ISSN:0975-9646

Comprehensive Review of Web Focused Crawling

Promila Devi[#1], Ravinder Thakur[*2]

[#] *M.Tech Scholar, Department of Computer Science & Engineering, LRIET Solan, HPTU Hamirpur, India*

* *Assistant Professor, Department of Computer Science & Engineering, LRIET Solan, HPTU Hamirpur, India*

***Abstract*— Finding useful information from the Web which has a huge and widely distributed structure requires efficient search techniques. Distributive and varying nature of Web resources is always major issue for search engines maintain latest index of the Web content as they have to crawl the Web after fixed interval of time. A focused driven crawler is a specific type of straggler that analyzes its crawl boundary to find the links that are to be in range for the crawl while avoiding undesired areas of the Web. Still many types of crawlers have been suggested that have different crawler strategies. To do this, focused crawler has an algorithm for classifying. In this paper we are review algorithm used to classify in focused crawlers. These algorithms may be based on page contents or uses a semantic classification or even on both.**

***Keywords*— Focused Crawler, Web Crawler, Search engine, Relevancy prediction.**

I. INTRODUCTION

The rapid growth of the World-Wide Web poses unpredictable challenges for general-purpose crawlers and search engines. A focused crawler or topical straggler is a web crawler that attempts to download only web pages that are related to a pre-defined topic or given set of topics. Topical crawling generally assumes that only the topic is given, while focused crawling also assumes that some labeled examples of required and not required pages are available. A focused crawler may be described as a crawler which returns relevant web pages on a given topic in the web. There are a number of issues related to existing focused crawlers, in particular the ability to ``tunnel" through lowly ranked pages in the search path to highly ranked pages related to a topic which might re-occur further down the search path. A focused crawler has the following main components:

(a) A way to determine if a particular web page is relevant to the given topic, and

(b) a way to determine how to proceed from a known set of pages.

An early search engine which deployed the focused crawling strategy was proposed in [1] based on the intuition that relevant pages often contain relevant links. It searches deeper when relevant pages are found, and stops searching at pages not as relevant to the topic. Unfortunately, the above crawlers show an important drawback when the pages about a topic are not directly connected in which case the crawling might stop pre-maturely.

This problem is tackled in [3] where reinforcement learning permits credit assignment during the search process, and hence, allowing off-topic pages to be included in the search path. However, this approach requires a large number of training examples, and the method can only be trained offline. In [2], a set of classifiers are trained on examples to estimate the distance of the current page from the closest on-topic page. But the training procedure is quite complex.

Our focused crawler aims at providing a simpler alternative for overcoming the issue that immediate pages which are lowly ranked related to the topic at hand. The idea is to recursively execute an exhaustive search up to a given depth d, starting from the ``relatives" of a highly ranked page. Hence, a set of candidate pages is obtained by retrieving pages reachable within a given perimeter from a set of initial seeds. From the set of candidate pages, we look for the page which has the best score with respect to the topic at hand. This page and its ``relatives" are inserted into the set of pages from which to proceed the crawling process. Our assumption is that an ``ancestor" with a good reference is likely to have other useful references in its descendants further down the lineage even if immediate scores of web pages closer to the ancestor are low. We define a degree of relatedness r with respect to the page with the best score. If r is large, we will include more distant ``cousins" into the set of seeds which are further and further away from the highest scored page.

This device overcomes the difficulties of using reinforcement learning in assigning credits, without the burden of solving a dynamic programming problem. These ideas may be considered as an extension to [1,2], as the use of a degree of relatedness extends the concept of child pages in [1] while avoiding the complex issue of inherence of scores, and the use of a perimeter is similar to the ``layer" concept used in [2].

II. RELATED WORK

The Focused crawling was first introduced by Chakrabarti in 1999[4]. One of the first web crawlers was proposed by Cho J et. al.[5] and they introduced a best first strategy. Fish-Search [1] is an example of early crawlers that prioritizes unvisited URLs on a queue for a specific search goal. The Fish-Search approach assigns priority values (1 or 0) to candidate pages using simple keyword matching. One of the disadvantages of Fish-Search is that all relevant pages are assigned the same priority value 1 based on keyword matching.

The Shark-Search [6] is a modified version of Fish-Search, in which, Vector Space Model (VSM) is used, and the priority values (more than just 1 and 0) are computed based on the priority values of parent pages, page content, and

anchor text. Shark-Search is a modification of Fish-search which differs in two ways: a child inherits a discounted value of the score of its parent, and this score is combined with a value based on the anchor text that occurs around the link in the Web page. Many researchers have written their approaches based on link analysis. For example, Effective Focused Crawling based on content and link structure analysis has been proposed for link analysis based on URL score, anchor score and relevance score and HAWK: A Focused Crawler with Content and Link Analysis.

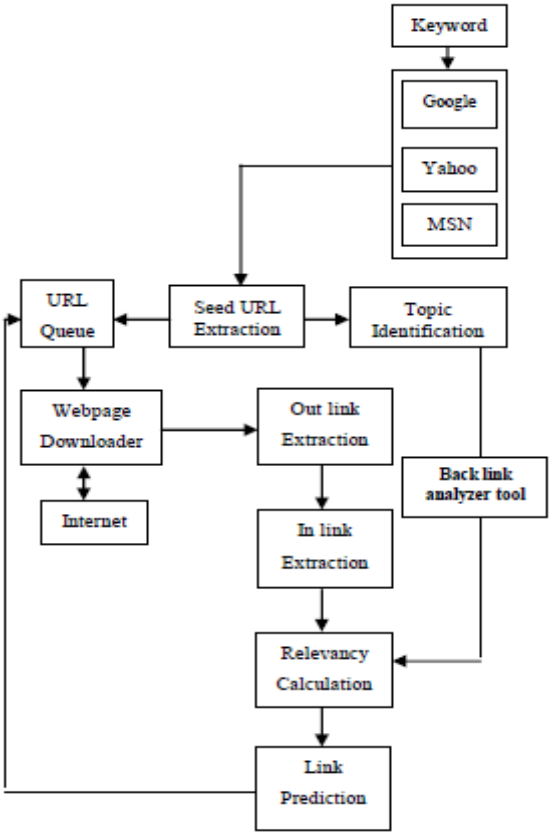

Figure 1: Work flow of Web Focused Crawler

Info Spiders and Best-First are additional examples of focused crawling methods [6]. The difference between them is that InfoSpiders uses Neural Networks, while Best-First method applies VSM to compute the relevance between candidate pages and the search topic. Shark-Search crawlers may be considered as a type of Best-First crawlers, but the former has a more complicated function for computing the priority values. In [6], Best-First was shown most successful due to its simplicity and efficiency. *N*-Best-First is generalized from Best-First, in which *N* best pages are chosen instead of one.

A relatively more recent type of focused crawlers adopts learning-based approaches to relevance prediction. This approach first trains a classifier using a training dataset and then applies the classifier to unvisited URLs. Our review crawler is a learning-based one with an enhanced relevance prediction model.

III. FOCUSED CRAWLING CLASSIFIER

In this section, we address two major issues. First, we discuss the training set preparation, which contains values of the aforementioned four relevance attributes: URL words relevancy, anchor text relevancy, parent pages relevancy, and surrounding text relevancy, besides the class label for each seed page (relevant=Yes or irrelevant=No). Second, we train the Naïve Bayesian classifier and Decision Tree Induction using the training set, and then apply the trained classifier to predict the relevancy of unvisited URLs with regard to the crawling topic.

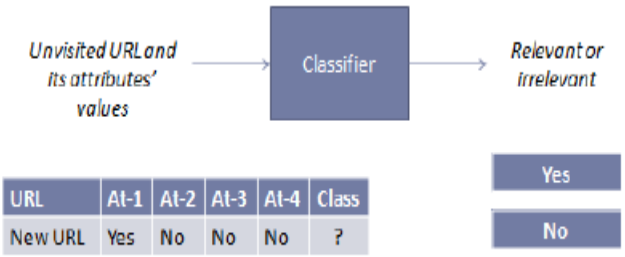

Figure 2: Classifiers input and Output

A. *Training Set Preparation*

1) *Relevant Seed URLs Generation:* As we first extract URLs which are common in all three search engine results.

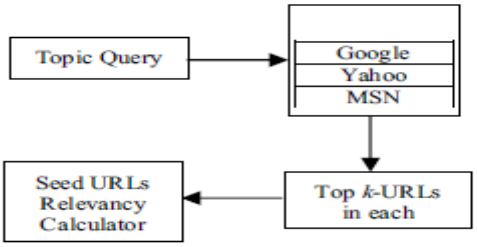

Figure 3: Relevant URL seed Generation

We assume that this common search result URLs are most relevant for this query and thus these URLs are grouped into most relevant group seed URLs. Now, we extract those URLs which are common in any two search engine results.

2) *Irrelevant Seed URLs Generation:* It is based on visited URLs property, we classify the unvisited URLs. There are positive and negative results in pertained data. So, for negative results, we put the query in site with negative (-) sign, and here we put a positive query "related words" and negative query –"Irrelevant words "in Site and find out resulting URLs.

3) *Creation of Topic Keywords Weights Table*: As Weight table defines the crawling target. The topic name can be sent as a query to the Google Web search engine and the first k results are retrieved. The retrieved pages are parsed. To avoid indexing useless words, a text retrieval system often associates a stop list with a set of documents. A stop list is a set of words that are deemed "irrelevant." Stop lists may vary per document set.

For example, database systems could be an important keyword in a newspaper. However, it may be considered as

a stop word in a set of research papers presented in a database systems conference. A group of different words may share the same word stem. A text retrieval system needs to identify groups of words where the words in a group are small syntactic variants of one another and collect only the common word stem per group.

B. Classifiers

1) *Naïve Bayesian Classifier:* This section discusses how Naïve Bayesian classifier is trained to predict the relevancy of unvisited URLs and proposes a new focused crawler algorithm. This approach is based on the Naïve Bayesian classification method described in [7]. The main method that we followed to calculate the probability of a given unvisited URL as relevant or irrelevant is reflected in Inequality (1) (as shown below). The left side of the inequality can be considered as the probability of X being relevant, and the right side as the probability of X being irrelevant, where X is the unvisited URL. Therefore, if the inequality holds, X is relevant, otherwise X is irrelevant. The way of computing the terms in (1) is explained by equations (2) and (3). We train our classifier in two different ways. First, we use a fixed training set – i.e., the training set never change once built. Second, we use an updatable training set that is constantly fed with new relevant and irrelevant URLs after built and during crawling process.

P(X | Relevant=Yes) * P (Relevant = Yes) >
P(X | Relevant = No) * P (Relevant = No) (3)

P (Relevant = Yes) = # of relevant/total (4)

P(X | Relevant =Yes) =
P (URL words relevancy | Relevant =Yes) *
P (Anchor text relevancy | Relevant =Yes) *
P (Parent pages relevancy | Relevant =Yes) *
P (Surrounding text relevancy | Relevant =Yes)

P (Relevant = No) = # of irrelevant/total (5)

P(X | Relevant =No) =
P (URL words relevancy | Relevant =No) *
P (Anchor text relevancy | Relevant = No) *
P (Parent pages relevancy | Relevant = No) *
P (Surrounding text relevancy | Relevant =No)

2) *Decision Tree Induction Classifier:* The DTI base classifiers calculate dissimilarity by computing the average distance between the set of non-linearly aligned feature vectors belonging to new Visited URL and the relevant words for the claimed owner. By taking the negative log-likelihood of this probability, a dissimilarity measure is obtained. The base classifiers subsequently convert the obtained dissimilarity measure into a confidence score, by using a sigmoid score normalization function. This normalization technique utilizes the writer statistics determined during model training.

Finally, a global decision threshold is imposed. If and only if the confidence score obtained for a questioned search is equal to or greater than the required threshold value, the claim of ownership is accepted. This crawling algorithm (predicting the relevance of unvisited URLs). In our implementation, relevant URLs and irrelevant URLs are maintained in separate tables: the *Relevant Table* keeps the identified relevant URLs and the *Irrelevant Table* keeps the identified irrelevant URLs.

The first step in our algorithm puts all out-links from seed pages in a *Queue*. Each URL taken from the *Queue* is sent to a function that computes its relevance attribute values (i.e., URL words relevancy, anchor text relevancy, parent pages relevancy, and surrounding text relevancy). Then, the classifier takes the URL with its attributes values as inputs and makes prediction of its relevancy to the search topic. Based on the predication made, the algorithm differentiates the following two cases accordingly.

In this Focused Crawler Algorithm, The first case: the URL is predicted as relevant. In this case, the URL is inserted into the *Relevant Table*, and the page is downloaded and all out-links from it are added to *Queue* that keeps a list of URLs waiting to be crawled. At the same time, the *level counter* is reset to 0 in this case. The level counter variable is increased by one if the crawler moves from an irrelevant page to another irrelevant page; and it is reset to 0 upon entering a relevant page.

The second case: the URL is predicted as irrelevant. In this case, the URL is inserted into the *Irrelevant Table*, and the page is downloaded (and all out-links are extracted into *Queue*) if the level counter is currently less than the *level limit* (another control variable to be explained shortly), otherwise the URL is ignored.

IV. CONCLUSION

A focused crawler seeks out and downloads web pages that are related to the topic to be search is given to the crawler. A learning-based focused crawler has learning ability to adapt to its search topic and to improve the accuracy of its prediction of relevancy of unvisited URLs. In this paper, we review the learning-based focused crawling approach that uses four relevance attributes to predict the relevance of unvisited URLs. The four attributes are the URL words, its anchor text, the parent pages, and the surrounding text. Our approach adopted Naïve Bayesian classification model, which can be extended to other more sophisticated models.

V. FUTURE SCOPE

As future work, we plan to do more extensive tests with larger volumes of web pages by using various classification models such as DTI and Neural Networks.

REFERENCES

[1] De Bra, P., Houben, G., Kornatzky, Y., Post, R. ``*Information retrieval in distributed hypertexts". Proc. 4th RIAO Conference*, 1994.

[2] Diligenti, M., Coetzee, F., Lawrence, S., Giles, L., Gori, M. ``*Focused crawling using context graphs". Proc. VLDB 2000*, Cairo, Egypt, 2000.

[3] McCallum, A., Nigam, K., Rennie, J., Seymore, K. ``*Automating the Construction of Internet Portals with Machine Learning". Information Retrieval*. Vol 3, 127-163, 2000.

[4] Chakrabarti S., Van den berg M, Dom B, *"Focused Crawling: A new approach to Topic Specific Web Resource discovery"*, Proceedings WWW 1999.

[5] Choj, Garcia Molina H, Page L., *"Efficient Crawling Through URL ordering "*, Proceedings WWW 1998.

[6] M. Hersovici, A. Heydon, M. Mitzenmacher, D. Peeleg, *"The Shark-Search Algorithm: An application Tailored website mapping"*, Proceedings of WWW conference, Brisbane, Australia, 1998, 317-326.

[7] Han, J. and Kamber, M. 2003. *"Data Mining: Concepts and Techniques."* San Francisco: Morgan Kaufman.

[8] Pal, A., Tomar, D. S., and Shrivastava, S. C. 2009. *Effective Focused Crawling Based on Content and Link Structure Analysis.* International Journal of Computer Science and Information Security (IJCSIS), Vol. 2, No. 1, June 2009.

[9] Sun, Y., Jin, P., and Yue, L. 2008. *A Framework of a Hybrid Focused Web Crawler*. 2nd International Conference on Future Generation Communication and Networking Symposia.

[10] Mejdl S. Safran Abdullah Althagafi and Dunren Che , *"Improving Relevance Prediction for Focused Web Crawlers",2012* IEEE/ACIS 11th International Conference on Computer and Information Science

[11] Debashis Hati ,Amritesh Kumar ,Lizashree Mishra," *Unvisited URL Relevancy Calculation in Focused Crawling Based on Naïve Bayesian Classification ", International Journal of Computer Applications (0975 – 8887) Volume 3 – No.9, July 2010.*