Bypass Rates: Reducing Query Abandonment using Negative Inferences

Atish Das Sarma*
College of Computing
Georgia Tech.
atish@cc.gatech.edu

Sreenivas Gollapudi
Microsoft Search Labs
Microsoft Research
sreenig@microsoft.com

Samuel Ieong†
Dept. of Computer Science
Stanford University
sieong@cs.stanford.edu

ABSTRACT

We introduce a new approach to analyzing click logs by examining both the documents that are clicked and those that are *bypassed*—documents returned higher in the ordering of the search results but skipped by the user. This approach complements the popular click-through rate analysis, and helps to draw *negative inferences* in the click logs. We formulate a natural objective that finds sets of results that are unlikely to be collectively bypassed by a typical user. This is closely related to the problem of reducing *query abandonment*. We analyze a greedy approach to optimizing this objective, and establish theoretical guarantees of its performance. We evaluate our approach on a large set of queries, and demonstrate that it compares favorably to the *maximal marginal relevance* approach on a number of metrics including *mean average precision* and *mean reciprocal rank*.

Categories and Subject Descriptors

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

General Terms

Algorithms, Experimentation

Keywords

Query abandonment, Relevance, Bypass rates, Similarity search, Random walk

1. INTRODUCTION

Query abandonment is a major challenge faced by many search engines today. When interacting with a search engine, a user typically starts by issuing a few keywords related to the documents she is looking for. Such query is often *ambiguous*—it may be related to a number of topics. If none of the returned documents is relevant to her intended query, the user may reformulate the query and try again. After a few attempts, if the user does not find any relevant document, she may decide to quit. We say that the user *abandons* the query. In this paper, we define *query abandonment* as the scenario in which the user does not find *any* of the search results to satisfy her intent.

Search engines have a clear incentive to reduce query abandonment. To do so, however, requires a better understanding of user intentions when queries are ambiguous. As search engines often keep detailed logs of user interactions (*click logs*), an important question is how best to data-mine these logs for user preferences. Research in log analysis has been very active over the past few years, and has played a major role in improving relevance in web search and ad placement.

The basic premise behind click log analysis is that users are more likely to click on results that they find relevant. Thus, clicks are like users' "votes" for websites, and sites with more votes are likely to be more relevant. In our view, one equally important and yet often neglected source of signals in logs are the websites that the users do *not* click on. Interesting negative inferences can be drawn from such observations. In this paper, we introduce a complementary notion known as the *bypass rates* (BPR). We show how BPR addresses certain shortfalls of CTR. We also formulate an objective in terms of bypass rates, and demonstrate empirically that optimizing the stated objective leads to an improvement of the quality of the returned results.

Since CTR normalizes the number of clicks by dividing it by the number of impressions, why is it inadequate in analyzing these negative signals? There are two main problems. First, CTR does not differentiate between the reasons behind why results are not clicked. Second, it does not draw any inferences about how the quality of the clicked and non-clicked results are related. We will now elaborate on these shortfalls, and explains how BPR addresses them.

Under the assumption that users read the returned results in a top-down fashion, there are three reasons why a user decides not to click on a result. The first reason is that the user does not find the result relevant. The second reason is that the user has already found a relevant result earlier in the list, and hence did not read further results. As mentioned, CTR does not differentiate between these two cases. The third reason, which we do not consider in this study, is that users often simply stop looking, presumably because they didn't think it promising to keep looking once they have seen the first few results. We make this clear in Section 3.1. In our analysis, we call a result *bypassed* only if it is not clicked

*Work done while author was an intern at Search Labs.
†Work done while author was an intern at Search Labs.

KDD'08, August 24–27, 2008, Las Vegas, Nevada, USA.

and a result further down the list is clicked. By focusing on only bypassed documents, BPR counts only those lack of clicks that can be meaningfully attributed to less relevance.

Another problem with simple CTR analysis is that the relative relevance of the clicked and bypassed documents is ignored. By piecing together the available information, stronger inferences can sometimes be drawn. For example, if a result is bypassed in favor of a document of low relevance, the bypassed result is likely to be of low relevance as well. By assigning different penalties to the bypassed results depending on what results are eventually clicked, BPR takes into account the extra available inferences.

We next consider optimization problems involving BPR, and how they can help to improve the quality of the returned results. Based on its definition, BPR is naturally related to the likelihood that a result is not relevant to the user. We believe that there are many natural meaningful objectives involving BPR, and propose and analyze one that is related to the idea of *query abandonment* in this paper. The basic idea is that a user may abandon a query if she finds none of the results relevant. Hence, one should return results that have low bypass rates. At the same time, if a user bypasses a given document, she is also likely to bypass other *similar* documents . Thus, our objective tries to balance the inclusion of results with low BPR and low similarity with one another. The formal objective is given in Section 4.

We analyze the stated objective both theoretically and empirically. From a theoretic standpoint, we show that while the objective is NP-hard to optimize, a greedy approach is likely to work well due to its close relation with clustering. From an empirical standpoint, we compare our algorithms with a well known measure for diversity, *viz*, *maximal marginal relevance* (MMR) and show that it consistently outperforms MMR under a number of metrics, including *mean average precision* (MAP) and *mean reciprocal rank* (MRR).

In the remainder of the paper, we start by reviewing some related work in Section 2. In Section 3 we introduce notation, our setting and the assumptions we make. We formulate the precise optimization problem in Section 4 targeted towards obtaining a result set for queries to minimize query abandonment. Our theoretical results, including NP-hardness and an efficient greedy algorithm for a special case are detailed in Section 5. Finally, we conclude with experimental results and possible directions to extend our work in Section 6.

2. RELATED WORK

Due to its simplicity and wide applicability, there has been a lot of study analyzing *click through rates* (CTR) [8, 9, 15, 16]. Most of this research is based on two common assumptions used in click analysis -

- *independence* - clicks or *skips* on a document are independent of the user actions on the other documents on the page; and
- *positive feedback* - treat a click as a "vote" approving the document as a relevant result for the query.

In our work, we address the dependent nature of clicks by considering the order in which search results are returned, and our analysis of bypassed results capture the tacit negative feedback the users provide.

In [9], Joachims *et al* suggest several rules for inferring user preferences from click logs. For example, the rule "`CLICK > SKIP ABOVE`" suggests that if a user skip several search results and then clicks on a later result, this should be interpreted as the user preference for the clicked document is greater than for those skipped above it. This kind of reasoning is very similar to what we propose. The main difference is that we explicitly quantify how important this inference is, by considering not only the position of where the documents are, but also the interdependence of the documents. We will explain the notion of bypass rate in Section 3.2.

Our work is also related to the problem of *subtopic retrieval*, proposed by Zhai, Cohen, and Lafferty in [20]. They study the problem of how to handle queries that may be related to multiple topics (subtopics), and suggests a solution that attempts to find diverse sets of results that cover as many topics as possible. Their approach is based on the *maximum marginal relevance* (MMR) technique, first introduced in Carbonell and Goldstein [1]. As we will compare our algorithm to this, we now explain MMR in more detail.

Let Sim_1 be a similarity function between queries and documents, and Sim_2 be that between pairs of documents. The MMR technique greedily selects documents to be added to the set of returned results based on two factors—*relevance* and *novelty*. Specifically, let Q be the query, R be the universe of documents, and S be the current selected set of documents. The next document to be added is computed as

$$\arg \max_{D_i \in R \setminus S} \left[\lambda Sim_1(D_i, Q) - (1-\lambda) \max_{D_j \in S} Sim_2(D_i, D_j) \right] \quad (1)$$

where λ is a parameter to the algorithm that controls the diversity of the set. For example, when λ equals 1, documents are selected purely based on relevance, whereas when λ equals 0, documents are selected purely for diversifying the results. As a result, the output of MMR is sensitive to the choice of the parameter λ. In contrast, our approach is parameter-free. Also, our approach also produces an explicit score that can be interpreted as the collective probability that a set of documents will be bypassed; MMR is purely algorithmic and only select documents to be added.

Using BPR, we propose a natural objective that is related to minimizing the probability that none of the returned results is relevant to the user. This can be seen as an instantiation of the general risk minimization framework proposed by Zhai and Lafferty in [21]. Using their terminology, the loss function we propose is one where utility is lost whenever no result is relevant. This is related to the problem of query abandonment.

Radlinksi, Kleinberg, and Joachims [14] study the problem of abandonment using the notion of dependence of a document's relevance on other documents. Specifically, they propose online learning algorithms that learn a diverse ranking of documents based on clicking behavior of the users.

An important work along the same line as ours is that of Chen and Karger [3]. Under the Bayesian information retrieval framework, they propose a greedy algorithm that naturally return diverse results. They evaluate their approach on the TREC data set, and show improvement in metrics including the %no metric, MRR, and MSL over approaches based on the *Probability Ranking Principle* [18]. However, their approach does not specify how the parameters that correspond to the likelihoods that documents are

relevant to users are estimated. Experimentally, they tune these parameters for performance. In contrast, our approach is parameter-free.

Finally, our approach also bears some relation to clustering, as we make use of similarity functions between pairs of documents to tacitly diversify the set of results. Clustering has also been proposed in [10] to obtain a diverse range of interpretations for a given query. Our work is more general, however, as we consider both *hard clustering* and *soft clustering* of the results through different similarity functions.

3. LOG ANALYSIS

In this section, we explain the information we are interested in learning from the click logs. First, we describe two basic user behavioral assumptions that drive our analysis. Based on these assumptions, we derive a formula for estimating the BPR of a single document. We then explain a notion of similarity based on random walks in click graphs, first proposed in [5]. These estimates will be combined to form our optimization objective in Section 4.

3.1 User Behavioral Assumptions

We make two basic but important assumptions about user behavior that allows us to reason about the click logs.

ASSUMPTION 3.1. *Users only click on documents that are* relevant *to their queries.*

In other words, we assume that there are no *erroneous* clicks. In practice, some filtering may be required to get a clean data set; we assume that the data is already clean.

ASSUMPTION 3.2. *Users are* top-down *readers. When presented with an* ordered *set of results* $\{u_1, u_2, \ldots, u_l\}$*, a user clicks on document* u_i *if she prefers* u_i *to all documents that come before it,* $\{u_1, u_2, \ldots, u_{i-1}\}$.

Note that we do not draw any inference regarding the user preference between the clicked document and those that come after it. This is because a user may have been satisfied with her query by the clicked document, and not proceed further with the rest of the list. Another point to note is that we assume that each record from the click log is associated with only one user click. Thus, if the user clicks on two documents for a given query, this will result in two records corresponding to each click.

3.2 Bypass Rate (BPR)

In defining BPR, we want to capture two factors that are not properly captured in CTR. First, whether a user *chooses* not to click on a document. We say a document is *bypassed* if it is not clicked while some other document that appears later in the ordered set is clicked. Based on Assumption 3.2, bypassed documents are not clicked by choice. Second, the relative quality of the documents that are bypassed and clicked. We want to capture the intuition that the relevance of the bypassed document can only less than the relevance of the clicked document.

The first factor can be addressed by careful accounting. The second, however, requires an estimate of the quality of the clicked documents. We choose to heuristically estimate these by *position-dependent CTR*. This is by no means definitive; it is possible that some other estimates will give BPR higher predictive power. For example, one possibility is to use (1 - BPR) as the estimate. However, this requires computing a fixed point over the set of the documents, and is likely to be computationally inefficient. Future research may shed more light on how best to capture this factor.

For a given query q, we define the position-dependent CTR of a document u in position i, $CTR_i(u, q)$, as

$$CTR_i(u, q) = \frac{C_i(u, q)}{N_i(u, q)}$$

where $N_i(u, q)$ is the number of *effective* impressions of document u in position i for query q, and $C_i(u)$ the corresponding number of clicks. We say an impression is effective if and only if either u is clicked or some other document that appears later in the ordering of the returned results is clicked. In particular, we do not count an impression if the user has not clicked on any document. Under the stated assumptions, this style of accounting ensures that the user has indeed chosen to click (or not to click) on a given document.

Let S_{ij} denote the set of all instances where u is displayed in position i and some document v got clicked on position j. We define the *bypass rate* of a document u for query q, $B(u, q)$, as

$$B(u, q) = \frac{\sum_v \sum_{S_{ij}: j>i} (1 - CTR_j(v, q))}{\sum_v \sum_{S_{ij}: j>i} 1}$$

The interpretation of BPR as defined is that each time a document u is bypassed and a document v further down the list is clicked, u pays a penalty depending on the quality of document v, which we approximates with $(1 - CTR_j(v))$. We normalize this by the total number of impressions, so that $B(u, q) \in [0, 1]$.

The reason why we normalize the BPR to lie in $[0, 1]$ is because the quantity now has a natural *probabilistic interpretation*: BPR can be viewed as the approximate probability that a document u will be bypassed by the user in favor of a random document. In the next section, we extend this analogy to collection of documents. Note that in the special case when every document is shown only in one position, the bypass rate of a document is proportional to the CTRs of the document in the other positions, i.e., $B(u, q) = \sum_j CTR_j(v, q)(1 - CTR_j(v, q)) / \sum_j CTR_j(v, q)$. However, we do observe "churn" in the relative ordering of documents resulting in a more complicated relationship between BPR and CTR.

Note that the "penalty" that a bypassed document u pays increases as the quality of the clicked document decreases. This is consistent with the intuition we try to capture. It is debatable whether this is the right penalty, since a document that only gets beaten by perfect document(s), with $CTR_j(v) = 1$, will have a bypass rate of zero. Our justification is that such data set does not allow us to draw any inference about the true quality of the document. In practice, no document will have perfect click-through rate, so bypassed documents will always pay some penalty.

Now that we have described the way relevance feeds into our approach, we turn to the aspect of novelty which comes out of the similarity computations.

3.3 Similarity

At an abstract level, a *similarity function*, $sim(u, v)$, is one that assigns a value in $[0, 1]$ to pairs of documents, u and v, such that pairs of documents with higher values are more

similar to one another than those with lower values. Given a similarity function, we may refer to its induced *distance* between documents, defined as $d(u,v) = 1 - sim(u,v)$. An important property of any similarity function is whether its induced distance forms a metric

DEFINITION 3.3 (METRIC). *Given a set of elements X, a distance function $d : X \times X \mapsto \mathcal{R}$ forms a* metric *if*

1. $\forall u \in X, d(u,u) = 0$;
2. $\forall u, v \in X, d(u,v) = d(v,u)$; *and*
3. $\forall u, v, w \in X, d(u,w) + d(w,v) \geq d(u,v)$.

Furthermore, d forms a uniform metric *if it forms a metric and $d(u,v) = 0$ or $d(u,v) = 1$.*

In this paper, we explore the choice of different similarity functions on the theoretical and empirical performance of our algorithm. A useful observation is that if the induced distance of a similarity function forms a *uniform metric*, then the similarity function is effectively performing clustering of documents. In other words, each cluster contains all documents that have similarity 1 amongst themselves. An example of such similarity function is one that assigns $sim(u,v) = 1$ to documents that share a common query. In contrast, general similarity function could assign any values between 0 and 1 to documents. An example is the *Jaccard measure* defined over words of documents.

3.3.1 Similarity via random walks

Our similarity computation is based on random walks in click graphs, motivated by work in measuring distance between nodes in graphs [7, 13, 2, 4, 12, 5, 6]. Click graphs are bipartite graphs where documents and queries constitute the set of nodes, and a document is linked to a query if it has been clicked when the query is issued. Similarity under random walks is influenced by two factors - having short paths between nodes, as well as having many paths between them. The intuition is that similarity between two documents are high if it is "easy" to "reach" one document from another in the click graph.

Using matrix notation, where A denote the bipartite adjacency matrix of the click graph, the similarity between two documents can be computed as A^tA . To compute a similarity value in the range $[0,1]$, one can use the column-normalized matrices A_n and A_n^t respectively. Sometimes we may also want to bias the walk by introducing self loops. This is achieved by strengthening the diagonal entries. Mathematically, this corresponds to altering the matrix product to $(1-\alpha)A_n^tA_n + \alpha I$, where α is a parameter in $[0,1]$. To increase coverage, we may also want walks of length $l > 2$. This is achieved by taking products of the matrix, *viz*, by computing the matrix $D = B^l$ where $B = (1-\alpha)A_n^tA_n + \alpha I$. We compute the similarity between documents u and v as

$$sim(u,v) \quad := \quad \frac{D(u,v)}{\sqrt{D(u,u)D(v,v)}}.$$

3.3.2 Computing long random walks

Unfortunately, the naïve approach of taking matrix products for random walks becomes prohibitively expensive for longer walks. An originally sparse matrix may quickly becomes dense due to exponentiation, and we can no longer take advantage of its structure. We nod discuss ways to work around this problem.

We exploit some properties of random walks that will allow us to reduce the number of nodes visited by a random surfer without really affecting the properties of the walk. Given two nodes u and v in the graph, some important measures that are related to similarity computed through random walks are:

- *hitting time, h_{uv}* - the expected number of steps before node v is visited starting from node u. The first passage time, t_{uv} is the number of steps before node v is visited for the first time starting from node u.
- *commute time, c_{uv}* - the expected number of steps in a random walk starting at u, hitting v, before reaching u again, i.e., $h_{uv} + h_{vu}$.

Thus, two nodes are considered to be similar if the commute time between them is small and vice versa. Let N_t^u denote the number of nodes with hitting time t from a node u. In fact, it is easy to show that this number can be at most linear in t.

PROPOSITION 3.4. $N_t^u \leq 4t$.

PROOF. We know that the hitting time $h_{uv} = E[t_{uv}]$. Using Markov's inequality, we have $\Pr[t_{uv} > 2h_{uv}] \leq 1/2$. Thus, a random walk of length $2t$ from u hits every node v, such that $h_{uv} < t$, with probability at least $1/2$. On the other hand, we know that the number of distinct nodes seen in a random walk of length $2t$ from u is at most $2t$. Thus, we have $E[N_t^u] = N_t^u/2 \leq 2t$ completing the proof. □

We know that if we are interested in nodes that are similar above a certain threshold to a source node, then we need to run random walks up to a certain length only. Notice that, even in our setting, the number of nodes reachable by traversing a distance k may be exponential in k. However, according to Proposition 3.4, only a small number of these nodes can have a small hitting time and hence high similarity. Therefore, we trim the set of nodes at distance k by computing a cutoff as the similarity value of the $n(1-1/2^k)$-ranked node in set of nodes reachable by a random walk of length k. Notice that such a trimming factor $\tau = (1-1/2^k)$ is conservative initially and is more stringent for higher values of k. This is precisely what we need to change in the random walk to still be able to capture most of the nodes with a high expected hitting time.

4. PROBLEM FORMULATION

Given the definitions of bypass rates and document similarities, we are now ready to formulate an objective that is related to the problem of minimizing query abandonment. While in practice it is often difficult to tell if a user has abandoned a query by analyzing the click logs alone, as stated earlier, we can say a user abandons a query if she clicks on none of the results. In other words, if a user bypasses all of the returned results. This motivates the objective we describe next.

4.1 Optimization Objective

In the previous section, we have defined the BPR of individual documents. As noted, this is related to the probability of a user finding a document not relevant. We now

extend this concept to a collection of documents, to approximate the probability of query abandonment.

Note that when we consider the BPR of a set of documents, simply aggregating the BPR by addition or multiplication is inadequate. This is because documents that are similar to one another will tend to be bypassed together by the user. The aggregation will need to integrate both the BPR and the similarity of the documents.

To build some intuition of how BPR of sets of documents is related to the underlying BPR and similarity, let $P(S, q)$ denote the probability that at least one of the documents in S is clicked, hence relevant, for query q. Consider the following two extreme cases.

- If $sim(u, v) = 0$, then we expect the probability that neither u nor v is clicked, $1 - P(u + v, q)$, to be the product of the probabilities that u and v is not clicked, $(1 - P(u, q))(1 - P(v, q))$,. Intuitively, since u and v are about different topics, we expect the probability of whether they are clicked to be *independent*.
- If $sim(u, v) = 1$, then we expect $1 - P(u + v, q)$ to be the probability of the "better" document not getting clicked, i.e., $\min\{1 - P(u, q), 1 - P(v, q)\}$. Intuitively, u and v are now about the same topic, so if the user is not interested in the better document, she will not be interested in the worse document either.

This motivates the following definition of BPR of an *ordered set* of documents, $b([\ldots], q)$, recursively as:

$$b([a_1, a_2, \ldots, a_n], q) = b([a_1, \ldots, a_{n-1}], q)B(a_n, q)^{1-Sim(a_n,S)}$$

where $[\ldots]$ denote an ordered set, and $Sim(u, S) = \max_{v \in S} sim(u, v)$. The generalized similarity function $Sim(u, S)$ has the simple interpretation that the similarity between document u and a set of documents S is the similarity between u and its most similar counterpart in S. As a sanity check, note that this definition of BPR satisfies both of the extreme cases mentioned earlier.

Next, we define the BPR of a set of documents, $B(S, q)$

$$B(S, q) = \min_{[a_1,\ldots,a_n] \in \Pi(S)} b([a_1, \ldots, a_n], q)$$

where $\Pi(S)$ is the set of all possible permutation of the set S. Note that the BPR of a set of documents is *permutation-invariant*. This definition separates the problem of selecting a good set of documents with low BPR from the problem of optimally *ordering* the documents.

The BPR of a set of documents approximate the likelihood that the query will be abandoned by a user. Hence, our goal is to find a set S that minimizes $B(S, q)$. We label this problem MinQueryAbandonment.

> MinQueryAbandonment(k): Given k, the number of documents to return, find a set of documents S with $|S| = k$ that minimizes $B(S, q)$.

5. THEORETICAL ANALYSIS

We now investigate the algorithmic aspect of the objective described. We will show that minimizing the objective is NP-hard. We will also analyze a greedy approach to the problem, and shows that it is optimal for a certain special case, and we expect it to work well in practice.

5.1 Hardness

Theorem 5.1. MinQueryAbandonment(k) *is NP-hard.*

Proof. We show this via a reduction from the *Minimum k-Spanning Tree* which is known to be NP-hard [11], [17]. A problem instance of the Minimum k-spanning tree consists of a graph $G = (V, E)$, an integer $k \leq |V|$, and a weight function $w : E \to N$. The optimal solution is a k-spanning tree, i.e., a subtree T of G of at least k nodes that minimizes the total weight of T, $\sum_{e \in T} w(e)$.

We reduce this to the problem of finding an optimal set S for $Obj(S)$ by translating the vertex set V to the set of documents D. By scaling, one can now assume without loss of generality that $w(e) \in [0, 1)$ for all $e \in E$. For every $e = (u, v) \in E$, we set $sim(u, v) = w(e)$. For every $(u, v) \notin E$, we set $sim(u, v) = 1$. So $sim(u, v) \in [0, 1]$ for all $u, v \in D$. Set $B(u, q) = b$ for all $u \in D$. As in the minimum k-spanning tree, here also we require that the set of documents S picked be of size k.

Consider the objective of minimizing $B(S, q)$ for the special case of $B(u, q) = c$ for some $c < 1$.

$$\begin{aligned}
&\min_{S \subseteq D} B(S, q) \\
=&\min_{S \subseteq D} \min_{[a_1,\ldots,a_n] \in \Pi(S)} b([a_1, \ldots, a_n], q) \\
=&\min_{S \subseteq D} \min_{[a_1,\ldots,a_n] \in \Pi(S)} c \cdot c^{1-sim(a_2,a_1)} c^{1-Sim(a_3,\{a_1,a_2\})} \ldots \\
=&c^{\max_{S \subseteq D, [a_1,\ldots,a_n] \in \Pi(S)} (k - \sum_{i=2}^{k} Sim(a_i, \{a_1, \ldots a_{i-1}\}))}
\end{aligned}$$

The last step switches from a minimization to a maximization because $c < 1$, so the larger the exponent, the smaller its value. Since k is a constant, our actual optimization becomes

$$\min_{S \subseteq D, [a_1,\ldots,a_n] \in \Pi(S)} (\sum_{i=2}^{k} Sim(a_i, \{a_1, \ldots a_{i-1}\}))$$

Let the optimal *ordered set* S returned by our problem be $[a_1, a_2, \ldots, a_k]$ in that order.

Claim 5.2. *The optimal k-tree, T, for the minimum k-spanning tree problem is formed by the edges* (a_i, a_{i+1}) *for all* $i = 1$ *to* $i = k - 1$.

Proof. First, it is easy to check that $(a_i, a_{i+1}) \in E$ unless $|E| < k$. Let the weight of T be $w(T)$. By construction, we have $w(T) = Obj(S)$. Now, assume that there is a k-tree of lower weight, say T'. Let the vertices in this tree be $b_1, b_2, \ldots, b_k$.

We construct a solution S' to find Obj with higher value as follows: Assume without loss of generality that b_1 has degree 1 in T' (as there exists at least one vertex of degree 1 in every tree). Choose b_2 as a neighbor of b_1 in T'. Choose b_3 as a neighbor of one of b_1 or b_2, and proceed this way. Let S' be the ordered set $\{b_1, b_2, \ldots, b_k\}$. By construction, and the choice of $sim(u, v)$, we have,

$$\begin{aligned}
Obj(S') &\leq b^{k-w(T')} \\
&< b^{k-w(T)} \\
&= Obj(S)
\end{aligned}$$

This contradiction proves the claim. □

This polynomial-time reduction shows that if we can solve our optimization problem exactly, we can find the minimum k-spanning tree in polynomial time. Hence, optimizing $Obj(S)$ is NP-hard.

5.2 A Greedy Algorithm

Since the objective is NP-hard to optimize, we instead look for efficient approximate solution to MinQueryAbandonment. Motivated by the similarity of the objective with clustering- and set-cover-like problems, we propose a greedy heuristic for the problem.

Let $R(q)$ be the results produced by executing the query on a classical ranking algorithm. We consider how best to reorder these documents using our objective as guideline.

The algorithm selects one document at a time. In every step, it chooses the document that has the highest value of $B(u,q)^{1-Sim(u,S)}$ given that a set S of documents have already been chosen. It uses as input the document-document similarity matrix $\mathcal{S}$ and the query-document BPR matrix $\mathcal{B}$.

Algorithm 1 GreedySelect

Input k, q, $R(q)$, $\mathcal{B}$, $\mathcal{S}$
Output set of documents S
1: $S \leftarrow S \cup \min_{u \in R(q)} B(u,q)$
2: **while** |S| < k **do**
3: $d^* \leftarrow argmin_{d \in R(q)}\ B(S+d,q)$ [ties broken arbitrarily]
4: $S \leftarrow S \cup \{d^*\}$
5: $R(q) \leftarrow R(q) \setminus \{d^*\}$
6: **end while**
7: **return** S

Note that the similarity and BPR matrices are used in choosing the next candidate document in step 3. Unfortunately, it is not clear how to perform the selection step of the algorithm in step 3 optimally, as this involves optimization over all permutations. Instead, we employ the following simplified approach. Instead of considering all permutations, we focus only on any particular permutation that has the first document, as the one with minimum BPR and greedily build up the ordered set thereafter, in an attempt to minimize $B(S,q)$. In particular, according to our notation, this reduces to minimizing $b[d_1, d_2, \ldots, d_i, q]$ at every i-th step of the greedy algorithm, for the set $S = \{d_1, d_2, \ldots, d_{i-1}\}$ rather than minimizing $B(S,q)$.

Algorithm 2 OrderedGreedySelect

Input k, q, $R(q)$, $\mathcal{B}$, $\mathcal{S}$
Output let $[S]$ be an ordered set of documents
1: $S \leftarrow S \cup \min_{u \in R(q)} B(u,q)$
2: **while** |S| < k **do**
3: $d^* \leftarrow argmin_{d \in R(q)}\ b([S,d],q)$ [ties broken arbitrarily], where $[S,d]$ is the ordered set with $\{d\}$ at the end of $[S]$.
4: $[S] \leftarrow [S, d^*]$
5: $R(q) \leftarrow R(q) \setminus \{d^*\}$
6: **end while**
7: **return** $[S]$

Ideally, we would like to establish a formal theoretical guarantee of the approximation ratio for OrderedGreedySelect. However, we have not been able to do so, and will leave this as an open question. Instead, we show that for some special forms of the similarity function, the algorithm is in fact optimal.

Theorem 5.3. *When $1 - sim(u,v)$ form a uniform metric,* GreedySelect *and* OrderedGreedySelect *are the same, and both are* optimal.

Proof. First, one can verify that if $(1-sim(u,v))$ forms a uniform metric, then the set of documents can be clustered. In other words, if $sim(u,v) = 1$, then u and v are in the same cluster; otherwise they belong to different classes. The metric requirement ensures that the clustering is *consistent*, i.e., there does not exist three documents u, v, w such that $d(u,v) = d(v,w) = 1$ but $d(u,w) = 0$.

Next, we make two observations:

Observation 5.4. *The optimal solution never picks two documents from the same cluster, as long as there are k different clusters.*

Proof. Assume, for the sake of contradiction, that the optimal solution contains u and v from the same cluster. So $sim(u,v) = 1$. Without loss of generality, let v come after u in the optimal ordering of this solution. Since $Sim(v,X) = 1$ for any set X containing u, v's contribution to the optimal objective will be a product of $B(v,q)^{1-sim(u,v)} = 1$. So, v does not contribute in reducing the objective at all. Hence, any other document w with $B(w,q) < 1$ from a different cluster could be used to obtain a better optimal solution. □

Observation 5.5. *In any cluster, if* OrderedGreedySelect *picks a document, it picks one with the lowest BPR.*

Proof. Consider two documents u and v from the same cluster, and assume that $B(u,q) > B(v,q)$. Suppose u is in an optimal solution. Replacing u by v in the same optimal solution yields a lower objective as $sim(u,w) = sim(v,w)$ for all w, since sim forms a uniform metric. □

Based on these two observations, it is easy to see that the optimal solution is to pick the k documents with the least bypass rates such that all these k documents are from different clusters. Further, one way to do this is to first pick the document with the smallest bypass rate, and then ignore all documents from that cluster; then repeat this procedure until k documents are picked.

One can see that for this special case of uniform metrics, both GreedySelect as well as OrderedGreedySelect run exactly this. □

In theory, once we relax the uniform metric requirements, we can no longer make any formal claims about the optimality guarantee of the OrderedGreedySelect. However, intuition suggests that the closer the universe of documents to separating in the nice clusters, the better the algorithm will perform. In other words, for environments where similarity between documents are either high or low, the algorithm is likely to work well.

For the experiments in this paper, we use the OrderedGreedySelect and even though the similarity function does not form a uniform metric, our experiments show that it appears to perform well when compared to other diversity-based orderings like MMR.

6. EXPERIMENTS

To evaluate the empirical performance of our algorithm, we conduct three sets of experiments on data collected by a commercial search engine. In the first experiment, we compare the output of our algorithm to that produced by maximal marginal relevance (MMR) on the metrics of MAP and MRR. In the second experiment, we study the sensitivity of the performance of our algorithm to how similarity between documents is computed. In the last experiment, we evaluate a method for enlarging the candidate set of results for each query, which helps to address problems caused by the sparsity of the click logs. We start by explaining the data sets and the metrics used in this section.

6.1 Data Sets

The click log used in the experiments contains queries with frequencies between $1K$ and $100K$, collected over one month. The relevant entries in the log are the query, the order of the returned documents, and the click(s) of the user. To ensure that the data is meaningful, we filter out entries with no clicks or few impressions in a top-10 position (we set the threshold to be 100). This resulted in approximately $2.0M$ entries spanning about $128K$ unique queries and $1.3M$ distinct documents.

Experiments are evaluated over a test set consisting of about $28K$ queries sampled from the logs, with about $66M$ results (URLs) associated with these queries. Of these URLs, about 1% have been rated by human judges regarding their relevance to the query. These judgements are on a six-point scale: "`detrimental`" (0), "`bad`", "`fair`", "`good`", "`excellent`", and "`definitive`" (5). The documents selected for judgments were *not* random; they were selected because at least one of the major commercial search engines has placed them in a top-10 position. Thus, the judged documents are biased towards higher relevance. As a result, in our experiments, we treat results without judgement as `detrimental`.

Ideally, in addition to MMR, we would also like to compare the performance of our algorithm to the work of Chen and Karger [3]. However, we are unable to do so, as the authors reported their result using TREC data, a data set that does not include the number of impressions or document order, both of which are necessary for computing BPR.

6.2 Evaluation Metrics—MAP and MRR

We evaluate the performance of our algorithm and that of MMR on two widely-studied metrics, *viz.*, the *mean average precision* (MAP), and the *mean reciprocal rank* (MRR) [19]. We observe that even though these metrics explicitly don't measure diversity, they do assigns value to any result set containing at least one relevant document.

The "*average precision at k*" of an ordered set of results is defined as

$$\frac{\sum_{j=1}^{k} P(j) * Relevance(j)}{\sum_{j=1}^{n} Relevance(j)},$$

where $Relevance(j)$ is the relevance of the document in position j, and $P(j) = \sum_{i=1}^{j} Relevance(i)/j$. Following the usual convention, in our experiments, we choose a binary-valued relevance function, where $Relevance(j)$ is set to be 1 if the document in position j has a human rating of `Fair` or better and 0 otherwise. The *mean average precision at k* (MAP@k) of a query set is computed by taking the mean of the average precisions at k of all queries in the query set.

The *reciprocal rank* (RR) is the inverse of the position of the first relevant document in the ordering. In the presence of a rank-threshold k, this value is 0 if there is no relevant document in the first k positions. The *mean reciprocal rank* (MRR) of a query set is the average reciprocal rank of all queries in the query set.

In our experiments, we report the MAP and MRR scores at the threshold values of 1, 3, and 10.

6.3 Comparison with MMR

In this first set of experiments, we compare the performance of OrderedGreedySelect with that of maximal marginal relevance (MMR). The details of the MMR algorithm can be found in Section 2. We remind the reader that MMR depends on a certain tuning parameter λ that determines the tradeoff between diversity and quality of the results (see Equation (1)). It also depends on how similarity between documents is computed. Our results here are reported for the random-walk-based similarity measure with parameters $\alpha = 0$ and $l = 2$.

The effect of the tuning parameter λ is reported in Table 1. Both MAP@10 and MRR@10 are at their highest when λ is set to 1, where MMR will focus solely on the quality of the results at the expense of diversity. This is to be expected, since showing a set of diverse results adversely affects traditional IR metrics that do not reward diversity, including both MAP and MRR.

λ	Uniform metric		Continuous similarity	
	MAP@10	MRR@10	MAP@10	MRR@10
0	0.5706	0.6086	0.5703	0.6074
0.5	0.5863	0.6266	0.5889	0.6305
0.7	0.5892	0.6274	0.5794	0.6207
1	0.5918	0.6285	0.5979	0.6339

Table 1: Performance of MMR with continuous and discrete similarity functions with $\alpha = 0$ and $l = 2$

Next, we compared the performance of OrderedGreedySelect with that of MMR. The experiment is conducted using a continuous similarity function with $\alpha = 0$ and $l = 2$, and $\lambda = 0.5$ for MMR. We have chosen this value for lambda as it balances the quality of the documents with diversity, similar to how OrderedGreedySelect operates. The results are reported in Figure 1. The results demonstrated that our algorithm consistently outperforms MMR on both metrics at all three thresholds.

6.4 Sensitivity to Similarity Computation

In the previous set of experiments, we have chosen a particular setting for how similarity between documents is computed. We now vary the parameters and measure how sensitive our results are to these choices. As explained in Subsection 3.3, the main parameters are α, the strength of self-similarity, and l, the length of the walk. We fix the trimming parameter at $\tau = (1 - 1/2^{l+2})$ throughout.

First, we measure the sensitivity of the parameters. The results are reported in Figure 2. Intuitively, a longer walk helps in finding candidates that are more dissimilar to those already chosen. On the effect of α on the BPR of sets of results, ignoring the self loops in the random walk seems to improve the BPR while only including the diagonal elements in the similarity matrix produces the worst value of $B(S, q)$.

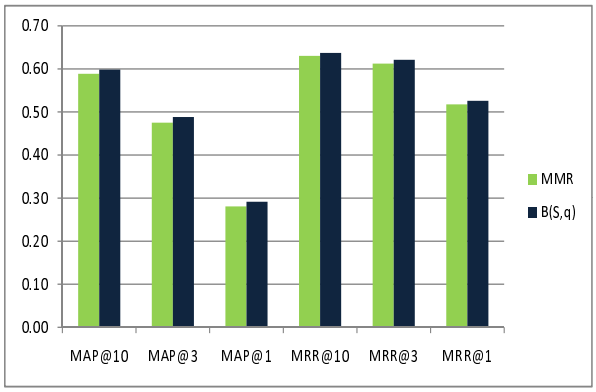

Figure 1: Relative performance of OrderedGreedySelect **and MMR for $l = 2$, $\alpha = 0$, and $\lambda = 0.5$**

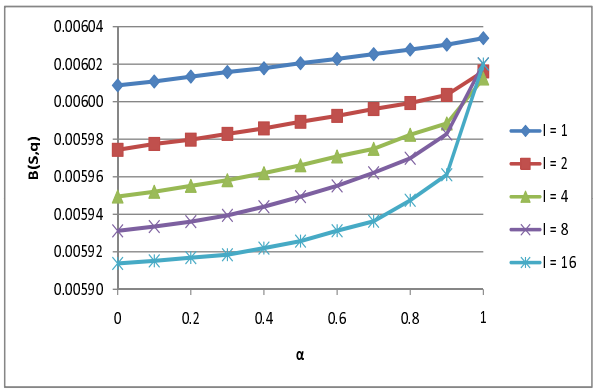

Figure 2: Effect of α and length of random walk on OrderedGreedySelect

Next, we measure the sensitivity of the MAP and MRR of the result sets produced by OrderedGreedySelect to the parameters. The results are summarized in Table 2. The sensitivity is relatively low, suggesting that these are not critical parameters to the algorithm, and hence little tuning is required. Close examination suggests that for longer walks, the uniform similarity function outperforms its continuous counterpart slightly. We are currently investigating whether this is significant. One possible explanation for this could be that the uniform metric does a better job in handling the instability of search results in the top positions of the result page.

Walk length (l)	Uniform metric		Continuous similarity	
	MRR@10	MAP@10	MRR@10	MAP@10
2	0.6372	0.5986	0.6372	0.5986
4	0.6368	0.5981	0.6372	0.5986
6	0.6380	0.5991	0.6368	0.5984

Table 2: performance of OrderedGreedySelect **with continuous and discrete similarity functions ($\alpha = 0$)**

6.5 Boosting the Candidate Set

One of the main drawbacks of using features that are based on the click graph is that the graph is often very sparse. Most of the clicks for a query are concentrated toward the highly ranked documents. This results in a very small number of unique clicked documents for each query (even taking into account the instability of search results). Therefore, we can only compute a ordering over these small set of documents. We propose a method to extend the set of relevant documents for a given query using the clicked documents for similar queries.

Assume we have access to a query similarity matrix S. We may infer the relevance of any document d for a query q as

$$R_{dq} = \sum_{q} S_{qq'} G_{dq'},$$

where $S_{qq'}$ denote the similarity between queries q and q' and G_{dq} is the relevance of a document d for query q. We also refer to the relevance score of a document as its *goodness* value of the document for the given query. Thus, the process of extending the relevant set of documents for a given query involves accumulating goodness values from similar queries by weighting them with their similarity values. Writing this in matrix form gives $R = SG$. The question then is how to obtain the similarity matrix S and the goodness matrix G.

The query similarity matrix is constructed as before. We construct the goodness graph from the BPR of the documents. Essentially we consider the click-graph with the edge weight on a click (q, u) as $1 - B(u, q)$. Here, the parameter α is set to 0, as the diagonal entries are of no interest.

Note that GG' is obtained by first looking at all paths of length 2 between two queries and then adding up the product of the goodness values on the edges over all the 2-length paths between the queries. This can be extended to paths of longer length, say l. This corresponds to the similarity matrix $S = (GG')^l$. The new goodness values based on this similarity matrix is given by $L = (GG')^l G$.

By running two iterations, i.e., $l = 2$ (including paths of length 4), we increase the number of the number of non-zero entries from around 1.5M to over 275M! Therefore, we use the trimming strategy described in Section 3.3.2 to reduce the number of non-zero entries at the end of each iteration. Instead of increasing the threshold gradually, we adopt a more conservative approach of using a constant τ in this experiment. For example, after performing a random walk of four steps, the average number of documents per query increases from 13.1 to 4026 with $\tau = 0.90$. Even at a higher value of $\tau = 0.95$, the average number of rated documents per query only decrease to 3029 - a reasonably large number! Figure 3 shows the relevance metrics for the extended ratings file for different values of τ. The values of MRR at all positions are better than those for the original data set. This implies the rank of the relevant results is higher in the extended data set than in the original data set. In the case of MAP, we observe the scores to be slightly less than those for the original data set. A possible explanation for this is the 'holes' or the unjudged documents in the extended ratings file. Even though the number of ⟨query, document⟩ pairs for a query increased by including documents belonging to similar queries, the total number of judged ⟨query, document⟩ pairs only increased from 65346 to 108959. This would suggest that there are a lot of unjudged documents in the data set, some of which could be more relevant than the documents which were judged "Fair" or lower in ratings. Filling these 'holes' in the ratings would improve the MAP scores shown in Figure 3.

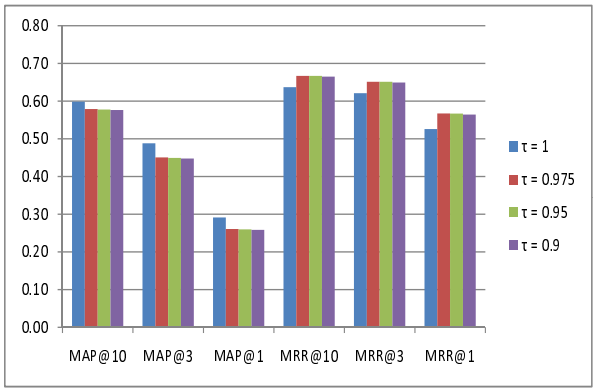

Figure 3: Effect of τ on the relevance of the extended data set

7. CONCLUDING REMARKS

We introduce bypass rates (BPR), a statistic of query-document pairs that can be computed from click logs. We believe that BPR are simple statistics that arise naturally as a result of how users interact with search engines. This statistic captures certain negative inferences in the log data, and is complementary to the well-studied statistic of click-through rates (CTR). We formulate an objective that minimizes the collective BPR of a set of documents, which we believe will help to reduce the severity of query abandonment in search engines. While finding the optimal set to return is NP-hard, we give a simple greedy algorithm with guarantees on its worst-case performance. We evaluate our algorithm experimentally using real data from a commercial search engine, and show that it outperforms the well-studied technique of MMR on two important metrics, MAP and MRR. We also investigate the sensitivity of its performance to certain algorithmic parameters.

8. ACKNOWLEDGEMENTS

We would like to thank D. Sivakumar for useful discussions on random walks and Marc Najork for providing us the tools to compute the MAP and MRR metrics. We would also like to thank the anonymous reviewers for their valuable suggestions.

9. REFERENCES

[1] Jaime G. Carbonell and Jade Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335–336, 1998.

[2] A. K. Chandra, P. Raghavan, W.L. Ruzzo, R. Smolensky, and P. Tiwari. The electrical resistance of a graph captures its commute and cover times (detailed abstract). In *STOC*, pages 574–586, 1989.

[3] Harr Chen and David R. Karger. Less is more: probabilistic models for retrieving fewer relevant documents. In *SIGIR*, pages 429–436, 2006.

[4] F. R. Chung, editor. *Spectral Graph Theory*. American Mathematical Society, 1997.

[5] Nick Craswell and Martin Szummer. Random walks on the click graph. In *SIGIR*, pages 239–246, 2007.

[6] F. Fauss, A. Pirotte, J-M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007.

[7] F. Gobel and A. Jagers. Random walks on graphs. *Stochastic Processes and their App.*, 2:311–336, 1974.

[8] Thorsten Joachims. Optimizing search engines using clickthrough data. In *KDD*, pages 133–142, 2002.

[9] Thorsten Joachims, Laura A. Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR*, pages 154–161, 2005.

[10] A. V. Leouski and W. B. Croft. An evaluation of techniques for clustering search results. Technical Report IR-76, University of Amherst, 1996.

[11] D. Lozovanu and A. Zelikovsky. Minimal and bounded tree problems. In *Tezele Congresului XVIII al Academiei Romano-Americane*, pages 25–26, 1993.

[12] M. Newman. A measure of betweenness centrality based on random walks. *Social Net.*, 27(1):39–54, 2005.

[13] P.G.Doyle and J.L.Snell, editors. *Random Walks and Electric Networks*. Math. Assoc. of America, 1984.

[14] F. Radlinksy, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *ICML*, 2008.

[15] Filip Radlinski and Thorsten Joachims. Minimally invasive randomization fro collecting unbiased preferences from clickthrough logs. In *AAAI*, 2006.

[16] Filip Radlinski and Thorsten Joachims. Active exploration for learning rankings from clickthrough data. In *KDD*, pages 570–579, 2007.

[17] R. Ravi, Ravi Sundaram, Madhav V. Marathe, Daniel J. Rosenkrantz, and S. S. Ravi. Spanning trees short or small. In *SODA*, pages 546–555, 1994.

[18] Stephen Robertson. The probability ranking principle in ir. *Journal of Documentation*, 33(4):294–304, 1977.

[19] H. Zaragoza, N. Craswell, M. Taylor, S. Saria, , and S. Robertson. Microsoft cambridge at trec-12: Web and hard track. In *Proc. of the 13th Text Retrieval Conference*, pages 418–425, 2003.

[20] C. Zhai, W. Cohen, and J.D. Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *SIGIR*, pages 10–17, 2003.

[21] ChengXiang Zhai and John D. Lafferty. A risk minimzation framework for information retrieval. *Info. Processing and Management*, 42(1):31–55, 2006.