FOCUSED WEB CRAWLING USING NAMED ENTITY RECOGNITION FOR NARROW DOMAINS

Sameendra Samarawickrama[1], Lakshman Jayaratne[2]

University of Colombo School of Computing, 35, Reid Avenue, Colombo, Sri Lanka
smsamrc@gmail.com, klj@ucsc.cmb.ac.lk

Abstract

Within recent years the World Wide Web (WWW) has grown enormously to a large extent where generic web crawlers have become unable to keep up with. As a result, focused web crawlers have gained its popularity which is focused only on a particular domain. But these crawlers are based on lexical terms where they ignore the information contained within named entities; named entities can be a very good source of information when crawling on narrow domains. In this paper we discuss a new approach to focus crawling based on named entities for narrow domains.

We have conducted experiments in focused web crawling in three narrow domains: baseball, football and American politics. A classifier based on the centroid algorithm is used to guide the crawler which is trained on web pages collected manually from online news articles for each domain. Our results showed that during anytime of the crawl, the collection built with our crawler is better than the traditional focused crawler based on lexical terms, in terms of the harvest ratio. And this was true for all the three domains considered.

Index Terms: *web mining, focused crawling, named entity, classification*

----------***----------

1. INTRODUCTION

As of today, the indexed web contains billions of web pages and continues to grow at a rapid pace. With such a large web space, exhaustive crawling has almost become impossible with traditional general purpose web crawlers. Even the largest crawlers are not capable of crawling the whole web. Also their requirements are very demanding in network bandwidth and storage if such a task is to be carried out.

As a result a new research area called focused web crawling emerged, where the crawler is specifically designed to crawl only a subset of the web graph that is of interest. In other words, focused crawler crawls the web in a particular domain (e.g., sports, health, politics)

1.1. Motivation

Web searching has been evolving ever since the start of the World Wide Web (WWW). Various searching techniques have been introduced since then to increase the popularity of web the among people. Web crawlers play a vital role inside a search engine, such as finding new pages to be indexed, periodically recrawling and updating the index with fresh information etc.

It is well known that general purpose search engines are not tailored at providing topic specific information. There are times we get irrelevant information or information at best marginally relevant. As a result, vertical search engines (a.k.a. topical search engines) have gained its popularity in recent years which are specifically designed to provide topic specific information. Focused crawler can be considered as the key component in a vertical search engine where it attempts to collect web pages relevant to a particular topic of interest, while filtering out the irrelevant.

In the same way, focused crawlers can be used to generate data for an individual user or a community in their interested topic or automated building of web directories like Yahoo![1], Open Directory Project[2] which still uses human expertise for the categorization.

Since the focused crawlers filters out pages and crawls only a subset of the web, it saves network bandwidth and space. They are also capable of giving a more up to date crawl since the crawling is focused; detection of changes are far more nimble than generic crawlers. A typical focused crawler meets with three major challenges: (i) needs to determine the relevance of a retrieved web page (ii) should predict and identify relevant Uniform Resource Locators (URLs) that can lead to topic relevant pages (iii) ranking and ordering the relevant URLs so that the crawler knows exactly what to follow next.

[1] http://www.yahoo.com

[2] http://www.dmoz.org

1.2. What is a Domain?

In web and data mining, a domain can be loosely defined as information specializing a particular area of interest. Domains can exist for broad areas as well as narrow areas, named as broad domains and narrow domains (closed domains) respectively. Broad domain describes a topic in general while narrow domain describes a topic more specifically and detail. Often, a broad domain can be thought as a collection of narrow domains. For instance, Sports and Music can be considered as broad domains while relatively, baseball, football and Sri Lankan music can be thought as narrow domains.

1.3. Focused Crawling Terminology

In the literature of focused crawling the term *harvest ratio* [5] comes as the primary metric in evaluating the crawler's performance. It measures the rate at which relevant pages are fetched and how effectively irrelevant pages are kept out of the crawl; harvest ratio, *H(t)*, after crawling first *t* pages is computed as:

$$H(t) = \frac{1}{t}\sum_{i=1}^{t} r_i$$

where r_i is 1 if page *i* is considered relevant or 0 otherwise. Usually, harvest ratio is computed at different points during the crawl to obtain a trajectory of crawler performance.

Authorities are pages that are rich and relevant in content to the user specified topic. *hubs* are places which have many links to other pages which might contain relevant information. Jon Kleinberg - the introducer of these two terms - argued that hubs and authorities exhibit a mutually reinforcing relationship [6] i.e. a good hub will point to many authorities and a good authority will be pointed at by many hubs. So in a focused crawler authorities should definitely sent to be indexed while correctly processing hubs to find authorities.

crawler frontier contains the list of URLs to be crawled.
A page from which a link was extracted is called the *parent page* and the page pointed by the url is called the *child page* or the *target page*.

Due to the complexity of web an irrelevant web page might refer to a highly relevant page. In this case the crawler has to traverse the irrelevant page to get the relevant web pages. This process is called the *tunneling*.

Seed set is the set of URLs that are known to be highly relevant to the particular topic of interest. It is collected manually or by the crawler with the help of an existing search engine/portal.

Figure 1 shows the usual variation of the harvest ratios for general and focused crawlers. It is clear that focused crawlers should always maintain a high harvest ratio than generic crawlers.

Figure 2 gives the high level overview of a simple focused crawler. URL downloader downloads web pages from WWW initiated with the seed URLs and sends them to the classifier. Classifier which is trained with the help of seed set makes relevance judgments on the pages it receives. URLs extracted from these relevant pages will be added to the crawler frontier thus to continue with the crawling process. URL downloader maintains a database (URL DB) of crawled pages. Upon retrieving a URL from the crawler frontier it will first check the URL DB to see whether that page has already been downloaded or not. URL DB may contain a local copy of the downloaded pages and will also serve the indexing process.

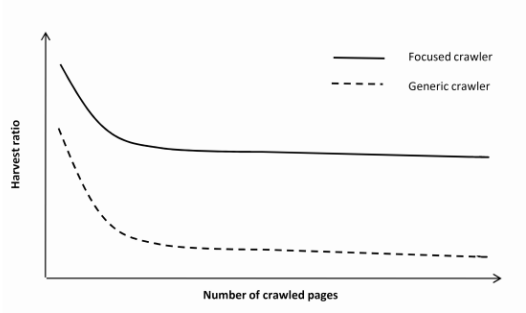

Fig - 1.1: Harvest ratio variation for focused and generic crawler

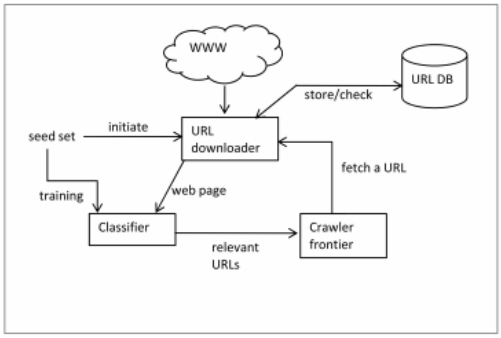

Fig - 1.2: System overview of a simple focused crawler

2. RELATED WORK

Early work on focused crawling was based on simple keyword matching, regular expression matching or binary classifiers. De Bra et al. [1] proposed the Fish-search algorithm in which, crawling is simulated by a *group of fish* migrating the web. Each URL corresponds to a fish whose survivability is dependent on visited page relevance and remote server speed. Page relevance is estimated using a binary classification by using simple keyword or regular expression matches. After

traversing a certain number of irrelevant pages, fish dies. Hersovici et al. [2] improves this algorithm into Shark-search in which the page relevance is calculated as a similarity between document and query in Vector Space Model (VSM). Cho et al. [3] proposed calculating the PageRank score on the graph induced by crawling the web and then using this score as a priority of URLs for the next crawl (about prioritizing the crawl frontier).

The two main approaches to modern focused crawling are based on content analysis and link structure analysis. Focused crawling based on content analysis is heavily dependent of automatic text classification techniques to determine the relevance of a retrieved web page to the crawling domain. Since web pages are noisy, these crawlers employ different noise removal techniques to extract only the useful textual content. Link analysis based approaches build a web graph and this graph is used to identify potential URLs that can lead to topic relevant pages. The underlying premise of them is the topical locality of web [4].

Chakrabarti et al. [5] was the first to utilize machine learning into focused crawling. They used an existing document taxonomy (Yahoo!) to train a Naive Bayes classifier and to classify retrieved web pages into categories. Use of a taxonomy helps in better modeling of irrelevant pages (the negative class). Distiller, another component of their crawler, identifies hub pages [6] pointing to many topic relevant pages. Naive Bayes classifier has been used in other topical crawlers as well [7], [8], [9] to name a few.

Assis et at. [10] use genre related information as well as the content related information. Crawler analyses a web page to identify the terms that correspond to genre and the topic separately, in which a traditional crawler doesn't identify as separate. Bazarganigilani et al. [11] propose a novel approach which uses genetic programming (GP) to efficiently discover the best similarity function which is a combination of Bag-of-words, Cosine, Okapi similarity measures. This is achieved with the fitness function used by the genetic algorithm.

FOCUS [12] is based on link structure analysis which uses link distance (how close a particular URL to the seed URLs) to rank pages directly rather than using a classifier to determine page relevance. Web is modeled as a layered graph with seed URLs as the first layer and their backlinks forming other layers. They use an ordinal regressor to find the corresponding rank of a newly fetched web page. In [13], Liu et al. make use of user's topic specific browsing patterns in predicting links leading to relevant pages based on Hidden Markov Model. First, users' browsing patterns are used to build a web graph, then this graph is clustered and the link structure among pages from different clusters is used to learn patterns which lead to topic specific pages.

3. NAMED ENTITY RECOGNITION

Named Entities (NE) are phrases that contain names of persons, organizations, locations, numeric expressions including time, date, money, percent expressions and names of other miscellaneous things. NER is an important task in many NLP tasks.

There are three approaches to NER[3]: (a) *rule based NER:* is the earliest approach to NER. It detects NEs by writing regular expressions to suit the user needs. However rule based NER is neither robust nor portable; (b) *Dictionary based NER:* where we have a complete list of NEs (dictionary) and the identifying NEs is about searching the dictionary and finding matches; (c) *Statistical NER*: recent research on NER is focused on statistic-based, machine learning approaches where a classifier is trained with human annotated phrases. Some relevant classification algorithms include HMMs [14], Maximum Entropy (ME) [15], Transformation Based error-driven Learning (TBL) [16], SVMs [17] Conditional Random Fields (CRF) [18] etc. The ability to recognize previously unknown entities is an essential part of machine learning based approaches.

Features help learning algorithms in identifying phrases containing named entities. Features can be divided into 3 categories namely, word level features, list lookup features and document and corpus features [19]. Word level features are related to the character makeup of words. Cases, punctuation, morphology, part of speech are to name some commonly used word level features.

List lookup features involves referring a list (also called gazetteer, lexicon and dictionary) containing common words occurring in named entities (e.g., "association" mostly relates to organization names), ambiguous words that can be named entities or lists containing famous named entities. Document and corpus features are defined based on both document content and document structure. This may include features like document meta information, frequency of words and phrases in the corpus and entity co-reference and alias among several others.

4. SYSTEM DESIGN

Our system consists of mainly 5 modules. Web pages are downloaded and their content is extracted. The extracted text and linguistic features are then fed to a classifier which determines if a page is relevant or not. If it is considered relevant, all the outlinks from that page is extracted and the crawling is continued. Figure 4.1 demonstrates the architecture

[3] http://alias-i.com/lingpipe/demos/tutorial/ne/read-me.html

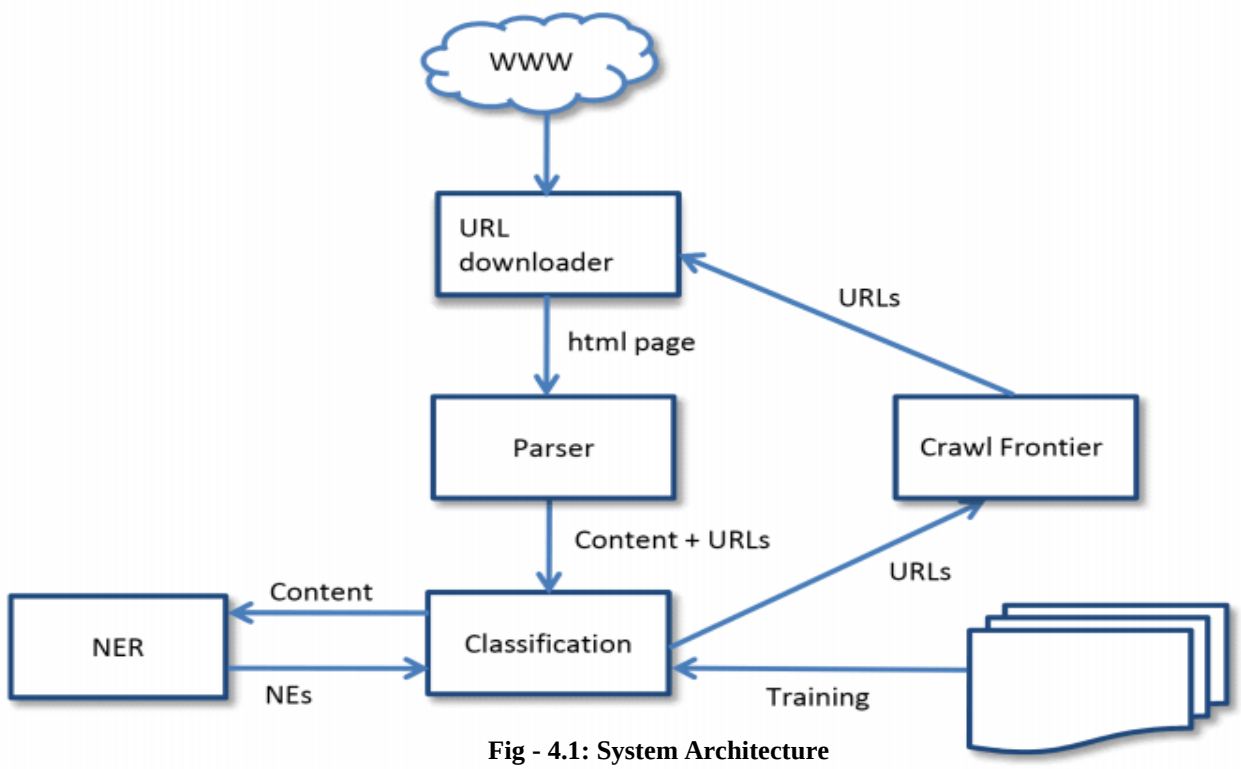

Fig - 4.1: System Architecture

of our system. We have used crawler4j[4], which is a Java based open source web crawler as the baseline crawler and custom modules are added to it to make it a focused crawler. In upcoming subsections we briefly discuss the functionality of these modules.

URL Downloader: downloads html pages from the web. Initially, the downloading is started with the user provided seed URLs and later refers the Crawler Frontier to get URLs for crawling. Implemented as a multi threaded application, makes web page downloading faster. URL Downloader is also responsible for enforcing certain ethics that prevents sending requests to the same server too frequently, thus it waits 500 milliseconds between 2 requests. It also respects the Robots Exclusion Protocol which allows site owners to give specific instructions to web crawlers whether the site is crawlable or only some parts.

Crawl Frontier: is the data structure which contains the URLs to be crawled and is implemented as a priority queue. All the URLs of a page will be added to the Crawl Frontier if that page is considered relevant at the classification stage. URL is assigned a relevance score which is the same score of its parent page (page where the URL is extracted) as assigned by the classifier. So the URLs extracted from highly relevant pages are given priority and crawled first. Crawl Frontier maintains a database of crawled URLs to make sure that same URL is not crawled twice.

Parser: Web pages are noisy as they contain banners, advertisements, and other sources of unwanted entities. So extracting useful textual content out of them is challenging. Parser extracts the textual content and URLs out of web pages. Parser can deal with typos in html, inconsistent html tags, comments and variety of other html errors. Parser ignores the content inside SCRIPT and STYLE tags in html pages, which do not contain any rich information useful for the classifier. Html META tags contain useful information about a particular page; we consider META title tag and the META description tag.

Named Entity Recognizer (NER): This module extracts NEs appearing on a page it receives. It recognizes NEs in three categories, namely, LOCATION, PERSON and ORGANIZATION. In our research we have used the Stanford NER [20] Java API for this task, Stanford NER uses Conditional Random Fields (CRFs) as its classifier. The model is trained on a mixture of CoNLL, MUC-6, MUC-7 and ACE named entity corpora, and thus the model is fairly robust across domains.

Classification: Classification phase is a very important stage in the system. We have considered both NEs and lexical terms as features for the classifier.

[4] http://code.google.com/p/crawler4j/

In most cases which involve a classifier, requires both positive and negative examples for training the classifier. But in this paper, since we are interested in retrieving information related only to a particular domain, a problem arises what to define as the negative training examples for the classifier. As a solution we have chosen the Centroid algorithm as the classifier and the cosine similarity measure to determine the relevance of a newly crawled web page to the centroid using a threshold value.

Stop word removal is done to improve the classification accuracy and to reduce the dimensionality of the feature vector.

Web page is represented in Vector Space Model (VSM) with TF-IDF weighting.

Centroid Algorithm

Centroid algorithm creates a classifier (centroid) based on the training data set. Then this classifier is used to determine the similarity of a new instance to the centroid using a similarity measure. Here, *S* denotes the set of training documents and *|S|* denotes its size. The centroid, *c*, which is the average of documents belonging to set S, is calculated as follows:

$$c = \frac{1}{|S|} \cdot \sum_{d \in S} d$$

Given a new document d_i, similarity between d_i and centroid c is the cosine of the angle between d_i and c and is computed using the equation below:

$$sim(d_i, c) = \frac{c.d_i}{|c||d_i|}$$

Since $0 \leq sim(d_i,c) \geq 1$, if a threshold value is t is chosen, then the algorithm reports that d_i belongs to that category if and only if $sim(d_i,c) \geq t$.

5. EXPERIMENTAL SETUP

5.1. Data Collection

Training data plays a vital role in any supervised learning model and can directly affect the test results as well. In this paper, the training data has been collected manually from online news providers in 3 domains: baseball, football and American politics. For each category we have collected around 200 web pages for training the classifier.

5.2. Preprocessing

Preprocessing is very important especially when it comes to web pages. Web pages are inherently noisy; to obtain the textual content that crawler is interested in, good parsing techniques are required. In this paper we remove all the JavaScript code inside SCRIPT tags, content within STYLE tags and all the other CSS code appearing on a page. Apart from the text inside BODY which is the most important, we consider the text inside TITLE tag and the META description tag. META keywords tag could also be used but it is ignored here as it can be susceptible to keyword spamming which can mislead the classifier. We remove all the numerical values in the dataset and finally remove all the files less than 400 byes. All the web pages have been converted to UTF-16 character encoding, since having different encodings gives problems when using programmatically.

These preprocessing steps are applied to web pages in the training dataset as well as when running the crawling online, by the Parser module.

5.3. Web Page Representation

In order to apply any machine learning algorithm on the data, we need to vectorize the data in some format. For textual data representation, one of the most widely used representations, bag-of-words is being used here to represent a web page in vector space model with TF-IDF weighting.

Thus a web page *P* is represented as a vector of weights $v_p = w_{1p}, w_{2p}, \ldots, w_{np}$, where w_{ip} is the weight of attribute *i* in page *P* and *n* is the size of the dictionary. TF-IDF weights are computed using the equation below:

$$W_{ip} = log(1 + f_{ip}).log\frac{|C|}{df_i}$$

Where f_{ip} is the number of times the attribute *i* appears on page *P*, *C* is the dataset containing all the web pages and df_i (document frequency) is the number of web pages containing attribute *i*. Here, an attribute might refer a lexical term or a named entity.

5.4. Feature Extraction and Selection

In this section we will discuss about feature extraction and the selection process. Feature extraction is done for 3 cases: (1) lexical terms (2) lexical terms + NEs (3) only NEs.

First case, which is how the features are extracted in most of the existing approaches, consider the tokens as features after splitting with a set of delimiters. The set of delimiters we have used here is - \r\t\n[]<>”?’()!#$%&*+-\/,;:.@ - which will mostly end up with white space separated words in a text and that is considered as the lexical terms (general terms).

For the second case, first we need to recognize the NEs appearing on text before its extraction. For that, we used the popular Stanford NER [20] for the recognition of NEs which is trained on news data and is fairly robust across domains. With the help of this tool, we recognize NEs in 3 classes: PERSONS, ORGANIZATIONS and LOCATIONS. Then we combine NEs that are longer than one word with an underscore sign, so that during tokenization we get all the NEs as well as

general terms. For example, the recognized NE, "New York Yankees" is reformatted to "New_York_Yankees".

For the third case, we consider only the extracted NEs. So that a particular web pages is represented by its NEs only.

Once the features are extracted a dictionary is created. For that we first consider the 4000 highest frequent terms (or NEs or combined) in the dataset after the removal of stop words. In order to speed up training, save time and for improved accuracy, it is common to remove terms that are likely not being informative or useful, which is called as the feature selection. There are different feature selection techniques are available for textual data classification but in the paper, Information Gain (IG) feature selection technique to reduce the dimensionality of dictionary to 2000 items with the highest information gain score. IG is proven to be a very good feature selection technique along with CHI square and document frequency thresholding [21]

6. RESULTS

For each domain, we have run 3 crawlers -- focused crawler based on lexical terms, focused crawler based on lexical terms combined with NEs and a Breadth First Crawler (BFC) – parallely starting with the same set of seed URLs.

BFC, which is a general crawler, was also run along with focused crawlers for the purpose of comparing focused crawling to general crawling.
Seeds in each case were considered the root web pages in each category.

As there is an infinite number of candidates to seed URLs, for each domain we have selected a few URLs, which we considered relevant to the domain. Seed URLs should be strong enough to initiate the crawl (many outlinks), should be rich in textual content and also be relevant to the crawling domain.

Harvest ratio has been used as the primary metric to evaluate the crawler performance which has been calculated at constant intervals during the crawl. Experimental results are based on the same classifier (Centroid classifier) which was used to guide the crawler.

6.1. Focused Web Crawling in Baseball Domain

Crawling was started with the following 2 seed URLs: http://www.nydailynews.com/sports/baseball, http://www.lfpress.com/sports/baseball/, and continued crawling for around 2000 web pages for each case. Figure 6.1 shows the variation of the harvest ratio with number of pages crawled. Figure 6.2 shows the number of topic relevant pages downloaded with number of pages crawled.

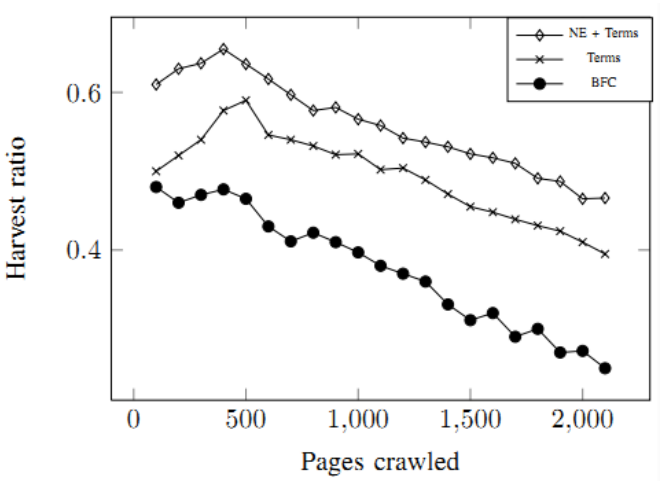

Fig - 6.1: Harvest ratio variation for baseball domain

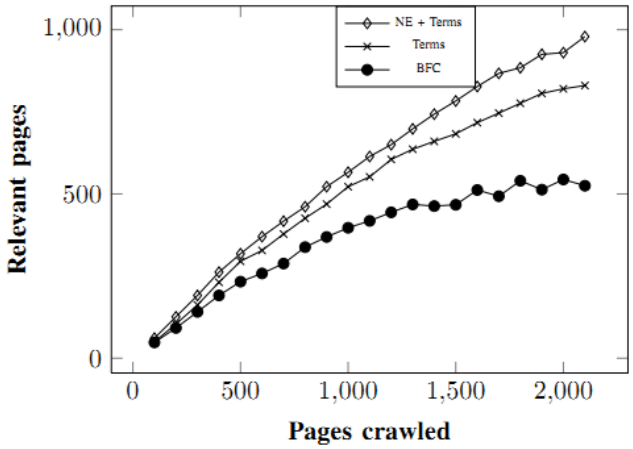

Fig - 6.2: Variation of number of pages downloaded for baseball domain

6.2. Focused Web Crawling in Football Domain

Second experiment was to investigate the effect of NEs in focused crawling related to football domain. Crawling was started with the following seed URLs: http://www.latimes.com/sports/football/nfl/, http://www.cbssports.com/nfl. Starting with these URLs, crawling was continued for about 1500 web pages. Figure 6.3 shows the variation of the harvest ratio with the number of pages crawled. Figure 6.4 shows the number of relevant pages downloaded against the total number of pages downloaded.

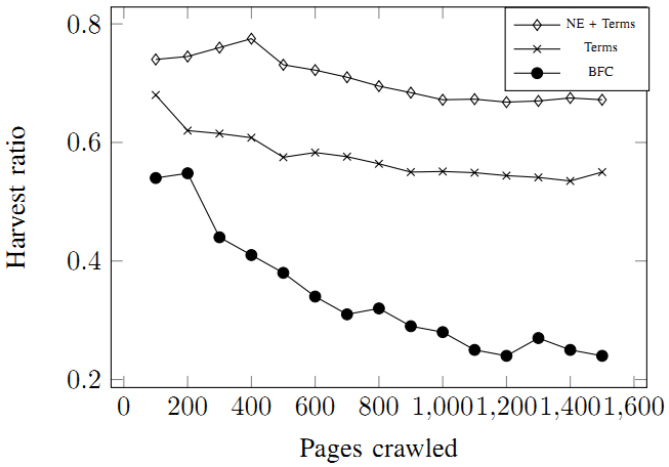

Fig - 6.3: Harvest ratio variation f or football domain

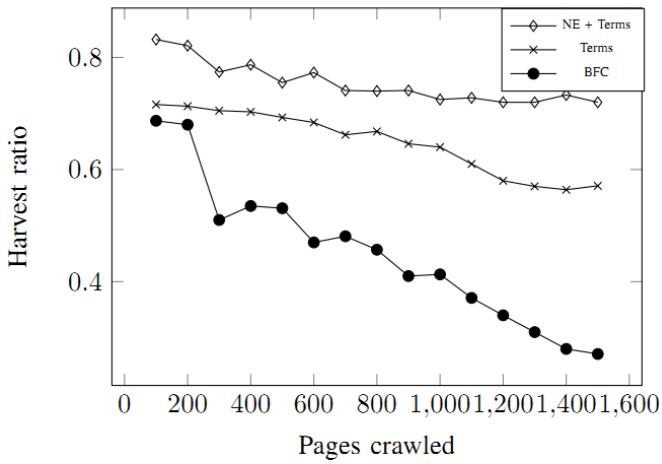

Fig - 6.5: Harvest ratio variation for American politics domain

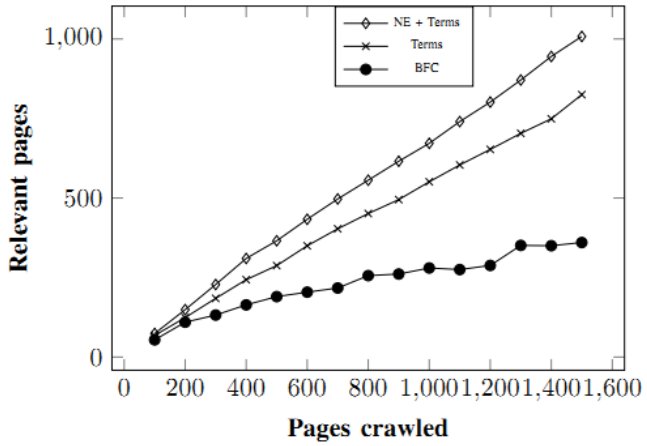

Fig - 6.4: Variation of number of pages downloaded for football l domain

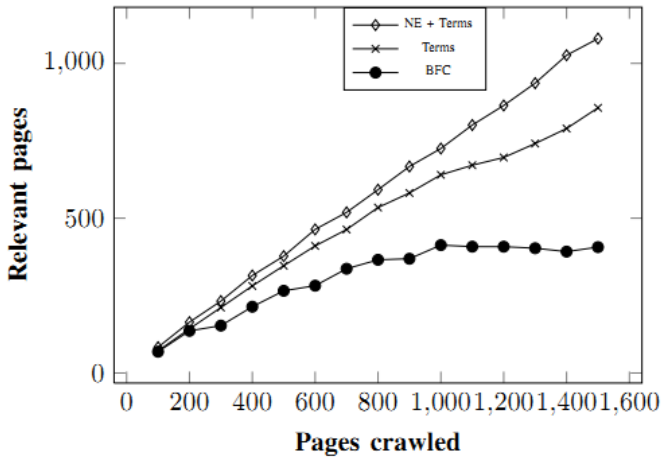

Fig - 6.6: Variation of number of pages downloaded for American politics domain

6.3. Focused Web Crawling in American Politics Domain

Finally we conducted experiments for the American politics domain. As same with the earlier experiments, we have collected training web pages manually from online news vendors and crawling was started with the following seed URLs: http://www.nytimes.com/politics/, http://www.americanpolitics.com/index.php. Crawling was continued for about 1500 web pages. Figure 6.5 shows the variation of harvest ratio with number of pages crawled and figure 6.6 shows the variation of relevant pages with the total number of pages crawled for all the 3 cases.

6.4. Discussion

These experiments showed the crawler performance for crawling domains baseball, football and American politics. Harvest ratio varied among different domains and seed sets, probably due to the linkage density of pages under a particular domain or the quality of the seed URLs.

Based on the results, we can see NE based focused crawler performs better than standard focused crawler (lexical terms based approach) and the breadth first crawler for all the three domains. Breath first crawler has the lowest harvest ratios for all the three domains as it doesn't deploy any strategies for identifying topic relevant pages. Harvest ratio is initially high for all the crawlers due to the presence of more topic relevant web pages near the seed URLs and gradually decreases before it gets stabilized.

7. Conclusions and Future Work

Current approaches to focused crawling are based solely on lexical terms when making relevance judgments and they disregard the information contained in NEs. But NEs can be a good source of information when it comes to crawling narrow domains. In this paper, we presented an extensive comparative study of the effect of named entities in focused web crawling for narrow domains.

Our experiments with focused crawling for three narrow domains -- baseball, football, American politics – showed that NEs enhance the crawler accuracy in terms of the harvest ratio. This was visible even for the duration of short crawls that we carried out. It indicated that during any time of the crawl, the collection built using NE approach is rich in content than the lexical terms based focused crawler.

Since NE recognition is performed on top of the main classifier used to guide the crawler, our approach is independent and allows to use with any focused crawler for improved harvest ratio when crawling narrow domains.

A possible drawback with this approach is that it can degrade the crawler efficiency as it involves extra steps for recognition and parsing of NEs. But as web crawling is an offline process, the tradeoff between improved harvest ratio to efficiency is worth at the end.

7.1. Future Work

This research brings up several themes for future research as well. We recognize NEs in web pages, but there is a possibility that some of them are unrelated to the crawling domain. If we can assign domain specific NEs and assign a higher weight for them, harvest ratio of the crawler can further be improved. Although in this paper relevance score is based on web pages, new scoring mechanisms like site based scoring and root directory based scoring are open to investigation. It is also worth investigating the behavior of the crawler on large scale crawls on different, NE rich domains as well.

REFERENCES:

[1]. P. D. Bra, G. jan Houben, Y. Kornatzky, and R. Post, “Information retrieval in distributed hypertexts,” in In RIAO, 1994, pp. 481–491.

[2]. M. Hersovici, M. Jacovi, Y. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur, “The shark-search algorithm - an application: tailored web site mapping,” in Proceedings of the 7th World Wide Web Conference, 1998.

[3]. J. Cho, H. Garcia-Molina, and L. Page, “Efficient crawling through url ordering,” Computer Networks and ISDN Systems, vol. 30, no. 1-7, pp. 161–172, 1998.

[4]. B. D. Davison, “Topical locality in the web,” in Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, ser. SIGIR ’00, 2000, pp. 272–279.

[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, no. 11-16, pp. 1623–1640, 1999.

[6]. J. Kleinberg, “Authoritative sources in a hyperlinked environment,” Journal of the ACM (JACM), vol. 46, no. 5, pp. 604–632, 1999.

[7]. Y. Z. H. W. Z. D. Wenxian Wang, Xingshu Chen, “A focused crawler based on naive bayes classifier,” pp. 517–521, 2010.

[8]. G. Pant, K. Tsioutsiouliklis, J. Johnson, and C. L. Giles, “Panorama: extending digital libraries with topical crawlers,” in Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries, ser. JCDL ’04, 2004, pp. 142–150.

[9]. M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori, “Focused crawling using context graphs,” in Proceedings of the 26th International Conference on Very Large Data Bases, ser. VLDB ’00, 2000, pp. 527–534.

[10]. G. T. De Assis, A. H. F. Laender, M. A. Gonc¸alves, and A. S. Da Silva, “Exploiting genre in focused crawling,” in Proceedings of the 14th international conference on String processing and information retrieval, ser. SPIRE’07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 62–73.

[11]. M. Bazarganigilani, A. Syed, and S. Burkir, “Focused web crawling using decay concept and genetic programming,” vol. 1, 2011.

[12]. R. Babaria, J. S. Nath, S. Krishnan, K. R. Sivaramakrishnan, C. Bhattacharyya, and M. N. Murty, “Focused crawling with scalable ordinal regression solvers,” in International Conference on Machine Learning, 2007, pp. 57–64.

[13]. H. Liu, J. Janssen, and E. Milios, “Using hmm to learn user browsing patterns for focused web crawling,” Data & Knowledge Engineering, vol. 59, pp. 270–291, November 2006.

[14]. G. Zhou and J. Su, “Named entity recognition using an hmm-based chunk tagger,” in Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2002, pp. 473–480.

[15]. A. Borthwick, “A maximum entropy approach to named entity recognition,” Ph.D. dissertation, New York University, 1999.

[16]. E. Brill, “Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging,” Computational linguistics, vol. 21, no. 4, pp. 543–565, 1995.

[17]. H. Isozaki and H. Kazawa, “Efficient support vector classifiers for named entity recognition,” in Proceedings of the 19th international conference on Computational linguistics-Volume 1. Association for Computational Linguistics, 2002, pp. 1–7.

[18]. A. McCallum and W. Li, “Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons,” in Proceedings of the seventh

conference on Natural language learning at HLT-NAACL 2003-Volume 4. Association for Computational Linguistics, 2003, pp. 188–191.
[19]. D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," Lingvisticae Investigationes, vol. 30, no. 1, pp. 3–26, 2007.
[20]. T. G. Jenny Rose Finkel and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005), 2005, pp. 363–370.
[21]. Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in Proceedings of the Fourteenth International Conference on Machine Learning, ser. ICML '97, 1997, pp. 412–420.

BIOGRAPHIES:

Sameendra Samarawickrama is currently a fourth year computer science undergraduate at University of Colombo School of Computing, Sri Lanka. His research interests include web mining, data mining, information retrieval and automatic text classification.

Lakshman Jayaratne obtained his B.Sc (Hons) in Computer Science from the University of Colombo, Sri Lanka in 1992. He obtained his PhD degree in Information Technology in 2006 from the University of Western Sydney, Sydney, Australia. He is working as a Senior Lecturer at the University of Colombo School of Computing (UCSC), University of Colombo. His research interest includes Multimedia Information Management, Multimedia Databases, Intelligent Human-Web Interaction, Web Information Management and Retrieval, and Web Search Optimization.