Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search

Taher H. Haveliwala

Stanford University
taherh@cs.stanford.edu

Abstract. The original PageRank algorithm for improving the ranking of search-query results computes a single vector, using the link structure of the Web, to capture the relative "importance" of Web pages, independent of any particular search query. To yield more accurate search results, we propose computing a *set* of PageRank vectors, biased using a set of representative topics, to capture more accurately the notion of importance with respect to a particular topic. For ordinary keyword search queries, we compute the topic-sensitive PageRank scores for pages satisfying the query using the topic of the query keywords. For searches done in context (e.g., when the search query is performed by highlighting words in a Web page), we compute the topic-sensitive PageRank scores using the topic of the context in which the query appeared. By using linear combinations of these (precomputed) biased PageRank vectors to generate context-specific importance scores for pages at query time, we show that we can generate more accurate rankings than with a single, generic PageRank vector. We describe techniques for efficiently implementing a large scale search system based on the topic-sensitive PageRank scheme.

Keywords: web search, web graph, link analysis, PageRank, search in context, personalized search, ranking algorithm

1 Introduction

Various link-based ranking strategies have been developed recently for improving Web-search query results. The HITS algorithm proposed by Kleinberg [22] relies on query-time processing to deduce the *hubs* and *authorities* that exist in a subgraph of the Web consisting of both the results to a query and the local neighborhood of these results. Bharat and Henzinger [4] augment the HITS algorithm with content analysis to improve precision for the task of retrieving documents related to a query topic (as opposed to retrieving documents that exactly satisfy the user's information need). Chakrabarti et al. [8] make use of HITS for automatically compiling resource lists for general topics.

The PageRank algorithm, introduced by Page et al. [26], precomputes a rank vector that provides a-priori "importance" estimates for all of the pages on the Web. This vector is computed once, offline, and is independent of the search query. At query time, these importance scores are used in conjunction with query-specific IR scores to rank the query results [7]. PageRank has a clear efficiency advantage over the HITS algorithm, as the query-time cost of incorporating the *precomputed* PageRank importance score for a page is low. Furthermore, as PageRank is generated using the entire Web graph, rather than a small subset, it is less susceptible to localized link spam. Figure 1 illustrates a system utilizing the standard PageRank scheme.

---

[0] Submitted July 15, 2002; revised Dec. 15, 2002.

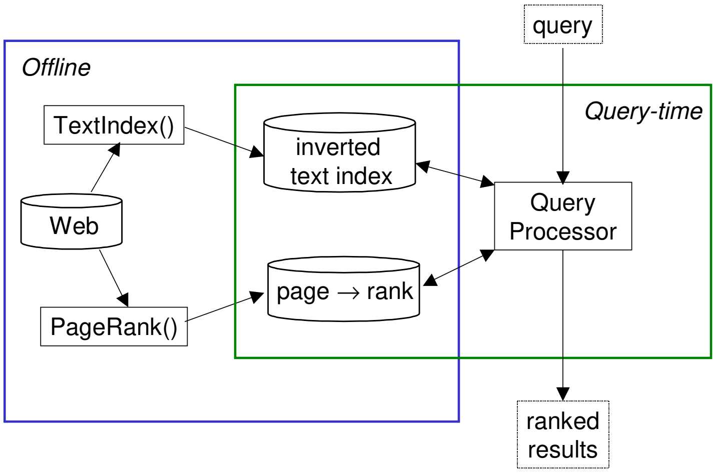

Fig. 1. Simplified diagram illustrating a simple search engine utilizing the standard PageRank scheme.

We propose an approach that (as with HITS) allows query-time information to influence the link-based score, yet (as with PageRank) requires minimal query-time processing. In our model, we compute offline a *set* of PageRank vectors, each biased with a different topic, to create for each page a *set* of importance scores with respect to particular topics [18]. The idea of biasing the PageRank computation was first suggested in [26] for the purpose of *personalization*, but was never fully explored. This biasing process involves the introduction of artificial links into the Web graph during the offline rank computation, and is described further in Section 2.

The recent work of Chakrabarti et al. [10] and Pennock et al. [27] demonstrates that the properties of the web graph are sensitive to page topic. In particular, [10] shows that pages tend to point to other pages that are on the same "broad" topic. Although this property helps explain why a query-independent PageRank score can be useful for ranking, it also suggests that we may be able to improve the performance of link-based computations by taking into account page topics. By making PageRank topic-sensitive, we avoid the problem of heavily linked pages getting highly ranked for queries for which they have no particular authority [3]. Pages considered important in some subject domains may not be considered important in others, regardless of what keywords may appear either in the page or in anchor text referring to the page. The *Hilltop* approach suggested by Bharat and Mihaila [5] has a similar motivation, but is designed to improve results for *popular* queries. Hilltop generates a query-specific authority score by detecting and indexing pages that appear to be good experts for certain keywords, based on their outlinks. However, query terms for which experts were not found will not be handled by the Hilltop algorithm.

Rafiei and Mendelzon [28] propose using the set of Web pages that contain some term as a bias set for influencing the PageRank computation, with the goal of returning terms for which a *given* page has a high reputation. An approach for enhancing search rankings by generating a PageRank vector for each possible query term was proposed by Richardson and Domingos [29] with favorable results. However, the approach requires considerable processing time and storage, and is not easily extended to make use of user and query *context*. Diligenti et al. [11] propose topic-specific PageRank scores for enhancing *vertical search*. Our approach to biasing the PageRank computation is novel in its use of a small number of representative basis topics, taken from the Open Directory, in conjunction with a multinomial naive-Bayes classifier for classifying the query and query context [18].

In our work we consider two scenarios. In the first, we assume a user with a specific information need issues a query to our search engine in the conventional way, by entering a query into a search box. In this scenario, we determine the topics most closely associated with the query, and use the appropriate topic-sensitive PageRank vectors for ranking the documents satisfying the query. This ensures that the "importance" scores reflect a preference for the link structure of pages that have some bearing on the query. As with ordinary PageRank, the topic-sensitive PageRank score can be used as part of a scoring function that takes into account other IR-based scores. In the second scenario, we assume the user is viewing a document (for instance, browsing the Web or reading email), and selects a term from the document for which he would like more information. This notion of *search in context* is discussed by Finkelstein et al. [14]. For instance, if a query for "architecture" is performed by highlighting a term in a document discussing famous building architects, we would like the result to be different than if the query "architecture" is performed by highlighting a term in a document on CPU design. By selecting the appropriate topic-sensitive PageRank vectors based on the context of the query, we hope to provide more accurate search results. Note that even when a query is issued in the conventional way, without highlighting a term, the *history* of queries issued constitutes a form of query context. Yet another source of context comes from the *user* who submitted the query. For instance, the user's bookmarks and browsing history could be used in selecting the appropriate topic-sensitive rank vectors. These various sources of search context are discussed in Section 5.

A summary of our approach follows. During the offline processing of the Web crawl, we generate 16 topic-sensitive PageRank vectors, each biased (as described in Section 2) using URLs from a top-level category from the Open Directory Project (ODP) [2]. At query time, we calculate the similarity of the query (and if available, the query or user context) to each of these topics. Then instead of using a single global ranking vector, we take the linear combination of the topic-sensitive vectors, weighted using the similarities of the query (and any available context) to the topics. By using a *set* of rank vectors, we are able to determine more accurately which pages are truly the most important with respect to a particular query or query-context. Because the link-based computations are performed offline, during the preprocessing stage, the query-time costs are not much greater than that of the ordinary PageRank algorithm. An illustration of our topic-sensitive PageRank system is given in Figure 2.

2 Review of PageRank

A review of the PageRank algorithm follows. The basic idea of PageRank is that if page u has a link to page v, then the author of u is implicitly conferring some importance to page v. Intuitively, *Yahoo!* is an important page, reflected by the fact that many pages point to it. Likewise, pages prominently pointed to from *Yahoo!* are themselves probably important. How much importance does a page u confer to its outlinks? Let N_u be the outdegree of page u, and let $Rank(p)$ represent the importance (i.e., PageRank) of page p. Then the link (u, v) confers $Rank(u)/N_u$ units of rank to v. This simple idea leads to the following iterative fixpoint computation that yields the rank vector $\boldsymbol{Rank}^*$ over all of the pages on the Web. If n is the number of pages, assign all pages the initial value $1/n$. Let B_v represent the set of pages pointing to v. In each iteration, propagate the ranks as follows:[1]

$$\forall_v Rank^{(i+1)}(v) = \sum_{u \in B_v} Rank^{(i)}(u)/N_u \tag{1}$$

We continue the iterations until $\boldsymbol{Rank}$ stabilizes to within some threshold. The final vector $\boldsymbol{Rank}^*$ contains the PageRank vector over the Web. This vector is computed only once after each crawl of the Web; the values can then be used to influence the ranking of search results [1].

[1] Note that for $u \in B_v$, the edge (u, v) guarantees $N_u \geq 1$.

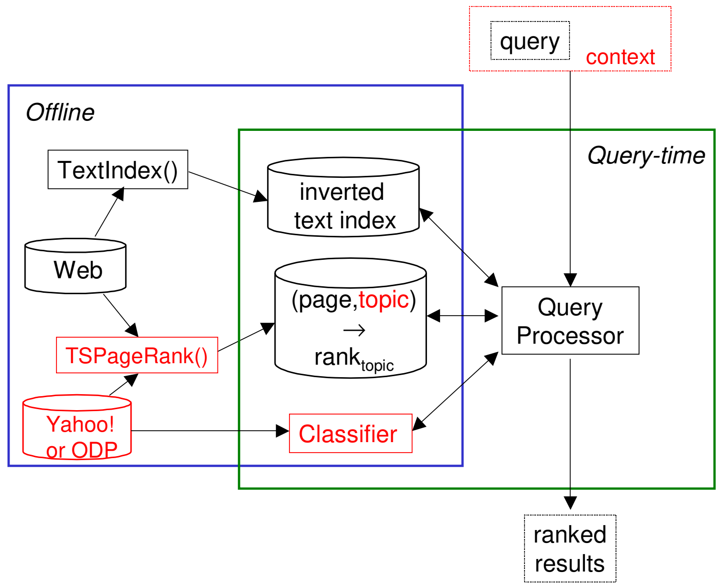

Fig. 2. Illustration of our system utilizing topic-sensitive PageRank.

The process can also be expressed as the following eigenvector calculation, providing useful insight into PageRank. Let M be the square, stochastic matrix corresponding to the directed Web graph G.[2] If there is a link from page j to page i, then let the matrix entry m_{ij} have the value $1/N_j$. Let all other entries have the value 0. One iteration of the previous fixpoint computation corresponds to the matrix-vector multiplication $M \times \boldsymbol{Rank}$. Repeatedly multiplying $\boldsymbol{Rank}$ by M yields the dominant eigenvector $\boldsymbol{Rank}^*$ of the matrix M. In other words, $\boldsymbol{Rank}^*$ is the solution to

$$\boldsymbol{Rank} = M \times \boldsymbol{Rank} \tag{2}$$

Because M^T is the stochastic transition matrix over the graph G, PageRank can be viewed as the stationary probability distribution for the Markov chain induced by a random walk on the Web graph.

One caveat is that the convergence of PageRank is only guaranteed if M is irreducible (i.e., G is strongly connected) and aperiodic [25]. The latter is guaranteed in practice for the Web, while the former is true if we (a) add a complete set of outgoing edges to nodes in G with outdegree 0, and (b) damp the rank propagation by a factor $1 - \alpha$ by adding a complete set of outgoing edges, with weight α/n, to all nodes. We can accomplish this task by constructing a new matrix M' in the following way. Let n be the number of nodes (i.e., pages) in the Web graph. Let $\boldsymbol{p}$ be the n-dimensional column vector representing a uniform probability distribution over all nodes:

$$\boldsymbol{p} = [\frac{1}{n}]_{n \times 1} \tag{3}$$

Let $\boldsymbol{d}$ be the n-dimensional column vector identifying the nodes with outdegree 0:

$$d_i = \begin{cases} 1 & \text{if } \deg(j) = 0, \\ 0 & \text{otherwise} \end{cases}$$

[2] Assume for now that all nodes in G have at least one outgoing edge.

Then we construct M' as follows:

$$\begin{aligned}
D &= \boldsymbol{p} \times \boldsymbol{d}^{\,T} \\
E &= \boldsymbol{p} \times [1]_{1\times n} \\
&= [\frac{1}{n}]_{n\times n} \qquad\qquad \textit{if } \boldsymbol{p} \textit{ is the uniform distribution} \\
M' &= (1-\alpha)(M+D) + \alpha E
\end{aligned} \tag{4}$$

This modification improves the quality of PageRank by introducing a decay factor $1-\alpha$ which limits the effect of rank *sinks* [26], in addition to guaranteeing convergence to a unique rank vector. Substituting M' for M in Equation 2, we can express PageRank as the solution to:[3]

$$\boldsymbol{Rank} = M' \times \boldsymbol{Rank} \tag{5}$$
$$= (1-\alpha)(M+D) \times \boldsymbol{Rank} + \alpha\boldsymbol{p} \tag{6}$$

with $\boldsymbol{p} = [\frac{1}{n}]_{n\times 1}$. The key to creating topic-sensitive PageRank is that we can bias the computation to increase the effect of certain categories of pages by using a nonuniform $n \times 1$ personalization vector for $\boldsymbol{p}$.[4] To ensure that M' is irreducible when $\boldsymbol{p}$ contains any 0 entries, nodes not reachable from nonzero nodes in $\boldsymbol{p}$ should be removed. This modification is not implementationally problematic. Note that the topic-based influencing involves introducing additional rank to the appropriate nodes in *each* iteration of the computation – it is not simply a postprocessing step performed on the standard PageRank vector.

In terms of the random-walk model, the personalization vector represents the addition of a complete set of transition edges where the probability on an artificial edge (u, v) is given by αp_v. We will denote the solution $\boldsymbol{Rank^*}$ of Equation 6, with $\alpha = \alpha^*$ and a particular $\boldsymbol{p} = \boldsymbol{p^*}$, as $\boldsymbol{PR}(\alpha^*, \boldsymbol{p^*})$. By appropriately selecting $\boldsymbol{p}$, the rank vector can be made to prefer certain categories of pages. The bias factor α specifies the degree to which the computation is biased towards $\boldsymbol{p}$.

3 Topic-Sensitive PageRank

In our approach to topic-sensitive PageRank, we precompute the importance scores offline, as with ordinary PageRank. However, we compute multiple importance scores for each page; we compute a set of scores of the importance of a page with respect to various topics. At query time, these importance scores are combined based on the topics of the query to form a composite PageRank score for those pages matching the query. This score can be used in conjunction with other IR-based scoring schemes to produce a final rank for the result pages with respect to the query. As the scoring functions of commercial search engines are not known, in our work we do not consider the effect of these IR scores (other than requiring that the query terms appear in the page).[5] We believe that the improvements to PageRank's precision will translate into improvements in overall search rankings, even after other IR-based scores are factored in. Note that the topic-sensitive PageRank score itself implicitly makes use of IR in determining the topic of the *query*. However this use of IR is not vulnerable to manipulation of *pages* by adversarial webmasters seeking to raise the score of their sites.

---

[3] Equation 6 makes use of the fact that $\|\boldsymbol{Rank}\|_1 = 1$.

[4] Page et al.[26] originally suggest setting $\boldsymbol{p}$ directly using the bookmarks of the user, although that approach is not practical for large numbers of users.

[5] For instance, most search engines use term weighting schemes which make special use of HTML tags.

3.1 ODP-biasing

The first step in our approach is to generate a set of biased PageRank vectors using a set of basis topics. This step is performed once, offline, during the preprocessing of the Web crawl. There are many possible sources for the basis set of topics. However, using a small basis set is important for keeping the preprocessing and query-time costs low. One option is to cluster the Web page repository into a small number of clusters in the hopes of achieving a representative basis. We chose instead to use the freely available, hand constructed Open Directory as a source of topics. To keep the basis set small, we made use of the 16 top level categories; using finer grained basis sets is part of our future work, and is discussed in Sections 6 and 7.

Let T_j be the set of URLs in the ODP category c_j. Then when computing the PageRank vector for topic c_j, in place of the uniform damping vector $\boldsymbol{p} = [\frac{1}{n}]_{n\times 1}$, we use the nonuniform vector $\boldsymbol{p} = \boldsymbol{v_j}$ where

$$v_{ji} = \begin{cases} \frac{1}{|T_j|} & i \in T_j, \\ 0 & i \notin T_j. \end{cases} \tag{7}$$

The PageRank vector for topic c_j is given by $\boldsymbol{PR}(\alpha, \boldsymbol{v_j})$. We also generate the single unbiased PageRank vector (denoted as NoBias) for the purpose of comparison. The choice of α will be discussed in Section 4.1.

We also compute the 16 class term vectors $\boldsymbol{D_j}$ consisting of the terms in the documents below each of the 16 top-level categories. D_{jt} simply gives the total number of occurrences of term t in documents listed below class c_j of the ODP.

As mentioned previously, one could envision using other sources of topics; however, the ODP data is freely available, and as it is compiled by thousands of volunteer editors, is less susceptible to influence by any one party. In Section 7 we describe a modification of the above construction that reduces the ability for even malicious ODP editors to affect scores in any non-negligible way.

3.2 Query-Time Importance Score

The second step in our approach is performed at query time. Given a query q, let q' be the context of q. In other words, if the query was issued by highlighting the term q in some Web page u, then q' consists of the terms in u. Alternatively, we could use only those terms in u nearby the highlighted term, as often times a single Web page may discuss a variety of topics. For ordinary queries not done in context, let $q' = q$. Using a multinomial naive-Bayes classifier [24],[6] with parameters set to their maximum-likelihood estimates, we compute the class probabilities for each of the 16 top-level ODP classes, conditioned on q'. Let q'_i be the ith term in the query (or query context) q'. Then given the query q, we compute for each c_j the following:

$$P(c_j|q') = \frac{P(c_j) \cdot P(q'|c_j)}{P(q')} \propto P(c_j) \cdot \prod_i P(q'_i|c_j) \tag{8}$$

$P(q'_i|c_j)$ is easily computed from the class term-vector $\boldsymbol{D_j}$. The quantity $P(c_j)$ is not as straightforward. We chose to make it uniform, although we could personalize the query results for different *users* by varying this distribution. In other words, for some user k, we can use a prior distribution $P_k(c_j)$ that reflects the interests of user k. This method provides an alternative framework for user-based personalization, rather than directly varying the damping vector $\boldsymbol{p}$ as had been suggested in [6, 7, 26].

[6] The multivariate-Bernoulli and the multinomial event models for naive-Bayes text classification are compared in [23]. The multinomial event model, which corresponds to a unigram language model, performs better in most scenarios.

Using a text index, we retrieve URLs for all documents containing the *original* query terms q. Finally, we compute the query-sensitive importance score of each of these retrieved URLs as follows. Let r_{jd} be the rank of document d given by the rank vector $\boldsymbol{PR}(\alpha, \boldsymbol{v_j})$ (i.e., the rank vector for topic c_j). For the Web document d, we compute the query-sensitive importance score s_{qd} as follows.

$$s_{qd} = \sum_j P(c_j|q') \cdot r_{jd} \tag{9}$$

The results are ranked according to this composite score s_{qd}.[7]

The above query-sensitive PageRank computation has the following probabilistic interpretation, in terms of the "random surfer" model [26]. Let w_j be the coefficient used to weight the jth rank vector, with $\sum_j w_j = 1$ (e.g., let $w_j = P(c_j|q)$). Then note that the equality

$$\sum_j [w_j \boldsymbol{PR}(\alpha, \boldsymbol{v_j})] = \boldsymbol{PR}\big(\alpha, \sum_j [w_j \boldsymbol{v_j}]\big) \tag{10}$$

holds, as shown in Appendix A. Thus we see that the following random walk on the Web yields the topic-sensitive score s_{qd}. With probability $1 - \alpha$, a random surfer on page u follows an outlink of u (where the particular outlink is chosen uniformly at random). With probability $\alpha P(c_j|q')$, the surfer instead jumps to one of the pages in T_j (where the particular page in T_j is chosen uniformly at random). The long term visit probability that the surfer is at page v is exactly given by the composite score s_{qd} defined above. Thus, topics exert influence over the final score in proportion to their affinity with the query (or query context).

4 Experimental Results

We conducted a series of experiments to measure the behavior of topic-sensitive PageRank. In Section 4.1 we describe the similarity measure we use to compare two rankings. In Section 4.2, we investigate how the induced rankings vary, based on both the topic used to bias the rank vectors as well as the choice of the bias factor α. In Section 4.3, we present results of a user study showing the retrieval performance of ordinary PageRank versus topic-sensitive PageRank. In Section 4.4, we provide an initial look at how the use of query context can be used in conjunction with topic-sensitive PageRank.

As a source of Web data, we used the latest Web crawl from the Stanford WebBase [19], performed in January 2001, containing roughly 120 million pages. Our crawl contained roughly 280,000 of the 3 million URLs in the ODP. For our experiments, we used 35 of the sample queries given in [12], which were in turn compiled from earlier papers.[8] The queries are listed in Table 1.

4.1 Similarity Measure for Induced Rankings

We use two measures when comparing rankings. The first measure, denoted $\mathrm{OSim}(\tau_1, \tau_2)$, indicates the degree of overlap between the top k URLs of two rankings, τ_1 and τ_2. We define the overlap of two sets A and B (each of size k) to be $\frac{|A \cap B|}{k}$. In our comparisons we will use $k = 20$. The overlap measure OSim gives an incomplete picture of the similarity of two rankings, as it does not indicate the degree to which the relative orderings of the top k URLs of two rankings are in agreement. Therefore, in addition to OSim, we use a second measure, KSim, based on Kendall's τ distance measure.[9]

[7] Alternatively, s_{qd} can be used as part of a more general scoring function.

[8] Several queries which produced very few hits on our repository were excluded.

[9] Note that the schemes for comparing top k lists recently proposed by Fagin et al. [13], also based on Kendall's τ distance measure, differ from KSim in the way normalization is done.

Table 1. Test queries used.

affirmative action	lipari
alcoholism	lyme disease
amusement parks	mutual funds
architecture	national parks
bicycling	parallel architecture
blues	recycling cans
cheese	rock climbing
citrus groves	san francisco
classical guitar	shakespeare
computer vision	stamp collecting
cruises	sushi
death valley	table tennis
field hockey	telecommuting
gardening	vintage cars
graphic design	volcano
gulf war	zen buddhism
hiv	zener
java	

For consistency with OSim, we will present our definition as a similarity (as opposed to distance) measure, so that values closer to 1 indicate closer agreement. Consider two partially ordered lists of URLs, τ_1 and τ_2, each of length k. Let U be the union of the URLs in τ_1 and τ_2. If δ_1 is $U - \tau_1$, then let τ_1' be the extension of τ_1, where τ_1' contains δ_1 appearing after all the URLs in τ_1.[10] We extend τ_2 analogously to yield τ_2'. We define our similarity measure KSim as follows:

$$\mathrm{KSim}(\tau_1, \tau_2) = \frac{|(u, v) : \tau_1', \tau_2' \text{ agree on order of } (u, v), u \neq v|}{(|U|)(|U| - 1)} \tag{11}$$

In other words, $\mathrm{KSim}(\tau_1, \tau_2)$ is the probability that τ_1' and τ_2' agree on the relative ordering of a randomly selected pair of distinct nodes $(u, v) \in U \times U$.[11]

4.2 Effect of ODP-Biasing

In this section we measure the effects of topically biasing the PageRank computation. First, note that the choice of the bias factor α, discussed in Section 2, affects the degree to which the resultant vector is biased towards the topic vector used for $\boldsymbol{p}$. Consider the extreme cases. For $\alpha = 1$, the URLs in the bias set T_j will be assigned the score $\frac{1}{|T|}$, and all other URLs receive the score 0. Conversely, as α tends to 0, the content of T_j becomes irrelevant to the final score assignment.

We heuristically set $\alpha = 0.25$ after inspecting the rankings for several of the queries listed in Table 1. We did not concentrate on optimizing this parameter; although α affects the induced rankings of query results, the differences *across* different topically-biased PageRank vectors, for a fixed α, are much higher. For instance, for $\alpha = 0.05$ and $\alpha = 0.25$, we measured the average similarity of the induced rankings across our set of test queries, for each of our PageRank vectors.[12] The results are given in Table 2. We see that the average overlap

[10] The URLs in δ are placed with the *same* ordinal rank at the end of τ.

[11] A pair ordered in one list and tied in the other is considered a disagreement.

[12] We used 25 iterations of PageRank in all cases.

Table 2. Average similarity of rankings for $\boldsymbol{\alpha} = \{\mathbf{0.05}, \mathbf{0.25}\}$.

Bias Set	$OSim$	$KSim$
NoBias	0.72	0.64
Arts	0.66	0.58
Business	0.63	0.54
Computers	0.70	0.60
Games	0.78	0.67
Health	0.73	0.62
Home	0.77	0.67
Kids & Teens	0.74	0.66
News	0.74	0.65
Recreation	0.62	0.55
Reference	0.68	0.57
Regional	0.60	0.52
Science	0.69	0.59
Shopping	0.66	0.55
Society	0.57	0.50
Sports	0.69	0.60
World	0.64	0.55

between the top 20 results for the two values of α is high. Furthermore, the high values for KSim indicate high overlap as well as agreement (on average) on the relative ordering of these top 20 URLs for the two values of α. Chakrabarti et al. [10] suggest that the ideal choice of α may differ for different topics; choosing the optimal α for each topic is an avenue for future study. All subsequent experiments in this paper use $\alpha = 0.25$.

We now discuss the difference between rankings induced by different topically-biased PageRank vectors. We computed the average, across our test queries, of the pairwise similarity between the rankings induced by the different topically-biased vectors. The 5 most similar pairs, according to our OSim measure, are given in Table 3, showing that even the most similar topically-biased rankings have little overlap. Having established that the topic-specific PageRank vectors each rank the results substantially differently, we proceed to investigate which of these rankings is "best" for specific queries.

Table 3. Topic pairs yielding most similar rankings.

Bias-Topic Pair	$OSim$	$KSim$
(Games, Sports)	0.18	0.13
(NoBias, Regional)	0.18	0.12
(Kids & Teens, Society)	0.18	0.11
(Health, Home)	0.17	0.12
(Health, Kids & Teens)	0.17	0.11

As an example, Table 4 shows the top 4 ranked URLs for the query "bicycling," using several of the topically-biased PageRank vectors. Note in particular that the ranking induced by the Sports-biased vector is of high quality. Also note that the ranking induced by the

SHOPPING-biased vector leads to the high ranking of websites selling bicycle-related accessories.

4.3 Query-Sensitive Scoring

In this section we look at how effectively we can utilize the ranking precision gained by the use of multiple PageRank vectors. Given a query, our first task is to determine which of the rank vectors can best rank the results for the query. We found that using the quantity $P(c_j|q)$ as discussed in Section 3.2 yielded intuitive results for determining which topics are most closely associated with a query. In particular, for most of the test queries, the ODP categories with the highest values for $P(c_j|q)$ are intuitively the most relevant categories for the query. In Table 5, we list for several of the test queries the 3 categories with the highest values for $P(c_j|q)$. When computing the composite s_{qd} score in our experiments, we chose to use the convex combination of only the rank vectors associated with the three topics with the highest values for $P(c_j|q)$, rather than all of the topics. Based on the data in Table 5, we saw no need to include the scores from the topic vectors with lower associated values for $P(c_j|q)$.

To compare our query-sensitive approach to ordinary PageRank, we conducted a user study. We randomly selected 10 queries from our test set for the study, and found 5 volunteers. For each query, the volunteer was shown 2 result rankings; one consisted of the top 10 results satisfying the query, when these results were ranked with the unbiased PageRank vector, and the other consisted of the top 10 results for the query when the results were ranked with the composite s_{qd} score.[13] The volunteer was asked to select all URLs which were "relevant" to the query, in their opinion. In addition, they were asked to mark which of the two rankings was the better of the two, in their opinion. They were not told anything about how either of the rankings was generated.

Let a URL be considered *relevant* if at least 3 of the 5 volunteers selected it as relevant for the query. The *precision* then is the fraction of the top 10 URLs that are deemed *relevant*. The precision of the two ranking techniques for each test query is shown in Figure 3. The average precision for the rankings induced by the topic-sensitive PageRank scores is substantially higher than that of the unbiased PageRank scores; 0.51 vs. 0.28. Furthermore, as shown in Table 6, for nearly all queries, a majority of the users selected the rankings induced by the topic-sensitive PageRank scores as the better of the two. These results suggest that the effectiveness of a query-result scoring function can be improved by the use of a topic-sensitive PageRank scheme in place of a generic PageRank scheme.[14]

4.4 Context-Sensitive Scoring

In Section 4.3, the topic-sensitive ranking vectors were chosen using the topics most strongly associated with the query term. If the search is done in context, for instance by highlighting a term in a Web page and invoking a search, then the context can be used instead of the query to determine the topics. Using the context can help disambiguate the query term and yield results that more closely reflect the intent of the user. We now illustrate with an example how using query-context can help a system which uses topic-sensitive PageRank.

Consider the query "blues" taken from our test set. This term has several different senses; for instance it could refer to a musical genre, or to a form of depression. Two Web pages in which the term is used with these different senses, as well as short textual excerpts from the pages, are shown in Table 7. Consider the case where a user reading one of these two pages

[13] Both the title and URL were presented to the user. The title was a hyperlink to a current version of the Web page.

[14] The effect on the relative ranking performance of our scheme when aggregated with Web-oriented IR scores is an avenue of future investigation.

Table 4. Top results for the query "bicycling" when ranked using various topic-specific vectors.

NoBias	Arts
"RailRiders Adventure Clothing" www.RailRiders.com www.Waypoint.org/default.html www.Gorp.com/ www.FloridaCycling.com/	"Photo Contest & Gallery (Bicycling)" www.bikescape.com/photogallery/ www.trygve.com/ www.greenway.org/ www.jsc.nasa.gov/Bios/htmlbios/young.html
Business	**Computers**
"Recumbent Bikes and Kit Aircraft" www.rans.com www.BreakawayBooks.com java.oreilly.com/bite-size/ www.carbboom.com	"GPS Pilot" www.gpspilot.com www.wireless.gr/wireless-links.htm www.linkstosales.com www.LiftExperts.com/lifts.html
Games	**Kids and Teens**
"Definition Through Hobbies" www.flick.com/~gretchen/hobbies.html www.BellaOnline.com/sports/ www.npr.org/programs/wesun/puzzle/will.html www.trygve.com/	"Camp Shohola For Boys" www.shohola.com www.EarthForce.org www.WeissmanTours.com www.GrownupCamps.com/homepage.html
Recreation	**Science**
"Adventure travel" www.gorp.com/ www.GrownupCamps.com/homepage.html www.gorp.com/gorp/activity/main.htm www.outdoor-pursuits.org/	"Coast to Coast by Recumbent Bicycle" hypertextbook.com/bent/ www.SiestaSoftware.com/ www.BenWiens.com/benwiens.html www.SusanJeffers.com/jeffbio.htm
Shopping	**Sports**
"Cycling Clothing & Accessories for Women" www.TeamEstrogen.com/ www.ShopOutdoors.com/ www.jub.com.au/books/ www.bike.com/	"Swim, Bike, Run, & Multisport" www.multisports.com/ www.BikeRacing.com/ www.CycleCanada.com/ www.bikescape.com/photogallery/

Table 5. Estimates for $P(c_j|q)$ for a subset of the test queries.

alcoholism	
Health	0.47
Kids & Teens	0.20
Arts	0.06

bicycling	
Sports	0.52
Regional	0.13
Health	0.07

blues	
Arts	0.52
Shopping	0.12
News	0.08

citrus groves	
Shopping	0.34
Home	0.21
Regional	0.18

classical guitar	
Arts	0.75
Shopping	0.21
News	0.01

computer vision	
Computers	0.24
Business	0.14
Reference	0.09

cruises	
Recreation	0.65
Regional	0.18
Sports	0.04

death valley	
Regional	0.28
Society	0.14
News	0.10

field hockey	
Sports	0.89
Shopping	0.03
Reference	0.03

graphic design	
Computers	0.36
Business	0.23
Shopping	0.09

gulf war	
Society	0.21
Kids & Teens	0.18
Regional	0.17

hiv	
Health	0.40
News	0.19
Kids & Teens	0.14

java	
Computers	0.53
Games	0.10
Kids & Teens	0.06

lyme disease	
Health	0.96
Regional	0.01
Recreation	0.01

mutual funds	
Business	0.77
Regional	0.05
Home	0.05

parallel architecture	
Computers	0.70
Science	0.10
Reference	0.07

rock climbing	
Recreation	0.54
Regional	0.13
Sports	0.07

san francisco	
Sports	0.27
Regional	0.16
Recreation	0.10

shakespeare	
Arts	0.34
Reference	0.21
Kids & Teens	0.15

table tennis	
Sports	0.53
Shopping	0.14
Regional	0.09

telecommuting	
Business	0.70
Kids & Teens	0.04
Society	0.03

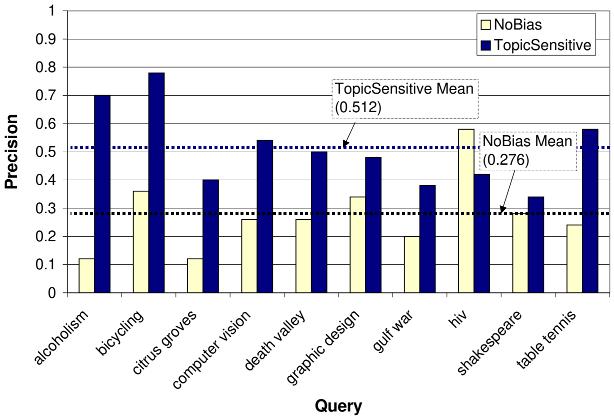

Fig. 3. Precision @ 10 results for our test queries. The average precision over the ten queries is also shown.

Table 6. Ranking scheme preferred by majority of users.

Query	Preferred by Majority
alcoholism	TopicSensitive
bicycling	TopicSensitive
citrus groves	TopicSensitive
computer vision	TopicSensitive
death valley	TopicSensitive
graphic design	TopicSensitive
gulf war	TopicSensitive
hiv	NoBias
shakespeare	Neither
table tennis	TopicSensitive

highlights the term "blues" to submit a search query. At query time, the first step of our system is to determine which topic best applies to the query in context. Thus, we calculate $P(c_j|q')$ as described in Section 3.2, using for q' the terms of the entire page,[15] rather than just the term "blues." For the first page (discussing music), $\text{argmax}_{c_j} P(c_j|q')$ is Arts, and for the second page (discussing depression), $\text{argmax}_{c_j} P(c_j|q')$ is Health. The next step is to use a text index to fetch a list of URLs for all documents containing the term "blues" — the highlighted term for which the query was issued. Finally, the URLs are ranked using the appropriate ranking vector that was selected using the $P(c_j|q')$ values (i.e., either Arts or Health). Table 8 shows the top 5 URLs for the query "blues" using the topic-sensitive PageRank vectors for Arts, Health, and NoBias. We see that as desired, most of the results ranked using the Arts-biased vector are pages discussing music, while all of the top results ranked using the Health-biased vector discuss depression. The context of the query allows the system to pick the appropriate topic-sensitive ranking vector, and yields search results reflecting the appropriate sense of the search term.

[15] It may be preferable to use only a window of terms surrounding the highlighted query terms. Determining the best window to use is left for future investigation.

Table 7. Two different search contexts for the query "blues".

That Blues Music Page	**Postpartum Depression & the 'Baby Blues'**
http://www.fred.net/turtle/blues.shtml	http://familydoctor.org/handouts/379.html
...If you're stuck for new material, visit Dan Bowden's Blues and Jazz Transcriptions - lots of older blues guitar transcriptions for you historic blues fans ...	...If you're a new mother and have any of these symptoms, you have what is called the "baby blues." "The blues" are considered a normal part of early motherhood and usually go away within 10 days after delivery. However, some women have worse symptoms or symptoms last longer. This is called "postpartum depression." ...

Table 8. Results for query "blues" using three different ranking vectors.

Arts	Health
Britannica Online www.britannica.com	Northern County Psychiatric Associates News www.baltimorepsych.com/news.htm
BandHunt.com Genres (Music) www.bandhunt.com/genres.html	Seasonal Affective Disorder www.ncpamd.com/seasonal.htm
Artist Information (Music) www.artistinformation.com/index.html	Women's Mental Health www.ncpamd.com/Women's_Mental_Health.htm
Billboard.com (Music charts) www.billboard.com	Wing of Madness Depression Support Group www.wingofmadness.com
Soul Patrol (Music) www.soul-patrol.com	Country Nurse Online www.countrynurse.com

NoBias
TUCOWS Themes news.tucows.com/themes/pastart.html
World's Most Popular MP3 Service www.emusic.com
Books, Music, DVD, and VHS Essentials www.johnholleman.com/amastatement.html
The Official Site of Major League Baseball www.majorleaguebaseball.com
MP3.com: Free MP3 Downloads www.mp3.com

5 Search Context

In Section 4.4, we gave an example of one possible source of context to utilize when generating the composite PageRank score, namely the document containing the query term highlighted by the user. There are a variety of other sources of context that may be used in our scheme. For instance, the history of queries issued leading up to the current query is another form of query context. A search for "basketball" followed up with a search for "Jordan" presents an opportunity for disambiguating the latter. As another example, most modern search engines incorporate some sort of hierarchical directory, listing URLs for a small subset of the Web, as part of their search interface.[16] The current node in the hierarchy that the user is browsing at constitutes a source of query context. When browsing URLs at TOP/ARTS, for instance, any queries issued could have search results (from the entire Web index) ranked with the ARTS rank vector, rather than either restricting results to URLs listed in that particular category, or not making use of the category whatsoever. In addition to these types of context associated with the query itself, we can also potentially utilize query independent *user* context. Sources of user context include the users' browsing patterns, bookmarks, and email archives. As mentioned in Section 3.2, we can integrate user context by selecting a nonuniform prior, $P_k(c_j)$, based on how closely the user's context accords with each of the basis topics.

When attempting to utilize the aforementioned sources of search context, mediating the personalization of PageRank via a set of basis topics yields several benefits over attempting to explicitly construct a personalization vector.

Flexibility: For any kind of context, we can compute the context-sensitive PageRank score by using a classifier to compute the similarity of the context with the basis topics and then weighting the topic-sensitive PageRank vectors appropriately. We can treat such diverse sources of search context such as email, bookmarks, browsing history, and query history uniformly.

Transparency: The topically-biased rank vectors have intuitive interpretations. If we see that our system is giving undue preference to certain topics, we can tune the classifier used on the search context, or adjust topic weights manually. When utilizing user context, the users themselves can be shown what topics the system believes represent their interests.

Privacy: Certain forms of search context raise potential privacy concerns. Clearly it is inappropriate to send the user's browsing history or other personal information to the search-engine server for use in constructing a profile. However a *client-side* program could use the user context to generate the user profile locally, and send only the summary information, consisting of the weights assigned to the basis topics, over to the server. The amount of privacy lost in knowing only that the user's browsing pattern suggests that he is interested in COMPUTERS with weight 0.5 is much less than actually obtaining his browser's history cache. When making use of query-context, if the user is browsing sensitive personal documents, they would be more comfortable if the search client sent to the server topic weights rather than the actual document text surrounding the highlighted query term.

Efficiency: For a small number of basis topics (such as the 16 ODP categories), both the query-time cost and the offline preprocessing cost of our approach is low, and practical to implement with current Web indexing infrastructure.

Given the rapid expansion of the Web, with no corresponding increase in query specificity, a large portion of the Web is effectively inaccessible to users who issue simple search queries. By utilizing search context, ranking functions can help solve the problem of allowing effective search with simple queries over a rapidly expanding Web.

[16] See for instance `http://directory.google.com/Top/Arts/` or `http://dir.yahoo.com/Arts/`.

6 Efficiency Considerations

In this section, we discuss the time and space complexity of both the offline and query-time components of a search engine utilizing the topic-sensitive PageRank scheme.

6.1 Offline Processing

We begin with a discussion of the space and time complexity for a straightforward implementation of the offline step. The offline processing was done on a machine with dual 1533 Mhz AMD Athlon CPUs, 480 GB RAID-5, and 2.5 GB of main memory. As mentioned previously, we used the Stanford WebBase repository, containing roughly 120 million pages, as our source of data. The link graph consisted of the crawled URLs, as well as URLs on the "frontier", for a total of 360 million URLs, and required 7.3 GB of storage (uncompressed). After removing 2 levels of dangling pages [26], 80 million URLs remained in the link graph, requiring 4.3 GB of storage. This subgraph was used for all but the final 2 PageRank iterations, for which the full graph was used. Using a straightforward implementation [16], running PageRank for 25 iterations on one CPU took roughly 5 hours, both for the standard vector as well as for each topical vector. Utilizing 2 dual-processor machines, we can generate all 16 topical vectors in roughly 20 hours.

For generating a small number of topic-specific vectors, the above approach is reasonable in terms of time complexity. However, reducing the running time can be very useful in minimizing the delay between the completion of a new Web crawl and the generation of an updated search index. To improve on the above running times, we use a scheme introduced by Kamvar et al. [21] to accelerate PageRank computations by using successive iterates to approximate and subtract out the nonprincipal eigenvectors from the current iterate. Intuitively, during the iterative computation, the algorithm periodically uses a sequence of intermediate iterates to extrapolate the value of the true rank vector. Using this scheme, called *Quadratic Extrapolation*, the convergence of PageRank can be sped up by 25-300%, depending on the setting of the parameter α.

For generating a larger number of topic-specific vectors, a different approach is required; speeding up the computation of individual rank vectors is insufficient. Jeh and Widom [20] propose a scheme for efficiently computing personalized PageRank vectors by exploiting the overlap in the computation of the different vectors. The intuition behind their scheme can be seen through an example. Consider the simple graph given in Figure 4. If we set the personalization vector $\boldsymbol{p}$ to add a complete set of artificial transitions terminating at A (Graph 1), the induced random walk is very similar to the case where we set $\boldsymbol{p}$ so that each of the children of A, namely B and C, are targets of a complete set of artificial transitions (Graph 2). By exploiting this simple observation, they construct a dynamic programming algorithm to generate a large basis set of personalized PageRank vectors simultaneously, which can then be used to compute arbitrary topic-specific vectors. As the time and space overhead of this latter algorithm is high, which of the above techniques is most suitable depends on the granularity of the topical basis desired.

6.2 Query-Time Processing

For efficient query-time processing, it is desirable to keep most (if not all) of the topic-specific ranking data in main memory. In Section 6.2, give an overview of a scalable keyword-search system to help make clear why this is the case.[17] In Section 6.2, we describe memory-efficient encodings for PageRank vectors that minimize the effect of the lossy encoding on final search rankings.

[17] For further information on large-scale search systems, we refer the reader to [9, 30].

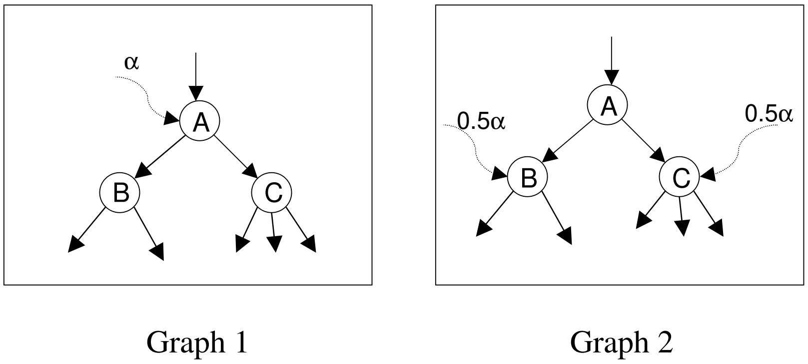

Fig. 4. Graph 1 personalizes on node A, Graph 2 personalizes on the children of A. Random walks on these graphs behave similarly, an observation that Jeh and Widom [20] use to compute a large number of personalized PageRank vectors simultaneously.

Overview of Keyword Search Processing As depicted in Figure 5, a Web search system utilizes an inverted text index $\mathcal{I}$ and a set of auxiliary, numeric ranking vectors $\{\boldsymbol{R_i}\}$. In our case, $\{\boldsymbol{R_i}\}$ includes a set of topic-specific PageRank vectors. For simplicity, consider a system with only the standard PageRank vector $\boldsymbol{R_p}$. The index $\mathcal{I}$ contains information about the occurrences of terms in documents and is used to retrieve the set of document IDs for documents satisfying some query $\mathcal{Q}$. The index $\boldsymbol{R_p}$ is then consulted to retrieve the PageRank for each of these candidate documents. Using the information retrieved from $\mathcal{I}$ and $\boldsymbol{R_p}$, a composite document score is generated for each candidate result, yielding a final ranked listing.

The inverted index $\mathcal{I}$ is constructed offline and provides the mapping $\{t \to f_{dt}\}$, where f_{dt} describes the occurrence of term t in document d. In the simplest case, f_{dt} could be the within-document frequency of t. The number of random accesses to $\mathcal{I}$ needed to retrieve the necessary information for answering a query $\mathcal{Q}$ exactly equals the number of terms in the query, $|\mathcal{Q}|$. Because queries are typically small, consisting of only a few words, it is practical to keep the index $\mathcal{I}$ on-disk and perform $|\mathcal{Q}|$ seeks for answering each query.

The auxiliary index $\boldsymbol{R_p}$ is also constructed offline, and provides the mapping $\{d \to r_d\}$, where r_d is the PageRank of document d. Note that in contrast to $\mathcal{I}$, the index $\boldsymbol{R_p}$ provides *per-document* information. The search system typically must access $\boldsymbol{R_p}$ once for *each* candidate document of the result set, which could potentially be very large. These random accesses would be prohibitively expensive, unless $\boldsymbol{R_p}$ can be kept entirely in main memory. Whereas the query length is the upper bound for the accesses to $\mathcal{I}$, the number of candidate results retrieved from $\mathcal{I}$ is the upper bound for accesses to $\boldsymbol{R_p}$. One way to reduce the number of random accesses required is to store the attribute values of $\boldsymbol{R_p}$ in $\mathcal{I}$ instead; e.g., create an index $\mathcal{I}'$ that provides the mapping $\{t \to \{f_{dt}, r_d\}\}$. However, this requires replicating the value r_d once for each distinct term that appears in d, generally an unacceptable overhead especially if *several* numeric properties (e.g., several topic-specific PageRank scores) are used.

Memory-Efficient Encoding of PageRank Much work has been done on compressing $\mathcal{I}$, although comparatively less attention has been paid to effective ways of compressing auxiliary numeric ranking vectors such as $\boldsymbol{R_p}$. The typical keyword search system has only one such auxiliary ranking vector, $\boldsymbol{R_l}$ — the document lengths needed in computing the query-document cosine similarity [30] — and can be kept in main memory without much difficulty. However, for our richer topic-sensitive PageRank scheme, much more consideration needs to be given to the encodings used for the attribute values.

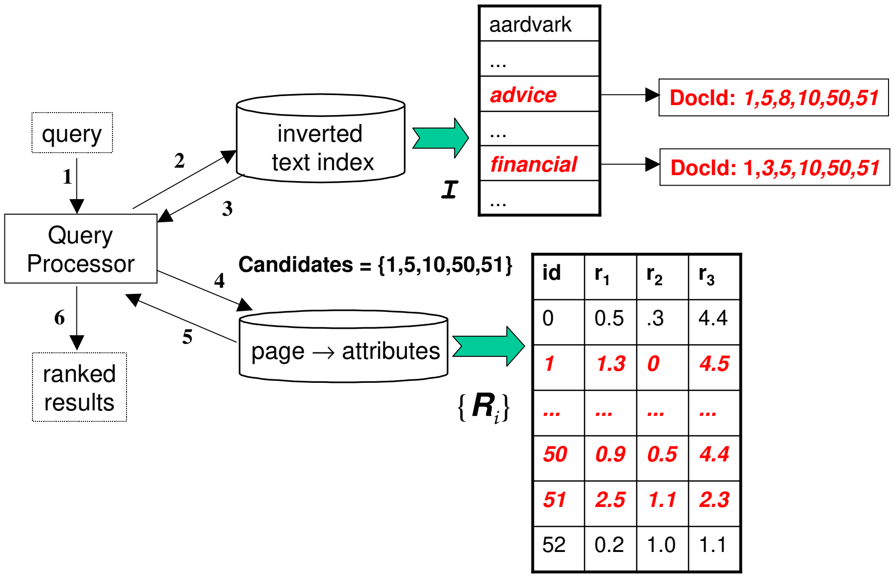

Fig. 5. A simplified illustration of a search engine with a standard inverted text-index and 3 auxiliary numerical attributes for each document. Note that the number of random accesses to $\mathcal{I}$ is typically small, whereas the number of accesses to $\{\boldsymbol{R_i}\}$ is large. Our goal is to minimize the space needed for the data structure $\{\boldsymbol{R_i}\}$.

Note that falling main memory prices do not alleviate the need for efficient encodings; increasingly affordable disk storage is leading to rapidly growing Web-crawl repositories, in turn leading to larger sets of documents that need to be indexed. Utilizing a rich set of per-document numeric ranking attributes for growing crawl repositories and growing numbers of users thus continues to require efficient encoding schemes.

Using a standard IEEE single-precision floating point representation, each final rank vector would require 4 bytes (32 bits) of storage per page — each PageRank vector for our 120 million page repository would require 460 MB. We make use of scalar quantization [15] to store instead approximate PageRank values for each page using fewer bits. Conceptually, we partition the range of PageRank values into intervals (or *cells*). For a page u that has rank r_u, instead of storing the value r_u, we store the codeword for the corresponding cell. The approimate value associated with the cell, called the *reproduction value*, is given by the midpoint of the interval. For a partition consisting of n cells, the cell is represented by a fixed-length codeword of length $\lceil \log_2 n \rceil$ bits. This coding scheme is lossy, since encoding and then decoding the value r_u will yield some reproduction value $q(r_u)$ rather than r_u itself. The key challenge is constructing the partition in a way that minimizes the effect of the lossy encoding on the search rankings. A scheme that partitions the range into cells of equal width, e.g., a uniform quantizer, tends to perform poorly in providing good approximations of the input data values. Note that instead of directly constructing a nonuniform partition, we can apply a nonlinear *compressor* function to the input values, which we then partition into equal-width intervals; there is no loss in expressivity [15]. This type of quantizer (e.g., a nonlinear compressor followed by uniform quantization) is known as a *compander*. A detailed discussion on how to determine the optimal partition (equiv., the optimal compander) in the context of search rankings can be found in [17].

Using a *distortion measure*, we can analyze the performance of a particular partition by looking at how the approximation affects search rankings. For simplicity, we give here the

Table 9. A description of 6 quantization strategies we compare. See [17] for additional details.

Strategy	Description
linear	A uniform quantizer (partition uses cells of equal width).
sqrt	Compressor function $G(x) \propto \sqrt{x}$
log	Compressor function $G(x) \propto \log x$
mse_optimal	Compressor function $G(x) \propto x^{-2.17}$. This optimizes for the mean squared error distortion.
approx_eq_depth	Compressor that approximates equal depth partitions.
eq_depth	Partition assigns an equal number of input values to each cell.

performance of various strategies in compressing the standard PageRank vector; results for the topic-specific vectors are similar.

For measuring the difference between two rankings τ_1 and τ_2, we use the distortion measure $\mathrm{KDist}(\tau_1, \tau_2)$, defined as $1 - \mathrm{KSim}(\tau_1, \tau_2)$, where KSim is the similarity measure defined in Section 4.1. By measuring the average distortion of search result rankings caused by quantizers, we can estimate quantizer performance and choose the quantizer that works best in practice. We next briefly describe some empirical results showing the average distortion for the quantizers given in Table 9. Details on these strategies can be found in [17].

Let $\cos(\mathcal{Q}, d)$ be the cosine similarity between a query $\mathcal{Q}$ and document d. Let r_d be the PageRank of document d. Let $q(r_d)$ be the approximation of r_d generated by a quantizer. We then let τ be the ordered list of the top 100 documents when results to the query $\mathcal{Q}$ are ranked by the composite score $cos(\mathcal{Q}, d) \cdot r_d$, and we let τ_q be the ordered list of the top 100 documents when query results are ranked by $cos(\mathcal{Q}, d) \cdot q(r_d)$ for some quantizer q. Note that $\tau \neq \tau_q$ because $q(r_d)$ is only an approximation of r_d. We then measure the distortion of q using the average value of $\mathrm{KDist}(\tau, \tau_q)$ over a large number of sample queries. As shown in Figure 6, in this scenario, the **log** compander performs the best for all codelengths in minimizing the mean KDist distortion.

Using 8 bits per value, the 16 topic-specific vectors for the 120 million page Web crawl can be stored in just under 2 GB of main memory; query-processing (including the disk-based text-index retrieval) takes roughly 1 second per query on our dataset, using the hardware described in Section 6.1.

7 Future Work

We are currently exploring several ways of improving our topic-sensitive PageRank approach. As discussed previously, discovering sources of search context is a ripe area of research. Another area of investigation is the development of the best set of basis topics. For instance it may be worthwhile to use a finer-grained set of topics, perhaps using the second or third level of directory hierarchies, rather than simply the top level. However, a fine-grained set of topics leads to additional efficiency considerations, as the cost of the naive approach to computing these topic-sensitive vectors is linear in the number of basis topics.

We are also currently investigating a different approach to creating the personalization vector $\boldsymbol{p}$ used to create the topic-specific rank vectors. This approach has the potential of being more resistant to adversarial ODP editors. Currently, as described in Section 3.1, we set the damping vector $\boldsymbol{p}$ for topic c_j to $\boldsymbol{v_j}$, where $\boldsymbol{v_j}$ is defined in Equation 7. In the modified approach, we instead first train a classifier for the basis set of topics using the ODP data as our training set, and then assign to *all* pages on the Web a distribution of topic weights. Let

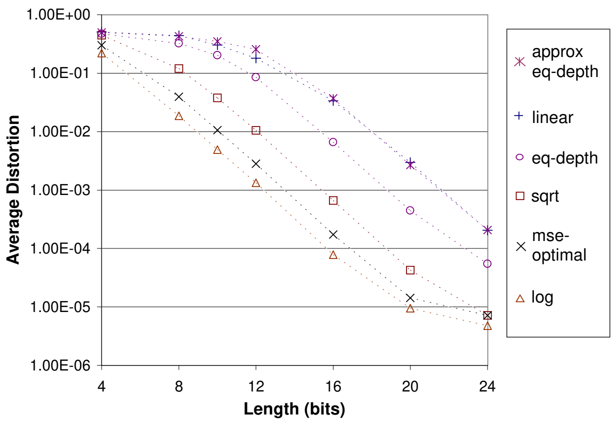

Fig. 6. The average distortion for the various strategies when using the KDist distortion measure for 86 test queries. The rankings compared were the top 100 results ranked by the scores $cos_{\mathcal{Q}d} \cdot r_d$ and $cos_{\mathcal{Q}d} \cdot q(r_d)$, resp.

this topic weight of a page u for category c_j be w_{uj}. Then we replace Equation 7 with

$$\forall_{i \in Web}[v_{ji} = \frac{w_{ij}}{\sum_k w_{kj}}] \tag{12}$$

In this way, we hope to ensure that the PageRank vectors generated are not overly sensitive to particular choices made by individual ODP editors.

8 Acknowledgments

I would like to thank Professor Jeff Ullman for invaluable comments and feedback. I would like to thank Glen Jeh and Professor Jennifer Widom for several useful discussions. I would also like to thank Aristides Gionis for his feedback. Finally, I would like to thank the anonymous reviewers for their insightful comments.

This work was done with the support of NSF Grant IIS-0085896 and an NSF Graduate Research Fellowship.

References

1. The Google Search Engine: Commercial search engine founded by the originators of PageRank. http://www.google.com/.
2. The Open Directory Project: Web directory for over 2.5 million URLs. http://www.dmoz.org/.
3. 'More Evil Than Dr. Evil?' http://searchenginewatch.com/sereport/99/11-google.html.
4. Krishna Bharat and Monika R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of the ACM-SIGIR*, 1998.
5. Krishna Bharat and George A. Mihaila. When experts agree: Using non-affiliated experts to rank popular topics. In *Proceedings of the Tenth International World Wide Web Conference*, 2001.
6. Sergey Brin, Rajeev Motwani, Larry Page, and Terry Winograd. What can you do with a web in your pocket. In *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 1998.
7. Sergey Brin and Larry Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International World Wide Web Conference*, 1998.

8. S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, and S. Rajagopalan. Automatic resource compilation by analyzing hyperlink structure and associated text. In *Proceedings of the Seventh International World Wide Web Conference*, 1998.
9. Soumen Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kaufmann Publishers, San Francisco, CA, 2002.
10. Soumen Chakrabarti, Mukul M. Joshi, Kunal Punera, and David M. Pennock. The structure of broad topics on the web. In *Proceedings of the Eleventh International World Wide Web Conference*, 2002.
11. Michelangelo Diligenti, Marco Gori, and Marco Maggini. Web page scoring systems for horizontal and vertical search. In *Proceedings of the Eleventh International World Wide Web Conference*, May 2002.
12. Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the Tenth International World Wide Web Conference*, 2001.
13. Ronald Fagin, Ravi Kumar, and D. Sivakumar. Comparing top k lists. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2003.
14. Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: the concept revisited. In *Proceedings of the Tenth International World Wide Web Conference*, 2001.
15. Robert M. Gray and David L. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, 44(6), October 1998.
16. Taher H. Haveliwala. Efficient computation of PageRank. *Stanford University Technical Report*, 1999.
17. Taher H. Haveliwala. Efficient encodings for document ranking vectors. *Stanford University Technical Report*, November 2002.
18. Taher H. Haveliwala. Topic-sensitive PageRank. In *Proceedings of the Eleventh International World Wide Web Conference*, May 2002.
19. J. Hirai, S. Raghavan, H. Garcia-Molina, and A. Paepcke. Webbase: A repository of web pages. In *Proceedings of the Ninth International World Wide Web Conference*, 2000.
20. Glen Jeh and Jennifer Widom. Scaling personalized web search. In *Proceedings of the Twelfth International World Wide Web Conference*, May 2003.
21. Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Extrapolation methods for accelerating PageRank computations. In *Proceedings of the Twelfth International World Wide Web Conference*, May 2003.
22. Jon Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 1998.
23. Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.
24. Tom Mitchell. *Machine Learning*, chapter 6, pages 177–184. McGraw-Hill, Boston, 1997.
25. Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, United Kingdom, 1995.
26. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. *Stanford Digital Libraries Working Paper*, 1998.
27. David Pennock, Gary Flake, Steve Lawrence, Eric Glover, and C. Lee Giles. Winner's don't take all: Characterizing the competition for links on the web. In *Proceedings of the National Academy of Sciences*, 2002.
28. Davood Rafiei and Alberto O. Mendelzon. What is this page known for? Computing web page reputations. In *Proceedings of the Ninth International World Wide Web Conference*, 2000.
29. Matthew Richardson and Pedro Domingos. *The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank*, volume 14. MIT Press, Cambridge, MA, 2002.
30. Ian H. Witten, Alistair Moffat, and Timothy C. Bell. *Managing Gigabytes*. Morgan Kaufmann Publishers, San Francisco, 1999.

A Convex Combinations of PageRank Vectors

In this section we derive the interpretation of convex combinations of PageRank vectors.[18] Consider a set of rank vectors $\{\boldsymbol{PR}(\alpha, \boldsymbol{v_j})\}$ for some fixed α.[19] For brevity let $\boldsymbol{r_j} = \boldsymbol{PR}(\alpha, \boldsymbol{v_j})$. Furthermore let $\boldsymbol{r'} = \sum_j [w_j \boldsymbol{r_j}]$, and $\boldsymbol{v'} = \sum_j [w_j \boldsymbol{v_j}]$. We claim that $\boldsymbol{r'} = \boldsymbol{PR}(\alpha, \boldsymbol{v'})$. In other words, $\boldsymbol{r'}$ is itself a PageRank vector, where the personalization vector $\boldsymbol{p}$ is set to $\boldsymbol{v'}$. The proof follows.

Because each $\boldsymbol{r_j}$ satisfies Equation 6 (with $\boldsymbol{p} = \boldsymbol{v_j}$), we have that

$$\boldsymbol{r'} \equiv \sum_j [w_j \boldsymbol{r_j}] \tag{13}$$

$$= \sum_j [w_j((1-\alpha)(M+D)\boldsymbol{r_j} + \alpha \boldsymbol{v_j})] \tag{14}$$

$$= \sum_j [(1-\alpha)w_j(M+D)\boldsymbol{r_j}] + \sum_j [\alpha w_j \boldsymbol{v_j}] \tag{15}$$

$$= (1-\alpha)(M+D)\sum_j [w_j \boldsymbol{r_j}] + \alpha \sum_j [w_j \boldsymbol{v_j}] \tag{16}$$

$$= (1-\alpha)(M+D)\boldsymbol{r'} + \alpha \boldsymbol{v'} \tag{17}$$

Thus $\boldsymbol{r'}$ satisfies Equation 6 for the personalization vector $\boldsymbol{p} = \boldsymbol{v'}$, completing our proof.

[18] The proof that follows is based on discussions with Glen Jeh (see the Linearity Theorem of [20]).

[19] See the end of Section 2 for the description of our notation.