Exploring Content and Linkage Structures for Searching Relevant Web Pages

Darren Davis and Eric Jiang

University of San Diego
5998 Alcala Park, San Diego, CA 92110, USA
{ddavis-08,jiang}@sandiego.edu

Abstract. This work addresses the problem of Web searching for pages relevant to a query URL. Based on an approach that uses a deep linkage analysis among vicinity pages, we investigate the Web page content structures and propose two new algorithms that integrate content and linkage analysis for more effective page relationship discovery and relevance ranking. A prototypical Web searching system has recently been implemented and experiments on the system have shown that the new content and linkage based searching methods deliver improved performance and are effective in identifying semantically relevant Web pages.

Keywords: Web mining, hyperlink analysis, information retrieval, singular value decomposition.

1 Introduction

In general, the search problem spans a spectrum of activities ranging from a precise query for a specific subject to a non-specific desire for learning what information is available. Keyword search systems are commonly used to address these needs and have worked well in some cases. However, they are limited by a vague conception of the user's intent as represented by the few words typical of a keyword search [1]. These limitations mean that keyword search may not be a panacea for information retrieval on the Web and different information search methodologies need to be developed. In this work, we investigate approaches of search where the input is a Web page URL, and the search system returns a list of pages that are relevant to the query URL.

The query URL presumably contains useful content, but the user may desire additional pages that elaborate upon the subject matter found in the query page. Such pages could provide additional information or another perspective on the topic. Thus, Web pages that are relevant to the query address the same topic as the original page, in either a broad or a narrow sense [2]. If the category in which the pages are related is too broad, however, the returned pages may not be very useful. In this work, we aim to find pages with the same specific topic. For example, if the input page describes a security vulnerability, a page that addresses the same topic may discuss procedures for removing the vulnerability.

R. Alhajj et al. (Eds.): ADMA 2007, LNAI 4632, pp. 15–22, 2007.

Relevant page search could have several important benefits. A web site URL often provides copious information that can be used to infer the user's information need. Thus, without significantly increasing the demands made on the user, the user can specify much more information about their intended results. Such a mechanism could be invaluable for document-specific needs such as cross-referencing or locating additional information. It could also be useful in a general case where a topic cannot be readily simplified to a few words or a phrase, but a site is available to provide an example of the topic. In addition to direct use in information retrieval, relevant page search can assist other content organization processes such as clustering, topic separation or disambiguation, or a Web page recommendation system based on pages in a user's search history.

The paper is organized as follows. Some related work is presented in Section 2. In Section 3, we discuss our approaches that integrate content and linkage analysis, thereby enhancing available information about page relationships and improving relevant page searching performance. A prototypical content and linkage based search system has been implemented and is described in Section 4. The experiments on the system and the evaluations of the ranking strategies are presented in Section 5. Some concluding remarks are provided in Section 6.

2 Background

Web searching differs from traditional information retrieval because of the Web's massive scale, diversified content and in particular, its unique link structure. The hyperlinks on the Web not only serve as navigational paths between pages, but also define the context in which a Web page appears and reflect the semantic associations of Web pages [7]. The page information embedded in hyperlinks is useful and can be used for ranking relevant Web pages.

Due to a growing interest in hyperlink analysis, several hyperlink-oriented methodologies have been developed and successfully applied to Web-related fields such as search engines, Web mining, clustering and visualization. The hyperlink analysis has also been used in finding Web pages that are relevant to a topic defined by a user-specified keyword set or Web page. One well-known and representative work in this area is the HITS (Hyperlink Induced Topic Search) algorithm [6] that applies an iteration process to identify pages with topic-related links (hubs) and topic-related content (authorities) within a page neighborhood. The HITS algorithm has offered an interesting perspective on page hyperlink analysis. However, it suffers from the problem of *topic drift* when the majority of neighborhood pages are on a different topic from the query. Several HITS extensions have been proposed that address the problem by either adding linkage weights or expanding the page neighborhood ([1], [2]).

More recently, [5] describes a more direct algorithm named LLI (Latent Linkage Information) for finding relevant pages through page similarity. For a query page, it constructs a page neighborhood in two steps. First, it builds one *reference page set* (Pu) by selecting a group of its parent (inlink) pages and the other reference page set (Cu) by a group of its child (outlink) pages. In the second

step, it builds one *candidate page set* (BPS) by adding a group of child pages from each of the Pu pages. Similarly, it builds the other candidate set (FPS) by adding a group of parent pages from each of the Cu pages. Both candidate sets BPS and FPS are presumably rich in relevant pages, and their pages are ranked and returned by the algorithm. The neighborhood page construction is finalized by merging some of the reference pages and their outlink sets.

For page ranking in LLI, two matrices (Pu-BPS, Cu-FPS) are constructed to represent the hyperlink relations among the pages. In both Pu-BPS and Cu-FPS matrices, each column represents a reference page and each row represents a candidate page. The binary matrix entries indicate if there are page links between the corresponding reference and candidate pages. LLI then applies the singular value decomposition (SVD) on the matrices to reveal deeper relationships among the pages in the rank-reduced SVD spaces.

3 New Approaches (MDP and QCS)

In this section, we present two approaches (MDP, QCS) for finding pages relevant to a given query page. They build on the LLI algorithm [5] and utilize both page linkage and content information for accurate page similarity assessment.

To gather Web page content, terms are read from the page title, meta-description and keywords, and body text, but we limit the term extraction to the first 1000 words, as in [1]. Next, common words and word suffixes are removed to improve term matching [8]. Finally, we use the vector space model [9] to represent page content and determine content similarity between pages. Each page has an associated content vector that is represented by the *log(tf)-idf* term weighting scheme [3], and the dot product between two content vectors produces a normalized similarity score for the two pages.

These content scores can be incorporated into LLI in various ways. During the neighborhood page construction, we use content scoring to eliminate candidate pages that have scores below the median score value. This decreases the computational load and potentially reduces the influence of irrelevant pages.

Our other content integration efforts concern page ranking. With content similarity available, dichotomous link/no-link matrix entries are replaced with content similarity scores between the two pages. There are three primary areas where we integrated content information into the page ranking algorithm. First is the representation of the query vector. Not all pages are equal in the number of relevant pages that they bring into the page neighborhood, and their influence should vary accordingly. The SVD techniques explored in LLI address this by assigning higher influences to reference pages that have more links to candidate pages, and we refine this by using content similarity scores. Reference pages deemed more relevant by content scoring should correspond to a larger value in the query vector. Secondly, the nonzero matrix entries are modified by content relevance. By keeping the zero-nonzero status of the entries, we can retain the linkage information between pages while allowing a continuous range of values to incorporate content information. Third, content is also used to influence the

integration between page sets and the overall ranking of the pages. This can be especially useful to reconcile scores from two sets that may differ widely in their scales. We describe two approaches for addressing the first two issues, followed by several variations in page score integration.

In the *multilevel dot product* (MDP) approach, content-vector dot products between two pages determine each nonzero entry. In the query vector representation, the ith entry is the dot product between the query and the ith parent or child page. In the matrices that represent the page sets, a nonzero entry is given by the dot product between the corresponding candidate (BPS or FPS) page and reference (Pu or Cu) page. This approach thus captures content relationship information between all page levels: the query, the first link level, and the candidate pages. However, some dot products could be zero, which would discard the hyperlink information between the two pages. Thus, the dot products, originally in the range $[0, 1]$ inclusive, are rescaled to $[\mu, 1]$, $0 < \mu \leq 1$. We use the empirically determined value $\mu = 0.01$ in our experiments. This allows the content similarity scores to influence the results while preserving the benefits obtained from the linkage structure. After these modifications, the SVD can be applied as before.

The second approach is *query-candidate scoring* (QCS). This method measures content similarity directly between the query page and the candidate pages, thus eliminating the need to store content information for the reference-level pages. This can reduce the memory requirement and accelerate the page ranking process. The query representation is determined as follows: for each entry, there exist pages in the candidate set that exhibit the appropriate linkage relationship with the corresponding reference page. Dot products are computed between the query content vector and all of these linked pages, and the mean score becomes the value of the entry. A nonzero matrix score is set as a weighted average of a content score and a link score. More specifically, a page's content score is the dot product between the query content vector and the associated candidate page, normalized by division with the maximum score in the set. A page's link score is the number of in- or out-links that the page has with pages in the corresponding reference set. This score is also divided by the maximum link score in the set. These normalization procedures ensure that the range of possible scores is filled, allowing score influences to be properly balanced. The final score for a page is given by a linear combination of both content and link scores. We use a content weight of 0.7 and a link weight of 0.3 in our experiments. As with MDP, the algorithm can then proceed with the SVD.

Once the SVD has been computed and the cosine similarity scores between the query vector and the projected candidate pages in SVD spaces are obtained, various options exist to produce the final page ranking. One option, used in [5], is to use the scores for page ranking without further processing. We call this the *direct* method. It should be noted that a list of scores is generated for each of the page sets (BPS and FPS) in an SVD space. Therefore, there is no guarantee that both lists are comparable. This is particularly the case when certain queries create large size differences between the two page sets, producing two SVD spaces

with widely different dimensions. There are several possible ways to mitigate this compatibility problem. One approach is to balance the scores between the two sets. In our experiments with MDP and QCS, we accomplished this by dividing each score in each candidate set by the mean of the top ten scores in the set.

We consider two additional options that allow further and fine-grained integration of the page scores. The first is the *weighted ordinal* method, where pages in each set are sorted by their cosine score and the unreturned pages with the highest score in each set are iteratively compared. During the comparison, each page is assigned a new composite score equal to the weighted sum of cosine, content, and link scores. The content and link scores are obtained in the same way as the QCS matrix entry weighting scheme, but they are not projected through a SVD space. In our experiments we used cosine, content, and link weights of 0.6, 0.3, and 0.1, respectively. This approach preserves within-set ordering but allows changes in between-set ordering. The second score integration option works similarly by computing the composite scores for all pages, but it does not take the orders of original cosine scores into consideration. In this approach, pages are returned according to their composite scores. This allows changes in both between- and within-set ordering but still benefits from the cosine information. It is referred to as the *weighted non-ordinal* method.

The MDP and QCS approaches are summarized as follows:

Algorithm (MDP, QCS)

1. Construct the reference page sets Pu and Cu (MDP, QCS)
2. Extract reference page content (MDP)
3. Construct the candidate page sets BPS and FPS (MDP, QCS)
4. Extract candidate page content (MDP, QCS)
5. Filter candidate pages by content (MDP, QCS)
6. Set the query vector by query-reference dot products (MDP) or by linked-page content scores (QCS)
7. Set the matrix entries of Pu-BPS and Cu-FPS by reference-candidate dot products (MDP) or by combinations of query-candidate dot products and linkage frequency (QCS)
8. Perform SVD on Pu-BPS and Cu-FPS (MDP, QCS)
9. Compute similarity scores between the query and candidate pages in the rank-reduced SVD spaces (MDP, QCS)
10. Balance the scores from the two SVD spaces (MDP, QCS)
11. Rank candidate pages by one of the score integration options (direct, weighted ordinal, weighted non-ordinal) (MDP, QCS)

4 A Prototypical System for Searching Relevant Pages

In order to evaluate and compare the performance of our algorithms, a prototypical relevant page searching system has been implemented. The system takes a query URL and identifies its reference and candidate pages. Next, the system

accepts a list of ranking strategies to be performed along with their corresponding parameters. The ranking strategies share the same set of initial pages, but each ranking trial is presented with the same set of initial pages regardless of its serial position. This independence includes the possible elimination of candidate pages from the page neighborhood, as discussed in Section 3. By making the initial page read common to all ranking approaches but keeping the rankings otherwise independent, we can ensure higher consistency in our evaluation of the ranking algorithms.

The ranking strategies implemented in the system are LLI, MDP, and QCS, along with several variations. LLI is coded as a replicated model in [5] that ranks pages by direct score comparison. Score balancing is applied to MDP and QCS, and one can also choose cosine, content, and link weights for the weighted ordinal and weighted non-ordinal rankings of these two approaches. The user is permitted to choose an arbitrary number of ranking configurations before the program exits. For MDP and QCS, all three page integration strategies (direct, weighted ordinal, weighted non-ordinal) are automatically performed. All page information is available throughout program execution, but no cache of page content is maintained between program runs.

5 Experiments

In this section, we compare and evaluate the performance of LLI and the two approaches, MDP and QCS, which are presented in this work. Note that, given the volatile nature of the Internet, we could not exactly replicate the results of LLI reported in [5]. The MDP and QCS algorithms are configured with the score integration techniques (direct, weighted-ordinal, and weighted-non-ordinal).

For performance comparison, we investigated the top ten results returned by the algorithms for various input URLs. Then, results were subjectively classified into four categories: pages whose content has no relevance to the query, pages whose content is relevant in a broad sense, pages that contain some narrow-topic related information, and pages that are exclusively devoted to the same topic as defined in a narrow sense. Fig. 1 summarizes the top returned pages from LLI and the variations of MDP and QCS for three quite different query pages.

The first query page[1] concerns deprecated thread methods in the Java API. The results in Fig. 1 have indicated an improved performance of MDP and QCS over LLI for the query. A further analysis illustrates a weakness in LLI that was addressed in our approaches. In this case, some pages contain very few in- or out-links, and after merging there may only be one or two reference pages left. The resulting matrix has only one or two columns, which inflates the scoring of all the corresponding candidate pages. This makes it difficult to integrate the rankings of the candidate pages from the different sets. Our methods address this issue by allowing a continuous range of scores within the linkage matrices, and by providing post-SVD adjustments to the ranking scores. Thus, while the benefits from the SVD-based rankings are preserved, further content and

[1] http://java.sun.com/j2se/1.3/docs/guide/misc/threadPrimitiveDeprecation.html

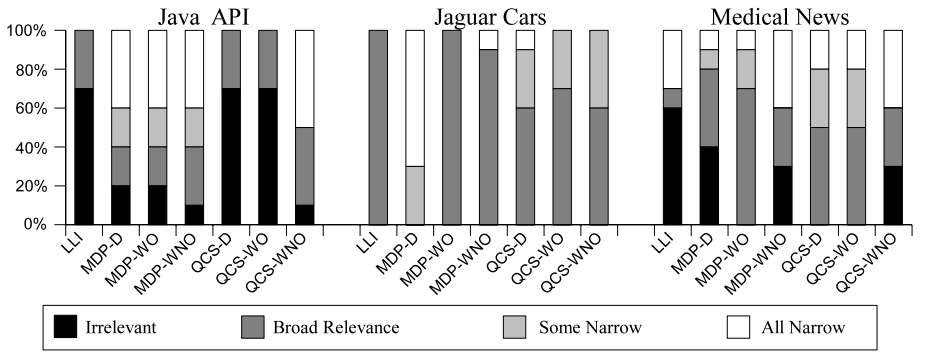

Fig. 1. Result relevance comparison for three query pages

link-based analysis, especially with the weighted non-ordinal method, can provide further differentiation between pages.

The second query presented in Fig. 1 is the Jaguar home page[2] and was also used in [5]. This is a good example to demonstrate the algorithm behavior when very little content (but much misleading) information is available on query pages. From Fig. 1, it can be seen that even with little useful content information in the query page, the performance of the MDP and QCS algorithms is still equal or superior to LLI.

The last query we present in Fig. 1 is a medical news query page[3] that discusses a potential risk factor for early-onset Parkinson's disease. It reflects a popular Web search activity that people turn to for finding health related information. Fig. 1 shows that in this case all MDP and QCS approaches perform competitively to LLI and in particular, both MDP and QCS with weighted-non-ordinal deliver the best results.

Across both techniques and all results, we found that the weighted-ordinal method produces results that are similar or identical to the direct method. An examination of program output indicates that this may be because pages with a high original ranking but lower composite ranking must be listed before all later pages from the same set, even if the later pages were assigned a higher composite relevance score. This blocking of later results is addressed by the weighted non-ordinal method, which generally delivers improved results as compared to the weighted-ordinal method. For both MDP and QCS, the weighted non-ordinal method also consistently performs better that LLI.

We have also observed another interesting property of our approaches. Although not indicated by the categorization of the results reported here, we found that our results generally favor pages with textual content about the topic in question, while LLI often favors link indices. For example, in the Jaguar case, many LLI results contain lists of links to official Jaguar sites for different countries. Our algorithms return some of these pages, but also include pages

[2] http://www.jaguar.com (likely the page has been updated since [5])
[3] http://www.medicalnewstoday.com/medicalnews.php?newsid=51960

such as a Jaguar fan club, which arguably offers content beyond pointers to other pages.

6 Conclusions

The Internet has become an enormous and popular information resource. Searching this information space, however, is often a frustrating task due to the massive scale and diversity of the digital data. It has been an active research field for developing efficient and reliable searching techniques. In this work, we investigate the approaches for searching relevant Web pages to a query page. Based on the algorithm LLI [5] that applies SVD to reveal a deep linkage association among pages, we propose two approaches (MDP and QCS) that integrate content and linkage analysis for more effective page relationship discovery and relevance ranking. Both approaches incorporate content and link information in building a neighborhood of pages and their rank-reduced SVD spaces, as well as in scoring page relevance. We have also developed a prototypical search system that implements LLI, MDP and QCS algorithms. The experiments of the system have indicated that the proposed content and linkage based searching methods (MDP and QCS) deliver improved performance and are effective in identifying more semantically relevant Web pages. As future work, we plan to conduct extensive testing of our approaches, and explore methodologies to associate and rank all neighborhood pages in a unified information space.

References

1. Bharat, K., Henzinger, M.: Improved Algorithms for Topic Distillation in a Hyperlinked Environment. In: Proceedings of 21st International ACM Conference on Research and Development in Information Retrieval, pp. 104–111 (1998)
2. Dean, J., Henzinger, M.: Finding Related Pages in the World Wide Web. In: Proceedings of 8th International World Wide Web Conference, pp. 389–401 (1999)
3. Dumais, S.: Improving the retrieval of Information from External Sources. Behavior Research Methods, Instruments, and Computers 23(2), 229–232 (1991)
4. Golub, G., Van loan, C.: Matrix Computations, 3rd edn. John-Hopkins, Baltimore (1996)
5. Hou, J., Zhang, Y.: Effectively Finding Relevant Web Pages from Linkage Information. IEEE Transactions on Knowledge and Data Engineering 15(4), 940–951 (2003)
6. Kleinberg, J.: Authoritative Sources in a Hyperlinked Environment. In: Proceedings of 9th ACM-SIAM Symposium on Discrete Algorithms, ACM Press, New York (1998)
7. Kleinberg, J., Kumar, R., Raghaven, P., Rajagopalan, S., Tomkins, A.: The Web as a graph: measurements, models, and methods. In: Asano, T., Imai, H., Lee, D.T., Nakano, S.-i., Tokuyama, T. (eds.) COCOON 1999. LNCS, vol. 1627, pp. 1–17. Springer, Heidelberg (1999)
8. Porter, M.F.: An Algorithm for Suffix Stripping. Program (14), 130–137 (1980)
9. Salton, G., Wong, A., Yang, C.S.: A Vector Space Model for Automatic Indexing. Communications of the ACM 18(11), 613–620 (1975)