Focused Crawling Using Latent Semantic Indexing - An Application for Vertical Search Engines

George Almpanidis, Constantine Kotropoulos, and Ioannis Pitas

Aristotle University of Thessaloniki, Department of Infomatics,
Box 451, GR-54124 Thessaloniki, Greece
{galba, costas, pitas}@aiia.csd.auth.gr
http://www.aiia.csd.auth.gr

Abstract. Vertical search engines and web portals are gaining ground over the general-purpose engines due to their limited size and their high precision for the domain they cover. The number of vertical portals has rapidly increased over the last years, making the importance of a topic-driven (focused) crawler evident. In this paper, we develop a latent semantic indexing classifier that combines link analysis with text content in order to retrieve and index domain specific web documents. We compare its efficiency with other well-known web information retrieval techniques. Our implementation presents a different approach to focused crawling and aims to overcome the size limitations of the initial training data while maintaining a high recall/precision ratio.

1 Introduction

Within the last couple of years, search engine technology had to scale up dramatically in order to keep up with the growing amount of information available on the web. In contrast with large-scale engines such as Google [1], a search engine with a specialised index is more appropriate to services catering for specialty markets and target groups because it has more structured content and offers a high precision. Moreover, a user visiting a vertical search engine or portal may have a priori knowledge of the covered domain, so extra input to disambiguate the query might not be needed [2]. The main goal of this work is to provide an efficient topical information resource discovery algorithm when no previous knowledge of link structure is available except that found in web pages already fetched during a crawling phase. We propose a new method for further improving targeted web information retrieval (IR) by combining text with link analysis and make novelty comparisons against existing methods.

2 Web Information Retrieval

The expansion of a search engine using a *web crawler* is seen as task of *classification* requiring supervised automatic categorisation of text documents into specific and predefined categories. The visiting strategy of new web pages usually characterises the purpose of the system. Generalised search engines that seek to cover as much proportion of the web as possible usually implement a *breadth-first* (BRFS) or *depth-first*

search (DFS) algorithm [3]. The BRFS policy is implemented by using a simple FIFO queue for the unvisited documents and provides a fairly good bias towards high quality pages without the computational cost of keeping the queue ordered [4]. Systems on the other hand that require high precision and targeted information must seek new unvisited pages in a more intelligent way. The crawler of such a system is assigned the task to automatically classify crawled web pages to the existing category structures and simultaneously have the ability to further discover web information related to the specified domain. A *focused* or *topic-driven crawler* is a specific type of crawler that analyses its crawl boundary to find the links that are likely to be most relevant for the crawl while avoiding irrelevant regions of the web. A popular approach for focused resource discovery on the web is the *best-first search* (BSFS) algorithm where unvisited pages are stored in a priority queue, known as frontier, and they are reordered periodically based on a criterion. So, a typical topic-oriented crawler performs keeps two queues of URLs; one containing the already visited links (from here on **AF**) and another having the references of the first queue also called *crawl frontier* (from here on **CF**) [5]. The challenging task is ordering the links in the CF efficiently. The importance metrics for the crawling can be either interest driven where the classifier for document similarity checks the text content and popularity or location driven where the importance of a page depends on the hyperlink structure of the crawled document.

2.1 Text Based Techniques in Web Information Retrieval

Although the physical characteristics of web information is distributed and decentralized, the web can be viewed as one big virtual text document collection. In this regard, the fundamental questions and approaches of traditional IR research (e.g. term weighting, query expansion) are likely to be relevant in web document retrieval [6]. The three classic models of text IR are probabilistic, Boolean, and vector space model (VSM). The language independent VSM representation of documents has proved effective for text classification [7]. This model is described with indexing terms that are considered to be coordinates in a multidimensional space where documents and queries are represented as binary vectors of terms. Various approaches depend on the construction of a term-by-document two-dimensional $m \times n$ matrix A where m is the number of terms and n is the number of documents in the collection. We present an extension of the classic method that can be used as classifier in focused crawling in Sect 3.2.

2.2 Link Analysis Techniques

Contrary to text-based techniques, the main target of link analysis is to identify the *importance* or *popularity* of web pages. This task is clearly derived from earlier work in bibliometrics academic citation data analysis where prestige ("impact factor") is the measure of importance and influence. More recently, link and social network analysis have been applied to web hyperlink data to identify authoritative information sources [8]. In the web, the impact factor corresponds to the ranking of a page simply by a tally of the number of links that point to it, also known as *backlink* (BL) count or *in-degree*. But BL can only serve as a rough, heuristic based, quality measure of a document, because it can favour universally popular locations regardless of the specific query topic.

PageRank (PR) is a more intelligent connectivity-based page quality metric with an algorithm that recursively defines the importance of a page to be the weighted sum of its backlinks' importance values [9]. An alternative but equally influential algorithm of modern hypertext IR is HITS, which categorises web pages to two different classes; pages rich and relevant in text content to the user's query (*authorities*) and pages that might not have relevant textual information but can lead to relevant documents (*hubs*) [10]. Hubs may not be indexed in a vertical engine as they are of little interest to the end user, however both kind of pages can collaborate in order to determine the visit path of a focused crawler.

2.3 Latent Semantic Indexing and SVD Updating

Latent semantic indexing (LSI) is a concept-based automatic indexing method that models the semantics of the domain in order to suggest additional relevant keywords and to reveal the "hidden" concepts of a given corpus while eliminating high order noise [11]. The attractive point of LSI is that it captures the higher order "latent" structure of word usage across the documents rather than just surface level word choice. The dimensionality reduction is typically computed with the help of Singular Value Decomposition (SVD), where the eigenvectors with the largest eigenvalues capture the axes of the largest variation in the data. In LSI, an approximated version of A, denoted as $A_k = U_k S_k V_k^T$, is computed by truncating its singular values keeping only the $k = rank(A_k) < k_0 = rank(A)$ larger singular values and their associated left and right eigenvectors are used for retrieval.

Unfortunately, the practical application of matrix decompositions such as SVD in dynamic collections is not trivial. Once an index is created it will be obsolete when new data (terms and documents) is inserted to the system. Adding new pages or modifying existing ones also means that the corpus index has to be regenerated for both the recall and the crawling phase. Depending on the indexing technique followed, this can be a computationally intensive procedure. But there are well-known relatively inexpensive methods such as fold-in and SVD updating that avoid the full reconstruction of the term-by-document matrix [12]. *Folding-in* is based on the existing latent semantic structure and hence new terms and documents have no effect on the representation of the pre-existing terms and documents. Furthermore, the orthogonality in the reduced k-dimensional basis for the column or row space of A (depending on inserting terms or documents) is corrupted causing deteriorating effects on the new representation. *SVD-updating*, while more complex, maintains the orthogonality and the latent structure of the original matrix [12].

3 Focused Crawling

3.1 Related Works in Focused Crawling

Numerous techniques that try to combine textual and linking information for efficient URL ordering exist in the literature. Many of these are extensions to PageRank and HITS. HITS does not work satisfactorily in cases where there is a mutually reinforcing

relationship between hosts (nepotism) [13]. An algorithm where nodes have additional properties and make use of web page content in addition to its graph structure is proposed. An improvement to HITS is probabilistic HITS (PHITS), a model that has clear statistical representations [14]. An application of PageRank to target seeking crawlers improves the original method by employing a combination of PageRank and similarity to the topic keywords [15]. The URLs at the frontier are first sorted by the number of topic keywords present in their parent pages, then they are sorted by their estimated PageRanks. The applicability of a BSFS crawler using PageRank as the heuristic is discussed in [16] and its efficiency against another crawler based on neural networks is tested. In [17] an interesting extension to probabilistic LSI (PLSI) is introduced where existing links between the documents are used as features in addition to word terms. The links can take the form of hyperlinks as is the case of HTML documents or they can take the form of citations, which is the case of scientific journal articles in citation analysis. The hypothesis is that the links contribute to the semantic context of the documents and thereby enhance the chance of successful applications. Two documents having a similar citation pattern are more likely to share the same context than documents with different citation patterns. An intelligent web crawler is suggested based on a principle of following links in those documents that are most likely to have links leading to the topic at interest. The topic is represented by a query in the latent semantic factor space. [18] proposes supervised learning on the structure of paths leading to relevant pages to enhance target seeking crawling. A link-based ontology is required in the training phase. Another similar technique is reinforcement learning [19] where a focused crawler is trained using paths leading to relevant goal nodes. The effect of exploiting other hypertext features such as segmenting Document Object Model (DOM) tag-trees that characterise a web document and propose a fine-grained topic distillation technique that combines this information with HITS is studied in [20]. Keyword-sensitive crawling strategies such as URL string analysis and other location metrics are investigated in [21]. An intelligent crawler that can adapt online the queue link-extraction strategy using a self-learning mechanism is discussed in [22]. Work on assessing different crawling strategies regarding the ability to remain in the vicinity of the topic in vector space over time is described in [23]. In [24] different measures of document similarity are evaluated and a Bayesian network model used to combine linkage metrics such as bibliographic coupling, co-citation, and companion with content-based classifiers is proposed. [25] also incorporates linking semantics additional to textual concepts in their work for the task of web page classification into topic ontologies. [26] uses tunnelling to overcome some of the limitations of a pure BSFS approach.

3.2 Hypertext Combined Latent Analysis (HCLA)

The problem studied in this paper is the implementation of a focused crawler for target topic discovery, given unlabeled (but known to contain relevant sample documents) textual data, a set of keywords describing the topics and no other data resources. Taking into account these limitations many sophisticated algorithms of the Sect. 2.2, such as HITS and context graphs, cannot be easily applied. We evaluate a novel algorithm called *Hypertext Content Latent Analysis* or **HCLA** from now onwards that tries to combine text with link analysis using the VSM paradigm. Unlike PageRank, where simple

eigen-analysis on globally weighted adjacency matrix is applied and principal eigenvectors are used, we choose to work with a technique more comparable with HITS. While the effectiveness of LSI has been demonstrated experimentally in several text collections yielding an increased average retrieval precision, its success in web connectivity analysis has not been as direct. There is a close connection between HITS and LSI/SVD multidimensional scaling [27]. HITS is equivalent to running SVD on the hyperlink relation (source, target) rather than the (term, document) relation to which SVD is usually applied. As a consequence of this equivalence, a HITS procedure that finds multiple hub and authority vectors also finds a multidimensional representation for nodes in a web graph and corresponds to finding many singular values for AA^T or A^TA, where A is the adjacency matrix. The main problem is that LSI proves inefficient when the dimensions of the term-document matrix A are small. But in the classification process of un/semi-supervised learning systems the accuracy of LSI can be enhanced by using unlabelled documents as well as labelled training data.

Our main assumption is that terms and links in an expanded matrix are both considered for document relevance. They are seen as *relationships*. In the new space introduced, each document is represented by both the terms it contains and the similar text and hypertext documents. This is an extension of the traditional "bag-of-words" document representation of the traditional VSM described in Sect. 2.1. Unlike [17], we use LSI instead of PLSI. The proposed representation, offers some interesting potential and a number of benefits. First, text only queries can be applied to the enriched relationships space so that documents having only linking information, such as those in CF, can be ordered. Secondly, the method can be easily extended for the case where we also have estimated content information for the documents in CF. This can be done using the anchor text or the neighbour textual context of the link tag in the parent's html source code, following heuristics to remedy for the problem of context boundaries identification [16]. Moreover, we can easily apply local weights to the terms/rows of the matrix, a common technique in IR that can enhance LSI efficiency. While term weighting in classic text IR is a kind of linguistic favouritism, here this can also been seen as a method of emphasizing either the use of linking information or text content. An issue in our method is the complexity of updating the weights in the expanded matrix, especially when a global weighting scheme is used. For simplicity, we do not use any weighting scheme here. Let A be the original term-document representation while $\begin{pmatrix} L_{m\times a} \\ G_{a\times a} \end{pmatrix}$ and $\begin{pmatrix} O_{m\times b} \\ R_{a\times b} \end{pmatrix}$ are the new document vectors projected in the expanded term-space having both textual (submatrices $L_{m\times a}$ and $O_{m\times b}$ and linking connectivity components (submatrices $G_{a\times a}$ and $R_{a\times b}$. The steps of our method are depicted in Fig. 1 and are described as follows.

- With a given text-only corpus of m documents and a vocabulary of n terms we first construct a text-document matrix $A_{m\times n}$ and perform a truncated Singular Value Decomposition A_k = SVD(A, k). Since this is done during the offline training phase an effort in finding the optimum k is highly suggested.

- After a sufficient user-defined number of pages (a) have been fetched be the crawler, we analyse the connectivity information of the crawler's current web graph and

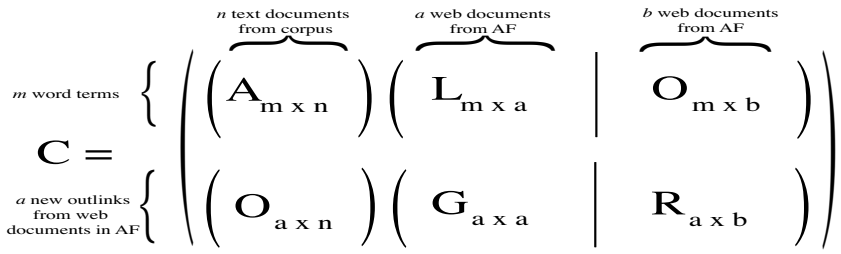

Fig. 1. Expanded connectivity matrix in HCLA. Matrix C is $[(m+a)\times(n+a+b)]$. AF=Already Fetched links, CF=Crawl Frontier docs

insert $a = |AF|$ new rows as "terms" (i.e. documents from AF) and $a+b = |AF|+|CF|$ web pages from both AF and CF as "documents" to the matrix. We perform the SVD-updating technique to avoid reconstructing the expanded index matrix. Because the matrices G and R in Fig. 1 are sparse, the procedure is simplified and the computation is reduced. We want to insert $t = a$ terms and $d = a + b$ documents, so we append submatrix $D_{(m+a)\times(a+b)} = \begin{pmatrix} L_{m\times a} & 0_{m\times b} \\ G_{a\times a} & R_{a\times b} \end{pmatrix}$ to $B_{[(m+a)\times n]} = \begin{pmatrix} A_{m\times n} \\ 0_{a\times n} \end{pmatrix}$ which is the new space after inserting terms from the AF.

- Because we do not have any information of direct relationship between any of these web pages and the text documents $\{d_i\}$ of the original corpus, we simply add a terms/rows at the bottom of the matrix A_k with zero elements. This allows the re-computing of SVD with minimum effort, by reconstructing the term-document matrix. If SVD(A, k) = $U_k S_k (V_k)^T$ is the truncated SVD of the original matrix A, and SVD$(B) = U_B S_B (V_B)^T$ the k-SVD of the matrix after inserting a documents, then we have:

$$U_B = \begin{pmatrix} U_{m\times k} \\ 0_{a\times k} \end{pmatrix}, S_B = S_k, V_B = V_k \tag{1}$$

The above step does not follow the SVD-updating technique since the full term-document matrix is recreated and a k-truncated SVD of the new matrix B is recomputed. In order to insert fetched and unvisited documents from the AF and CF queues as columns in the expanded matrix we use an SVD-updating technique to calculate the semantic differences introduced in the column and row space. If we define $SVD(C) = U_C S_C V_C^T$, $F = \left(S_k | U_B^T D\right)$ and $SVD(F) = U_F S_F V_F^T$ then, matrices U_C, S_C and V_C are calculated according to [12]:

$$V_C = \begin{pmatrix} V_B & 0 \\ 0 & I_{a+b} \end{pmatrix} V_F, \quad S_C = S_F, \quad U_C = U_B V_F \tag{2}$$

Accordingly, we project the driving original query q in the new space that the expanded connectivity matrix C represents. This is done by simply appending a rows of zeroes to the bottom of the query vector: $q_C = \begin{pmatrix} q_{m\times 1} \\ 0_{a\times 1} \end{pmatrix}$. By applying the driving

query q_C of the test topic we are able to compute a total ranking of the expanded matrix C. Looking at Fig. 1 we deduce that we only need to rank the last $b = |CF|$ columns. The scores of each document in CF are calculated using the cosine similarity measure:

$$\cos\theta_j = \frac{e_j^T V_C S_C (U_C^T q_C)}{||S_C V_C^T e_j||_2 ||q_C||_2} \tag{3}$$

where $||\cdot||_2$ is the L_2 norm. Once similarity scores are attributed to documents, we can reorder the CF, select the most promising candidate and iterate the above steps.

4 Implementation – Experimental Results - Analysis

In this work we evaluate five different algorithms. BRFS is only used as a baseline since it does not offer any focused resource discovery. The rest are cases of BSFS algorithms with different CF reordering policies. The 2^{nd} algorithm is based on simple BL count [21]. Here the BL of a document v in CF is the current number of documents in AF that have v as an outlink. The 3^{rd} algorithm (SS1) is based on the Shark-Search algorithm, a more aggressive variant of Fish-Search [28]. The 4^{th} algorithm (SS2) is similar to SS1 but the relevance scores are calculated using a pre-trained VSM that uses a probability ranking based scheme [7]. Since we work with an unlabelled text corpus, we use the topic query to extract the most relevant documents and use them as sample examples to train the system. The 5^{th} algorithm is based on PageRank. Here, no textual information is available, only the connectivity between documents fetched so far and their outlinks. A known problem is that pages in CF do not have known outlinks since they have not been fetched and parsed yet. In order to achieve convergence of the PR we assume that from nodes with no outlinks we can jump with probability one to every other page in the current web graph. In this application, the exact pagerank values are not as important as the ranking they induce on the pages. This means that we can stop the iterations fairly quickly even when the full convergence has not been attained. In practice we found that no more than 10 iterations were needed. The 6^{th} algorithm (HCLA) is the one this paper proposes. In the training phase choosing $k = 50$ for the LSI of the text corpus (matrix A) yielded good results.

The fact that the number of public available datasets suitable for combined text and link analysis is rather limited denotes the necessity of further research efforts in this field. In our experiments we used the WebKB corpus [29]. This has 8275 (after eliminating duplicates) web documents collected from universities and manually classified in 7 categories. For algorithms SS1, SS2, HCLA we selected each time three universities for training the text classifier and the fourth for testing. Documents from the "misc" university were also used for HCLA since the larger size of the initial text corpus can enhance the efficiency of LSI. Although the WebKB documents have link information we disregarded this fact in the training phase and choose to treat them only as textual data but for the testing phase we took into account both textual and linking information. The keyword-based queries that drive the crawl are also an indicative description of each category. These were formed by assigning 10 people the task of retrieving relevant documents for each category using Google and recording their queries. In each case as seeds we considered the root documents in the "department" category. This entails

the possibility of some documents being unreachable nodes in the vicinity tree by any path starting with that seed, something that explains the $< 100\%$ final recall values in Fig. 2, 4 and 4. Categories having relatively limited number of documents (e.g. "staff") were not tested. We repeated the experiments for each category and for every university. Evaluation tests measuring the overall performance were performed by calculating the average ratio of relevant pages retrieved out of the total ground-truth at different stages of the crawl. Due to the complexity of PR and HCLA algorithms we chose to follow a BSFSN strategy, applying the reordering policy every N documents fetched for all algorithms (except BRFS). This is supported by the results of [30] which indicate that explorative crawlers outperform their more exploitive counterparts. We experimented with values of N = 10, 25, 50. The preprocessing involved fixing HTML errors, converting text encoding and filtering out all external links (outlinks that are not found inside the corpus), stemming [31], and a word stoplist for both the train and test text documents.

The results in Fig. 2 depict the superiority of our method especially at higher recall ranges. We must also consider that in our implementation we didn't use term weighting, which is argued to boost LSI performance [11]. BRFS performance matched or exceeded in some cases SS1 and BL. This can be attributed to the structure of the WebKB corpus and the quality of the seed documents. The unimpressive results of PR justify the assertion that it is too general for use in topic-driven tasks due to its minimal exploitation of the topic context [16], [23]. In a BSFS strategy it is crucial that the time needed for reorganising the crawl frontier is kept at a minimum. According to [32], the best algorithms for SVD computation of an $m \times n$ matrix take time that is proportional to is $O(P \cdot m^2 \cdot n + Q \cdot n^3)$ (P and Q are constants which are 4 and 22 for a Riemannian SVD algorithm (R-SVD)). This means that the performance of a LSI-based BSFS crawler suffers when new documents and terms are inserted in each iteration. In our work, we do not need to recompute the SVD of the highly dimensional matrix C, but perform calculations on the reduced matrices of Sect. 3.2. Also, we follow a BSFSN algorithm where the reordering of the CF, and consequently the term-by-document matrix expansion and SVD computation, are performed every N documents fetched. Naturally, value N has a significant influence in the processing time of the algorithm and the efficiency of the reordering analysis [30]. For the results presented here it is $N = 50$. From Fig. 4 we deduce that reordering the CF in higher frequency does not necessarily yield better results. A parameter not well documented is the choice of k (number of important factors) in LSI. While trial and error offline experiments can reveal an optimum value for the text corpus (matrix A), there is no guarantee this will remain optimal for the expanded matrix C. In Fig. 4 we see that selecting too many features can have in fact deteriorating results.

5 Conclusions

This work has been concerned with a statistical approach to text and link processing. We argue that content- and link-based techniques can be used for both the classifier and the distiller of a focused crawler and propose an alternative document representation where terms and links are combined in an LSI based algorithm. A positive point in our method

Table 1. WebKB Corpus topic queries

Category	Topic keywords
course	course, university, homework, lesson, assignment, lecture, tutorial, book, schedule, notes, grading, handout, teaching, solutions, exam
faculty	faculty, university, professor, publications, papers, research, office
project	project, university, demonstration, objective, overview, research, laboratory
student	student, university, interests, favourite, activities, graduate, home

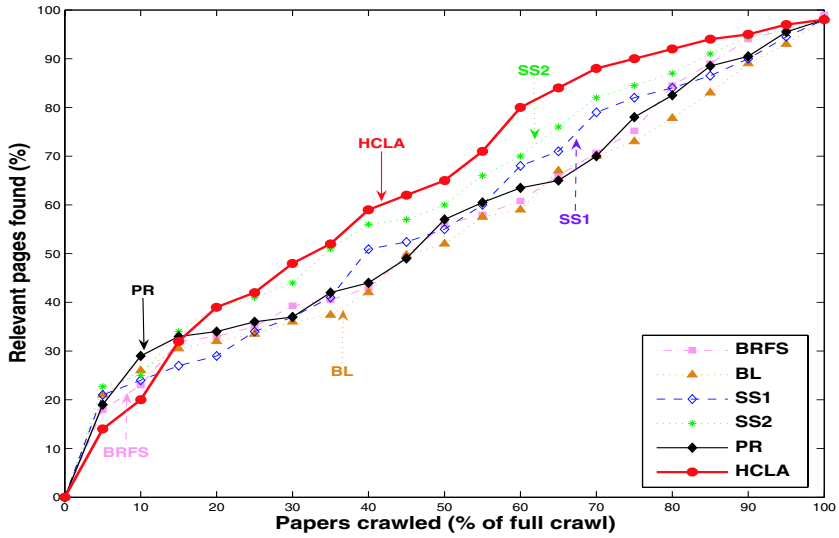

Fig. 2. Algorithm performance for WebKB

is that its training is not dependent on a web graph using a previous crawl or an existing generalised search service but only on unlabeled text samples making the problem a case of unsupervised machine learning. Because LSI performance is sensitive to the size of the trained corpus performance can suffer severely when little data is available. Therefore, starting a crawl with a small text-document matrix A is not recommended since at early stages the extra linking-text information from the crawl is minimal. Appending extra text documents in the training phase, even being less relevant to the topics of the current corpus, can enhance the crawling process. At later stages when more information is available to the system these documents can be removed and the model retrained. We also believe that a hybrid strategy where HCLA is facilitated in the early stages of the crawl by a more explorative algorithm can be a practical alternative.

The question remains whether the extra performance gain justifies the complexity it induces in the development of a focused web crawler. Both HCLA and PR methods proved significantly slower requiring more processor power and memory resources. Practically, HCLA was up to 100 times slower than the simple BRFS on some tests and

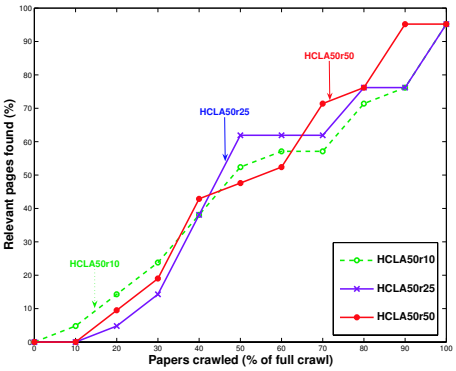

Fig. 3. HCLA performance for category project and university washington of WebKB for different BSFSN strategies. HCLA50r10 means we use $k = 50$ features for LSI analysis and reorder the CF every $N = 10$ documents

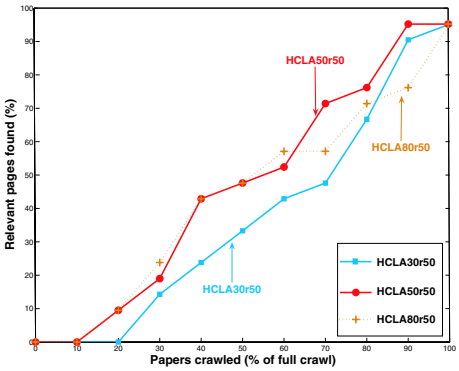

Fig. 4. HCLA performance for category project and university washington of WebKB for different BSFSN strategies. HCLA50r10 means we use $k = 50$ features for LSI analysis and reorder the CF every $N = 10$ documents

PR performed similarly, something that has been attested by [16]. The dynamic nature of the crawler means that computational complexity increases as more documents are inserted in AF and CF. A solution to the problem is to limit the size of both queues and discard less authoritative or relevant docs at the bottom of the queues during the reordering phase. Another idea worth exploring in the future is using a "folding-in" technique instead of SVD-updating during the reorganisation step of HCLA to reduce the complexity of the algorithm.

[33] also proposes an expanded adjacency matrix that allows for different weighting schemes in different directions and explores the use of eigen-analysis in the augmented matrix. There, not only term-document similarity is modelled but also term-term and document-document. It will be interesting to apply the assumption of word-link semantic equivalence in this representation of web documents. As a first step we can expand the original term-document matrix $A_{m \times n}$ during training by considering the documents as terms, i.e. add n rows to the bottom of A. In the new column vector space, a document is represented as a bag of both terms and citations (outlinks). The significance of this representation will be realised when we there is link connectivity previous knowledge between documents available, for example when deploying an incremental crawler. This can lead to semantically richer query definition.

References

1. Google Search Technology. Online at http://www.google.com/technology/index.html
2. R. Steele, "Techniques for Specialized Search Engines", in Proc. *Internet Computing*, Las Vegas, 2001.
3. S. Chakrabarti, M. Berg, and B. Dom, "Focused crawling: a new approach to topic-specific Web resource discovery", *Computer Networks*, vol. 31, pp. 1623-1640, 1999.
4. M. Najork and J. Wiener, "Breadth-first search crawling yields high-quality pages", in Proc. 10^{th} *Int. World Wide Web Conf.*, pp. 114-118, 2001.

5. A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan, "Searching the Web", *ACM Transactions on Internet Technology*, vol. 1, no. 1, pp. 2-43, June 2001.
6. K. Yang, "Combining text- and link-based methods for Web IR", in Proc. 10^{th} *Text Rerieval Conf. (TREC-10)*, Washington 2002, DC: U.S. Government Printing Office.
7. G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing". *Communications of the ACM*, vol. 18, no. 11, pp. 613-620, 1975.
8. A. Ng, A. Zheng, and M. Jordan, "Stable algorithms for link analysis", in Proc. *ACM Conf. on Research and Development in Infomation Retrieval*, pp. 258-266, 2001.
9. S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine", *WWW7 / Computer Networks,* vol. 30, no. 1-7, pp.107-117, 1998.
10. J. Kleinberg, "Authoritative sources in a hyperlinked environment", in Proc. 9^{th} *Annual ACM-SIAM Symposium Discrete Algorithms*, pp. 668-677, Jan. 1998.
11. M. Berry and M. Browne. *Understanding Search Engines: Mathematical Modeling and Text Retrieval*, Philadelphia, PA: Society of Industrial and Applied Mathematics, 1999.
12. G. O'Brien, Information Management Tools for Updating an SVD-Encoded Indexing Scheme. Master's thesis, University of Tennessee, Knoxville, TN. 1994.
13. K. Bharat and M. Henzinger, "Improved algorithms for topic distillation in hyperlinked environments", in Proc. *Int. Conf. Research and Development in Information Retrieval*, pp. 104-111, Melbourne (Australia), August 1998.
14. D. Cohn and H. Chang, "Learning to probabilistically identify authoritative documents", in Proc. 17^{th} *Int. Conf. Machine Learning*, pp. 167-174, 2000.
15. P. Srinivasan, G. Pant, and F. Menczer, "Target Seeking Crawlers and their Topical Performance", in Proc. *Int. Conf. Research and Development in Information Retrieval*, August 2002.
16. M. Chau and H. Chen, "Comparison of three vertical search spiders", *Computer,* vol. 36, no. 5, pp. 56-62, 2003.
17. D. Cohn and T. Hoffman, "The Missing Link-A probabilistic model of document content and hypertext connectivity", *Advances in Neural Information Processing Systems*, vol. 13, pp. 430-436, 2001.
18. M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori, "Focused crawling using context graphs", in Proc. 26^{th} *Int. Conf. Very Large Databases (VLDB 2000)*, pp. 527-534, Cairo, 2000.
19. J. Rennie and A. McCallum, "Using reinforcement learning to spider the Web efficiently", in Proc. 16^{th} *Int. Conf. Machine Learning (ICML99)*, pp. 335-343, 1999.
20. S. Chakrabarti, "Integrating the Document Object Model with hyperlinks for enhanced topic distillation and information extraction", in Proc. 10^{th} *Int. World Wide Web Conf*, pp. 211-220, Hong Kong, 2001.
21. J. Cho, H. G. Molina, and L. Page, "Efficient Crawling through URL Ordering", in Proc. 7^{th} *Int. World Wide Web Conf.*, pp. 161-172, Brisbane, Australia 1998.
22. C. Aggarwal, F. Al-Garawi, and P. Yu, "Intelligent Crawling on the World Wide Web with Arbitrary Predicates", in Proc. 10^{th} *Int. World Wide Web Conf.*, pp. 96-105, Hong Kong, 2001.
23. F. Menczer, G. Pant, M. Ruiz, and P. Srinivasan. "Evaluating topic-driven web crawlers", in Proc. *Int. Conf. Research and Development in Information*, pp. 241-249, New Orleans, 2001.
24. P. Calado, M. Cristo, E. Moura, N. Ziviani, B. Ribeiro-Neto, and M.A. Goncalves, "Combining link-based and content-based methods for web document classification", in Proc. 12^{th} *Int. Conf. Information and Knowledge Management*, pp. 394-401, New Orleans, Nov. 2003.
25. I. Varlamis, M. Vazirgiannis, M. Halkidi, and B. Nguyen, "THESUS: Effective thematic selection and organization of web document collections based on link semantics", *IEEE Trans. Knowledge & Data Engineering*, vol. 16, no. 6, pp. 585-600, 2004.

26. D. Bergmark, C. Lagoze, and A. Sbityakov, "Focused Crawls, Tunneling, and Digital Libraries", in Proc. 6^{th} *European Conf. Research and Advanced Technology for Digital Libraries*, pp. 91-106, 2002.
27. S. Chakrabarti, *Mining the Web: Discovering Knowledge from Hypertext Data*. San Francisco, CA: Morgan Kaufmann Publishers, 2002.
28. M. Hersovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur, "The shark-search algorithm. An Application: tailored Web site mapping", *Computer Networks and ISDN Systems*, vol. 30, pp. 317-326, 1998.
29. CMU World Wide Knowledge Base and WebKB dataset.
Online at http://www-2.cs.cmu.edu/∼webkb
30. G. Pant, P. Srinivasan, and F. Menczer, "Exploration versus exploitation in topic driven crawlers", in Proc. 2^{nd} *Int. Workshop Web Dynamics*, May, 2002.
31. M. Porter, "An algorithm for suffix stripping". *Program*, vol. 14 no. 3, pp. 130-137, 1980.
32. G. Golub and C. Van Loan. *Matrix Computations*, 3/e. Baltimore: Johns Hopkins University Press, 1996.
33. B Davison. "Unifying text and link analysis", in Proc. *IJCAI-03 Workshop Text-Mining & Link-Analysis (TextLink)*, Acapulco, August 9, 2003.