Using Web-Graph Distance for Relevance Feedback in Web Search

Sergei Vassilvitskii
Stanford University
Stanford, CA
sergei@cs.stanford.edu

Eric Brill
Microsoft Research
Redmond, WA
brill@microsoft.com

ABSTRACT

We study the effect of user supplied relevance feedback in improving web search results. Rather than using query refinement or document similarity measures to rerank results, we show that the web-graph distance between two documents is a robust measure of their relative relevancy. We demonstrate how the use of this metric can improve the rankings of result URLs, even when the user only rates one document in the dataset. Our research suggests that such interactive systems can significantly improve search results.

Categories and Subject Descriptors: H.3.3 [information Search and Retrieval]: Relevance feedback

General Terms: Algorithms.

Keywords: Web Search, Relevance Feedback, Link Analysis

1. INTRODUCTION

Web search today is still an unsolved problem. Although relevance of results has steadily improved over the past decade, anecdotal evidence indicates that many queries still go unanswered. A major obstacle to further improvement of search results is understanding user intent from the query. The problem is exacerbated by the fact that the average query length is between two and three words, and average query sessions consist of only a handful of queries.

The brevity of most search queries is a recognized problem, and several methods have been proposed to deal with this scenario. The approaches can be divided into passive and active systems. Passive systems observe user behavior and try to infer more information about the users' intent from their previous actions. In this area numerous personalized search algorithms have been suggested, and implemented in practice (see, for example, [8, 18]). These algorithms reweigh the results for each query asked by the users based on their stated preferences or past search histories.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.
Copyright 2006 ACM 1-59593-369-7/06/0008 ...$5.00.

Active systems, on the other hand, engage in some interaction with the user that better elicits user intent. A decade ago, Koenemann and Belkin showed that interaction makes users more effective searchers [10]. This interaction can take many different forms. For example, the Vivisimo search engine [20] clusters the results and lets the user navigate the cluster hierarchy. Yahoo's mindset search [21] lets users rerank the results, giving more influence to "Shopping" or "Research" URLs. Other work has explored the space of query suggestion. Here a list of possible query expansions is presented to help the user further narrow down the results (see for example [2] and the references therein).

Relevance feedback is an interactive approach that has been shown to work particularly well in classical information retrieval [3, 16] and more recently in Internet search settings [13, 19]. In this setting, after receiving results for a query the user is asked to rate the relevancy of some of them. This information is passed to the search engine, and the engine reranks the results to better suit the user's intent. This type of interaction is particularly well suited to the web search domain, as the relevance of a result can often be judged from a quick look at the text snippet returned. The performance of different term-reweighing schemes has been studied previously [19]; in this work we focus on using new information available only in the web search domain to further improve performance.

1.1 Our Results

We show that classical text-based relevance feedback approaches (in particular Rocchio's term reweighing method) do not perform well in web search scenarios. We propose a novel algorithm that is based on the underlying link structure of the web rather than the page text and show that it outperforms standard relevance feedback methods. This notion of relevance propagation is at the core of the original PageRank and HITS algorithms [9, 14] and remains a hot topic in the community [15, 17]. A similar approach was also used by Dean and Henzinger [4] to find related webpages. However, it has not been previously applied in an *interactive* search setting to directly improve the quality of the search results.

To validate the hypothesis of relevance propagation we show that for each query, relevant pages tend to point to other relevant pages, while irrelevant pages are pointed to by other irrelevant pages. It is intuitive that this relationship holds for direct links; we show that this "relevance signal" is preserved even as the distance between pages increases.

We tested our system on a set of human-labeled data and demonstrate that web-graph based relevance feedback in-

creases the average relevancy of the query results, while classic techniques are of limited help and can sometimes be detrimental to the overall rankings. Our results are made more remarkable by the fact that the major search engine used as a baseline for the experiments in this work (MSN Search) uses link analysis as one of the factors in its result ranking.

We begin by describing in more detail the interactive search system and our main hypothesis behind it. In Section 3 we describe the labeled data necessary to objectively evaluate the performance of different systems and present our experimental setup. In Section 4 we validate our main hypothesis and in Section 5 we present the exact relevance feedback algorithm. In Section 6 we present the results of our experiments and discuss the tradeoffs between the various parameters. Finally, we conclude with some future work directions.

2. HYPOTHESIS

The intended scenario for our system is as follows. Upon receiving the list of search results, the user informs the system of the relevancy of some of the results. The search engine then uses this information to produce a new set of results, or to rerank the initial results to have more relevant pages appear near the top of the returned list.

The hypothesis behind the web-graph reranker is twofold:

- Relevant pages *point to* other relevant pages.
- Irrelevant pages are *pointed to* by other irrelevant pages.

We begin with some examples demonstrating the intuition behind this hypothesis:

Example 1: Word Sense Disambiguation Consider the query "Jaguar." Major search engines return URLs about Jaguar the British car, Jaguar the Apple OS and the wild animal. To limit the results to one of the word meanings, we must add special keywords like "car" or "mac." However, this information is already encoded in the web graph. The web graph distance between two pages talking about the new Jaguar car model is typically much smaller than the distance between an automotive page and a page describing the life-cycle of jaguars. For example, many of the car pages link to the official website at jaguar.com.

Example 2: Travel Websites It is well known that the travel search results are often spammed by travel agency affiliates, all using the same back-end search engine to advertise their fares. Again, the web graph distance between these sites is usually small as they all link to the same searching back-end, whereas almost no highly relevant page would point to any of these spam sites.

Example 3: Vague User Intent Consider the query "Cancer." The list of relevant results may include general information about the disease, scientific papers on latest research breakthroughs, homeopathic remedies, or local support groups. The intent of the user is not clear from the search term, yet all of these results form well linked clusters in the web-graph. (It is highly unlikely that scientific papers would link to support groups, for example.)

We emphasize that the direction of the links plays an important role in the hypothesis. While relevant pages tend to point to other relevant pages, the converse is not true. The set of pages pointing *to* relevant pages need not be relevant. Consider an authoritative page for a particular topic. The author of the page decides upon and selects the list of outlinks, but has no control over the incoming links. There may be many pages pointing to an authoritative source, but they themselves may not be relevant to the query. When dealing with irrelevant pages, the opposite is true. Consider a spam page, S, full of advertisements for other pages. The inlinks to S often come from link farms, or pages specifically set up to increase S's score and raise its standing in the search engine results. Again, while S itself may be pointing to relevant pages, the pages pointing to it are most certainly not relevant.

3. DATA

To objectively measure the performance of different relevance feedback techniques, we used the following dataset. A set of 9500 queries were selected at random from those posed to MSN Search [12]. For each of the queries, between 4 and 30 of the top URL results were presented to a set of human judges trained for result evaluation. The number of results varied, depending on the specificity of the query: for many of the queries 30 results were selected; while some of the more obscure queries produced only a handful of results. The judges rated the relevance of each page on a scale of 1(poor) to 5 (perfect). The test data is a set of 150,000 (query, url, score, rating) four-tuples. The score of a (query, url) pair is the relevancy score returned by the search engine. Note that both the ratings and the scores are query dependent and a particular URL can be rated as perfect for one query and poor for another.

3.1 Modeling User Behavior

We use the data to evaluate the performance of the relevance feedback system and to simulate user interaction. For each query, we split the rated URLs into a training set and a test set. The human supplied relevance scores are given to the algorithm for the URLs present in the training set. The algorithm is then evaluated on its performance on URLs in the test set. We varied the split into training and test sets to simulate different user scenarios. For example, we changed the number of documents in the training set from one to five, corresponding to a user providing feedback for one to five documents for a query, and also experimented with different selection rules for URLs in the training set. See Section 6.3 for further discussion.

4. WEB-GRAPH RERANKER

4.1 Hypothesis Validation

We begin by validating our original hypotheses. Recall that the data is a set of (query, url, score, relevance) four-tuples, where the URLs are those returned as top results for the query. Let P be the distribution of relevance scores over all of the results for a fixed query. If we were to examine the relevance of a random result page, it would be drawn from this distribution. Formally, $P(1)$: the probability of a random result page having a poor relevance score of 1 is $=$ *# of URLs with relevance 1 / total number of URLs examined for the query.* For example, if there were ten URLs examined for a particular query with relevances $\{5, 3, 4, 3, 1, 3, 2, 2, 2, 1\}$ then $P(1) = 0.2, P(2) = 0.3$ etc. P will be the baseline distribution on relevance in our experiments.

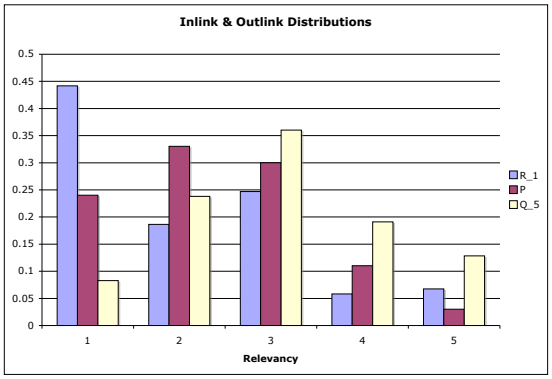


Figure 1: The distribution of pages that can reach an irrelevant page (R_1) is skewed towards being irrelevant, while the distribution of pages reached from a relevant page (Q_5) is skewed towards being relevant.

We define a reachability constraint over the set of URLs, signifying those that are reached from, or can reach a page of particular relevance. For two URLs u and v we say that u can reach v (or v can be reached by u) and write $u \rightsquigarrow v$ if there exists a directed path of length at most 4 in the web graph from u to v[1]. To examine the effect of distance from relevant pages on a URLs relevancy, we define two sets of distributions: Q and R. Let Q_i be the distribution P, restricted to those pages that can be reached from a page with relevance i. In the preceeding example, if only the first 5 results can be reached from a page with relevance 3 then $Q_3(1) = 0.2, Q_3(2) = 0, Q_3(3) = 0.4$ etc. Similarly, we let R_i to be the restriction of P to only those pages that can reach a page of relevance i.

If our hypotheses are correct we expect to see a skew in the distribution of Q_i as compared to P. In particular, if relevant pages tend to point to other relevant pages then the average value of Q_5, the average relevance of the targets of highly relevant pages, should be higher than baseline. Likewise if only irrelevant pages point to known irrelevant pages then a skew towards lower values should be observed in R_1.

We plot P, Q_5 and R_1 in Figure 1, and observe the desired behavior. The probability that an irrelevant page is pointed to by a relevant page ($Q_5(1)$) falls to less than 8%, whereas the baseline probability is almost 25%. Similarly, the probability that a page pointing to an irrelevant page is itself irrelevant ($R_1(1)$) jumps to almost 45% from the 25% baseline. Although the signal provided by the web-graph is somewhat noisy, there is a clear skew in the two distributions from the baseline, P, validating our hypotheses.

We performed similar experiments while ignoring the directionality of the graph edges, so that the path from u to v could follow edges both forwards and backwards. In the undirected model, the relevance signal disappeared entirely, and the distributions P, Q_i, and R_i were almost identical. Thus, as expected, the directionality of the links plays a key role in the hypothesis.

[1]We describe our reasoning for the choice of 4 as the reachability parameter in the next section.

4.2 Constructing the Web-graph

The main trade-off in the design of the algorithm is the choice of the hard parameter *distance* ≤ 4 in the definition of connectivity between two URLs u and v. To describe the reasons behind its selection we present the framework used to compute such distances and the effect the parameter has on the results.

The fundamental question that the reranker needs answered is "What is the shortest path distance between pages u and v?" Although shortest path algorithms are a classic topic in computer science, shortest path computation on very large graphs remains a challenging problem. Familiar methods, like Djikstra's shortest path algorithm exhibit random access over the whole graph, and thus perform poorly when the data is streamed from disk. External memory algorithms (see for example [1, 11]) provide theoretical improvements, but are still too slow to be useful in an online setting. Recent work by Goldberg et. al [5, 6] drastically improves the computational speeds for long paths in low maximum degree graphs, which is the opposite setting from the problem we are considering.

To compute the shortest path distance between two URLs we crawled the Web graph in the neighborhood of the URLs in our dataset. For each URL present in the dataset, the crawler saved the link structure following links both forward and backward for two hops. (This allowed us to perform bidirectional breadth first search to answer the connectivity question.) Even this crawl was very time consuming, especially when the crawler came across highly linked pages with thousands of in- and out-links (e.g. msn.com).

To make the calculations feasible, we seek to limit the degree of the graph and avoid the exponential growth in the size of the neighborhood set. To that end, we reduced the number of links followed by the crawler from each page. From every visited page the crawler was limited to following at most r incoming and outgoing links chosen at random. To investigate the effect that this pruning of the web-graph had on our results, we performed two crawls. The small crawl restricted r to 35, while the larger crawl had r set at 70. Since the average degree of the web is smaller than 35, in the majority of the cases no sampling was performed. The pruning of the webgraph had a minor effect on its overall performance of the system. We discuss the effect of the link sampling on the results in full detail in Section 6.4

The choice of 4 as a threshold parameter relied on balancing two opposing forces. When the parameter was set lower the relevance signal was much stronger, but the total number of interlinked pages was significantly smaller — less than 5% of the result sets had two pages within 3 hops of each other, as compared to more than 50% for 4 hops. Thus restricting the reachability to be no more than 3 hops had a detrimental effect on the recall of the system. On the other hand, setting the parameter higher became computationally infeasible to compute, and, more importantly, almost completely diluted the relevance signal. We therefore use 4 as the threshold for all of our experiments.

5. RERANKING ALGORITHM

We introduce the reranker algorithm in full detail. Let *Ranked* be the set of pages whose relevance is revealed by

the user. Recall that P is the baseline probability over page relevances and Q_i (respectively R_i) is the probability distribution for the page with an *incoming* (respectively *outgoing*) path to a page of relevance i.

The algorithm maintains a relevance distribution for each page in the result set. It is first initialized to the baseline distribution and is then updated every time a 4-hop path exists between the page and one of the user rated URLs[2]. Finally the most probable relevance score is selected for each URL. The final score is a weighted sum of the original search engine score and the relevance score obtained by the reranking algorithm. The value of γ controls the relative weight of the two scores. The full algorithm is presented below.

Algorithm 1 WEB-GRAPH-RERANKER

for all Unrated URLs u in returned set **do**
  Let P_u be the baseline distribution P
  for all Rated URLs v **do**
    if v has relevance ≥ 3 and $v \rightsquigarrow u$ **then**
      $P_u = P_u + Q_{rating(v)}$
    else if v has relevance ≤ 2 and $u \rightsquigarrow v$ **then**
      $P_u = P_u + R_{rating(v)}$
    end if
  end for
  NewScore(u) = score(u) + $\gamma \cdot \arg\max(P_u)$
end for
Return results sorted by NewScore

The algorithm is based on the original hypotheses. Notice that if a URL p in the result set can be reached from a highly relevant page, then during the update step, the distribution P_u will become more skewed towards the highly relevant pages (by virtue of addition of Q_3, Q_4 or Q_5). Likewise, if p can reach a page of low relevance, then P_u is skewed towards the lower values.

The value of γ was determined by tuning the algorithm on a held out set of examples, and was set to 0.1 for all of the experiments.

6. RESULTS

6.1 Performance Evaluation

To compare the performance of the system, we use the Normalized Discounted Cumulative Gain measure (NDCG) [7]. Formally, NDCG is defined as

$$Score = N \sum_i (2^{r(i)} - 1)/\log(1+i)$$

where the sum is over all of the URLs for the query in the dataset, $r(i)$ is the relevance of document in rank position i and N is a normalizing constant chosen so that the score is always between 0 and 100. (It is easy to compute the normalizing constant since a sort by relevance will always produce the highest NDCG.) There are several properties of NDCG (see [7] for a longer discussion) that make it a desirable performance metric. First, NDCG is very sensitive to the position of the highest rated page. This characteristic is crucial in web search, where the majority of the users rarely look past the first page of results before reformulating their query. This is further modeled by the discounting factor, $\log(1+i)$, which increases with the position of the result. Finally, the fact that the NDCG of the result set is always between 0 and 100, *regardless* of the number of documents returned for the query, makes it easy to compare accuracy on queries with different numbers of returned results.

We remark that in all of our experiments the NDCG is computed only on the set of URLs that were not explicitly rated by the user. This allows us to measure the effect of the user's rating of some URLs on the overall performance of the search engine.

The main objective of the method is to improve the relevance of the results over the initial search ranking, since extra information is provided by the user to aid the search engine. We report average NDCG changes when the reranker changed the ordering of the results. The NDCG of the search engine ranges throughout the dataset from perfect (score of 100) on some queries to fair (score less than 50) on a small percentage of queries. The average baseline NDCG over all queries in our sample is 89. To further investigate the effects of the reranker we divide its performance into three categories: average change over all of the queries, average change for queries where some improvement is possible (those where NDCG < 100) and average change for queries where major improvement was possible (NDCG < 85).

Since a method that works on only a small percentage of the queries would be of limited use, we also measure the recall of the system. In this context, the recall of the reranker is the percentage of queries on which the reranker *attempts* to change the ordering of the results (equivalently, where the value of P_u in the algorithm is not identically P for all pages in the result set).

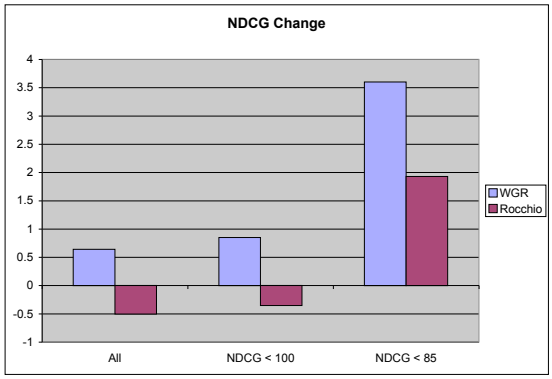
We observe that in many cases even when $P_u \neq P$ for some pages in the result set the overall ranking remains the same. Consider the result of reranking when the top-ranked result is pointed to by the only page rated by the user. If the rating is positive, then P_u is skewed towards the high relevance marks, thereby attempting to promote the page higher in the ordering. But since this is already the top ranked page, the order in which results are presented will be identical. We measure the percentage of queries where the ordering of the results changes based on user feedback, and call this measure the *observed recall* of the system.

Finally, there are situations when we can predict that the web-graph based reranker will not change the ordering of the results, no matter what user feedback is provided. This is the case when all of the result pages are spread out in the web-graph, and therefore none are within 4 hops of another. We can take these situations into account when we measure the *predictive recall* of the system: the percentage of pages where user feedback could result in a change in the ordering and a change in the ordering was observed. Note that observed recall will always be lower than the predictive recall since the latter is overly optimistic (e.g. counting when the top ranked result is further promoted by the user).

6.2 Main Findings

We present the complete set of results for the Web Graph Reranker method when the user rates five results selected by the algorithm. (See the following section for a discussion on parameter selection.) We compare our results against two benchmarks. The first is the ranking of the underlying

[2]We note that one option for the update rule is to perform a convolution of the two distributions. Empirically, however, direct addition of distributions (used in the algorithm) worked very well.



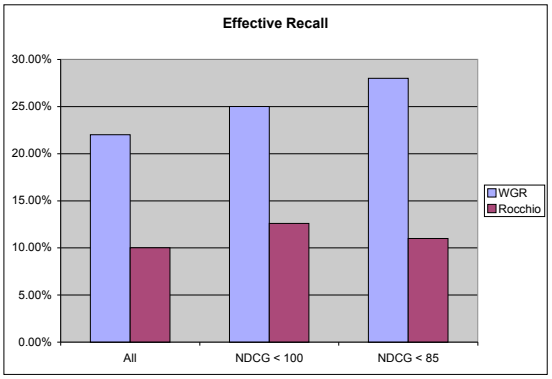


Figure 2: Comparing the performance of the web-graph reranker with Rocchio's relevance feedback measure. (a) The average change in NDCG from baseline (in queries where change occurred). The user rated 5 URLs selected at random from the result list. (b) Observed recall (i.e. percent of time the ordering changed due to user feedback) for the two methods.

search engine (here the URLs are sorted by the scores provided in the dataset). The algorithm should improve upon this benchmark since it can utilize the original search engine scores in its reranking function.

Further, we compare our results to those of classical relevance feedback algorithms, in particular the Rocchio metric [16][3]. It is an established fact that this metric often helps in many information retrieval situations. The input to the Rocchio metric is the full text of the highly rated pages in the training set. For the pages in the test set a cosine similarity score to the training set is computed, and is combined with the original search engine score to achieve the final score for the page.

We plot the average NDCG change of the two methods against the baseline in Figure 2. The reranker consistently outperforms the Rocchio metric in all three situations. Observe that the major downside of the Rocchio metric is its tendency to change the ranking of the results even when they are in perfect order. This is demonstrated by the negative NDCG change from baseline. The Web-Graph Reranker (WGR), on the other hand, consistently improves the average NDCG of the queries.

It is important to note that the improvement is greatest when it would be of most help to the user. In other words, when the original results are ranked in a poor order, and the perfect result is buried somewhere underneath, the reranker is the most helpful. The Rocchio feedback measure also leads to larger improvements when the original NDCG is low. However, it *decreases* the average NDCG when the original score is high, whereas the web-graph reranker does not change the ordering on those pages.

When selecting five URLs at random for the user to rate the reranker improves the quality of the results, but has a relatively low observed recall of 22%. (The recall is 52%, but the reranker often agrees with the ordering provided by the search engine). The predictive recall for this dataset was 42%. This issue is present in the Rocchio metric as well, which has an observed recall of 12%. The details are provided in Figure 2.

[3]In the future we plan to test against other relevance feedback algorithms.

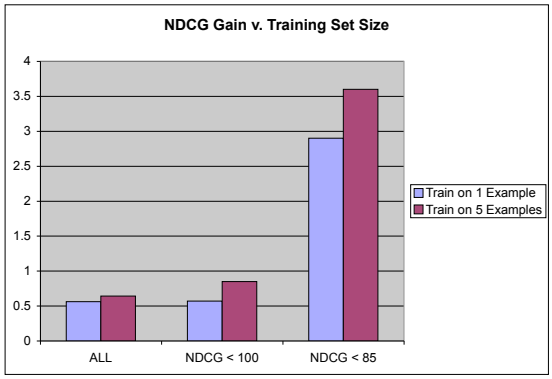
6.3 Training Set Size and Selection

One of the main parameters implicit in this experiment is the number of URLs that the user is required to rate for the method to achieve good improvements. Our studies show that many of the results are highly interconnected in the web-graph. As expected, the larger the size of the user rated results, the better the performance of the reranker, both in the NDCG and recall metrics. A testament to the connectivity of the web is the surprising result that even ranking a single URL can produce noticeable improvements in the quality of the ranking. The results are shown in Figure 3, where the rated results were chosen uniformly at random from the result set.

The other tunable parameter is the selection of the URL to be ranked. Clearly ranking some of URLs is going to be more beneficial than others. In this section we consider the changes by testing four different training set selection methods. In all cases the training set consisted of one URL.

- Random: Pick a URL uniformly at random from the result set.
- Top-Ranked: Select the top ranked URL, since this is the URL that the user will always see and evaluate.
- Most-Linked. In this scheme we select the URL that has the most number of connections to other URLs in the result set. This is the URL that we would expect to affect the ranking the most.
- Oracle: For every query, we run the experiment selecting every possible URL for the training set, and recording the best result.

The results of these experiments are presented in Figure 4. Of the three non-oracle selection methods, it is not surprising that *most-linked* achieves the highest observed recall.



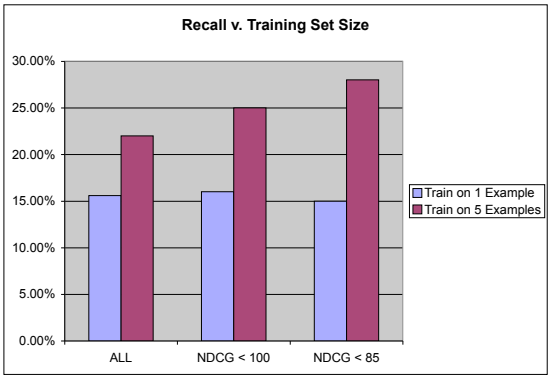
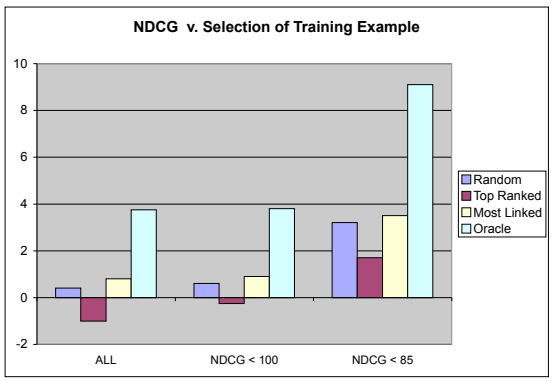


Figure 3: Increasing the size of the training set from 1 user rated result to 5 user rated results increases both (a) the NDCG change and (b) the observed recall.



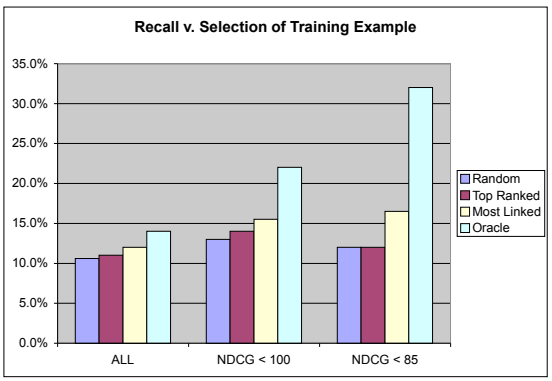


Figure 4: The effect of different selection rules on performance of the algorithm. (a) Asking the user to rate the top ranked element can be detrimental to the relevance feedback mechanism. (b) Not surprisingly, the rating of the most linked result gives the highest recall. The gap between that and the Oracle numbers indicates a better selection method may be possible

This method also achieves the highest NDCG change. What is surprising however, is the performance of rating the top-ranked result. Our experiments show that in many cases the top result is "special" in a sense that rating the top result can be detrimental to the overall search reranking. This may be due to the fact that search engines utilize extra heuristics in selecting the top element in the list. This implies that it may be best to present a set of URLs to rate to the user, rather than allowing him to pick URLs to rate. The latter method will invariably lead to the user rating the top result, since this is the result most often evaluated when judging the results.

As a sidenote, observe that while the above results measure the NDCG change in the test set, the test sets for the different metrics are not identical. Recall that the test set is the set of URLs that were not rated by the user. Therefore, while the top ranked result may be included in the test set by the "Random" selection method, it will always be part of the *training* (and not test) set for the "Top-Ranked" method.

6.4 Crawl Sizes

Recall that in an effort to limit the computational intensity of the reranker, we performed two crawls around the URLs in the experiment set. In the small crawl the total in- and out-degrees were limited to 35. In the large crawl the total degrees were limited to 70. The effects of different crawl sizes are present in Figure 5. Although on the majority of the pages no sampling was performed, we expected to see a change in the recall rates when running the experiment over the two different crawls. However, the recall over the two datasets was identical. This leads us to believe that our hypothesis is robust: if two URLs are connected in the web-graph by a short path, they are typically connected by

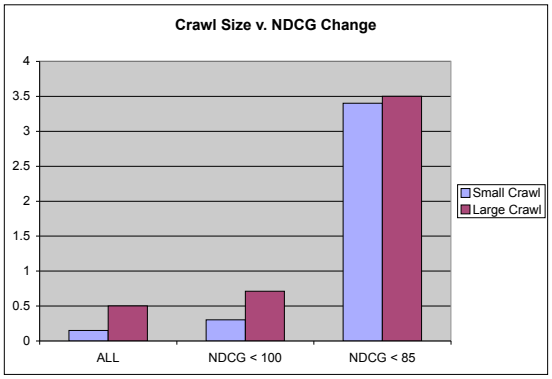


Figure 5: The effect of crawl size on relevance feedback. The web-graph distance measure is robust, as even with a small crawl of the system, the average NDCG improvement is still dramatic, and the recall of the overall system is unchanged.

multiple paths and thus the sampling of links in each webpage does not change the recall property.

While recall was not affected, the overall NDCG change was different by approximately 0.3 on all levels. This is due to the fact that the web-graph distance is a noisy signal about the relative relevance of two URLs. However, the number of paths found typically increases with the crawl size, which reduces the error rate of the algorithm due to noise in the signal.

7. CONCLUSION AND FUTURE WORK

Our research indicates that interactive search systems can improve the average quality of the rankings. The web-graph reranker has several attractive properties:

- Search engines strive for diversity in search results. The reranker offers the user a simple way to let the user narrow down the search to the desired category.
- Spam search results are often tightly interlinked. This is due to the individual sites trying to increase their PageRank and participating in link farms. The web-graph reranker can cut out large chunks of links farms from the search results with a single rating from the user.
- The rating of the results improves the user experience, while at the same time provides useful rating data to the search engine designers.

There are several improvements possible to the web-graph based reranker. One of the biggest open questions is the improvement of the recall metric. Even when the original query NDCG < 85, and thus the result ordering is far from optimal, the observed recall of our system is around 30%. In large part this is due to some of the results not being nearby in the web-graph. In these situations a hybrid method analyzing the link structure of the web as well as the text on the individual web pages can probably increase the recall without sacrificing the NDCG gain of the algorithm.

In addition, new relevance feedback methods have been shown to perform better than Rocchio's original scheme, especially in web search scenarios [19]. We plan a comprehensive set of experiments comparing text based relevance feedback methods with the web graph reranker.

Overall, we view this work as a step towards a more interactive search environment. Figuring out user intent from two or three query words has been recognized as a difficult problem and interactive methods appear to be a good stepping stone towards eliciting more information from the user when necessary.

8. REFERENCES

[1] Lars Arge, Gerth Stølting Brodal, and Laura Toma. On external-memory MST, SSSP, and Multi-way Planar Graph Separation. In *SWAT*, pages 433–447, 2000.

[2] Doug Beeferman and Adam L. Berger. Agglomerative clustering of a search engine query log. In *KDD*, pages 407–416, 2000.

[3] N. Belkin, C. Cool, J. Koenemann, K. Ng, and S. Park. Using relevance feedback and ranking in interactive searching, 1996.

[4] J. Dean and M. Henzinger. Finding related pages in the World Wide Web, 1999.

[5] Andrew V. Goldberg and Chris Harrelson. Computing the shortest path: search meets graph theory. In *SODA*, pages 156–165, 2005.

[6] Andrew V. Goldberg, Haim Kaplan, and Chris Harrelson. Reach for A*: Efficient point-to-point shortest path algorithms. In *ALENEX*, 2006.

[7] Kalervo Jarvelin and Jaana Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 41–48, New York, NY, USA, 2000. ACM Press.

[8] Glen Jeh and Jennifer Widom. Scaling personalized web search. In *WWW*, pages 271–279, 2003.

[9] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[10] Jurgen Koenemann and Nicholas J. Belkin. A case for interaction: A study of interactive information retrieval behavior and effectiveness. In *CHI*, pages 205–212, 1996.

[11] Kurt Mehlhorn and Ulrich Meyer. External-memory breadth-first search with sublinear I/O. In *ESA*, pages 723–735, 2002.

[12] MSN search. `http://search.msn.com`.

[13] Seda Ozmutlu, Amanda Spink, and Huseyin C. Ozmutlu. A day in the life of web searching: an exploratory study. *Inf. Process. Manage.*, 40(2):319–345, 2004.

[14] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[15] Tao Qin, Tie-Yan Liu, Xu-Dong Zhang, Zheng Chen, and Wei-Ying Ma. A study of relevance propagation for web search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 408–415, New York, NY, USA, 2005. ACM Press.

[16] J. J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Indexing*, pages 324–336. Prentice Hall, Englewood Cliffs, NJ, 1971.

[17] Azadeh Shakery and ChengXiang Zhai. Relevance propagation for topic distillation UIUC TREC 2003 Web Track Experiments. In *TREC*, pages 673–677, 2003.

[18] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456, New York, NY, USA, 2005. ACM Press.

[19] Vishwa Vinay, Kenneth R. Wood, Natasa Milic-Frayling, and Ingemar J. Cox. Comparing relevance feedback algorithms for web search. In Allan Ellis and Tatsuya Hagino, editors, *WWW (Special interest tracks and posters)*, pages 1052–1053. ACM, 2005.

[20] Vivisimo search engine. `http://www.vivisimo.com`.

[21] Yahoo mindset. `http://mindset.research.yahoo.com`.