 1))\Big). \qquad (9)
\end{aligned}
$$

Now we can substitute $p(A_{q,i} = A|a_{q,i})$ for the values of A: $p(A_{q,i} = 1|a_{q,i}) = a_{q,i}$ and $p(A_{q,i} = 0|a_{q,i}) = 1 - a_{q,i}$ yielding:

$$
\begin{aligned}
&p(a_{q,i}|C_{q,i} = 0, L_{j-1}) \propto \\
&\qquad p(a_{q,i}|L_{j-1})(1 - a_{q,i} \cdot p(E_{q,i} = 1)). \qquad (10)
\end{aligned}
$$

Thus, (10) is the Bayesian update rule for attractiveness. When $p(E_{q,i} = 1) = 1$, e.g., if the document was clicked, the posterior is a Beta distribution because plugging $p(E_{q,i} = 1) = 1$ in (10) yields a term conjugate to the parametrization in (5). Otherwise, we use $p(E_{q,i}|L_j)$, as computed in the first step via a message-passing algorithm that takes into account the satisfactoriness of the last clicked document and the attractivenesses of the unexamined documents. Instead of a Beta distribution, this yields a more general polynomial function. The normalization constant can be found by integration: since the function is a polynomial with known coefficients, it can be integrated analytically and then the definite integral evaluated on the interval $[0, 1]$. The cumulative distribution function is therefore also a polynomial and easy to compute. The same holds for the expectation of the distribution.

Analogously, we can derive an update rule for satisfactoriness:

$$
\begin{aligned}
&p(s_{q,i}|C_{q,i+1} = 0, L_{j-1}) \propto \\
&\qquad p(s_{q,i}|L_{j-1})(p(E_{q,i+1} = 1) + s_{q,i}p(E_{q,i+1} = 0)). \qquad (11)
\end{aligned}
$$

Algorithm 1 summarizes the steps involved in computing $p(a_{q,i}|L)$ and $p(s_{q,i}|L)$. First, the attractiveness and satisfactoriness of all document query pairs are assigned uniform $Beta(1, 1)$ priors. Then, the log file is processed session by session and the attractiveness and satisfactoriness of all document query pairs in the session are updated using the Bayesian update rules described in this section: (6) and (10) for attractiveness and (11) for satisfactoriness.

5.2 The Switching Problem

To decide whether or not to switch from R_p to R_c, we must determine whether (1) holds. Determining this from L requires evaluating the following double integral:

$$
\begin{aligned}
&p(M_c + \epsilon \geq M_p \mid L) = \\
&\int_{M_c=0}^{M_c=M^*} p(M_c|L) \int_{M_p=0}^{M_p=M_c+\epsilon} p(M_p|L)\, dM_c\, dM_p, \qquad (12)
\end{aligned}
$$

Algorithm 2 BARACO-SP$(R_p, R_c, L, \epsilon, \delta)$

```
1: $p(a|L), p(s|L) \leftarrow computePosteriors(L)$          ▷ §5.1
2: for $k$ in 1:$K$ do                                         ▷ §5.2
3:     $M_p^k \leftarrow sample(R_p, p(a|L), p(s|L))$
4:     $M_c^k \leftarrow sample(R_c, p(a|L), p(s|L))$
5:     if $M_c^k + \epsilon \geq M_p^k$ then
6:         $N_{M_c+\epsilon \geq M_p} \leftarrow N_{M_c+\epsilon \geq M_p} + 1$
7: $p(M_c + \epsilon \geq M_p) \leftarrow \frac{N_{M_c+\epsilon \geq M_p}}{K}$
   return $p(M_c + \epsilon \geq M_p) \geq 1 - \delta$
```

where M^* is the maximum possible value of the metric defined in (4). Evaluating this integral requires knowing the posterior probabilities $p(M_p|L)$ and $p(M_c|L)$, which can be computed from (3) and (4) given $p(a_{q,i}|L)$ and $p(s_{q,i}|L)$ for each ranker, query, and document.

Computing (12) analytically is hard, but it can be estimated using a Monte-Carlo sampling scheme. Algorithm 2 describes this scheme, which yields a version of BARACO that solves the Switching Problem.

First, we compute the posteriors $p(a|L)$ and $p(s|L)$ using the approach described in §5.1 (line 1). Then, each sampling iteration k consists of drawing a sample $a_{q,i}$ and $s_{q,i}$ for each ranker, document, and query from $p(a_{q,i}|L)$ and $p(s_{q,i}|L)$. Using these samples, we compute M_c^k and M_p^k, the values of the metrics given the sampled probabilities from the k-th iteration (lines 3–4). Estimating (12) then reduces to computing the fraction of sampling iterations for which $M_c^k + \epsilon \geq M_p^k$ (lines 6–7). If this fraction is greater than $1 - \delta$, we can safely switch to the candidate ranker.

However, sampling $a_{q,i}$ and $s_{q,i}$ from $p(a_{q,i}|L)$ and $p(s_{q,i}|L)$ is itself hard because their cumulative distribution functions are high degree polynomials that are hard to invert. Therefore, we employ the Metropolis-Hastings algorithm [1] with the expected value of the sampled distribution, which can be calculated analytically through integration, as the starting position of the random walk. The proposal distribution is a Gaussian with fixed variance.

5.3 The Difference Estimation Problem

If we are interested only in the expected value of a ranker R_x, and not the uncertainty of this estimate, we can compute it as follows:

$$
\mathbb{E}[M_x] = \textstyle\int_{M_x=0}^{M_x=M^*} M_x p(M_x)\, dM_x.
$$

Using the same sampling procedure described above, we can solve the Difference Estimation Problem by estimating this expected value:

$$
\mathbb{E}[M_x] \approx \tfrac{1}{K} \textstyle\sum_{k=1}^{K} M_x^k, \qquad (13)
$$

where K is the number of sampling iterations. We can similarly estimate the expected difference between M_c and M_p:

$$
\mathbb{E}[M_c - M_p] \approx \tfrac{1}{K} \textstyle\sum_{k=1}^{K} (M_c^k - M_p^k). \qquad (14)
$$

Algorithm 3 summarizes the resulting version of BARACO that solves the Difference Estimation Problem. It is essentially the same as Algorithm 2 except that it estimates the expected difference between the production and the candidate rankers' metrics.

6. EXPERIMENTAL SETUP

In this section, we describe the experimental setup used to evaluate BARACO. This evaluation is complicated by the fact that the user's information need, and therefore the true relevance labels, are unknown. We present two evaluations which deal with this problem

Algorithm 3 BARACO-DEP($R_p, R_c, L, \epsilon, \delta$)

```
$p(a|L), p(s|L) \leftarrow computePosteriors(L)$    ▷ §5.1
$\sum_{\Delta_M} \leftarrow 0$
for $k$ in $1{:}K$ do    ▷ §5.3
    $M_p^k \leftarrow sample(R_p, p(a|L), p(s|L))$
    $M_c^k \leftarrow sample(R_c, p(a|L), p(s|L))$
    $\sum_{\Delta_M} \leftarrow \sum_{\Delta_M} + M_c^k - M_p^k$
$\mathbb{E}[M_x] \leftarrow \frac{\sum_{\Delta_M}}{K}$
return $\mathbb{E}[M_x]$
```

in two different ways. The first evaluation, based on the LETOR datasets [17], uses manual relevance assessments as ground-truth labels and synthetic clicks as feedback to BARACO. The second evaluation, based on the WSDM 2014 Web search personalization challenge,[1] uses dwell time as ground-truth labels and real clicks as feedback to BARACO.

Another possibility would be to evaluate BARACO using explicit relevance labels provided by human assessors to measure metrics such as Normalized Discounted Cumulative Gain (NDCG) but this approach is known to have low agreement with metrics based user behavior such as A/B testing or interleaving [4, 6, 19, 24], so it is natural to expect that it would have a low agreement with BARACO as well. It would also be possible to evaluate BARACO using A/B tests or interleavings, but A/B tests have low agreement with interleavings and different metrics collected during A/B tests such as click through rate, clicks@1 and others have low agreement with each other [19]. The only definitive way to evaluate BARACO would be in an industrial setting that measures long-term metrics such as engagement. Such results, however, would be difficult to reproduce. Consequently, the LETOR evaluation is the most reliable and reproducible, as it depends on indisputably unbiased ground truth. In addition, it allows us to explore how the discrepancy between the click model and user behavior affects BARACO's performance. However, the WSDM evaluation, though less reliable, is nonetheless useful because it gives insight into how BARACO performs in a real-world setting with real users.

We compare BARACO to a baseline that, in lieu of our Bayesian approach, uses the EM-based approach described in [3] to compute maximum a posteriori estimates of (4) for M_p and M_c. These estimates can then be directly used in the Difference Estimation Problem. Because this approach does not compute full posteriors, it does not quantify uncertainty in the resulting estimates and therefore cannot be directly used in the Switching Problem. Instead, the baseline method, which we call *Manual Thresholding (MT)*, resorts to a heuristic approach to determine whether (1) holds. In particular, R_c is deemed safe when $\hat{M}_c - \hat{M}_p > \epsilon_m$, where $\hat{M}_c$ and $\hat{M}_p$ are the maximum a posteriori estimates of M_c and M_p produced by the EM-based method and ϵ_m is a threshold parameter whose value must be tuned manually. Because we want to switch whenever $M_c - M_p > -\epsilon$, the quantity $\epsilon + \epsilon_m$ acts as a buffer, i.e., an extra gap that R_c must exceed to be considered safe. Adding this buffer heuristically accounts for the uncertainty in $\hat{M}_c$ and $\hat{M}_p$.

The need to tune ϵ_m for MT poses significant difficulties in practice. To understand the effect of ϵ_m on the behavior of MT would require access to ground truth about a set of candidate rankers, i.e., whether they are in fact no more than ϵ worse than the production ranker. While such ground truth could be obtained using off-line or on-line evaluations, such evaluations pose exactly the difficulties that motivate the need for methods like BARACO: the former is expensive and may not reflect real user preferences. The latter requires showing potentially poor rankers to real users. Furthermore, while such ground truth, even if it could be obtained, would shed light on how ϵ_m affects to which candidate rankers MT switches, it would still not make it possible to select the ϵ_m that is best for a given δ supplied by the system designers. Doing so would require a quantification of the uncertainty about each candidate ranker that is inherent to BARACO but absent in MT.

Importantly, BARACO does *not* require tuning ϵ_m or any analogous parameter. On the contrary, δ and ϵ, which are supplied by the system designers, are simply quantifications of their risk tolerance.

6.1 LETOR Evaluation

The first evaluation is based on the LETOR datasets [17], which include manual relevance assessments. However, they do not include user clicks. In addition, even if they did, these would likely correlate only poorly with the relevance assessments, since the assessors may not interpret the users' information need correctly. To address this difficulty, we instead axiomatically define the relevance assessments to be correct ground-truth labels and use click models to generate the clicks. Since BARACO also relies on click models, we evaluate it in settings where the clicks are generated and interpreted using different click models, to assess BARACO's robustness to errors in its modeling assumptions.

LETOR is split into six sub-datasets: HP2003, HP2004, NP2003, NP2004, TD2003, and TD2004. For each run of each algorithm, we use the data from one sub-dataset. First we train a ranker R_{Ada} using AdaRank [22] on all the data in this sub-dataset. AdaRank, which performs reasonably on all these datasets [22], trains a linear ranking function, i.e., a linear combination of the ranking features for a given query-document pair. The documents are then sorted based on the values produced by this ranking function.

To ensure that some candidate rankers will be better than the production ranker, we craft R_p by "damaging" R_{Ada}, i.e., randomly by adding random vectors to it. These vectors were generated by randomly sampling a normal distribution with mean equal to 0 and standard deviation of 0.2. Finally, to generate a population of candidate rankers R_c, we again perturb R_p 1000 times using the same sampling method. This ranker generation methodology is motivated by the gradient descent algorithm [25]; the standard deviation value was chosen so that some of the rankers would be similar enough to the production ranker and some too different from it for the algorithm to be confident about their performance.

The next step is to generate the log file. To this end, we *generate* user click interaction data using the DBN, sDBN and UBM click models with three user model settings: perfect, navigational and informational. The clicks are then *interpreted* using the DBN click model. The parameters of the click models are summarized in Table 1, where $p(C|R)$ and $p(C|NR)$ denote the probability of a user clicking a relevant document and an irrelevant document, respectively, and $p(s|R)$ and $p(s|NR)$ denote the probability of abandoning the SERP after clicking a relevant document and an irrelevant document, respectively. The closer $p(C|R)$ is to $p(C|NR)$ and $p(s|R)$ to $p(s|NR)$, the more noise there is in the feedback and the more difficult inference becomes. The user interaction data is generated for the production ranker by randomly sampling 500 queries and generating the SERPS and clicks.

The perfect user model setting is used to obtain an upper bound in performance: the user clicks all relevant documents and no irrelevant ones. The navigational and informational user models are based on typical user behavior in web search [11]. The navigational

[1] Personalized Web Search Challenge 2013 https://www.kaggle.com/c/yandex-personalized-web-search-challenge

Table 1: Overview of the user model settings.

Model	$p(C\|R)$	$p(C\|NR)$	$p(s\|R)$	$p(s\|NR)$
Perfect	1.0	0.0	0.0	0.0
Navigational	0.95	0.05	0.9	0.2
Informational	0.9	0.4	0.5	0.1

user model reflects user behavior while searching for an item they know to exist, such as a company's homepage. Because it is easy for users to distinguish between relevant and irrelevant documents, the noise levels are low. The informational user model reflects user behavior while looking for information about a topic, which can be distributed over several pages. Because this task is more difficult, there is more noise in the feedback.

The clicks are generated and interpreted using either the same or different click models. When they are generated and interpreted using the same click model, our findings are not affected by the accuracy of the assumptions in the click model, allowing us to focus on the differences between BARACO and the baseline method. Of course, some assumptions made by DBN, which is used to interpret clicks, may not always hold in practice. For example, DBN assumes that the document that was clicked and led to abandonment is the document that satisfied the user. This assumption typically holds for navigational queries, but may not be valid for informational queries. Therefore, experiments in which the clicks are generated and interpreted using different click models help measure the robustness of BARACO to settings whether the assumptions underlying DBN do not always hold.

In the LETOR Evaluation setup, we compare the performance of BARACO and MT using area under roc curves (AUC), Pearson correlation, and the square root of the mean squared error (RMSE). The rankers are compared using the metric rrMetric (3).

6.2 WSDM Evaluation

We additionally evaluate BARACO using an anonymized click log released by Yandex for the WSDM 2014 Web search personalization challenge. The click log contains sessions with queries submitted by a user. For each query there is a list of search results returned by the engine and the clicks produced by the user. The queries and the clicks have timestamps. However, the units of time are not disclosed. The queries, query terms, documents, and users are represented by IDs in order to protect the privacy of users.

The organizers of the challenge have defined three levels of relevance based on the clicks that the documents receive: documents that receive clicks with dwell time less than 50 time units have relevance 0, clicks with dwell time between 50 and 150 units have relevance 1, and clicks with dwell time of more than 2 time units as well as clicks that are the last clicks for a given query have relevance 2. We use all but the last session for a query for training and use the last session for extracting the relevance labels for query-document pairs.

The candidate rankers are generated by perturbing the result lists observed in the click log in a random way. In order to ensure that some of the candidates are better than the production ranker, the relevant documents have a higher chance to be promoted to top than the irrelevant ones.

In the WSDM Evaluation setup, we compare the performance of BARACO and MT using the following metrics: AUC and Pearson correlation as before. But we do not use RMSE because the graded relevance and the estimated relevance have different scales from 0 to 2, and from 0 to 1 respectively. The candidates are compared using DCG instead of the metric in (3) because (3) requires a mapping from relevance labels to attractiveness and satisfactoriness that is not available for the graded relevance in the click log—it could be computed using, e.g., DBN but then the evaluation would be less sound because the same click model would be used for training the parameters of the documents and for training the metric.[2]

6.3 Parameter Settings

Both BARACO and MT are instantiated with the user persistence parameter $\gamma = 0.9$, as in [3, 7], $\epsilon = 0.01$. For the Metropolis-Hastings sampling procedure described in §5.2, the variance of the proposal distribution was set to 0.1, which was determined experimentally to provide a rejection ratio of around 0.6. For each query/document pair, $N_{samples} = 1000$ samples are drawn and shuffled to reduce autocorrelation in order to evaluate (1). All results are averaged over 30 independent runs for each algorithm. In order to check the significance of observed differences between results, we perform Wilcoxon signed-rank tests; in the result tables, ▲ denotes a significant difference at $p = 0.01$, and △ at $p = 0.05$.

7. RESULTS

In this section, we present our experimental results aimed at answering RQ1 and RQ2. In §7.1, we analyse the performance of BARACO and MT on the LETOR data; in §7.2, we analyse their performance on the WSDM data.

7.1 LETOR Results

In §7.1.1, we compare BARACO and MT on the Switching Problem; in §7.1.2, we compare BARACO and the EM-based approach [3] that underlies MT on the Difference Estimation Problem.

7.1.1 Switching Problem Results

To address RQ1, we compare the ROC curves of BARACO and MT on the Switching Problem. Such curves show how the true and false positives of both methods change when we fix $\epsilon = 0.01$ and vary across different values of $\delta \in [0, 1]$ for BARACO and $\epsilon_m \in [-4\epsilon, \epsilon]$ for MT. In this context, a true positive occurs when the algorithm recommends switching to R_c and $M_c + \epsilon \geq M_p$, while a false positive occurs when it recommends switching but $M_c + \epsilon < M_p$. We then compare the AUC of both methods for different datasets and user and click models.

Note that this comparison is fundamentally unfair to BARACO because its parameter, δ, does not require tuning but instead is input by the system designers as a quantification of their risk tolerance. By contrast, ϵ_m is a parameter that requires manual tuning and cannot be derived from δ, which MT ignores. As discussed in §6, tuning this parameter is quite difficult in practice. Because the ROC curves show performance across different values of δ and ϵ_m, they allow MT to "cheat" by optimizing its critical parameter, a step that is unnecessary in BARACO. In other words, these ROC curves answer the following question: for each value of δ that a system designer could input to BARACO, how well can MT match the quality of BARACO's decisions about when to switch to R_c *if an oracle provides MT with the best possible value of* ϵ_m? Thus, BARACO can be considered a success if it can match the performance of MT when MT is given this advantage.

Fig. 3 plots the area under the ROC curves for BARACO and MT for all six data sets, three click models, and three user model settings. Fig. 4 shows the ROC curve for the TD2004 dataset, informational user model with the DBN model used for generation. The other ROC curves, omitted for brevity, are qualitatively similar.

[2] In the case of the LETOR evaluation experiments, this mapping is known and can be read from Table 1.

The results in Fig. 3a show that, even at the best value of ϵ_m, BARACO substantially outperforms MT on four of the six datasets (HP2003, HP2004, NP2003, TD2004). On the other two datasets (NP2003 and TD2003), the two methods perform similarly. Analysis of the latter two datasets shows that the production ranker was at a local minimum. Hence, nearly all candidate rankers are better than the production ranker and the best performance is obtained by always switching. As this degenerate policy can be represented just as well by MT as by BARACO, the two perform similarly.

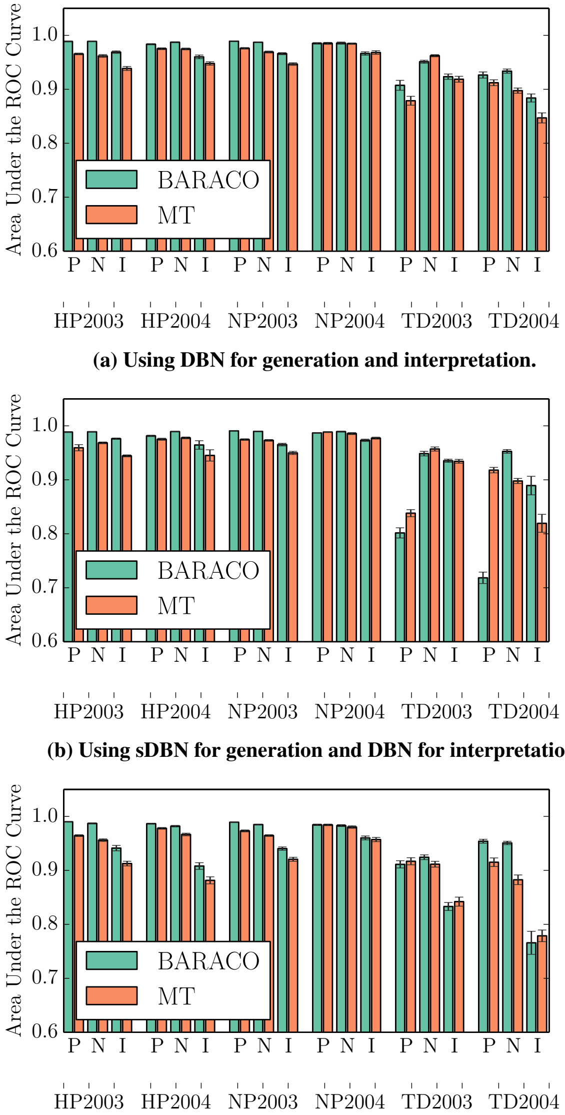
Furthermore, the fact that BARACO performs nearly as well when clicks are generated and interpreted with different models, as shown in Figs. 3b and 3c, shows that BARACO is robust to violations of its modeling assumptions. The baseline substantially outperforms BARACO only for the perfect sDBN user model on the TD2003 and TD2004 datasets. The baseline shows superior performance because there are many more relevant documents in the TD datasets, and many of them are not presented to the user by the production ranker. In the case of the perfect user model, the user only clicks on relevant documents. Therefore, there are many queries for which no clicks are produced because no relevant documents were shown. The baseline essentially removes such queries from consideration through condensing [6], which may be an effective strategy in this case. Overall, these results demonstrate that BARACO can offer a robust and effective means for deciding when to switch rankers: especially in cases where its modeling assumptions hold, it outperforms MT with a tuned ϵ_m parameter for nearly all combinations of dataset and user model.

The values of ϵ_m shown here were chosen because their inflection point lies in the interval $\delta \in [0, 1]$. These are all negative values of ϵ_m because MT has a consistent negative bias: almost all candidate rankers receive a score lower than the production ranker. This bias is a consequence of condensing [6]: almost all candidate rankers have unseen documents that do not contribute to the metric. This further highlights the brittleness of MT: as more data is collected, the relative number of unseen documents may decrease, which would reduce the effect of condensing and therefore the amount of bias, necessitating a retuning of ϵ_m.

7.1.2 Difference Estimation Problem Results

To address RQ2, we compare BARACO to the EM-based method [3] that underlies MT, on the Difference Estimation Problem. BARACO uses (14) to estimate the expected difference while the EM-based method uses $\hat{M}_c - \hat{M}_p$, where $\hat{M}_x$ is the maximum a posteriori estimate of M_x. First, we consider how the RMSE scores of the two approaches differ. Error is defined here to be the difference between the true and estimated value of $M_c - M_p$. The true values are computed from the expert-generated relevance labels in the datasets. Table 2 summarizes the MSE of BARACO and MT. These results show that BARACO consistently outperforms the EM-based approach, in many cases by an order of magnitude. The only exception is the TD2004 dataset for clicks generated using UBM. This exception occurs because there are many unseen relevant documents in the TD2004 dataset and, when the user model assumptions do not hold, the baseline's condensing strategy [6] may be more effective because it does not rely on these assumptions.

However, the fact that BARACO has lower RMSE scores is important only if we are interested in the absolute values of the metric differences. Instead, if we want to be able to rank the candidate rankers by their metric values, we need a different way to compare the methods. To this end, we measure the Pearson's correlation between the ground truth value $M_c - M_p$ and the estimates produced by BARACO or MT. For example, if the correlation with the ground truth was perfect, ordering all the candidate rankers by their ground truth difference with the production ranker would be the same as ordering them by the estimated difference. Thus, the correlation with the ground truth is more informative than RMSE in cases where we care about preserving the ground-truth ranking. This occurs, e.g., when several candidate rankers confidently outperform the production ranker. In such cases, it is desirable to switch to the one that outperforms it by the largest margin, while the exact absolute values of the estimated metrics are not important.



(a) Using DBN for generation and interpretation.

(b) Using sDBN for generation and DBN for interpretation.

(c) Using UBM for generation and DBN for interpretation.

Figure 3: AUC for all data sets, user and click models. The error bars are standard errors of the means. P - perfect user model setting, I - informational, N - navigational (LETOR evaluation).

Table 3 summarizes the correlations between the ground truth difference of rankers and the difference of rankers computed by BARACO and the EM-based method. Higher correlations with the ground truth mean that the way the rankers are ranked is closer to the ground truth. These results show that BARACO again outperforms the EM-based method. The negative correlation in the informational setting of NP2003 dataset is due to a heavily skewed distribution of candidate rankers when the production ranker is at a

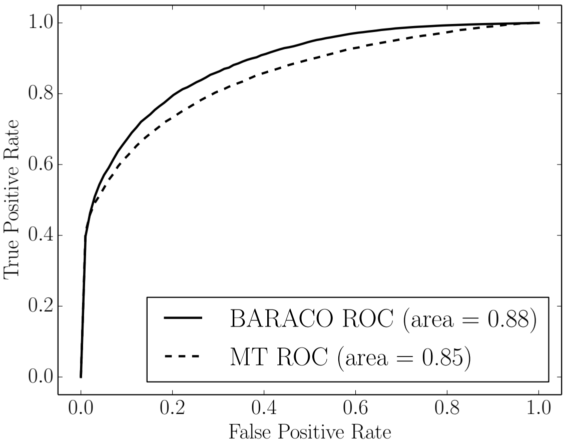


Figure 4: The ROC curves for BARACO and MT, using DBN for generation and interpretation, TD2004 dataset, informational user model (LETOR evaluation).

Table 2: RMSE between the predicted outcome of the comparison and the ground truth, P - perfect user model setting, I - informational, N - navigational (LETOR evaluation).

		DBN		sDBN		UBM	
Dataset	UM	BARACO	EM	BARACO	EM	BARACO	EM
HP2003	P	3.1e−5$^\blacktriangle$	3.0e−4	3.8e−5$^\blacktriangle$	3.1e−4	3.5e−5$^\blacktriangle$	3.2e−4
	N	2.8e−5$^\blacktriangle$	3.5e−4	2.6e−5$^\blacktriangle$	3.1e−4	4.7e−5$^\blacktriangle$	3.0e−4
	I	6.1e−5$^\blacktriangle$	2.7e−4	4.1e−5$^\blacktriangle$	2.7e−4	1.1e−4$^\blacktriangle$	2.6e−4
HP2004	P	2.7e−5$^\blacktriangle$	2.3e−4	4.3e−5$^\blacktriangle$	2.0e−4	2.7e−5$^\blacktriangle$	2.4e−4
	N	1.6e−5$^\blacktriangle$	2.4e−4	1.9e−5$^\blacktriangle$	2.3e−4	6.0e−5$^\blacktriangle$	1.9e−4
	I	6.2e−5$^\blacktriangle$	2.0e−4	5.3e−5$^\vartriangle$	1.6e−4	2.2e−4	2.4e−4
NP2003	P	1.3e−5$^\blacktriangle$	2.9e−4	1.5e−5$^\blacktriangle$	2.8e−4	1.4e−5$^\blacktriangle$	2.9e−4
	N	1.4e−5$^\blacktriangle$	3.0e−4	1.3e−5$^\blacktriangle$	2.8e−4	3.3e−5$^\blacktriangle$	2.5e−4
	I	3.7e−5$^\blacktriangle$	2.8e−4	3.8e−5$^\blacktriangle$	2.7e−4	1.1e−4$^\blacktriangle$	2.7e−4
NP2004	P	4.1e−6$^\blacktriangle$	6.2e−5	6.6e−6$^\blacktriangle$	5.8e−5	4.3e−6$^\blacktriangle$	6.6e−5
	N	4.9e−6$^\blacktriangle$	8.9e−5	3.6e−6$^\blacktriangle$	7.8e−5	7.9e−6$^\blacktriangle$	6.4e−5
	I	1.5e−5$^\blacktriangle$	9.8e−5	1.5e−5$^\blacktriangle$	1.0e−4	2.5e−5$^\blacktriangle$	1.2e−4
TD2003	P	7.0e−5$^\blacktriangle$	1.1e−4	3.7e−4$^\blacktriangledown$	1.2e−4	6.8e−5$^\blacktriangle$	1.7e−4
	N	8.0e−5$^\blacktriangle$	3.5e−4	6.2e−5$^\blacktriangle$	3.8e−4	7.6e−5$^\blacktriangle$	1.9e−4
	I	6.3e−5$^\blacktriangle$	2.7e−4	6.9e−5$^\blacktriangle$	2.7e−4	1.5e−4$^\blacktriangle$	3.0e−4
TD2004	P	4.0e−4$^\blacktriangle$	7.8e−4	1.2e−3$^\blacktriangledown$	5.9e−4	5.1e−4$^\blacktriangle$	9.2e−4
	N	3.2e−4$^\blacktriangle$	2.3e−3	2.5e−4$^\blacktriangle$	2.3e−3	6.0e−4$^\blacktriangle$	1.2e−3
	I	4.7e−4$^\blacktriangle$	2.0e−3	4.2e−4$^\blacktriangle$	1.7e−3	1.2e−3$^\blacktriangle$	1.5e−3

local minimum of the ranker space and almost all candidate rankers are better for the queries in which the production ranker has not presented any relevant documents. This situation is unlikely to occur in the real world since production rankers are typically highly engineered and thus more likely to be local maxima than local minima. As with our earlier results, we see that the performance on the TD2004 dataset using UBM for generation is qualitatively different from the other conditions for the reasons mentioned earlier.

To answer RQ2, we observe that both the RMSE and correlation results show that BARACO outperforms MT: BARACO achieves better estimates in both absolute and relative terms, except on the TD2004 dataset with the UBM click model for generation, whose special nature has been recognized before.

7.2 WSDM Results

In this section, we compare BARACO and MT on the Switching Problem and Difference Estimation Problem, respectively, using the WSDM experimental setup

Table 3: Correlation between the predicted outcome of the comparison and the ground truth, P - perfect user model setting, I - informational, N - navigational (LETOR evaluation)

		DBN		sDBN		UBM	
Dataset	UM	BARACO	EM	BARACO	EM	BARACO	EM
HP2003	P	0.961$^\blacktriangle$	0.846	0.950$^\blacktriangle$	0.858	0.963$^\blacktriangle$	0.857
	N	0.967$^\blacktriangle$	0.856	0.961$^\blacktriangle$	0.856	0.950$^\blacktriangle$	0.808
	I	0.914$^\blacktriangle$	0.763	0.930$^\blacktriangle$	0.744	0.510$^\blacktriangledown$	0.664
HP2004	P	0.963$^\blacktriangle$	0.933	0.946	0.937	0.969$^\blacktriangle$	0.940
	N	0.980$^\blacktriangle$	0.926	0.981$^\blacktriangle$	0.943	0.960$^\blacktriangle$	0.908
	I	0.931$^\blacktriangle$	0.825	0.935$^\vartriangle$	0.825	0.262$^\blacktriangledown$	0.612
NP2003	P	0.979$^\blacktriangle$	0.890	0.982$^\blacktriangle$	0.901	0.982$^\blacktriangle$	0.897
	N	0.973$^\blacktriangle$	0.878	0.982$^\blacktriangle$	0.891	0.969$^\blacktriangle$	0.856
	I	0.902$^\blacktriangle$	0.777	0.887$^\blacktriangle$	0.769	0.523$^\blacktriangledown$	0.678
NP2004	P	0.922$^\blacktriangle$	0.841	0.915$^\blacktriangle$	0.871	0.925$^\blacktriangle$	0.861
	N	0.923$^\blacktriangle$	0.809	0.943$^\blacktriangle$	0.835	0.864$^\blacktriangle$	0.751
	I	0.700$^\blacktriangle$	0.571	0.678$^\blacktriangle$	0.546	0.464$^\vartriangle$	0.393
TD2003	P	0.738$^\vartriangle$	0.678	0.398$^\blacktriangledown$	0.619	0.732	0.723
	N	0.838$^\blacktriangle$	0.813	0.873$^\vartriangle$	0.830	0.758$^\blacktriangle$	0.675
	I	0.777$^\blacktriangle$	0.689	0.776$^\blacktriangle$	0.702	0.009 67$^\blacktriangledown$	0.404
TD2004	P	0.846$^\vartriangle$	0.783	0.434$^\blacktriangledown$	0.817	0.887$^\blacktriangle$	0.795
	N	0.862$^\blacktriangle$	0.776	0.890$^\blacktriangle$	0.753	0.858$^\blacktriangle$	0.693
	I	0.755$^\blacktriangle$	0.633	0.829	0.637	−0.156$^\blacktriangledown$	0.464

7.2.1 *Results for the Switching Problem*

To address RQ1, we compare the ROC curves of BARACO and MT on the Switching Problem. The AUC of these curves for both methods are stated in Table 4. The AUCs illustrate that both methods are able to distinguish between strong and weak candidates. Both methods suffer from a weak, systematic bias—they consistently underestimate the quality of the candidates because the relevance labels used in the evaluation are biased towards top results.

Table 4: AUC and Correlation between the predicted outcome of the comparison and the ground truth (WSDM evaluation).

	BARACO	Manual Thresholding
AUC	0.936	0.934
Correlation	0.751	0.735

The real proportion of candidate rankers that are better than the production ranker across different probability levels computed by BARACO is summarized in Table 5. The probabilities output by BARACO are not perfectly calibrated and instead tend to be underestimates. Thus, not all risk prescribed by δ and ϵ can be utilized, making the system somewhat overly conservative.

Unfortunately, a limitation of these WSDM experiments is that there is no way to ascertain how much of this bias is due to discrepancies between the relevance labels and the true information needs of the users who generated the clicks and how much is due to discrepancies between BARACO's click model and those users' behavior. However, because the bias is consistent, correcting for this bias, e.g., by learning a constant offset, is straightforward.

7.2.2 *Results for the Difference Estimation Problem*

The correlations between the true and estimated value of $M_c - M_p$ computed by BARACO and MT are also stated in Table 4. The estimated difference between rankers is strongly correlated with the ground truth in the WSDM dataset, suggesting that both methods can estimate the difference between rankers well given the logged user interactions with the production ranker.

Table 5: **Proportion of candidate rankers that beat the production ranker across probability levels computed by BARACO (WSDM evaluation).**

Probability Level	0–0.1	0.1–0.2	0.2–0.3	0.3–0.4	0.4–0.5	0.5–0.6	0.6–0.7	0.7–0.8	0.8–0.9	0.9–1
Proportion	0.004	0.037	0.139	0.312	0.546	0.742	0.886	0.971	0.993	1.0

In both experimental setups, i.e., the LETOR and WSDM setup, both BARACO and MT have good performance in most cases and high agreement with the ground truth. In the LETOR setup the performance is often superior to that in the WSDM experiments, especially when there is little noise and no discrepancy between the user behavior and the click model. However, when there is much noise and the clicks are generated and interpreted using different click models, the performance drops to levels lower than in the WSDM experiments. Overall, these results show that, given a reasonable click model, BARACO makes it possible to make informed decisions whether or not to switch to a candidate ranker given historical user interaction data obtained using the production ranker.

8. CONCLUSIONS AND FUTURE WORK

We presented BARACO, a new click model-based method of ranker comparison with two key features: (1) it compares the performance of rankers using only historical data, and (2) it quantifies the uncertainty in such comparisons. Using BARACO, it is possible to decide, using only historical data collected with the current production ranker, whether one can confidently replace the production ranker with a candidate ranker. The algorithm takes as input ϵ, the degree to which the candidate ranker is allowed to be worse than the production ranker; δ, the probability with which the candidate ranker is allowed to fall outside this threshold; and user interaction logs collected with the production system.

Our experiments show that BARACO can correctly and confidently identify candidate rankers that are ϵ as good as the production ranker. BARACO outperforms or has performance comparable to that of the MT baseline that requires manual tuning of the threshold ϵ_m through offline assessments or online user experiments.

A natural application of BARACO is within online learning to rank algorithms, many of which require an evaluation subroutine. For example, in Dueling Bandit Gradient Descent (DBGD) [25], the algorithm randomly picks a candidate ranker from the neighborhood of the current ranker, compares it to the current ranker, and, if the candidate ranker appears to be better, updates the current ranker so that is is closer to the candidate ranker. The evaluation step is usually done using interleaving, which requires showing potentially poor rankings to the user. Using BARACO, DBGD could be performed while restricting the rankings shown to users to those generated by production rankers or candidate rankers in which we have sufficient confidence.

Acknowledgements. This research was supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 312827 (VOX-Pol), the Netherlands Organisation for Scientific Research (NWO) under project nrs 727.011.005, 612.001.116, HOR-11-10, 640.006.013, 612.066.930, CI-14-25, SH-322-15, Amsterdam Data Science, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), the Netherlands eScience Center under project number 027.012.105, the Yahoo! Faculty Research and Engagement Program, the Microsoft Research PhD program, and the HPC Fund.

REFERENCES

[1] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[2] B. Carterette. System effectiveness, user models, and user utility: a conceptual framework for investigation. In *SIGIR '11*, pages 903–912. ACM, 2011.

[3] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *WWW '09*, 2009.

[4] O. Chapelle, D. Metzler, Y. Zhang, and P. Grinspan. Expected reciprocal rank for graded relevance. In *CIKM '09*, pages 621–630. ACM, 2009.

[5] O. Chapelle, T. Joachims, F. Radlinski, and Y. Yue. Large-scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Systems*, 30(1):6, 2012.

[6] A. Chuklin, A. Schuth, K. Hofmann, P. Serdyukov, and M. de Rijke. Evaluating aggregated search using interleaving. In *CIKM '13*, pages 669–678. ACM, 2013.

[7] A. Chuklin, P. Serdyukov, and M. de Rijke. Click model-based information retrieval metrics. In *SIGIR '13*, page 493. ACM, 2013.

[8] C. W. Cleverdon, J. Mills, and E. Keen. Factors determining the performance of indexing systems (Volume 1: Design). *Cranfield: College of Aeronautics*, 1966.

[9] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR '08*, pages 331–338. ACM, 2008.

[10] F. Guo and Y.-m. Wang. Efficient multiple-click models in web search. In *WSDM '09*. ACM, 2009.

[11] F. Guo, L. Li, and C. Faloutsos. Tailoring click models to user goals. In *Proceedings of the 2009 workshop on Web Search Click Data*, pages 88–92. ACM, 2009.

[12] J. He, C. Zhai, and X. Li. Evaluation of methods for relative comparison of retrieval systems based on clickthroughs. In *CIKM '09*, pages 2029–2032. ACM, 2009.

[13] K. Hofmann, F. Behr, and F. Radlinski. On caption bias in interleaving experiments. In *CIKM '12*, pages 115–124. ACM, 2012.

[14] K. Hofmann, A. Schuth, S. Whiteson, and M. de Rijke. Reusing historical interaction data for faster online learning to rank for IR. In *WSDM '13*, pages 183–192. ACM, 2013.

[15] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02*, pages 133–142. ACM, 2002.

[16] R. Kohavi, R. Longbotham, D. Sommerfield, and R. Henne. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery*, 18:140–181, 2009.

[17] T. Qin, T.-Y. Liu, J. Xu, and H. Li. LETOR: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval*, 13(4):346–374, 2010.

[18] F. Radlinski and N. Craswell. Comparing the sensitivity of information retrieval metrics. In *SIGIR '10*, pages 667–674. ACM, 2010.

[19] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *CIKM '08*, pages 43–52. ACM, 2008.

[20] M. Sanderson. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4(4):247–375, 2010.

[21] A. Schuth, F. Sietsma, S. Whiteson, D. Lefortier, and M. de Rijke. Multileaved comparisons for fast online evaluation. In *CIKM '14*, pages 71–80. ACM, 2014.

[22] J. Xu and H. Li. Adarank: A boosting algorithm for information retrieval. In *SIGIR '07*, pages 391–398, New York, NY, USA, 2007. ACM.

[23] E. Yilmaz, M. Shokouhi, N. Craswell, and S. Robertson. Expected browsing utility for web search evaluation. In *CIKM '10*, pages 1561–1564. ACM, 2010.

[24] E. Yilmaz, M. Verma, N. Craswell, F. Radlinski, and P. Bailey. Relevance and effort: an analysis of document utility. In *CIKM '14*, pages 91–100. ACM, 2014.

[25] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML '09*, pages 1201–1208. ACM, 2009.