Quality-Biased Ranking of Web Documents

Michael Bendersky
Dept. of Computer Science
University of Massachusetts
Amherst, MA
bemike@cs.umass.edu

W. Bruce Croft
Dept. of Computer Science
University of Massachusetts
Amherst, MA
croft@cs.umass.edu

Yanlei Diao
Dept. of Computer Science
University of Massachusetts
Amherst, MA
yanlei@cs.umass.edu

ABSTRACT

Many existing retrieval approaches do not take into account the content quality of the retrieved documents, although link-based measures such as PageRank are commonly used as a form of document prior. In this paper, we present the quality-biased ranking method that promotes documents containing high-quality content, and penalizes low-quality documents. The quality of the document content can be determined by its readability, layout and ease-of-navigation, among other factors. Accordingly, instead of using a single estimate for document quality, we consider multiple content-based features that are directly integrated into a state-of-the-art retrieval method. These content-based features are easy to compute, store and retrieve, even for large web collections. We use several query sets and web collections to empirically evaluate the performance of our quality-biased retrieval method. In each case, our method consistently improves by a large margin the retrieval performance of text-based and link-based retrieval methods that do not take into account the quality of the document content.

Categories and Subject Descriptors

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

General Terms

Algorithms, Experimentation

Keywords

Quality-biased ranking, document quality

1. INTRODUCTION

Existing document retrieval models usually make the assumption that the quality of all documents in the corpus is equal. This assumption is reasonable in newswire corpora that have been used in TREC evaluation by the information retrieval community. These corpora are relatively small (typically less than a million documents) and are usually homogeneous, as all documents come from the same source (e.g., a news agency like Associated Press).

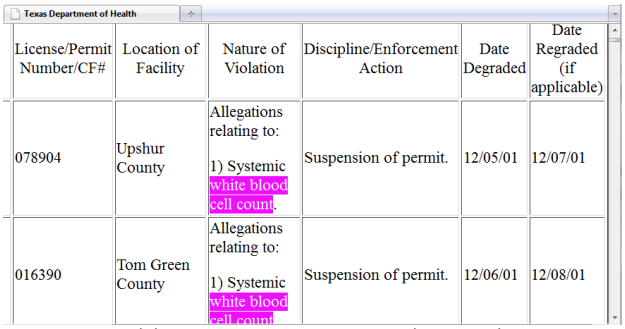
The equal quality assumption does not, however, hold for large (hundreds of millions or, in some cases, billions of documents) web corpora, which have become the focus of information retrieval research in the last few years [9]. These corpora are heterogeneous, since web documents come from many sources that vary significantly in terms of their authority, credibility, goals, and publishing standards. Therefore, there are very large variations in the quality of web pages contained in these corpora. As any web user knows, not all web pages are equal, and they differ by the quality and the type of the information they provide to their readers, as well as by the way they present this information.

Most published current research on document quality in web search focuses on *link analysis*. Graph algorithms, including PageRank [6], HITS [15] and SALSA [26] among others, have been used to estimate document quality by examining their neighborhood in the link graph. While highly successful, these algorithms do not explicitly take into account the actual *content* of the document, including its layout and presentation. Graph algorithms rely solely on the "votes" from the neighbors of the document in the link graph to determine the quality of the document.

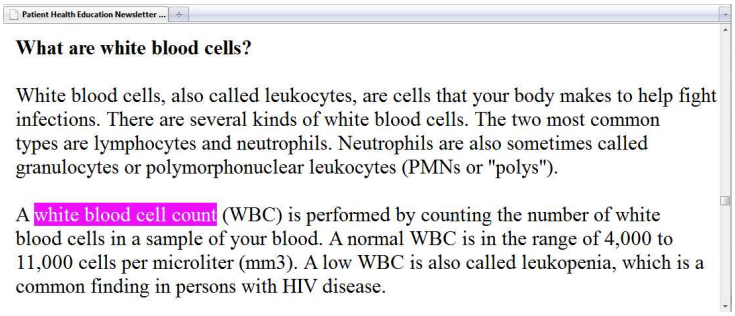
This link analysis approach is similar to the *collaborative filtering* based recommendation systems. Generally, in these systems, the actual content of the recommended item is disregarded, and user recommendations are produced based on neighborhoods in a user-item graph. While collaborative filtering systems have certainly been very successful, recent research shows that hybrid systems that combine collaborative-based and content-based features can significantly improve recommendations for textual items [19]. Analogously, we hypothesize that directly modeling content-based document quality can enhance the performance of existing information retrieval systems that use link-based document quality estimates.

The quality of a web page is determined by a combination of many distinct factors. First, it has to contain original, trustworthy, and up-to-date content of genuine value. It should also provide metadata that accurately describes the content of a page, and contain links that can point people to other related resources. Finally, web page layout should be consistent and follow the principles of user-centric web design, by allowing readers to effortlessly navigate to the relevant information on the page [12]. As document quality is influenced to some degree by all of these factors, the

WSDM'11, February 9–12, 2011, Hong Kong, China.



(a) Non-relevant document (1st rank)



(b) Highly relevant document (3rd rank)

Figure 1: Two documents retrieved by a standard retrieval method [23] in response to the query *low white blood cell count*. (a) Document retrieved at the first rank and marked as *non-relevant*; (b) Document retrieved at the third rank and marked as *highly relevant*.

quality of a page should not be viewed as a dichotomy, but rather as a continuous spectrum.

At one end of this quality spectrum are well known resources for high-quality web documents such as Wikipedia. Wikipedia articles are constantly monitored and updated by editors, have a consistent layout and usually contain links to other related Wikipedia articles and web pages of interest. On the other end of this spectrum are *spam pages* that employ techniques such as content duplication, link schemes, content cloaking and keyword stuffing to artificially inflate their search engine ranking and provide no useful content (or even fraudulent and harmful content) to their readers.

Most of the pages on the web, however, are somewhere in between these two extremes on the quality spectrum. Many web pages do not have the same level of editorial supervision as Wikipedia, and might contain some outdated information, but still provide useful content to their readers. Many of the web pages also do not have a consistent easy-to-follow layout, making it harder to locate relevant portions of the text. However, these pages of lesser quality are still relevant to some user queries. This is especially true for rare and "niche" user information needs that often lack proper coverage by high quality resources such as Wikipedia. Therefore, it is important to explicitly incorporate the information about the quality of the page into the ranking produced by the search engine. However, to the best of our knowledge, there is surprisingly little publicly available research on modeling document content quality in the context of web search.

Figure 1 demonstrates the importance of document quality information. Figure 1 (a) shows a web document retrieved at the first position by a standard retrieval method [23] for a query *low white blood cell count* from GOV2 — a TREC collection containing a crawl of the *.gov* domain. It is easy to see that while this document receives a high textual match score, it provides no relevant information for the query (and is labeled as *Non-Relevant* by the TREC judges). On the other hand, a document shown at Figure 1 (b), which is retrieved at the third position, and has fewer query term matches, actually contains much more relevant information (and is labeled as *Highly Relevant*).

While the documents in Figure 1 have similar PageRank in GOV2, and both contain some useful information, they significantly differ by their readability, layout and content presentation. Taking these factors into account can potentially improve the performance of a given retrieval method.

Based on this insight, we propose a *quality-biased ranking* approach that directly introduces document quality as a part of the ranking function. This quality-biased ranking approach achieves significant improvements over the baselines that do not use any page quality information, or use information solely from link-based quality measures such as PageRank. For instance, for the query described in Figure 1, the highly relevant page (b) is promoted to the first rank by our quality-biased ranking, while the non-relevant page (a) is demoted to the 11th position.

The rest of this paper is organized as follows. First, in Section 2 we formulate the principles of quality-biased ranking, based on the Markov Random Field model for Information Retrieval [23]. Next, in Section 3 we discuss the content-based quality features used by our method. We provide a detailed discussion of our implementation of the feature extraction process in Section 4. Related work is described in Section 5. In Section 6, we evaluate the performance of the quality-biased ranking method using two large web collections. Finally, we summarize our findings in Section 7.

2. QUALITY-BIASED RANKING

In this section, we formulate the principles of *quality-biased* ranking, based on the Markov Random Field model for Information Retrieval (MRF-IR), first proposed by Metzler and Croft [23]. MRF-IR has consistently demonstrated state-of-the-art retrieval effectiveness in a variety of search tasks, and especially for search over large web collections [3, 23]. Several top performing submissions at the Text Retrieval Conference (TREC) in the web search tracks (Terabyte Track 2004-2006 [8, 25], Million Query Track 2007-2008 [1]) have used this model in the last five years. Currently, the MRF-IR model is one of the most effective publicly disclosed text-based retrieval models for web search.

However, to the best of our knowledge, there is no published research on successfully incorporating the notion of document quality into the MRF-IR model. Accordingly, in this section, we discuss the integration of features representing the quality of the document content into this model.

The rest of this section is organized as follows. We discuss the general MRF-IR model in Section 2.1. In Section 2.2 we describe the integration of the document quality features into the MRF-IR model. Finally, we describe a technique for parameter estimation in the resulting model in Section 2.3.

Feature Function	Description
$f_T(q_i, D) = \log\left[\frac{tf_{q_i,D}+\mu\frac{cf_{q_i}}{\|C\|}}{\|D\|+\mu}\right]$	Weight of unigram q_i in document D.
$f_O(q_i, q_{i+1}, D) = \log\left[\frac{tf_{\#1(q_i,q_{i+1}),D}+\mu\frac{cf_{\#1(q_i,q_{i+1})}}{\|C\|}}{\|D\|+\mu}\right]$	Weight of exact phrase "q_i q_{i+1}" in document D.
$f_U(q_i, q_{i+1}, D) = \log\left[\frac{tf_{\#uw8(q_i,q_{i+1}),D}+\mu\frac{cf_{\#uw8(q_i,q_{i+1})}}{\|C\|}}{\|D\|+\mu}\right]$	Weight of unordered window q_i q_{i+1} (size = 8) in document D.
$f_{\mathcal{L}}(D) = \sum_{L\in\mathcal{L}} \lambda_L f_L(D)$	A weighted sum of quality features associated with document D.

Table 1: Summary of feature functions used in a quality-biased sequential dependence model. $tf_{e,D}$ is the number of times e has a match in document D, $cf_{e,D}$ is the number of times concept e matches in the entire collection, $|D|$ is the length of document D, and $|C|$ is the total length of the collection. μ is a weighting function hyperparameter that is set to 2500, following prior work [23, 3].

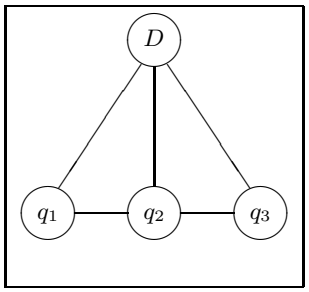


Figure 2: MRF model with a sequential dependence assumption.

2.1 Markov Random Fields for IR

A Markov random field (MRF) is a graphical model in which a joint distribution over a set of random variables is represented using an undirected graph G, where the nodes in the graph represent random variables and the edges define the dependence semantics between the random variables. Metzler and Croft [23] proposed using MRF to model a joint relevance distribution over a query $Q = q_1, \ldots, q_n$ and a document D. Figure 2 shows an example of an MRF for a document and a three-term query. In the depicted model, adjacent query terms are dependent on each other since they share an edge, but non-adjacent query terms (e.g., q_1 and q_3) are independent given D.

In the MRF, given the undirected graph G, the joint distribution over the graph is calculated using non-negative potential functions ψ defined over the set of cliques $C(G)$ in the graph G. That is, for a given document D and a query Q, the joint relevance distribution is expressed as

$$P_{G,\Lambda}(Q, D) = \frac{1}{Z_\Lambda} \prod_{c\in C(G)} \psi(c; \Lambda), \tag{1}$$

where Z_Λ is a normalizing constant, and Λ is a set of free parameters that are used within the potential functions. Commonly, the potential functions take the form

$$\psi(c; \Lambda) = e^{\sum_i \lambda_i f_i(c)}.$$

MRF-IR defines the score of a document D given a query Q as [23]

$$\begin{aligned} score(Q, D) &= \log P_{G,\Lambda}(D|Q) \\ &= \log P_{G,\Lambda}(Q, D) - \log P_{G,\Lambda}(Q) \\ &= \sum_{c\in C(G)} \log\psi(c; \Lambda) - \log Z_\Lambda - \log P_{G,\Lambda}(Q) \\ &\overset{rank}{=} \sum_{c\in C(G)} \log\psi(c; \Lambda). \end{aligned} \tag{2}$$

Therefore, to instantiate the MRF model, one must define a set of cliques $c \in C(G)$ and a set of potential functions $\psi(c; \Lambda)$ over these cliques. There are several possible instantiations, based on the different dependence assumptions between the document and the query terms [23], however in this work we employ the *sequential dependence* instantiation, as it has been shown to provide a good balance between effectiveness and efficiency [23, 3]. The sequential dependence instantiation of the MRF model, depicted in Figure 2, assumes dependence only between the adjacent query terms.

Under the sequential dependence assumption, there are three types of cliques over which the potential functions are defined. First, there are cliques involving a single term node and the document node. The potentials for these cliques are defined as follows:

$$\log\psi(q_i, D; \Lambda) = \lambda_T f_T(q_i, D)$$

Here, $f_T(q_i, D)$ is a feature function defined over the query term q_i and the document D, and λ_T is a free parameter.

The second type of cliques over which we define the potentials, are cliques that contain a bigram (two adjacent query terms nodes) and the document node. The potentials over these cliques are defined as:

$$\log\psi(q_i, q_{i+1}, D; \Lambda) = \lambda_O f_O(q_i, q_{i+1}, D) + \lambda_U f_U(q_i, q_{i+1}, D)$$

where $f_O(q_i, q_{i+1}, D)$ and $f_U(q_i, q_{i+1}, D)$ are feature functions, and λ_O and λ_U are free parameters. These potentials are made up of two distinct components. The first considers ordered (i.e., exact phrase) matches and is denoted by the O subscript. The second, denoted by the U subscript, considers unordered matches.

Finally, the third clique type over which we define the potential functions, is the clique that contains only the document node. In previous work, this clique type was effectively ignored for the purpose of ranking, by setting its potential function to zero [23]. In other words, the query-independent information about the document quality was

Feature	Description
numVisTerms	Number of visible terms on the page (as rendered by a web browser)
numTitleTerms	Number of terms in the page <*title*> field.
avgTermLen	Average length of visible terms on the page.
fracAnchorText	Fraction of anchor text on the page.
fracVisibleText	Fraction of visible text on the page (as rendered by a web browser) [33].
entropy	Entropy of the page content.
fracStops	Stopword/non-stopword ratio.
stopCover	Fraction of terms in the stopword list that appear on the page.
urlDepth	The depth of the URL path (number of backslashes in the URL).
fracTableText	Fraction of table text on the page.

Table 2: Detailed description of the extracted document quality features.

disregarded, which is analogous to the uniform document prior assumption, often made in other probabilistic retrieval approaches [29]. In contrast, in this work, we define the query-independent potential function based on a set of quality-based factors $\mathcal{L}(D)$ associated with the document node D.

$$\log \psi(D;\Lambda) = \sum_{L \in \mathcal{L}(D)} \lambda_L f_L(D).$$

2.2 Ranking with Quality Bias

We are now ready to fully specify the *quality-biased* ranking function by using the feature functions defined in the previous section. Using the three types of potential functions in the sequential dependence model (defined over term-document, bigram-document and document-only cliques) in Equation 2, the query-document score is

$$\begin{aligned} score(Q,D) &= \lambda_T f_T(q_i, D) \\ &+ \lambda_O f_O(q_i, q_{i+1}, D) + \lambda_U f_U(q_i, q_{i+1}, D) \\ &+ \sum_{L \in \mathcal{L}} \lambda_L f_L(D) \end{aligned} \quad (3)$$

Table 1 specifies the set of feature functions used in Equation 3. The functions f_T, f_O and f_U are based on weighting functions, which have been successfully used by researchers in the past [23, 3]. Functions f_L are based on the document quality features, which are summarized in Table 2 and described in detail in Section 3.

2.3 Parameter Estimation

Given the quality-biased ranking function in Equation 3, we now need to estimate the set of free parameters Λ in the ranking formula such that its retrieval performance is optimized. It is clear that Equation 3 takes a form of a linear combination of features $f(\cdot)$ that either depend on the query-document pair or the document itself. Overall, there are 13 features (the number of document quality features in $\mathcal{L}(D)$ plus three query-document based features f_T, f_O, f_U). Therefore, exhaustive search to find the best parameter setting (as was done in the original MRF-IR model [23]) is no longer feasible when the quality bias is introduced.

To address this problem, we employ the coordinate ascent algorithm proposed by Metzler and Croft [22], a simple yet effective learning-to-rank [20] method that directly optimizes the retrieval metric of choice (e.g., nDCG or MAP). This algorithm iteratively optimizes a multivariate objective function (in our case, $score(Q,D)$) by performing a series of one-dimensional line searches, using a training set of queries. It repeatedly cycles through each parameter λ_i, holding all other parameters fixed while optimizing λ_i. This process is performed iteratively over all parameters in Λ until the gain in the target metric is below a certain threshold.

Although we use the coordinate ascent algorithm primarily for its simplicity and efficiency, any other learning to rank approach that estimates the parameters for linear models (such as RankSVM [13] or RankNet [7]) can be used to optimize the quality-biased ranking function in Equation 3.

3. QUALITY FEATURES

In this section, we focus on the set of features $\mathcal{L}$ that was used to assign a document quality score in Equation 3. In order to estimate the quality of a web page from its content, the HTML source of the web page is processed. Multiple features that might correlate with the quality of the page are then extracted from the page source. As previously described, the quality of the page is defined by many factors. Page quality can be related, among other things, to the textual content of the page visible in the browser, page metadata, anchor text of the page and the HTML markup that defines the page layout. Accordingly, we collect multiple quality features for each document, hypothesizing (based on previous findings in web page content analysis [27]) that their combination will be more effective in determining the true quality of the document than each feature on its own.

We use a mix of both novel quality features and quality features used in previous work on content analysis [32, 33], readability [14] and content-based spam detection [27] to represent the quality of each document. Table 2 provides a summary of the extracted quality features. A more detailed description of quality features is given below.

- `numVisTerms` Number of visible terms on the page. Visible terms are terms that are rendered by a web browser; these are the terms that are not a part of HTML markup, javascript or comments on the page. The `numVisTerms` feature provides an estimate of the length of the page content, as is seen by the page reader.

- `numTitleTerms` Number of terms in the page <*title*> field. This feature provides an estimate of the descriptiveness of the metadata on the page.

- `avgTermLen` Average length (number of characters) of visible terms on the page. A simple estimate of the page readability [14].

- `fracAnchorText` Fraction of anchor text on the page. An estimate of how much information a page provides about other potentially relevant pages. However, an excessive amount of anchor text on a page may indicate that it contains no useful content of its own [27].

- `fracVisText` Fraction of visible text on the page (as rendered by a web browser), compared to the full source of

the page. This feature is also known as the *information-to-noise ratio* of the page, and has been used as an estimate of the page quality in previous research [33, 24, 32].

- `entropy` Entropy of the page content. The entropy of document D is computed over the individual document terms as

$$-\sum_{w \in D} p_D(w) \log p_D(w),$$

where the probability of word w_i is computed using a maximum likelihood estimate $p_D(w_i) = \frac{tf_{w_i,D}}{\sum_{w_j \in D} tf_{w_j,D}}$. We use entropy as an estimate of the cohesiveness of the page — pages with smaller entropy will tend to be more cohesive and more focused on a single topic.

- `fracStops` Stopword/non-stopword ratio of the page — percentage of the terms on the page that are in the *stopword list*. The stopword list is constructed using the top-100 most frequent alphabetic unigrams in a large web corpus [5].

- `stopCover` Fraction of terms in the stopword list that appear on the page. Estimate of how well the text of the page follows term distribution in standard texts. The features `fracStops` and `stopCover` can be viewed as efficient approximations of the divergence between the document and the collection language models, which was used as a document quality predictor in the previous work [32, 24].

- `urlDepth` The depth of the URL path (number of backslashes in the URL). This feature is an estimate of the level of the page in the domain hierarchy, and was previously used in entry page search [16].

- `fracTableText` Fraction of table text on the page. Approximation of the layout of the page content. In addition, documents that contain a large fraction of table text, are unlikely to contain useful and readable content.

Overall, there were two main requirements for the feature to be included in the list of features above. First, the feature had to correlate with one of the factors that determine document quality. These factors include, among others, content clarity and readability (`avgTermLen, numVisTerms, fracVisText, fracStops`), provision of useful links (`fracAnchorText`) and ease of navigation (`numTitleTerms, fracTableText, urlDepth`).

Second, the feature had to be efficient to compute and amenable to parallelization when computed on a large web corpus. Accordingly, the list of features above was restricted to aggregate features that can be extracted in linear (in the number of terms) time over a single document. In the next section, we describe the process of feature extraction, which is based on these restrictions, and can be easily parallelized to handle millions of documents.

4. FEATURE EXTRACTION PROCESS

Since in this paper we deal with large-scale web collections, it was important to ensure that our implementation of the feature extraction process was as efficient as possible for a single web page, and that it could be easily scaled to handle millions of pages. Figure 3 outlines our implementation of the feature extraction process for a single document. The extraction process consists of two consecutive phases: *tokenization phase* and *generation phase*, which are described next.

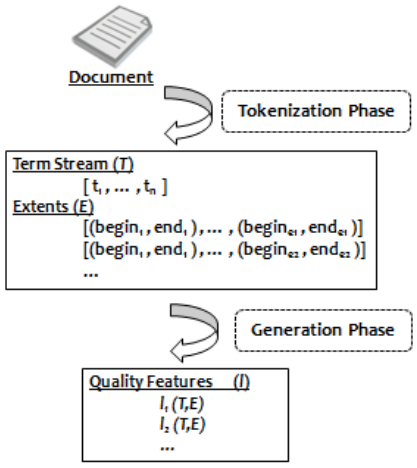


Figure 3: Feature extraction process.

```
<head>
<title>The Quick Fox Tale </title>
</head>
<body>
The quick <a href="wiki/Fox">brown fox</a>
jumps over the lazy <a href="wiki/Dog">dog</a>.
</body>
```

Stream	Stream Content
T	[the, quick, fox, tale, the, quick, brown, fox, jumps, over, the, lazy, dog]
E_{title}	[(0,3)]
E_a	[(6,7), (12,12)]

Figure 4: A mock-up of a web page, and the corresponding term and extent streams.

4.1 Tokenization Phase

Each document in the corpus is tokenized into two streams: stream of terms (T) and stream of extents (E). The stream of terms is a list of normalized terms that are visible to the user when the page is rendered by the browser. Namely, term stream T excludes all the text contained in HTML tags, as well as style definitions, javascript and comments.

The generalized stream of extents, E, is a list containing several separate streams E_f, one stream for each field f in the document. For instance, we keep separate extent streams for text in the <*title*>, <*a*> and <*td*> fields of the document (title text, anchor text and table text, respectively). Additional extents can be easily added, if more features need to be computed. Each of the extent streams E_f contains a list of pairs $(begin, end)$. Each such pair is a pointer to the position of the terms from field f in the term stream T.

A simple example of a web page in Figure 4 illustrates the use of term and extent streams. For instance, the extent stream E_{title} for the <*title*> field points to the terms *[the, quick, fox, tale]* in the term stream T, while the extent stream E_a has two pointers to the anchor text in T (*[brown, fox]* and *[dog]*).

4.2 Generation Phase

During the generation phase, streams T and E are consumed by an aggregation function that computes quality fea-

tures using a single pass over these streams. For instance, to compute a single feature `avgTermLen` we iterate over the term stream T and sum up the number of characters in each term. Finally, to get the feature, we divide this sum by $|T|$.

In practice, during the feature generation step we compute all the aggregates for all the extents simultaneously, by iterating over all the streams at once. Thus, almost all of our features can be computed after a single pass over the streams. The only exception is the `entropy` feature. It requires a single pass over the term stream to construct a language model (normalized count of each unique term) and a second pass over the language model to compute the entropy. Note, however, that the second pass is much shorter than the first pass, since it is done over the *unique* terms counts, not the entire stream. Overall, using our method we are able to process ~ 300 documents/second per node.

4.3 Parallelization

The two-phase extraction algorithm described above can be easily parallelized for large corpora. Assuming that the documents are grouped in batch files (such as WARC files used to store the ClueWeb corpus [9] or TREC-WEB files used to store the GOV2 corpus [8]) we first dispatch a single batch to all available nodes. Then, we monitor the job status of the batches on each of these nodes. Once one of the jobs is complete, we send the next available batch to the free node. This process is repeated until all batches are processed.

5. RELATED WORK

Associating a document with an estimate of probability of being relevant to any query (also known as *document prior*) is an important problem in many IR tasks, and has been extensively studied by researchers in the past. In particular, link-based priors such as PageRank [6], HITS [15] and SALSA [26] are often used in web search. Incorporating these priors into the scoring function has been demonstrated to improve retrieval on large web collections [11, 16, 28]. Click-based priors, which leverage the information about how frequently users visit a certain page to estimate its prior probability of relevance [21, 30], were also found to benefit web search.

While successful, the link-based and click-based priors in web search do not explicitly take into account the quality of the textual content of the document. In contrast, some work on small and homogeneous collections showed that using features based on the document content such as document length [31, 4] and information-to-noise ratio [33] can lead to improvements in retrieval performance.

However, improvements achieved by using content-based quality features in retrieval over large heterogeneous web collections were not shown to be as consistent. In some cases, information-to-noise ratio combined with collection-document distance improved precision at the top ranks when combined with a bag-of-words retrieval model [32]; in other cases, these features were reported to hurt the retrieval performance when combined with a more complex retrieval model, which took term proximities into account [25]. Compared to these inconsistent results, our quality-biased ranking unequivocally demonstrates that integrating content quality based features into the scoring function significantly improves retrieval performance, even when a state-of-the-art retrieval method — which uses exact phrases, term proximities, and link-based priors — is used as a baseline.

Another area of research where examining the quality of the content of web pages was found to be beneficial is a *spam detection* task [27, 10]. Spam detection is a crucial component in web search engines, and can have a significant impact on retrieval performance in large web collections [10, 18]. The spam detection task requires training data in the form of pages labeled as "spam" by human annotators [10, 18, 26]. Thus, most existing retrieval methods that employ spam detection operate in two separate stages: (i) retrieval stage and (ii) spam-filtering stage. [10, 18].

In contrast, our quality-biased ranking approach is able to improve the retrieval performance even *after* an explicit spam filtering stage. Our method learns (using relevance data) a combination of document quality features that severely penalizes low-quality documents that were not classified as spam. In addition, since our approach does not depend on spam labels, it can also be applied to specialized web collections, which do not contain spam, but do contain documents of differing quality.

6. EVALUATION

6.1 Corpora and Query Sets

To evaluate the performance of our quality-biased ranking method, we used two standard web collections developed by TREC[1] (Text REtrieval Conference). The first collection, GOV2, is a crawl of *.gov* domain from 2004. It contains $\sim$ 25 million documents. The second collection, ClueWeb (Category B), is a part of a large recent crawl of the entire web [9] and contains $\sim$ 50 million English web documents. Both GOV2 and ClueWeb collections have a set of queries as well as documents judged for relevance for these queries associated with them. Each query set is named by the last year in which it was used by TREC. The following table details the query sets that we used in this work

Query Set	Collection	# Queries	# Judged Docs
TREC-06	GOV2	150	135,352
TREC-07	GOV2	1,778	73,015
TREC-09	ClueWeb	50	13,118

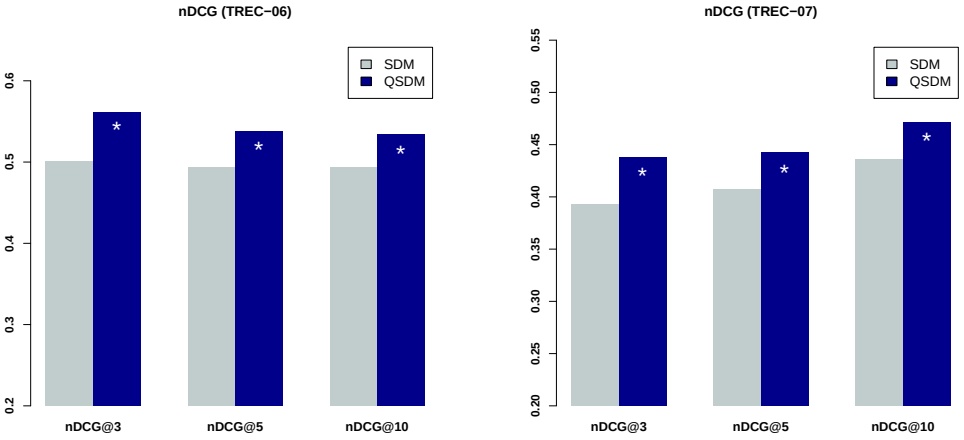
TREC-06 is a set of queries collected over three years of a Terabyte TREC (2004-2006), and was used for evaluation in previous work on the MRF-IR model [23, 3]. The *TREC-07* set of queries is based on the Million Query Track 2007. The aim of this track was to create an evaluation of a retrieval system based on many queries with shallow judgment pools (in contrast to the *TREC-06* query set, which was based on few queries with deep judgment pools). Finally, the *TREC-09* set of queries is a set of queries used in the latest 2009 Web Track to evaluate the performance of the participating retrieval systems on ClueWeb, the largest publicly available web crawl.

6.2 Experimental Setup

For the purposes of evaluation we use the corpora and the query sets described in Section 6.1. The corpora are indexed using an open-source search engine Indri[2]. During indexing, the documents are stemmed using Porter stemmer. Queries are stopped using a short list of 35 common stopwords.

[1] `http://trec.nist.gov/`

[2] `http://www.lemurproject.org/indri/`



	prec@10	MRR	MAP
SDM	59.46	77.74	32.71
QSDM	**63.36***	**84.01***	**34.34***

(a)

	prec@10	MRR	MAP
SDM	37.06	55.51	45.90
QSDM	**39.81***	**59.18***	**48.14***

(b)

Figure 5: Retrieval evaluation using the GOV2 collection: (a) *TREC-06* query set and (b) *TREC-07* query set. * denotes statistically significant difference with SDM (two-sided Wilcoxon sign test, $\alpha < 0.05$).

As a competitive baseline method we use the sequential dependence model [23], which is equivalent to setting the weights of all the parameters λ_L in Equation 3 to zero. We denote this baseline `SDM`. The `SDM` retrieval model is implemented using the structured Indri query language, which natively supports term proximities.

Our evaluation of the quality-biased ranking is performed as follows:

1. An initial candidate set of top-1000 documents for each query Q is retrieved using the `SDM` retrieval model [3].
2. For each document in the candidate set, the set of quality features $L(D)$ is retrieved from the quality features database (see Section 4).
3. The initial list of candidate documents is re-ranked using Equation 3.

We refer to this evaluation process as `QSDM` and compare its performance to `SDM` — the method that is used to create the initial candidate list. Parameters λ_T, λ_O, and λ_U in Equation 3 are set to 0.85, 0.1 and 0.05, respectively, as this setting was found optimal in our experiments as well as in previous work [23]. Parameters λ_L in Equation 3 are estimated using the coordinate ascent method described in Section 2.3, with the target metric being the normalized discounted cumulative gain of the entire ranked list. The evaluation is done using 10-folds cross-validation to avoid overfitting.

The performance of `QSDM` is evaluated using four standard IR metrics: (i) normalized discounted cumulative gain (nDCG) at positions 1-10; (ii) precision at top 10 retrieved documents (prec@10); (iii) mean reciprocal rank of the first relevant document (MRR); and (iv) mean average precision at all ranks for all the queries (MAP).

[3]For the *TREC-07* query set, which has very shallow judgment pools, the candidate set consists, instead, of all the documents with available relevance judgments.

6.3 Quality-Biased Ranking Performance

In this section, we analyze the performance of the quality-biased ranking using the two collections and the three query sets described in Section 6.1. First, we evaluate the retrieval performance of our method on a smaller, specialized web collection, GOV2. Second, we perform the evaluation on ClueWeb, which is a large general-purpose web collection.

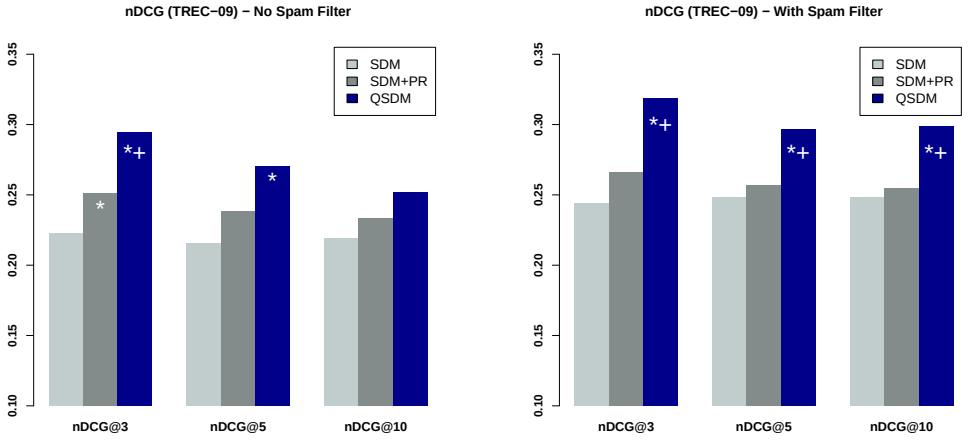
6.3.1 Performance on the GOV2 collection

GOV2 is a relatively small and homogeneous collection by web standards. It contains only documents from the *.gov* domain, which is restricted to use only by the government entities in the United States. Therefore, it is not expected to contain a lot of spam, and spam-filtering techniques such as those described by Lin et al. [18] and Cormack et al. [10] are not expected to produce significant relevance gains. Previous work on this collection [32, 25] has shown only very limited improvements — mostly in precision at top ranks — in retrieval performance when either link-based or content-based quality features were used.

In contrast to this previous work, `QSDM` achieves significant improvements in all retrieval metrics, when compared to the `SDM` baseline[4]. Figure 5 compares the effectiveness of these two methods on the two query sets based on the GOV2 corpus (*TREC-06* and *TREC-07*).

Even in this specialized corpus that does not contain spam and other extremely low quality documents, the quality-biased ranking significantly improves the retrieval performance. These improvements are very visible at the top ranks (the improvements in MRR are as high as 8% and 7% for *TREC-06* and *TREC-07* respectively), and also significant across the entire ranked list (on both query sets, close to 5% improvements in MAP are achieved). The improvements are

[4]It is important to note that the `SDM` method is among the most effective retrieval methods used with the GOV2 collection [1, 24, 25].



	prec@10	MRR	MAP
SDM	28.40	52.79	17.82
PR+SDM	30.00	56.58	17.87
QSDM	**32.40***	**59.03***	**19.71**$^{*}_{+}$

(a) No Spam Filter

	prec@10	MRR	MAP
SDM	34.00	55.17	16.05
PR+SDM	35.20	57.94	15.76
QSDM	**38.40**	**63.67***	**17.05**

(b) With Spam Filter

Figure 6: Retrieval evaluation for the ClueWeb corpus with and without spam filtering. * denotes a statistically significant difference with SDM. $^{+}$ denotes a statistically significant difference with PR+SDM (two-sided Wilcoxon sign test, $\alpha < 0.05$).

also consistent at different ranks, as the nDCG plots in Figure 5 show. For instance, there is around 9% improvement in nDCG@5 for both query sets.

The results in Figure 5 show that by using a learning-to-rank approach, which seeks to directly optimize some retrieval metric, we are able to find a weighted combination of document quality features that significantly improves the performance of SDM on GOV2 — a collection for which it is already considered to be a state-of-the-art retrieval method.

6.3.2 *Performance on the ClueWeb collection*

ClueWeb is a large general-purpose web collection, and as such it contains many low-quality and spam pages. The proliferation of spam on ClueWeb has severely hindered the performance of existing retrieval algorithms, and recently researchers have shown that filtering out spam pages can significantly improve their performance [18, 10].

Due to these findings, we use the spam data for ClueWeb provided by Cormack et al. [10] to filter out pages that are likely to be spam from the initial candidate list retrieved by SDM. Following Cormack et al. [10] we use the *50% filter* (50% of the documents with the highest spam scores are removed), which optimizes the precision at high ranks without a severe impact on mean average precision for the ClueWeb (Category B) collection.

Since ClueWeb is a large web collection, which contains potentially useful link data, we also enhance the standard SDM method with a PageRank [6] prior, which is computed using all the 500 million English documents in ClueWeb. This method, denoted SDM+PR, is conceptually similar to QSDM, but uses only a single feature (PageRank) to estimate document quality[5]. On the other hand, the QSDM combines all the content-based quality features of a document (see Table 2) with its PageRank to produce the final ranking.

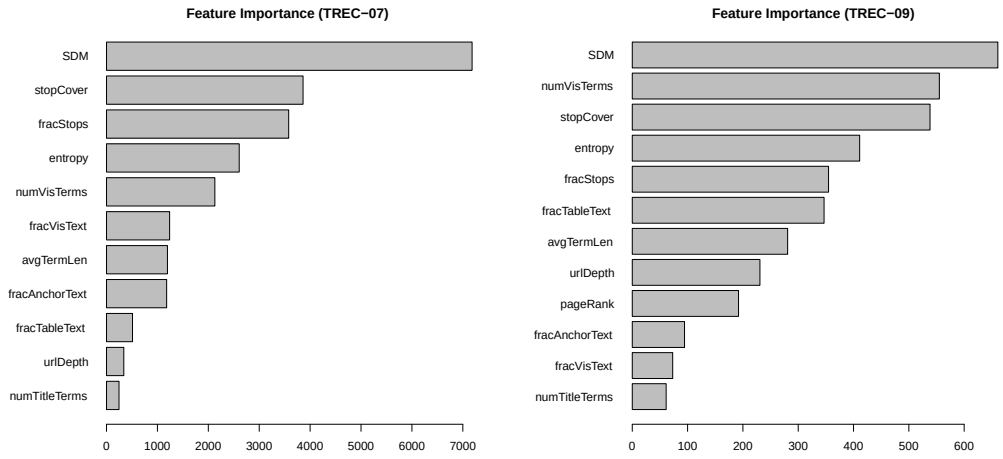
Figure 6 compares the effectiveness of the methods SDM, SDM+PR and QSDM on a subset of ClueWeb used in TREC 2009 (Category B). Figure 6 (a) reports the performance of these three methods on the unfiltered candidate set of retrieved documents, while Figure 6 (b) reports their performance when a 50% spam filter is applied.

The first thing to note in Figure 6 is that the spam filtering results in a 20% increase in precision at 10, while incurring only a 10% loss in MAP. In addition, applying QSDM to the filtered candidate set compensates for much of this loss, by improving the performance for up to 6% over the SDM baseline (Figure 6 (b)).

Our quality-biased ranking QSDM outperforms the SDM on all the retrieval metrics (in most cases to a statistically significant degree) for both the unfiltered and the filtered candidate sets. It achieves the highest performance overall both in terms of MAP (11% improvement over SDM on the unfiltered set) and early precision (15% improvement over SDM in MRR on the filtered set). These results unambiguously confirm our initial hypothesis (see Section 1) that modeling the finer-grained document quality aspects beyond the spam dichotomy may be beneficial for information retrieval in general, and specifically in the context of web search.

It is interesting to note that the QSDM ranking is especially beneficial for improving the ranking of the highly relevant retrieved documents. In the TREC evaluation of the web corpora, documents are judged as *non-relevant*, *relevant* or *highly-relevant*. Binary retrieval metrics such as precision at 10 collapse the *relevant* and the *highly-relevant* categories, while the nDCG metric differentiates between them. We can

[5]A similar SDM+PR combination was applied to the GOV2 collection as well. However in the case of GOV2 collection, no significant improvements were observed over SDM when SDM+PR was used, and hence the results are omitted.



(a) (b)

Figure 7: Quality feature importance for query sets (a) *TREC-07* and (b) *TREC-09*.

infer the positive effect of QSDM on the ranking of the highly relevant documents by observing the nDCG metric in the barplots in Figure 6. For instance, there is a 20% improvement in nDCG@10 in Figure 6 (b) over SDM, compared to the 12% improvement in precision at 10.

Finally, we compare the performance of QSDM to that of the SDM+PR method. In general, QSDM is more effective than SDM+PR in all retrieval metrics. The performance gap (in terms of nDCG) between QSDM and SDM+PR is around 15% at ranks 1-10. In addition, unlike SDM+PR, which improves the early precision but has no significant positive effect on MAP, QSDM is beneficial for all retrieval metrics. This demonstrates the importance of taking into account content-based features, in addition to the link-based ones, when determining document quality.

6.4 Further Analysis

6.4.1 Feature Importance

In this section, we investigate the relative importance of the quality features described in Section 3 for retrieval performance. To this end, for each feature in Table 2, we compute the value of the χ^2 statistic with respect to the relevant class (documents judges as *relevant* or *highly-relevant*) in each query set.

Figure 7 shows the feature importance (based on the χ^2 statistic) diagram for the *TREC-07* [6] and the *TREC-09* query sets (with spam filtering applied to the *TREC-09* results). As a reference, these diagrams also show the χ^2 statistic for the query-document score obtained by the SDM.

Although there are differences between the feature importance diagrams for the two query sets (for instance, the features fracTableText and urlDepth are more important for *TREC-09* than for *TREC-07*), the two diagrams in Figure 7 are similar enough to draw some general conclusions. The most important features for both query sets are the stopword-based features (stopCover and fracStops), document length (numVisTerms) and term entropy (entropy).

[6]The feature importance diagram for the *TREC-06* query set is very similar to the *TREC-07* query set, and is, therefore, omitted.

Feature	Source	Mean	Std. Dev.
fracStops	General Web	0.22	0.11
	Wikipedia	0.27	0.07
stopCover	General Web	0.40	0.22
	Wikipedia	0.47	0.19
fracAnchorText	General Web	0.25	0.21
	Wikipedia	0.38	0.15
avgTermLen	General Web	5.09	0.71
	Wikipedia	5.17	0.48

Table 3: Distribution of quality features on a sample of the general web and a Wikipedia sample.

These features, while conceptually simple, serve as reliable surrogates for the quality of the document content.

The presence of stopwords in the text (modeled by the features stopCover and fracStops) is positively correlated with how informative the text is [14, 26], and documents with very few stopwords are unlikely to be relevant. The importance of document length (numVisTerm) for determining the document relevance is in line with previous research on document length priors [4, 16, 31]. Similarly, incorporating document cohesiveness (modeled in this work by the entropy feature) into the retrieval models was found to be beneficial in the past [2, 17].

6.4.2 Quality of Wikipedia Pages

The ClueWeb collection, which was used in our experiments, also contains a snapshot of English Wikipedia. Unlike general web pages, which often contain incomplete pieces of information on a variety of topics, each Wikipedia page is dedicated to a complete encyclopedic article on a particular subject. Hence, we expect the Wikipedia pages to differ significantly in both the quality of the content and their structure from the general web pages.

To test this hypothesis, we randomly sample 100,000 general web pages and 100,000 Wikipedia pages from ClueWeb collection. Table 3 compares the distribution of several quality features on these two samples.

It is clear from Table 3 that the quality features for the Wikipedia pages differ from those for the general web pages:

Wikipedia pages have a higher fraction of stopwords, more anchor text and slightly higher average term length. All of these factors confirm better readability and higher quality of the Wikipedia pages, compared to the general web.

We can also compare how often the evaluated methods retrieve Wikipedia pages, and how high they rank them. For comparison, when using the SDM method with the *TREC-09* query set and applying the spam filtration, the average rank of a retrieved Wikipedia page is 327, and the total number of times a Wikipedia page appears among the top 10 results is 18. When using the QSDM method, these numbers are 226 and 88, respectively. That is, QSDM is 5 times more likely to retrieve a Wikipedia article in the top 10 results than SDM. This demonstrates that while our quality-biased ranking has no explicit preference for the Wikipedia pages, it does recognize their quality based on the content-based features and promotes them accordingly.

7. CONCLUSIONS

In this work, we examined the importance of content-based document quality features for web search. We extended the state-of-the-art sequential dependence retrieval model, SDM, to include document quality features, and formulated a quality-biased ranking method, QSDM, which promotes high-quality documents and penalizes documents that contain low-quality content.

To implement the QSDM method, we extract several content-based features that are associated with various aspects related to document quality such as content readability, provision of useful links and ease-of-navigation. Our feature extraction process is highly efficient and can be easily scaled to handle millions of documents.

We performed a thorough empirical evaluation of the QSDM method on two standard web collections. Our experimental results show that QSDM consistently and significantly improves the retrieval performance of text-based and link-based retrieval methods that do not take into account the quality of the document content. Statistically significant improvements in retrieval performance were attained for both ClueWeb — a general web collection, in which our method was able to improve the retrieval effectiveness and to promote relevant Wikipedia pages even after an application of a standard spam filter — as well as for GOV2 — a specialized corpus, which contained documents of differing quality, but no explicit spam.

8. ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval, in part by ARRA NSF IIS-9014442 and in part by NSF grants IIS-0746939 and IIS-0812347. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor. We would like to thank David Fisher for his valuable feedback.

9. REFERENCES

[1] J. Allan, J. Aslam, B. Carterette, V. Pavlu, and E. Kanoulas. Million Query Track 2008 overview. In *Proc. of TREC*, 2008.

[2] M. Bendersky and O. Kurland. Utilizing passage-based language models for document retrieval. In *Proc. of ECIR*, pages 162–174, 2008.

[3] M. Bendersky, D. Metzler, and W. B. Croft. Learning concept importance using a weighted dependence model. In *Proc. of WSDM*, pages 31–40, 2010.

[4] R. Blanco and A. Barreiro. Probabilistic document length priors for language models. In *Proc. of ECIR*, pages 394–405, 2008.

[5] T. Brants and A. Franz. Web 1T 5-gram Version 1, 2006.

[6] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, April 1998.

[7] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proc. of ICML*, pages 89–96, 2005.

[8] C. L. A. Clarke, N. Craswell, and I. Soboroff. Overview of the TREC 2004 Terabyte Track. In *Proc. of TREC*, 2004.

[9] C. L. A. Clarke, N. Craswell, and I. Soboroff. Overview of the TREC 2009 Web Track. In *Proc. of TREC*, 2009.

[10] G. V. Cormack, M. D. Smucker, and C. L. A. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. *Arxiv.org*, Apr 2010.

[11] N. Craswell, S. Robertson, H. Zaragoza, and M. Taylor. Relevance weighting for query independent evidence. In *Proc. of SIGIR*, pages 416–423, 2005.

[12] M. Ivory and M. Hearst. Improving web site design. *Internet Computing, IEEE*, 6(2):56–63, 2002.

[13] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. of KDD*, pages 133–142, 2002.

[14] T. Kanungo and D. Orr. Predicting the readability of short web summaries. In *Proc. of WSDM*, pages 202–211, 2009.

[15] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999.

[16] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In *Proc. of SIGIR*, pages 27–34, 2002.

[17] O. Kurland and L. Lee. Pagerank without hyperlinks: structural re-ranking using links induced by language models. In *Proc. of SIGIR*, pages 306–313, 2005.

[18] J. Lin, D. Metzler, T. Elsayed, and L. Wang. Of Ivory and Smurfs: Loxodontan MapReduce experiments for web search. In *Proc. of TREC*, 2009.

[19] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *Proc. of IUI*, pages 31–40, 2010.

[20] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3), 2009.

[21] Y. Liu, B. Gao, T. Y. Liu, Y. Zhang, Z. Ma, S. He, and H. Li. BrowseRank: letting web users vote for page importance. In *Proc. of SIGIR*, pages 451–458, 2008.

[22] D. Metzler and W. Bruce Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274, 2007.

[23] D. Metzler and W. B. Croft. A Markov random field model for term dependencies. In *Proc. of SIGIR*, pages 472–479, 2005.

[24] D. Metzler, T. Strohman, and W. B. Croft. Indri at TREC 2005: Terabyte track. In *Proc. of TREC*, 2005.

[25] D. Metzler, T. Strohman, and W. B. Croft. Indri at TREC 2006: Lessons learned from three Terabyte tracks. In *Proc. of TREC*, 2006.

[26] M. A. Najork. Comparing the effectiveness of HITS and SALSA. In *Proc. of CIKM*, pages 157–164, 2007.

[27] A. Ntoulas and M. Manasse. Detecting spam web pages through content analysis. In *Proc. of WWW*, pages 83–92, 2006.

[28] J. Peng and I. Ounis. Combination of document priors in web information retrieval. In *Proc. of ECIR*, pages 732–736, 2007.

[29] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proc. of SIGIR*, pages 275–281, 1998.

[30] M. Richardson, A. Prakash, and E. Brill. Beyond PageRank: machine learning for static ranking. In *Proc. of WWW*, pages 707–715, 2006.

[31] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proc. of SIGIR*, pages 21–29, 1996.

[32] Y. Zhou and W. B. Croft. Document quality models for web ad hoc retrieval. In *Proc. of CIKM*, pages 331–332, 2005.

[33] X. Zhu and S. Gauch. Incorporating quality metrics in centralized/distributed information retrieval on the world wide web. In *Proc. of SIGIR*, pages 288–295, 2000.