Scientific Journal of Impact Factor (SJIF): 4.14

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 3, Issue 5, May -2016

An Adaptive Smart Crawler for Locating Deep Web Interfaces

Sonali Zol[1], Ms. N.S.Patil[2]

[1,2] Department of Computer Engineering, D.Y.Patil COE, Akurdi,Savitribai Phule Pune University, Pune

Abstract — *As search engine database stores a huge amount of information so searching on the internet is dragging a net across the surface of the ocean that means everytime it is not possible to get relevant information related with our query entered in the search engine. As there is a huge amount of information most of the information is hidden, burried far down on dynamically generated sites and standard search engine fails to find it. Traditional search engine create indices by crawling it is necessary that the page should be static. Such static pages has been discovered by search engine as, dynamically generated pages cannot be discovered which results in an increment of hidden data. So it is necessary to use a two-stage framework for efficient harvesting a deep web and which will also avoid to visit large number of page*

Keywords- Deep web, two-stage crawler, feature selection, ranking, adaptive learning, prequery ,post query.

I. INTRODUCTION

A web crawler is a program which collects and stores the data in a database over the internet for further analysis and arrangement. The process of web crawling involves gathering pages from the web and arranging them in such a way that the search engine can retrieve them efficiently. Deep web consist of data that exist on the web but they are inaccessible on text search engine. To locate the deep web databases is became a big challenge, because they are not registered with any search engines, mostly they are sparsely distributed, and changes constantly. Considering this problem, previous work has proposed two types of crawlers, generic crawlers and focused crawlers. Generic crawlers fetch all searchable forms and cannot focus on a specific topic.

Focused crawlers such as Form-Focused Crawler (FFC) and Adaptive Crawler for Hidden-web Entries (ACHE) can automatically search online databases on a given topic. FFC consists of link, page, and form classifiers for focused crawling of web forms, and it is further extended by ACHE consisting an additional components for form filtering and adaptive link learner. The link classifiers in these crawlers play an important role in achieving higher crawling efficiency than the best-first crawler. The link classifiers are used to predict the distance to the page containing searchable forms. As a result, the crawler fails to search targeted forms. SmartCrawler is a focused crawler consisting of two stages: efficient site locating and balanced in-site exploring. It perfoms site-based locating by reversely searching the known deep web sites for center pages, which can effectively find many data sources for sparse domains. By ranking collected sites and by focusing the crawling on a topic.

To improve more accuracy an Adaptive Smart Crawler is proposed in this paper. It is a two-stage framework, for efficiently harvesting deep web interfaces. In the first stage, the Crawler will performs site-based searching for center pages with the help of search engines, avoiding visiting a large number of pages. The Crawler ranks websites and priority is given to highly relevant sites for a given topic so that it can achieve accuracy for a focused crawl. In the second stage, Crawler will search the searchable forms from the given set of seed sites by classifying topic relevant links and domain specific searchable forms. For this we can use External search form detector that is those forms which provides external web search sites such as to Google for the sake of users. Also a prequery approach is considered for filtering out query form and non query form.

As Deep websites usually contains few searchable forms To achieve accuracy and wide coverage for a focused crawler an effective deep web harvesting framework was proposed. The proposed system consists of the crawler which is divided into two stages: site locating and locating searchable form stage. The site locating stage helps to achieve wide coverage of sites for a focused crawler, and the second stage can efficiently perform searches for web forms within a site. A two-stage framework to address the problem of searching for hidden-web resources is proposed

III. RELATED WORK

Host-ip clustering technique for deep web characterization:- This technique is proposed [3] which aimed at more accurate estimation of main parameters of the deep Web by sampling one national web domain. The Host-IP clustering sampling technique is proposed which addresses drawbacks of existing approaches i.e random sampling technique of IP addresses to characterize the deep Web. The Host IP web search consists of the following major steps.

First is Resolving, clustering and sampling which resolves a large number of hosts relating to their IP addresses, then it group the hosts based on their IPs, and finally it generate a sample of random IP addresses from a list of all resolved IPs. And second is Crawling the deep web and its site identification.

Searching for hidden-web databases:- Locating relevant data sources has been largely overlooked from few years. As the nature of the web is dynamic where data sources are constantly changing it is necessary to automatically discover these resources. Therefore Luciano Barbosa and Juliana Freire [15] proposed the crawling strategy which automatically locate hidden web databases by focusing the crawl on given topic by choosing links to follow within a topic that are more likely to lead to the pages that contain forms by employing a proper stopping criteria.

Crawling the hidden web [21] introduced a generic operational model of a hidden Web crawler which describes that how this model is realized in HiWE (Hidden Web Exposer), a prototype crawler built at Stanford also a new Layout-based Information Extraction Technique (LITE) is introduced which demonstrate its use in automatically extracting semantic information from search forms and response pages. As Crawlers retrieve content only from the publicly indexable Web, i.e., the set of Web pages that are searchable purely by following hypertext links, by ignoring search forms and pages that require authorization or prior registration. In most cases, the crawler ignore the tremendous amount of high quality content "hidden" behind search forms, in large searchable electronic databases. In this paper, the problem of designing a crawler which is capable of extracting content from this Hidden Web is addressed.

Hierarchical classication of Web content [22] based on the combination of both textual and visual features is described in this technique. This combination is achieved by multiple classier combination. In this technique weights are alloted for combinations, which has gained better results as compared to the ordinary combination based on general voting schema.

On Building a Search Interface Discovery System [5] is. the system for finding and classifying search interfaces forms. The I-Crawler consists of four main components: Site/Page Analyzer, Interface Identification, Interface Classification, and Form Database. Interface indentification includes three parts: Interface Detector, Structure Extractor, and Binary Classier. The Interface Detector is responsible for detecting a form within a web page. I crawler is known as Interface Crawler which aims to detect efficiently and automatically whether a particular web form is searchable or non-searchable and then to identify a main subject of a database accessible via a given searchable form.

IV. SYSTEM ARCHITECTURE

The proposed system deals with the two stage architecture for an Adaptive crawler to efficiently and effectively discover deep web data sources. The first stage finds the relevant site for a given topic and then in second stage it will find searchable forms from the site. In the first stage there will be a set of seed sites, can be called as candidate sites given for a crawler to start crawling process. The crawler will start working by following URLs from choosen set of seed sites in the site database to find other pages and other domains. A threshold value will be there, if the number of URLs for the seed site is less than that of threshold value then the crawler will perform reverse searching for center pages.

For example, "If there is a site of an sbi bank and we have to search for the query say 'home loan ' the crawler will start by following the URLs of a choosen site if the number of URLs is less than the threshold it will search over the content which is given on the home page, crawler will search the links, hyperlinks which is given for the home page(center pages). " The algorithm of the crawler is given below:-

Algorithm1: Two stage crawling algorithm for site search- ing and site classifying

Input: seed sites and deep websites
Output: Extracting relevant sites

1) Initialize the algorithm with seed set of pages p
2) T be the set of the best score pages s such that
   s □ p
3) site = getDeepWebsite(siteDatabase, seedSites)
4) **if** candidate site is less than threshold **then**
5) resultPage = reverseSearch(site)
6) links = extractLinks(resultPage)
7) for each link in links do
8) page= downloadPage(link)
9) relevant= classify(page) by using Naive Bayes Classifier
10) **if** relevant **then**
11) relevantSites=extractUnvisitedSite(page)
12) getRelevantSites
13) end
14) end
15) end

Algorithm 2: Reverse searching for more sites

Input: seed sites and harvested deep websites
Output: relevant sites

(a) while number of candidate sites less than a threshold do
(b) site = getDeepWebSite(siteDatabase,seedSites)
(c) resultPage = reverseSearch(site)
(d) links = extractLinks(resultPage)
(e) page = downloadPage(link)
(f) for each link in links do
(g) relevant = classify(page)
(h) if relevant then
(i) relevantSites = extractUnvisitedSite(page)
(j) Output relevantSites
(k) end
(l) end
(m) end

In first stage the modified adaptive crawler consist of three components:

- Frontier
- Ranker
- Classifier

The component frontier fetches the homepage URLs of highly relevant sites from site database, as site database consist of seed sets of web pages. Ranker is use to prioritize highly relevant sites it is done by learning the features of deep web site. the classifier is used to categorize the URLs as topic relevant or irrelevant according to the content of the homepage. In the second stage we use to find searchable forms i.e. the form which contains the relevant information. In this stage links of the sites are stored in link frontier component, its corresponding pages are fetched and forms (HTML,XML,..)are classified. There are different types of HTML forms such as:

- **External search forms:-** Forms which provides external web search sites such as to Google. It means when user search in the search textbox of that particular web site it uses Google search engine or some other search engine database for convenience of user.
- **No search forms:-** Forms for login, subscription, registration, polling, or blogging is classifies as No search forms.
- **Site-search form :-** The site search forms are what many web sites nowadays provide for searching their own HTML texts on their sites. These pages simply scan non HTML tag text and provide information. They are not dynamically produced using information generated from a database.

Fig.1:- Site Search Form

For classification of text the naive bayes classifier is used for training the data. Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class.

A. Naive Bayes Classification

. The nave Bayesian classifier, works as follows:

- Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector, X = (x1, x2,..... xn), depicting n measurements made on the tuple from n attributes, respectively, A1, A2,.... , An.

- Suppose that there are m classes, C1, C2, ..., Cm. Given a tuple, X, the classifier will predict that X belongs to the class having the highest posterior probability, conditioned on X. That is, the nave Bayesian classifier predicts that tuple X belongs to the class Ci if and only if. P(Ci |X) > P(Cj |X) for $1 \leq j \leq m; j \neq i$

  Thus we maximize P(Ci |X). The class Ci for which P(Ci |X) is maximized is called the maximum posteriori hypothesis. By Bayes theorem:-

$$P(Ci|X) = \frac{P(X|Ci)P(Ci)}{P(X)}$$

- As P(X) is constant for all classes, only P(X | Ci)P(Ci) need be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, (P(C1) = P(C2) = ... = P(Cm)) and would therefore maximize P(X | Ci) Otherwise, we maximize P(X | Ci)P(Ci). The class prior probabilities may be estimated by P(Ci)=| Ci,D | / | D | where | Ci,D | is the number of training tuples of class Ci in D.

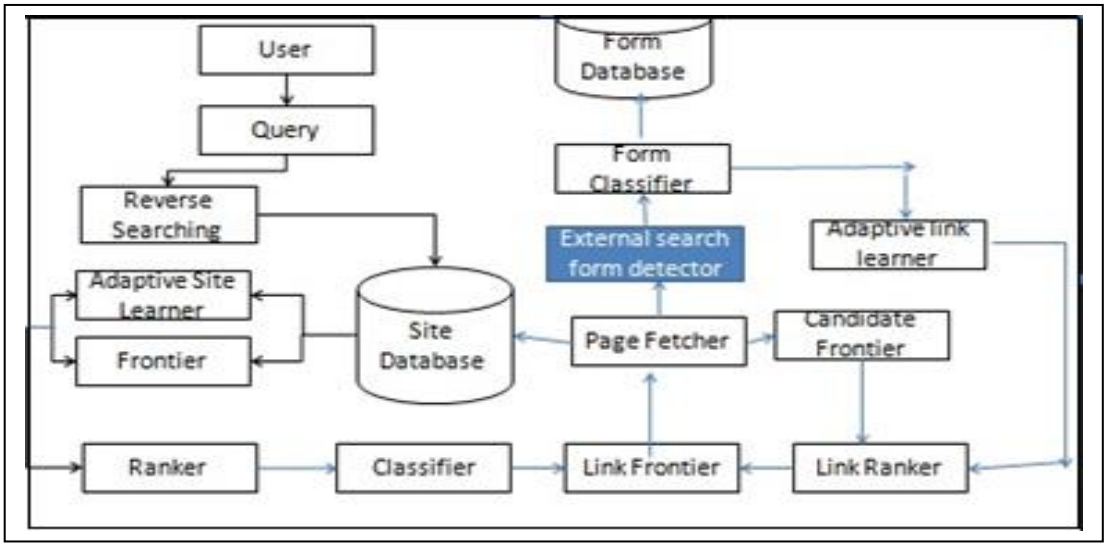

Fig2:- System Architecture of Two Stage Framework

In the given systen architecture the link frontier stores the link of the sites whereas the form classifier classify whether the form is searchable or not. In addition to that the links in these pages are extracted into candidate frontier. Ranking is given for prioritizing the link. So link ranker is use to prioritize links so that crawler can discover searchable forms quickly. For ranking purpose crawler ranks URLs to prioritize deep sites of given topic. Ranking is given on the basis of site similarity. It can be calculated as:

$$Site\ similarity(U, A,T)=Cos\ \frac{[v1.v2]((U, Us)+(A,As)+(T,Ts))}{|v1|.|v2|} \quad (1)$$

where *U, A,T* i.e. URL, anchor and text are the vectors corresponding to the feature context of URL and *Us, As, Ts* are the homepage URL of new site.

For prioritizing link similarity is calculated. It considers the path part since links have the same domain. The link similarity of known links with searchable form can be calculated as:

$$Link\ similarity\ (P, A,T)=Cos\ \frac{[v1.v2]((P, Pl)+(A, Al)+(T, Tl))}{|v1|.|v2|} \quad (2)$$

where P is the vector related to path of URL, A and T are the vectors corresponding to the feature context of URL and Pl, Al, Tl are new links. The feature context can be represented by a vector in terms of specific weight. The weight w of term t is defined as:

$$\mathbf{W}t,d = \mathbf{1+\ logT,F}t,d \quad (3)$$

where TFtd is frequency of the term t appears in document d and d can be URL, path, anchor or text.

The site frequency can be calculated by measuring number of times a site appears in other sites. Finally the rank of new site can be calculated as:

$$\mathbf{Rank(newsite)=\alpha*(Sitesimilarity(U,A,T))+(1\alpha)*log(1+Site\ frequency)} \quad \mathbf{(4)}$$

B. Stopping Criteria

The stopping criteria is used to avoid unproductive crawling:

- SC1: The maximum depth of crawling is reached.
- SC2: The maximum crawling pages in each depth are reached.
- SC3: A predefined number of forms found for each depth is reached.
- SC4: If the crawler has visited a predefined number of pages without searchable forms in one depth, it goes to the next depth directly.
- SC5: The crawler has fetched a predefined number of pages in total without searchable forms

V. RESULTS

The results shows number of deep website harvested versus various domain for SCDI (i.e. site base crawler for deep web interface) and Adaptive Smart Crawler. As SCDI does not consider reverse searching. It uses adaptive link prioritizing strategy for locating sites and links. The Adaptive smart crawler performs reverse searching as well as it classifies external search forms. It uses stopping criteria for avoiding unnecessary forms.

Hence, the adaptive smart crawler has higher crawling rate as compare to site base crawler for deep web interface. The horizontal axis shows the various domains of searchable sites and the vertical axis shows the number of deep web sites search in fig.3

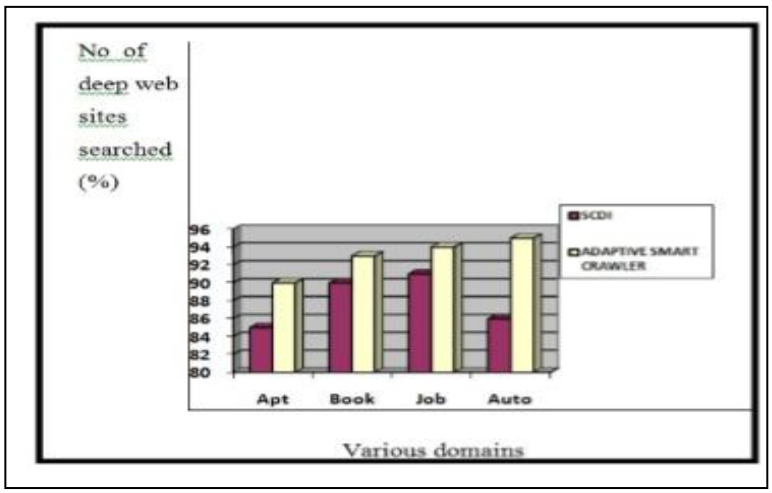

Fig.3 Total The number of deep websites harvested for various domain by SCDI and Adaptive smart crawler in percentage

The Total number of searchable Forms Found by various crawlers i.e existing and proposed crawlers for time =2 minutes is given below in fig. 4

Domains	ACHE (number of search form)	SCDI (number of search form)	Adaptive Smart Crawler (number of search form)
Airfare	8	123	320
Movies	15	750	2052
Rental	4	14	27
Book	2	13	63
Hotel	2	16	44

Fig.4: Total no. of searchable form found in time T=2min

The total number of links and searchable form classified by Smart Crawler in time T=2 hour is given below.

Domains	Total number of links	Total number of searchable form classified
Agriculture	804	8061
Air	288	2464
Apartment	895	6587
Books	172	1730
Computer	5	36
Hotel	272	2760
Movies	7927	61372
Music	597	4981
Rentals	233	1926
Route	16	114
Sports	141	1291

Fig5. Total number of searchable form found in time T= 2hr

VI. CONCLUSION and FUTURE SCOPE

As Deep web grows An effective deep web harvesting framework, namely SmartCrawler, for achieving both wide coverage and high efficiency for a focused crawler is used. The Two stage crawler performs site-based locating by reversely searching the known deep web sites for center pages, which can effectively find many data sources for sparse domains. In future we can work on postquery approach of form identifier and in future we can work on postquery approach of form identifier and The prequery form identifier filters out the non query form from the searchable set of forms. Hence, The proposed algorithm with prequery form identifier will give more accurate result than SCDI(Site base crawler for domain interface)

ACKNOWLEDGMENT

We express our deepest gratitude to the college authorities for technical guidance and infrastructure. And we are thankful to IJAERD authorities for giving chance to publish our research work. Lastly we wish to thank the researchers and reviewers for their contributions because of which we could complete this work

REFERENCES

[1] Yeye He, Dong Xin, Venkatesh Ganti, Sriram Rajaraman, andNirav Shah.Crawling deep web entity pages. In Proceedings of the sixth ACM international conference on Web search and data mining, pages 355364. ACM, 2013
[2] Denis Shestakov. Databases on the web: national web domain survey. In Proceedings of the 15th Symposium on International Database Engineering and Applications, pages 179184. ACM, 2011.
[3] Denis Shestakov and TapioSalakoski. Host-ip clustering technique for deep web characterization. In Proceedings of the12th International Asia-Pacific Web Conference (APWEB), pages378380. IEEE, 2010.
[4] Denis Shestakov and TapioSalakoski. On estimating the scale of national deep web. In Database and Expert Systems Applications, pages 780789. Springer, 2007.
[5] Shestakov Denis. On building a search interface discovery system. In Proceedings of the 2nd international conference one source discovery, pages 8193, Lyon France, 2010. Springer.
[6] Luciano Barbosa and Juliana Freire.An adaptive crawler for locating hidden-web entry points. In Proceedings of the16th international conference on World Wide Web, pages 441450.ACM, 2007.
[7] Infomine. UC Riverside library.http://lib-www.ucr.edu/,2014.
[8] Clustys searchable database dirctory. http://www.clusty.com/, 2009.
[9] Books in print. Books in print and global books in print ac- cess.http://booksinprint.com/, 2015.
[10] Kevin Chen-Chuan Chang, Bin He, and Zhen Zhang. Toward large scale integration: Building a metaquerier over databases on the web. In CIDR, pages 4455, 2005.
[11] Roger E. Bohn and James E. Short. How much information?2009 report on american consumers. Technical report, Universityof California, San Diego, 2009.
An Adaptive Crawler for Locating Deep Web Interfaces
[12] DMartin Hilbert. How much information is there in the informationso- ciety? Significance, 9(4):812, 2012.
[13] Idc worldwide predictions 2014: Battles for dom- inance and survival on the 3rd platform. http://www.idc.com/research/Predictions14/index.jsp, 2014.

[14] Michael K. Bergman. White paper: The deep web: Surfacing hidden value. Journal of electronic publishing, 7(1), 2001.

[15] Luciano Barbosa and Juliana Freire.Searching for hidden-web databases. In WebDB, pages 16, 2005.
[16] Peter Lyman and Hal R. Varian. How much information? 2003.Technical report, UC Berkeley, 2003.
[17] SoumenChakrabarti, Martin Van den Berg, and Byron Dom. Focused crawling: a new approach to topic-specific web resource discovery. Computer Networks, 31(11):16231640, 1999.
[18] Jayant Madhavan, David Ko, ucjaKot, VigneshGanapathy,Alex Rasmussen, and Alon Halevy. Googles deep web crawl.Proceedings of the VLDB Endowment, 1(2):12411252, 2008.
[19] Olston Christopher and Najork Marc. Web crawling.Foundationsand Trends in Information Retrieval, 4(3):175246, 2010.
[20] Dhawan, Sanjeev. ”Usability and Performance Estimation of Informa- tion Retrieval Systems.”
[21] Raghavan, Sriram, and Hector Garcia-Molina. "Crawling the hidden web." (2000).
[22] Dumais Susan and Chen Hao. Hierarchical classification of Web content. In Proceedings of the 23rd Annual International ACM SIGIR conference on Research and Development in Informa- tion Retrieval, pages 256–263, Athens Greece, 2000.

