PROJECT : CTRNet Focused Crawler

CS5604 Information Storage and Retrieval
Fall 2012
Virginia Tech

Focused Crawler

Mohamed Magdy Gharib Farag
mmagdy@vt.edu
Virginia Tech

Mohammed Saquib Akmal Khan
mohak12@vt.edu
Virginia Tech

Gaurav Mishra
mgaurav@vt.edu
Virginia Tech

Prasad Krishnamurthi Ganesh
kgprasad@vt.edu
Virginia Tech

Edward A. Fox
fox@vt.edu
Virginia Tech

ABSTRACT:

Finding information on WWW is difficult and challenging task because of the extremely large volume of the WWW. Search engine can be used to facilitate this task, but it is still difficult to cover all the webpages on the WWW and also to provide good results for all types of users and in all contexts. Focused crawling concept has been developed to overcome these difficulties. There are several approaches for developing a focused crawler. Classification-based approaches use classifiers in relevance estimation. Semantic-based approaches use ontologies for domain or topic representation and in relevance estimation. Link analysis approaches use text and link structure information in relevance estimation. The main differences between these approaches are: what policy is taken for crawling, how to represent the topic of interest, and how to estimate the relevance of webpages visited during crawling. We present in this report a modular architecture for focused crawling. We separated the design of the main components of focused crawling into modules to facilitate the exchange and integration of different modules. We will present here a classification-based focused crawler prototype based on our modular architecture.

Keywords – *Focused Crawler, Crawler, Naive Bayes, Support Vector Machine*

1. INTRODUCTION

A web crawler is defined as an automated program that methodically scans through Internet pages and downloads any page that can be reached via links. With the exponential growth of the Web, fetching information about a special-topic is gaining importance.

A focused crawler is a web crawler that attempts to download only web pages that are relevant to a predefined topic or set of topics. In order to determine a web page is about a particular topic, focused crawlers use classification techniques. Topical crawling generally assumes that only the topic is given, while focused crawling also assumes that some labeled examples of relevant and not relevant pages are available. Topical crawling was first introduced by Menczer.

A focused crawler ideally would like to download only web pages that are relevant to a particular topic and avoid downloading all others. Therefore a focused crawler may predict the probability that a link to a particular page is relevant before actually downloading the page. A possible predictor is the anchor text of links. In a review of topical crawling algorithms show that such simple strategies are very effective for short crawls, while more sophisticated techniques such as Reinforcement Learning and evolutionary adaptation can give the best performance over longer crawls. Some Researchers propose to use the complete content of the pages already visited to infer the similarity between the driving query and the pages that have not been visited yet.

In another approach, the relevance of a page is determined after downloading its content. Relevant pages are sent to content indexing and their contained URLs are added to the crawl frontier; pages that fall below a relevance threshold are discarded.

The performance of a focused crawler depends mostly on the richness of links in the specific topic being searched, and focused crawling usually relies on a general web search engine for providing starting points.

***Basic Algorithm for Crawling**:*

- Get a URL from priority queue
- Check if URL is:
- Visited (i.e., its web page is already downloaded)
- In Queue (i.e., no need to put it again in queue)
- Download
- Extract text and URLs
- Estimate relevance
- Put URLs in priority queue
- Stop if queue is empty or reach pages limit

The basic operation of any hypertext crawler (whether for the Web, an intra-net or other hypertext document collection) is as follows. The crawler begins with one or more URLs that constitute a *seed set*. It picks a URL from this seed set, then fetches the web page at that URL. The fetched page is then parsed, to extract both the text and the links from the page (each of which points to another URL). The extracted text is fed to a text indexer. The extracted links (URLs) are then added to a *URL frontier*, which at all times consists of URLs whose corresponding pages have yet to be fetched by the crawler. Initially, the URL frontier contains the seed set; as pages are fetched, the corresponding URLs are deleted from the URL frontier. The entire process may be viewed as traversing the web graph. In continuous crawling, the URL of a fetched page is added back to the frontier for fetching again in the future. The algorithms uses max heap data structure of maintain a priority queue.

***Note:** The architecture and details of crawling process are defined in Implementation section of this report.*

2. RELATED WORK

Machine learning techniques have been used in focused crawling [1-11]. Most focused crawler approaches use classification algorithms to learn a model from training data. The model is then used by the focused crawler to determine the relevancy of the unvisited URLs. The effectiveness of classifier-based focused crawler depends mainly on the training data used for learning the model. The training data has to broad enough to cover all the aspects of the topic of interest. Some of the most used text classification algorithms are Naïve Bayes, and SVM. Semi-supervised clustering was also used for focused crawling. Hidden Markov Models (HMM) were also used for learning user behavior patterns or for learning link patterns for relevant and non-relevant URLs.

Semantic web techniques also were used in focused crawling [12-22]. Semantic focused crawlers used ontology for describing the domain or topic of interest. The domain ontology can be built manually by domain experts or automatically, using concepts extraction algorithms from text. Once the ontology is

built, it can be used for estimating the relevance of unvisited URLs by comparing the concepts extracted from the target webpage with the concepts that exists in the ontology. The performance of semantic focused crawling also depends on how well the used ontology describes and covers the domain or topic of interest.

Link analysis [23-34] was used in focused crawling to incorporate the link structure of the already visited webpages to the decision making process. Link analysis adds the feature of popularity of webpages as another parameter in the relevance estimation process. Link analysis includes analyzing linkage structure of webpages, and the link context. Link context is the part of text that surrounds the anchor of the URL and gives the context of having this URL in the webpage. As extreme case, each URL can have the whole text in the webpage as its context. The link context can be represented by context graphs, which is a formal representation of the concepts in the context text using Formal Concept Analysis (FCA).

Another work has concentrated on solving the problem of tunneling in focused crawling [35-37]. Basically, focused crawling ignores non-relevant webpages and their outgoing URLs. Although this is necessary for saving resources, it is sometime the case where relevant webpages are linked through non-relevant one. So we need to use the non-relevant webpages, based on the fact that we may reach to a relevant webpages through them. This is approached by having a limit on the number of consecutive non-relevant to be included before ignoring them. For example, if we set this limit to 3, then we will use the URLs of the first non-relevant webpage and if one of them is non-relevant we will use the URLs in its webpage too. If in the last set of URLs we have non-relevant webpages, we will not include their URLs in next phases as we have reached the limit of consecutive non-relevant webpages to be included.

Most of the work evaluates the performance of focused crawling by measuring the precision and recall of their collection. There are a lot of measures to be used for evaluation. Using these measures for evaluation depends on the purpose of using the focused crawling and the application it is used for. A lot of studies have been made [27, 38-42] for developing a general evaluation framework for focused crawling.

3. PROBLEM STATEMENT

Given a set of webpages that describe (represent) the topic of interest, how can we build a model that can distinguish between relevant and non-relevant webpages in a modular architecture? Using classification techniques and collections built using IA[43], we will build classifier-based focused crawler prototype. The data set will be divided into training and test data. The test data will be used for evaluation. The focused crawler will also be tested on real unseen data.

4. IMPLEMENTATION

4.1 Details

The Focused Crawler is implemented in Python , general-purpose, interpreted high-level programming language. Our project makes extensive use of NLTK and Scikit-learn. NLTK is a leading platform for building Python programs to work with human language data. Scikit-learn is a Python module integrating classic machine learning algorithms in the tightly-knit scientific Python world (numpy, scipy, matplotlib).

Pre-requisites:

System Requirements

- Python Interpreter 2.7
- Python libraries: ntlk, scikit-learn, gensim, and beautifulsoup
- External software: Hanzo warc tools[1]

Project Requirements

- Crisis, Tragedy and Recovery network (CTRnet) collections for seed URLs
- Sikkim Earthquake collection

Installation

You need to have python 2.7 with the dependencies mentioned above installed. You also will need to download Hanzo warc tools for extracting warc files of IA collections. Once all these are installed you can run the focused crawler file using the following command:

Python FocusedCrawler.py

You need to change some parameters if you want to use the program for your purpose. You have to specify the seed URLs and the training data for classification. The training data is specified using two text files, one is called "html_files.txt" and the other is called "labels.txt". The first one stores the path of the files on the disk that will be used for training. The other file stores the class labels for each corresponding document in the "html_files" file. You have also to specify the number of webpages to download.

File Inventory

1- Focused crawling framework files

- FocusedCrawler.py (the main file)
- crawler.py (crawling module)
- Filter.py (filtering training data to positive (relevant) and negative (non-relevant) documents)
- NBClassifier.py (Naïve Bayes classifier)
- priorityQueue.py (priority queue for storing URLs)
- scorer.py (default relevance estimation module)
- SVMClassifier.py (SVM classifier)
- tfidfScorer.py (TF-IDF based relevance estimation)
- url.py (URL representation)
- webpage.py (webpage representation)

4.2 System Architecture

The Focused Crawler begins with one or more URLs that constitute the seed set. The crawler fetches the web page in the seed URL and it is parsed. The extracted text is then represented as vector of identifiers using the normalized TF weighting scheme and it serves as the topic vector (vector space model for the set of documents). This is the training phase of the crawler.

In the testing phase, the web page is fetched at the URL. The web page is parsed, to extract both the text and the links from the page. The Figure 1 shows the baseline crawler implemented using cosine similarity in the top section and further enhanced by adding classifier models in the bottom section for relevance estimation.

[1] http://code.hanzoarchives.com/warc-tools/wiki/Home

Baseline crawler implementation

The cosine similarity (measure of similarity between two vectors) of the webpage vector is determined for relevance estimation. If relevant, the extracted links is added to the URL frontier which is implemented as a priority queue, which at all times consists of links whose corresponding pages are yet to be fetched by the crawler. The entire process is recursive may be viewed as traversing the web graph and there should no duplications in the queue.

Classifier Models for Relevance Estimation:

CTRnet

Crisis, Tragedy and Recovery network (CTRnet) [44], is a digital library network for providing a range of services relating to different kind of tragic events. CTRnet collections about school shootings and natural disasters have been developed from collaboration with the Internet Archive.

Support Vector Machine and Naive Bayes

CTRnet collection is filtered to get the training collections and it is pre-processed to get the collection vectors and is fed to the classifier which generates a model which is used for relevance estimation. The classifier models that are being supported are Support vector machines and Naïve Bayes classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model"

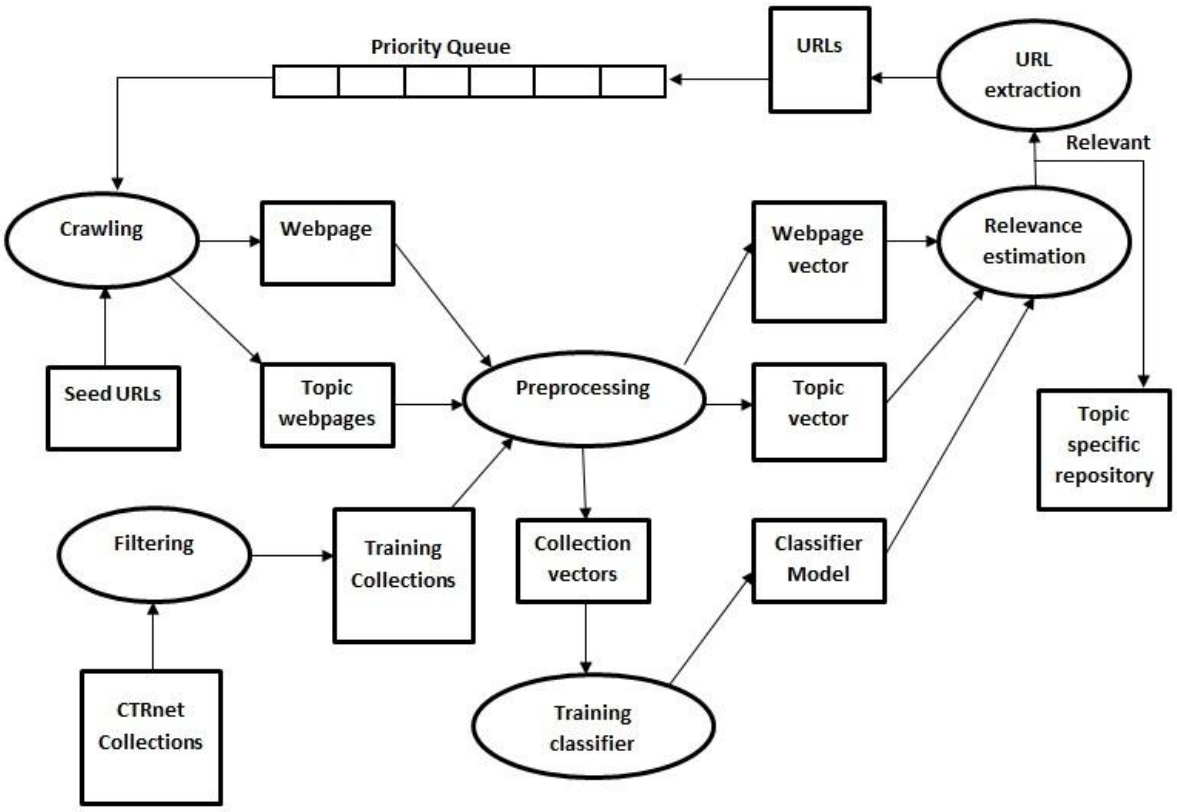

Figure 1: Focused Crawler Architecture

4.3 Key Modules

The details of the key modules are as follows:

4.3.1 Crawling

The crawling module includes the following modules

- The URL frontier, containing URLs yet to be fetched in the current crawl .It is implemented as a priority queue and the priority is determined by the relevance score.
- A DNS resolution module that determines the web server from which to fetch the page specified by the URL.
- A fetch module that uses the http protocol to retrieve the web page at the URL.
- A duplicate elimination module that determines whether an extracted link is already in the URL frontier or has recently been fetched.
- **The output of the crawler (topic-specific repository in Figure 1) will be a repository of webpages that are relevant to the topic of interest.**

4.3.2 Fetching data from World Wide Web

- The urllib module provides a high-level interface for fetching data across the World Wide Web through built in functions like open() , urlopen() etc
- The html5lib is a Python package that implements the HTML5 parsing algorithm which is heavily influenced by current browsers and based on the WHATWG HTML5 specification.

4.3.3 URL Extraction

- Get the linked URLs for the web page (A HREF Tag for getting hyper-links) and generate a python list.
- Amend relative paths by absolute paths.
- BeautifulSoup provides a few simple methods for navigating, searching, and modifying a parsed HTML DOM tree.

4.3.4 Preprocessing

The preprocessing module includes the following tasks

- Tokenize the text from the web page
- Remove stop words and stem the tokens
- Lemmatize the different inflected forms of a token
- Calculate the normalized term frequency and the inverse term frequency for the terms in the collection

4.3.5 Tokenizing Text

- NLTK supports Porter Stemming Algorithm and Lancaster Stemming Algorithm
- Remove punctuations ,dashes and special characters
- Apply NTLK's tokenize procedure to obtain tokens
- Stem the tokens for removing and replacing word suffixes to arrive at a common root form of the word.
- Remove Stop Words from the stemmed tokens.
- Lemmatize the tokens, lemmas differ from stems in that a lemma is a canonical form of the word

4.3.6 TF-IDF

- Calculate the frequency of occurrence of stemmed tokens in the document
- Compute the normalized term frequency
- Compute the TF-IDF (Term frequency , Inverse Document Frequency) for all the terms in the corpus.

4.3.7 Classifier Models

The classifier module includes the following tasks

- Takes inputs from the CTRnet collection
- Feature Selection: TF-IDF for terms in the collection to get the collection vectors.
- Train and Build a model
- Use the model to do the relevance estimation of the web page.

The models that are being supported are Support vector machines and Naïve Bayes classifier

Support Vector Machine

- Non-Probabilistic Approach
- Large Margin Classifier
- Linear/ Non-Linear

Naive Bayes

- Uses Bayes Theorem with strong independence assumptions
- Probabilistic Approach

4.3.8 Relevance Estimation

The Relevance Estimation module includes the following tasks

- Determine the relevance of the web page using the topic specific repository.
- The web page is given a relevance score of 0 or 1
- Scikit is used to determine the precision, recall, F1 score and support
- These are determined by using methods like metrics.precision_score()

4.3.9 Filtering

The filtering phase includes preparing training data for the classifier.
They are

- Warc files build by IA. Extracted using Hanzo warc tools
- We used Sikkim earthquake collection.
- Seed URLs ~2000 HTML files out of 9000 files
- Keywords, selected manually. Relevance →if k or more words from keywords
- K = 1 ~50% relevance (high recall, low precision)
- We used k = 5

5. EXPERIMENTAL METHODOLOGY

The scope of our experimental methodology encompasses all modules. We desire to test the effectiveness of each module in order to achieve the desired results.

In case of Classifier, we will test out baseline classifier and use the results to evaluate the other classifiers - Naïve Bayes and Support Vector Machine (SVM). We used the same training (Sikkim Earthquake) for the baseline as well as the other classifiers and calculate the precision, recall and f1-score for each of them respectively. This approach will lead us to choose the classifier model that suits

best with our proposed focused crawler.

For the baseline crawler we get the seed URLs and build a topic vector using TF-IDF. The crawler will extract URLs by traversing the seed URLs. Once these URLs are extracted, we determine their relevance by comparing it with the topic vector using cosine similarity.

The performance of the classifier is evaluated on basis of the test data. We have used cross validation to select the best parameters. We have used Cross Validation to select the best parameters for feature selection. Ordering of URLs in the Priority Queue is also an important task as we place the those urls higher in the queue which leads us to more relevant pages.

In order to test our classifiers we have used Automatic Filtering. We test our Naïve Bayes and Support Vector Machine (SVM) classifier with and without feature selection. For SVM, we used Chi-square feature selection.

6. EVALUATION

The baseline crawler was initially tested with the Egyptian Revolution (2011) data. We set the threshold for the URLs as 0.1, **Precision** = **0.52**

Table 1: Baseline focused crawler performance for Egyptian revolution

URL	Score
http://botw.org/top/Regional/Africa/Egypt/Society_and_Culture/Politics/Protests_2011/	1
http://live.reuters.com/Event/Unrest_in_Egypt?Page=0	1
http://www.aljazeera.com/indepth/spotlight/anger-in-egypt/	1
http://www.guardian.co.uk/world/series/egypt-protests	1
http://www.huffingtonpost.com/2012/06/24/egypt-uprising-election-timeline_n_1622773.html	1
http://www.washingtonpost.com/wp-srv/world/special/egypt-transition-timeline/index.html	1
http://www.guardian.co.uk/news/blog/2011/feb/09/egypt-protests-live-updates-9-february	0.5242834
http://www.guardian.co.uk/world/blog/2011/feb/05/egypt-protests	0.50552904
http://www.guardian.co.uk/world/blog/2011/feb/11/egypt-protests-mubarak	0.50212776
http://www.guardian.co.uk/news/blog/2011/feb/08/egypt-protests-live-updates	0.47775149

The first six URLs are the seed URLs which have been assigned the relevance score of 1, from these seed URLs we gather collect the remaining URLs in the above table. The relevance score these URLs are given in the table. The total URLs are 10 due to the fact that we have set the page limit to 10 to till the Priority queue get empty. The precision we achieved for this set of data is as follows:

For Sikkim Earthquake collection we had the following results:

For Baseline approach, **Precision = 0.15**

URL	Score
http://articles.timesofindia.indiatimes.com/2011-09-21/india/30184028_1_construction-site-teesta-urja-gangtok	1
http://earthquake-report.com/2011/09/18/very-strong-earthquake-in-sikkim-india/	1
http://www.ndtv.com/topic/sikkim-earthquake	1
http://zeenews.india.com/tags/Sikkim_earthquake.html	1
http://zeenews.india.com/entertainment/gallery/bipasha-scorches-maxims-cover_1729.htm	0.598282933235
http://zeenews.india.com/blog/guns-n-roses-play-on-sweet-idols-o-mine_5901.html	0.530184864998
http://zeenews.india.com/entertainment/gallery/bollywood-shines-at-marrakech-film-festival_1716.htm	0.499703705311
http://zeenews.india.com/exclusive/india-vs-england-2012-can-dhoni-s-men-give-it-back-to-the-english-team_5833.html	0.499164909124
http://timesofindia.indiatimes.com/topic/National-Disaster-Relief-Force	0.490525960922
http://zeenews.india.com/blog/fifty-shades-of-grey-the-lobbyists-and-service-providers-of-india_5899.html	0.490444749594

For SVM-based focused crawler, **Precision = 0.69**

URL	Score
http://earthquake-report.com/2011/09/18/very-strong-earthquake-in-sikkim-india/	1
http:// earthquake-report.com/2011/09/18/very-strong-earthquake-in-sikkim-india/	1
http://www.ndtv.com/topic/sikkim-earthquake	1
http://zeenews.india.com/tags/Sikkim_earthquake.html	1
http://www.ndtv.com/topic/quake-in-sikkim	0.99750957
http://www.ndtv.com/topic/earthquake-in-sikkim	0.99112698
http://earthquake-report.com/nepal-sikkim-himalaya-1809/	0.99104695
http://www.ndtv.com/topic/sikkim-quake	0.98757532
http://earthquake-report.com/2011/11/25/earthquake-induced-landslides-in-the-sikkim-darjeeling-himalayas/	0.94213802
http://www.ndtv.com/article/india/earthquake-in-sikkim-42-dead-across-india-134783	0.86038476

For Support Vector Machine (SVM), the evaluation results are as follows:

1. Without feature selection, k = 5 for filtering ('k' denotes the number of words form the list of keywords built manually for Automatic Filtering)

Accuracy: 0.845

Table 2: SVM Classifier performance, without feature selection

Document	Precision	Recall	F1-Score	Support
0	0.85	1.00	0.92	191
1	0.00	0.00	0.00	35
Avg/Total	0.71	0.85	0.77	226

'0' denotes negative documents, whereas '1' denotes positive documents. As we can see from the above table we were unable to get the positive documents. This due to the noisy data present in urls.

2. With Chi-Square feature selection, k = 5 for filtering

Accuracy: 0.898

Table 3: SVM Classifier performance, with feature selection

Document	Precision	Recall	F1-Score	Support
0	0.89	1.00	0.94	191
1	1.00	0.34	0.51	35
Avg/Total	0.91	0.90	0.88	226

As we can see from the above table we were able to get the positive documents. Thus, SVM with feature selection (Chi-Square) performs better that without feature selection.

Similarly, we used the same data and methodology to test the Naïve Bayes classifier. The performance of NB classifier was same with and without feature selection and k = 5 for filtering.

Table 4: Naive Bayes classifier performance

Document	Precision	Recall	F1-Score	Support
0	0.85	1.00	0.92	191
1	0.00	0.00	0.00	35
Avg/Total	0.71	0.85	0.77	226

The table above clearly shows that we were unable to fetch positive documents form the Sikkim earthquake collection. This is also due to the noisy data present in urls. One approach to resolve this using manual feature selection, i.e., to get the positive documents and find the most frequent words and use them as features to classify the documents.

7. ISSUES AND CHALLENGES

We faced some issues related to test data setup. These were related to Warc extraction and are listed below:

- Files saved on disk, original URL, wayback machine URL
- Encoding problems of saved on disk files
- Original URLs, not working (page not found)
- Wayback machine URLs, web pages have extra content that needs to be parsed and removed

We also faced issues related to text extraction, like

- Extract visual text of a web page
- Heuristic approaches
- Scripts and comments tags remain after
- Extraction (need explicit manipulation)
- Invalid HTML tags (missing closing brackets)

8. FUTURE WORK

Our framework can be extended to handle using ontologies as domain representation and providing modules for using ontologies in relevance estimation. More classification algorithms can be examined like logistic regression, K-NN, and tree-based algorithms. Link context can also be included in relevance estimation. New modules need to be included for building link context for different approaches.

Our framework accepts html files extracted from warc files. We can add modules for handling input in different formats.

We can also make use of evaluation frameworks for evaluating our focused crawler on different evaluation measures.

9. CONCLUSION

Generic crawlers and search engines are like public libraries: they try to cater to everyone, and do not specialize in specific areas. Serious web users are increasingly feeling a need for highly specialized and filtered 'university research libraries' where they can explore their interest in depth. Unlike public libraries, web libraries have little excuse not to specialize, because it is just a matter of locating and linking to resources.

We have demonstrated that goal-directed crawling is a powerful means for topical resource discovery. The focused crawler is a system that learns the specialization from examples, and then explores the Web, guided by a relevance and popularity rating mechanism. It filters at the data-acquisition level, rather than as a post-processing step. Our system selects work very carefully from the crawl frontier. A consequence of the resulting efficiency is that it is feasible to crawl to a greater depth than would otherwise be possible. This may result in the discovery of some high-quality information resources that might have otherwise been overlooked. The focused crawler also gives better user experience as it tries to give results which are more relevant to user’s information needs, thus leading to user satisfaction

which is one of the parameters measuring success of IR system. The quantity of data on web is increasing exponentially so it is critical for IR system to be very efficient while handling user queries which normally require high precision, thus focused crawler plays a crucial role over here.

We showed that SVM-based focused crawler with feature selection is more suitable for our approach. The SVM focused crawler performed better than the Naïve Bayes classifier, based on the evaluation results shown above, and better than the baseline approach (TF-IDF with cosine similarity).

10. ACKNOWLEDGEMENT

We would like to thank Dr. Fox for his continuous support and all our colleagues in the class for their comments and feedback.

11. REFERENCES

[1] M. Theobald, R. Schenkel, and G. Weikum, "Classification and focused crawling for semistructured data," *Intelligent Search on XML Data*, pp. 145-157, 2003.

[2] C. Li, L. Zhi-shu, Y. Zhong-hua, and H. Guo-hui, "Classifier-guided topical crawler: a novel method of automatically labeling the positive URLs," presented at the Proceedings of the 5th International Conference on Semantics, Knowledge and Grid (SKG), Zhuhai, China, 2009.

[3] H. Liu, E. Milios, and J. Janssen, "Focused Crawling by Learning HMM from User's Topic-specific Browsing," presented at the Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI), Beijing, China, 2004.

[4] T. K. Shih, "Focused crawling for information gathering using hidden markov model," Master's thesis, Computer Science and Information Engineering, National Central University, Taiwan, 2007.

[5] S. Chakrabarti, M. Van den Berg, and B. Dom, "Focused crawling: a new approach to topic-specific Web resource discovery," *Computer Networks*, vol. 31, pp. 1623-1640, 1999.

[6] Y. Ye, F. Ma, Y. Lu, M. Chiu, and J. Z. Huang, "iSurfer: A focused web crawler based on incremental learning from positive samples," presented at the Advanced Web Technologies and Applications, 2004.

[7] G. Pant and P. Srinivasan, "Link contexts in classifier-guided topical crawlers," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, pp. 107-122, 2006.

[8] I. Partalas, G. Paliouras, and I. Vlahavas, "Reinforcement learning with classifier selection for focused crawling," presented at the Proceedings of the 18th European Conference on Artificial Intelligence (ECAI) Amsterdam, The Netherlands, 2008.

[9] H. Zhang and J. Lu, "SCTWC: An online semi-supervised clustering approach to topical web crawlers," *Applied Soft Computing*, vol. 10, pp. 490-495, 2010.

[10] F. Menczer, G. Pant, and P. Srinivasan, "Topic-driven crawlers: Machine learning issues," *ACM TOIT*, 2002.

[11] D. Hati, A. Kumar, and L. Mishra, "Unvisited URL Relevancy Calculation in Focused Crawling Based on Naïve Bayesian Classification," *International Journal of Computer Applications IJCA*, vol. 3, pp. 23-30, 2010.

[12] S. Sizov, S. Siersdorfer, M. Theobald, and G. Weikum, "The BINGO! focused crawler: From bookmarks to archetypes," presented at the Proceedings of the 18th International Conference on Data Engineering San Jose, CA , USA, 2002.

[13] A. Batzios, C. Dimou, A. L. Symeonidis, and P. A. Mitkas, "BioCrawler: An intelligent crawler for the semantic web," *Expert Systems with Applications*, vol. 35, pp. 524-530, 2008.

[14] S. Thenmalar, "Concept based Focused Crawling using Ontology," *International Journal of Computer Applications*, vol. 26, pp. 29-32, 2011.
[15] S. Chang, G. Yang, Y. Jianmei, and L. Bin, "An efficient adaptive focused crawler based on ontology learning," presented at the Proceedings of the 5th International Conference on Hybrid Intelligent Systems (HIS), 2005.
[16] Z. Zhang, O. Nasraoui, and R. V. Zwol, "Exploiting Tags and Social Profiles to Improve Focused Crawling," presented at the Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01 2009.
[17] G. Almpanidis, C. Kotropoulos, and I. Pitas, "Focused crawling using latent semantic indexing–An application for vertical search engines," *Research and Advanced Technology for Digital Libraries*, pp. 402-413, 2005.
[18] L. Kozanidis, "An Ontology-Based Focused Crawler," in *Natural Language and Information Systems*. vol. 5039, E. Kapetanios, V. Sugumaran, and M. Spiliopoulou, Eds., ed: Springer Berlin / Heidelberg, 2008, pp. 376-379.
[19] S. Ganesh, M. Jayaraj, V. Kalyan, S. Murthy, and G. Aghila, "Ontology-based Web Crawler," presented at the Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC), Las Vegas, Nevada, USA, 2004.
[20] M. Ehrig and A. Maedche, "Ontology-focused crawling of Web documents," presented at the Proceedings of the 2003 ACM symposium on applied computing, Melbourne, Florida, 2003.
[21] A. Maedche, M. Ehrig, S. Handschuh, L. Stojanovic, and R. Volz. (2002). *Ontology-Focused Crawling of Web Documents and RDF-based Metadata*. Available: http://projekte.l3s.uni-hannover.de/pub/bscw.cgi/S4893f6f4/d5269/Maedche_Ehrig-Focused_Crawler-ISWC2002sub.pdf
[22] J. J. Jung, "Towards open decision support systems based on semantic focused crawling," *Expert Systems with Applications*, vol. 36, pp. 3914-3922, 2009.
[23] D. Hati, B. Sahoo, and A. Kumar, "Adaptive focused crawling based on link analysis," presented at the Proceedings of the 2nd International Conference on Education Technology and Computer (ICETC), 2010.
[24] T. Peng, F. He, W. Zuo, and C. Zhang, "Adaptive Topical Web Crawling for Domain-Specific Resource Discovery Guided by Link-Context," in *MICAI 2006: Advances in Artificial Intelligence*. vol. 4293, A. Gelbukh and C. Reyes-Garcia, Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 963-973.
[25] G. Almpanidis and C. Kotropoulos, "Combining text and link analysis for focused crawling," presented at the Proc. (Part 1) Pattern Recognition and Data Mining, Third International Conference on Advances in Pattern Recognition, ICAPR 2005, Bath, UK, August 22-25, Springer LNCS 3686, 2005.
[26] A. Pal, D. S. Tomar, and S. Shrivastava, "Effective Focused Crawling Based on Content and Link Structure Analysis," *Arxiv preprint arXiv:0906.5034*, 2009.
[27] A. Cami and N. Deo, "Evaluation of a Graph-based Topical Crawler," presented at the International Conference on Internet Computing, 2006.
[28] J. Dong, W. Zuo, and T. Peng, "Focused crawling guided by link context," presented at the Proceedings of the 24th IASTED international conference on Artificial intelligence and applications (AIA) Innsbruck, Austria, 2006.
[29] F. M. C. M. Diligenti, S. Lawrence, C.L. Giles, M. Gori, "Focused Crawling Using Context Graphs," in *26th Internaiional Conference on Very Large Databases (VLDB 2000)*, ed, 2000, pp. 527-534.
[30] C. Xiaoyun and Z. Xin, "HAWK: A Focused Crawler with Content and Link Analysis," presented at the Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE), Xi'an, China, 2008.

[31] M. Yuvarani, N. C. S. N. Iyengar, and A. Kannan, "LSCrawler: A Framework for an Enhanced Focused Web Crawler Based on Link Semantics," presented at the Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, 2006.
[32] M. Jamali, H. Sayyadi, B. B. Hariri, and H. Abolhassani, "A method for focused crawling using combination of link structure and content similarity," presented at the Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI), Hong Kong, China, 2006.
[33] C. Wang, Z. Guan, C. Chen, J. Bu, J. Wang, and H. Lin, "On-line topical importance estimation: an effective focused crawling algorithm combining link and content analysis," *Journal of Zhejiang University-Science A*, vol. 10, pp. 1114-1124, 2009.
[34] Y. Yang, Y. Du, J. Sun, and Y. Hai, "A Topic-Specific Web Crawler with Concept Similarity Context Graph Based on FCA," *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, pp. 840-847, 2008.
[35] N. Luo, W. L. Zuo, and F. Y. Yuan, "Gray Tunneling Based on Block Relevance for Focused Crawling," presented at the Proceedings on Intelligent Systems and Knowledge Engineering, ISKE2007 2007.
[36] N. Luo, W. Zuo, F. Yuan, and C. Zhang, "A New Method for Focused Crawler Cross Tunnel," in *Rough Sets and Knowledge Technology*. vol. 4062, G.-Y. Wang, J. Peters, A. Skowron, and Y. Yao, Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 632-637.
[37] T. Peng, C. Zhang, and W. Zuo, "Tunneling enhanced by web page content block partition for focused crawling," *Concurrency and Computation: Practice and Experience*, vol. 20, pp. 61-74, 2008.
[38] T. Peng, W. Zuo, and Y. Liu, "Characterization of Evaluation Metrics in Topical Web Crawling Based on Genetic Algorithm," in *Advances in Natural Computation*. vol. 3611, L. Wang, K. Chen, and Y. Ong, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 432-432.
[39] P. Srinivasan, F. Menczer, and G. Pant, "Defining evaluation methodologies for topical crawlers," presented at the Proceedings of the SIGIR Workshop on Defining Evaluation Methodologies for Terabyte-Scale Collections, Toronto, Canada, 2003.
[40] P. Srinivasan, F. Menczer, and G. Pant, "A general evaluation framework for topical crawlers," *Information Retrieval*, vol. 8, pp. 417-447, 2005.
[41] T. Peng, W. L. Zuo, and Y. L. Liu, "Genetic Algorithm for Evaluation Metrics in Topical Web Crawling," in *Computational Methods*, G. R. Liu, V. B. C. Tan, and X. Han, Eds., ed: Springer Netherlands, 2006, pp. 1203-1208.
[42] F. Menczer, G. Pant, and P. Srinivasan, "Topical web crawlers: Evaluating adaptive algorithms," *ACM Transactions on Internet Technology (TOIT)*, vol. 4, pp. 378-419, 2004.
[43] Internet_Archive. (2000). *Internet Archive* [Web page]. Available: http://www.archive.org
[44] CTRnet. (2011). *Crisis, Tragedy, and Recovery Network website*. Available: http://www.ctrnet.net/