A survey on effective webmining algorithms

M. Renuka Devi1 and S.Saravanan2

1MCA Department, Sree Saraswathi Thyagaraja College, Pollachi, Bharathiar University, Coimbatore, Tamil Nadu, India.
2Department of Computer Science, Nehru Arts and Science College, Coimbatore, Bharathiar University, Coimbatore, Tamil Nadu, India.

ABSTRACT
This paper presents three algorithms and one proposed algorithm to find relevant pages for a given Web page (URL). The first algorithm comes from the Pagerank analysis of the Web pages. It is intuitive and easy to implement. The second one takes advantage of HITS to identify relevant pages more precisely and effectively. The third one shows the advantages and limitations of Weighted Pagerank algorithm and the final proposed algorithm based on Topic Sensitive Pagerank algorithm. These algorithms could be used for various Web applications, such as enhancing Web search. The ideas and techniques in this work would be helpful to other Web-related researches.

Introduction
In general, the World Wide Web (www) is a system of interlinked hypertext documents [1]. WWW provides an architectural framework for accessing linked documents spread out over millions of machines all over the Internet [2]. Retrieving useful information from the vast sea of data on World Wide Web has been one of the most challenging tasks. Web search engines have surfaced as a useful technique that helps in searching for useful information on the World Wide Web using search strings provided by the user. The search results of a search engine are generally presented as a list often referred to as search engine results pages (SERPs). To locate any information from the web, the user accesses his favorite search engine, issues queries and clicks on the returned pages [3]. The search results returned by a search engine are a mixture of large amount of relevant and irrelevant information [4]. Any user cannot read all web pages returned in response to the user’s query. Hence, search engines help users trace relevant pages worth considering by displaying the resultant pages in a ranked order using different page rank algorithms [4]. Web-page ranking is a search-engine optimization technique used by search engines for ranking hundred thousands of web pages in a relative order of importance. Conventional search engine technology can be broadly classified into two main categories of search engines: the crawler based engine and the human-powered directories based engine [5]. A human-powered directory, for instance the Open Directory depends on humans for its listings [6]. The web pages in such a setting are stored in different directories on the basis of their category. When a query is fired, it is categorized first and then the appropriate directory is searched to locate the web page. They are constructed when the owner of a website submits the site for a review along with a short description of the site [3]. A search is generally based on the matches only in the descriptions submitted.

Crawler-based search engines, for instance Google, create their listings automatically [6]. They “crawl” or “spider” the web, to search for pages matching user requests. Once they generate result sets, people can navigate through the results. Crawler-based search engines retrieve contents of web pages using indexers [6]. Indexers are used to store and index information regarding retrieved pages. The Ranker determines the importance of web pages returned and the Retrieval Engine performs lookups on index tables. The web page ranking algorithms play their role at the last component [7]. Exactly what information the user wants is unpredictable. So the web page ranking algorithms are designed to anticipate the user requirements from various static (e.g., number of hyperlinks, textual content) and dynamic (e.g., popularity) features [7]. They are important factors for making one search engine better than another [7]. Web search ranking algorithms play an important role in ranking web pages so that the user could get the good result which is more relevant to the user’s query [8]. The following figure illustrates the working of a typical search engine, which shows the flow graph for a searched query by a web user.

In this paper, we introduce and discuss various algorithms and techniques that we developed for web community mining and analysis.

Page Rank Algorithm
Page Rank algorithm is the most commonly used algorithm for ranking the various pages. Working of the Page Rank algorithm depends upon link structure of the web pages. The Page Rank algorithm is based on the concepts that if a page contains important links towards it then the links of this page towards the other page are also to be considered as important pages. The Page
Rank considers the back link in deciding the rank score. If the addition of the all the ranks of the back links is large then the page then it is provided a large rank [9][10][11][12]. A simplified version of PageRank is given by:

$$PR(u) = \frac{PR(v)}{L(v)} \sum_{v \in Bu}$$

Where the PageRank value for a web page u is dependent on the PageRank values for each web page v out of the set Bu (this set contains all pages linking to web page u), divided by the number L(v) of links from page v. An example of back link is shown in Figure 1.1 below. U is the back link of V & W and V & W are the back links of X.

![Figure 1.1: Illustration of back links](image)

**Hits Algorithm**

HITS algorithm ranks the web page by processing in links and out links of the web pages. In this algorithm a web page is named as authority if the web page is pointed by many hyper links and a web page is named as HUB if the page point to various hyperlinks. An Illustration of HUB and authority are shown in Figure 2.1.

![Figure 2.1: Illustration of Hub and Authorities](image)

**Weighted Page Rank Algorithm**

Weighted Page Rank [9] algorithm is proposed by Wenpu Xing and Ali Ghorbani. Weighted page rank algorithm (WPR) is the modification of the original page rank algorithm. WPR decides the rank score based on the popularity of the pages by taking into consideration the importance of both the in-links and out-links of the pages. This algorithm provides high value of rank to the more popular pages and does not equally divide the rank of a page among its out-link pages. Every out-link page is given a rank value based on its popularity. Popularity of a page is decided by observing its number of in links and out links. Simulation of WPR is done using the Website of Saint Thomas University and simulation results show that WPR algorithm finds larger number of relevant pages compared to standard page rank algorithm. As suggested by the author, the performance of WPR is to be tested by using different websites and future work include to calculate the rank score by utilizing more than one level of reference page list and increasing the number of human user to classify the web pages.

**Topic Sensitive Pagerank**

In Topic Sensitive PageRank, several scores are computed: multiple importance scores for each page under several topics that form a composite PageRank score for those pages matching the query. During the offline crawling process, 16 topic-sensitive PageRank vectors are generated, using as a guideline the top-level category from Open Directory Project (ODP). At query time, the similarity of the query is compared to each of these vectors or topics; and subsequently, instead of using a single global ranking vector, the linear combination of the topic-sensitive vectors is weighed using the similarity of the query to the topics. This method yields a very accurate set of results relevant to the context of the particular query.

**Proposed Methodology**

Our technique Topic sensitive weighted page rank makes use of a subset of the ODP category structure that is associated with individual information needs. This subset is processed locally, aiming at enhancing generic results offered by search engines. Processing entails introducing importance weights to those parts of the ODP structure that correspond to user-specific preferences. The results of local processing are subsequently combined with global knowledge to derive an aggregate ranking of web results. In the following subsection we describe in more detail the theoretical model used and the algorithmic steps deployed.

**A. Background**

Our proposed techniques is the following. Consider an arbitrary search engine uses a graphs structure G(V,E) of categories, in order to categorize web pages. Graph G consists of
nodes $v \in V$ that denote categories and every edge $vi, vj \in E$ denotes that $vj$ is a subcategory of $vi$ and is assigned a weight $d(vi, vj) \in [0, 1)$. It assumed that every web page is tagged with a specific category. Overall, the proposed ranking approach introduces the idea of incrementally selecting subgraph $G_{Sub}$ of $G$. This subgraph can be constructed according to set of some basic topics chosen from ODP. In the extreme case $G_{Sub} = G$. Every category $v \in G$ will be assigned a relevance-importance weight $\beta \in [0, 1)$. These weights are used in order to categorize pages returned to the end user, when posing a query. In particular, the position (rank) of a page pin the result-set of an arbitrary user query will be given by a function of the form: $\Phi(\beta, p, \sigma, p)$. In the above function, $(p)$ is the category that a page $p$ belongs to, $(\sigma(p))$ is the relevance-importance according to the ranking algorithm of the engine, and function $\Phi()$ indicates how the final ranking will be biased towards machine ranking or category importance defined (for example, $\Phi = \alpha + \beta/2$). In general, we have introduced the function $\Phi()$ that combines search engine ranking and our proposed ranking techniques in order to provide better scalability of our solution’s.

Figure 5.1 Proposed System Architecture

B. Offline Methodology Roadmap

In our approach, the first step is to generate a biased weighted pagerank vectors using a set of some basis topics. This step is the pre-processing step of the web crawler. This step is performed offline. We select these topics from freely available Open Directory Project as dmoz.

Let $T_j$ be the set of URLs in the ODP category $c_j$. Then we will compute the Weighted PageRank vector $v_j$ for topic $c_j$ where

$$v_j = \begin{cases} \frac{1}{|T_j|} & i \in T_j \\
0 & i \notin T_j \end{cases}$$

The Weighted PageRank vector for topic $c_j$ is given $WPR(\alpha, \beta)$ where $\alpha$ is bias factor. We also compute the same class term vectors $D_j$ consisting of the term in document below each of the top-level categories.

C. Compute Importance Score At Query Time

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

$$P(c_j|q') = P(c_j|q) \approx \pi(c_j) \cdot \frac{(q'|c_j)}{(q'|c_j)}$$

($$(q'|c_j)$$) is easily computed from the class term-vector $D_j$. The quantity $(c_j)$ is not as straight forward. We chose to make it uniform, although we could personalize the query results for different users by varying this distribution. In other words, for some user $k$, we can use a prior distribution $P_k(c_j)$ that reflects the interests of user $k$. Using a text index, we retrieve URLs for all documents containing the original query terms $q$. Finally, we compute the query-sensitive importance score of each of these retrieved URLs as follows.

Let $rank_jd$ be the rank of document $d$ given by the rank vector $WPR(\alpha, \beta)$ (i.e., the rank vector for topic $c_j$). For the document $d$, we compute the query-sensitive importance score $sqd$ as follows.

$$sqd = \sum_j P(c_j|q') \cdot rank_jd$$

The results are ranked according to this composite score $sqd$. The above query-sensitive Weighted PageRank computation has the following probabilistic interpretation, in terms of the “random surfer” model [26]. Let $w_j$ be the coefficient used to weight the jth rank vector, with $w_j = 1$ (e.g. $w_j(c_j|q')$). Then note that the equality

$$\sum_j [w_j \cdot WPR(\alpha, \beta) | v_j] = WPR(\alpha, \sum_j [w_j | v_j])$$

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

Conclusion

In this paper, we have discussed three algorithms advantages and limitations and we have proposed a new concept based on Topic-Sensitive PageRank and Weighted PageRank for web page ranking. Out this approach is based on the PageRank algorithm, and provides a scalable approach for search rankings using Link analysis. For each Web page, compute an importance score per topic. At query time, these importance scores are combined based on the topics of the query and associated context to form a composite PageRank score for those pages matching the query. This score can be used in conjunction with other scoring schemes to produce a final rank for the result pages with respect to the query. This algorithm will improve the order of web pages in the result list so that user may get the relevant pages easily.

References


Bibliography

Dr. M. Renuka Devi has nearly 10 years of post-graduate teaching experience in Computer Science. She has indulged in training the post graduate students to complete real time projects and also guides research scholars in Computer Science. Currently she is working as Assistant Professor in the Department of MCA at Sree Saraswathi Thyagaraja College (Autonomous), and an ISO 9001 Certified/ NAAC Accredited Institution, Pollachi, Coimbatore (DT), Tamil Nadu, India.

Mr. S. Saravanan, has nearly 4 years of Under Graduate teaching experience in Computer Science. Currently he is doing his research (Part time) at Sree Saraswathi Thyagaraja College (Autonomous), and an ISO 9001 Certified/ NAAC Accredited Institution, Pollachi, Coimbatore (DT), Tamil Nadu, India. Also he is working as Assistant Professor in Nehru Arts and Science College, Coimbatore, Tamil Nadu, India.