Scientific Journal of Impact Factor (SJIF): 4.72

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

International Journal of Advance Engineering and Research Development

Volume 5, Issue 01, January -2018

A Research on-An Efficient method for Deep Web Crawler based on accuracy

MissPranali Zade[1] Dr.S.W.Mohod[2]

Deptt.Computer Science & Engineering,B.D.College of Engg.Sewagram,Wardha,India[1]
Prof.,Deptt. Computer Science & Engineering,B.D.College of Engg.Sewagram,Wardha,India[2]

Abstract—*Due to the large volume of web resources and the dynamic nature of deep web, achieving wide coverage and high efficiency is a challenging issue. We propose three-stage framework, for efficient harvesting deep web interfaces. In the first stage, web crawler performs site-based searching for centre pages with the help of search engines, avoiding visiting a large number of pages. To achieve more accurate results for a focused crawl, Web Crawler ranks websites to prioritize highly relevant ones for a given topic. In the second stage the proposed system opens the web pages internally in application with the help of Jsoup API and pre-process it. Then it performs the word count of query in web pages. In the third stage the proposed system performs frequency analysis based on TF and IDF. It also uses a combination of TF*IDF for ranking web pages. To eliminate bias on visiting some highly relevant links in hidden web directories, In proposed work, we design a link tree data structure to achieve wider coverage for a website. Project experimental results on a set of representative domains show the agility and accuracy of our proposed crawler framework, which efficiently retrieves deep-web interfaces from large-scale sites and achieves higher harvest rates than other crawlers using Naïve Bayes algorithm.In this paper we included the work up to the second stage. The proposed system uses KNNalgorithm, opens the web pages internally in application with the help of Jsoup API and pre-process it. Then it performs the word count of query in web pages.*

Keywords:—*Deep web, web mining, feature selection, ranking*

I. INTRODUCTION

The deep (or hidden) web refers to the contents lie behind searchable web interfaces that cannot be indexed by searching engines. Based on extrapolations from a study done at University of California, Berkeley, it is estimated that the deep web contains approximately 91,850 terabytes and the surface web is only about 167 terabytes in 2003. More recent studies estimated that 1.9 petabytes were reached and 0.3 petabytes were consumed worldwide in 2007. An IDC report estimates that the total of all digital data created, replicated, and consumed will reach 6 petabytes in 2014[19]. A significant portion of this huge amount of data is estimated to be stored as structured or relational data in web databases — deep web makes up about 96% of all the content on the Internet, which is 500-550 times larger than the surface web. These data contain a vast amount of valuable information and entities such as Infomine, Clusty, Books In Print may be interested in building an index of the deep web sources in a given domain (such as book). Because these entities cannot access the proprietary web indices of search engines (e.g., Google and
Baidu), there is a need for an efficient crawler that is able to accurately and quickly explore the deep web databases.

It is challenging to locate the deep web databases, because they are not registered with any search engines, are usually sparsely distributed, and keep constantly changing. To address 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 specific topic. FFC is designed with link, page, and form classifiers for focused crawling of web forms, and is extended by ACHE with additional components for form filtering and adaptive link learner.

The link classifiers in these crawlers play a pivotal role in achieving higher crawling efficiency than the best-first crawler. However, these link classifiers are used to predict the distance to the page containing searchable forms, which is difficult to estimate, especially for the delayed benefit links (links eventually lead to pages with forms). As a result, the crawler can be inefficiently led to pages without targeted forms. Besides efficiency, quality and coverage on relevant deep web sources are also challenging.Our main contributions are:

In the propose work we design a novel three-stage framework to address the problem of searching for hidden-web resources. Our site locating technique employs a reverse searching technique (e.g., using Google's"link:" facility to get pages pointing to a given link) and incremental three-level site prioritizing technique for unearthing relevant sites, achieving more data sources. During the in-site exploring stage, i design a link tree for balanced link prioritizing, eliminating bias toward web pages in popular directories.

In the propose work an adaptive learning algorithm that performs online feature selection and uses these features to automatically construct link rankers. In the site locating stage, high relevant sites are prioritized and the crawling is focused on a topic using the contents of the root page of sites, achieving more accurate results. During the in site exploring stage, relevant links are prioritized for fast in-site searching.

II. LITERATURE REVIEW

There is a rich literature, here we discuss the most related work.

Feng Zhao et al.[1] presents a two-stage Crawler for Efficiently Harvesting Deep-Web Interfaces.Deep web grows at a very fast pace, there has been increased interest in techniques that help efficiently locate deep-web interfaces. However, due to the large volume of web resources and the dynamic nature of deep web, achieving wide coverage and high efficiency is a challenging issue. Here propose a two-stage framework, namely SmartCrawler, for efficient harvesting deep web interfaces. In the first stage, SmartCrawler performs site-based searching for center pages with the help of search engines, avoiding visiting a large number of pages. In the second stage, Smart Crawler achieves fast in-site searching by excavating most relevant links with an adaptive link-ranking. To achieve more accurate results for a focused crawl, SmartCrawler ranks websites to prioritize highly relevant ones for a given topic. In the second stage, SmartCrawler achieves fast in-site searching by excavating most relevant links with an adaptive link-ranking.

Jianxiao Liu et al.[2]proposed an Approach of Semantic Web Service Classification Based on Naive Bayes,proposed a method to classify and organize the semantic Web services to help users find the services to meet their needs quickly and accurately is a key issue to be solved in the era of service-oriented software engineering. This paper makes full use the characteristics of solid mathematical foundation and stable classification efficiency of naive bayes classification method. It proposes a semantic Web service classification method based on the theory of naive bayes. It elaborates the concrete process of how to use the three stages of bayesian classification to classify the semantic Web services in the consideration of service interface and execution capacity.

Bo Tang, presents an approach toward Optimal Feature Selection In Naive Bayes For Text Categorization[3]author proposed that, automated feature selection is important for text categorization to reduce the feature size and to speed up the learning process of classifiers. In this paper, author present a novel and efficient feature selection framework based on the Information Theory, which aims to rank the features with their discriminative capacity for classification. Author first revisit two information measures: Kullback-Leibler divergence and Jeffreys divergence for binary hypothesis testing, and analyze their asymptotic properties relating to type I and type II errors of a Bayesian classifier.
Here author introduced new feature selection approaches based on the information measures for naive Bayes classifiers, aiming to select the features that offer the maximum discriminative capacity for text classification.They also derived the asymptotic distributions of these measures, which leads to the other version of the Chi-square statistic approach for feature selection.

Amruta Pandit and Prof. Manisha Naoghare,proposed work for Efficiently Harvesting Deep Web Interface with Reranking and Clustering[4].The rapid growth of the deep web poses predefine scaling challenges for general purpose crawler and search engines. There are increasing numbers of data sources now become available on the web, but often their contents are only accessible through query interface. Here author proposed a framework to deal with this problem, for harvesting deep web interface. Here Parsing process takes place. To achieve more accurate result crawler calculate page rank and Binary vector of pages which is extracted from the crawler to achieve more accurate result for a focused crawler give most relevant links with an ranking. This experimental result on a set of representative domain show the agility and accuracy of this proposed crawler framework which efficiently retrieves web interface from large scale sites.

Anand Kumar et al.[5]presents a two-phase system, to be specific SmartCrawler, for productive gathering profound web interfaces.
In this paper, author proposed, web develops at a quick pace, there has been expanded enthusiasm for procedures that assistance effectively find profound web interfaces. Be that as it may, because of the expansive volume of web assets and the dynamic way of profound web, accomplishing wide scope and high proficiency is a testing issue.In the primary stage, SmartCrawler performs site-based hunting down focus pages with the assistance of web crawlers, abstaining from going to a substantial number of pages.

Akshaya Kubba[7],mentioned that, Web mining is an important concept of data mining that works on both structured and unstructured data. Search engine initiates a search by starting a crawler to search the World Wide Web (WWW) for documents .Web crawler works in a ordered way to mine the data from the huge repository. The data on which the crawlers

were working was written in HTML tags, that data lags the meaning. It was a technique of text mapping. Semantic web is not a normal text written in HTML tags that are mapped to the search result, these are written in Resource description language. The Meta tags associated with the text are extracted and the meaning of content is find for the updated information and give us the efficient result in no time.

Monika Bhide et al. [8] focus on accessing relevant web data and represents significant algorithm i.e. adaptive learning algorithm, reverse searching and classifier.
the web stores huge amount of data on different topics.The main goal is to locating deep web interfaces. To locating deep web interfaces uses techniques and methods.The locating deep web interfaces system works in two stages. In the first stage apply reverse search engine algorithm and classifies the sites and the second stage ranking mechanism use to rank the relevant sites and display different ranking pages.

The work is based on crawl ordering that reveals the incremental crawler performance, better and is more powerful because it allows revisitation of pages at different rates and efficiently accessing web data. It can eliminate bias toward certain web site directories for wider coverage. This will provide the most relevant sites and accurate results to the users.

Raju Balakrishnan,and Subbarao Kambhampati,[10] proposed, selecting the most relevant web databases for answering a given query. The existing database selection methods (both text and relational) assess the source quality based on the query-similarity-based relevance assessment. When applied to the deep web these methods have two deficiencies. First is that the methods are agnostic to the correctness (trustworthiness) of the sources. Secondly, the query based relevance does not consider the importance of the results. These two considerations are essential for the open collections like the deep web. Since a number of sources provide answers to any query, author conjuncture that the agreements between these answers are likely to be helpful in assessing the importance and the trustworthiness of the sources.

While computing the agreement, we measureand try to adjust for any collusion between the sources.This adjusted agreement is modeled as a graph with sources at the vertices. On this agreement graph, quality score of a source that we call SourceRank, is calculated as the stationary visit probability of a random walk. We evaluate SourceRank in multiple domains, including sources derived from Google Base, with sizes up to 675 sources. They demonstrate that the SourceRank tracks source corruption. Our relevance evaluations show that SourceRank improves precision by 22-60% over the the Google Base and the existing methods.

III. PROPOSED WORK

The proposed work is planned to be carried out in the following manner:

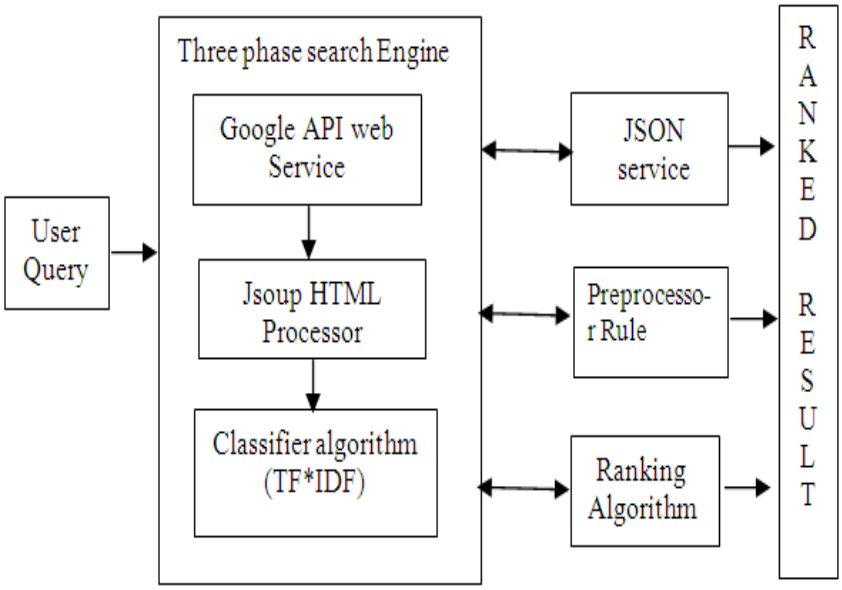


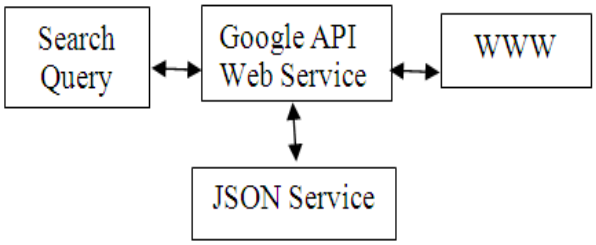
Fig 1: Proposed System Architecture

To efficiently and effectively discover deep web data sources, Crawler is designed with a three-stage architecture, as shown in above Fig 1. The first site locating stage finds the most relevant site for a given topic, the second in-site exploring stage uncovers searchable forms from the site and then the third stage apply naïve base classification ranked the result.

Specifically, the site locating stage starts with a seed set of sites in a site database. Seeds sites are candidate sites given for Crawler to start crawling, which begins by following URLs from chosen seed sites to explore other pages and other domains. When the number of unvisited URLs in the database is less than a threshold during the crawling process, Crawler performs "reverse searching" of known deep web sites for center pages (highly ranked pages that have many links to other domains) and feeds these pages back to the site database. Site Frontier fetches homepage URLs from the site database, which are ranked by Site Ranker to prioritize highly relevant sites.

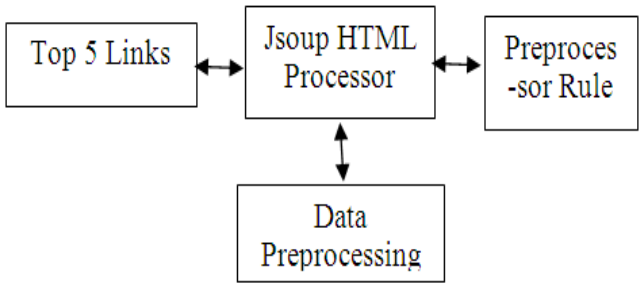
First Phase: Fetching Results from Google

In first phase the proposed system fetches results from Google search engine with the help of Google developer API and JSON (Java Script Object Notation).



Second Phase: Fetching the Word count from HTML Pages

In second phase the proposed system opens the web pages internally in application with the help of Jsoup API and preprocess it. Then it performs the word count of query in web pages. Up to this stage we use KNN algorithm to get word count.



Third Phase: Frequency Analysis

In third phase the proposed system performs frequency analysis based on TF and IDF. It also uses a combination of TF*IDF for ranking web pages. The ranking algorithm use here is Naïve Bays.

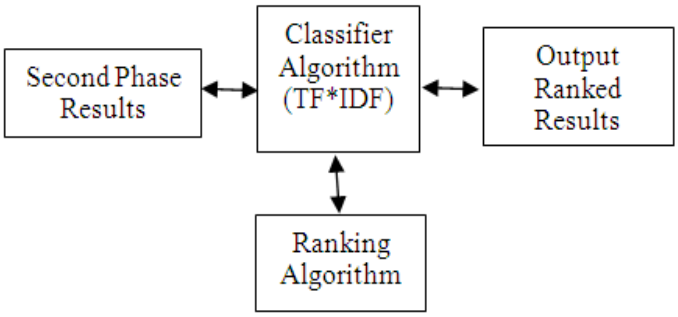


Fig 1: Search Query

The above Fig. 1, includes Textbox for searching query and search your query button. When user want to search query then he need to enter query in textbox first after that resulting top five URL/web will display by clicking on Search your query button.

The URL/web will display through JSON API. It is commonly used for transmitting data in web applications e.g., sending some data from server to the client so it can display on web pages and vice versa. In above screenshot the query"Latest mobile" is enter. After clicking on search your query button, via JSON API the top five URL for latest mobile will display.

When user click on search query button, the tittle and URL fetch from google via JSON API. JSON (JavaScript object Notation) is data interchange format that has vastly improved server to browser communication. The top five result of tittle and URL are store in Arraylist.

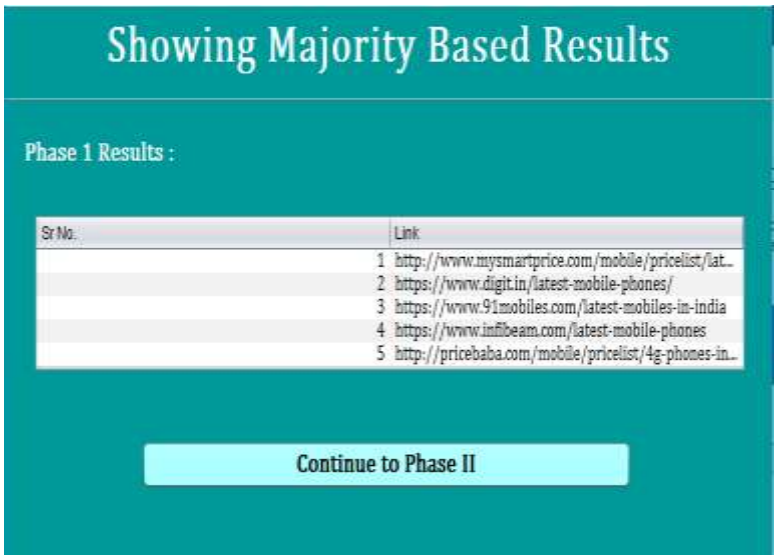


Fig 2: Data access from Google

The Fig 2: Data access from Google displays the top five URL. The first column display serial number and second column display Top five URLs. The above screenshot display URL for the query "Latest mobile" which is enter in screenshot 1: search Query

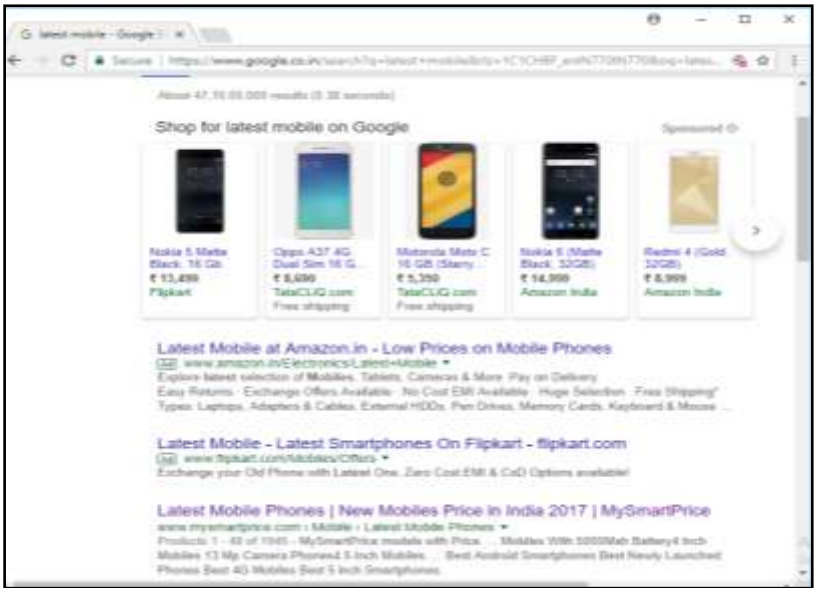


Fig 3: URL from google

The advantage of the propose work is that it can avoid the advertisement. We can see the comparison between the existing crawler and the proposed crawler in first phase. From the snap it is came to know that the advertisement for latest mobile is skip in fig. 2.

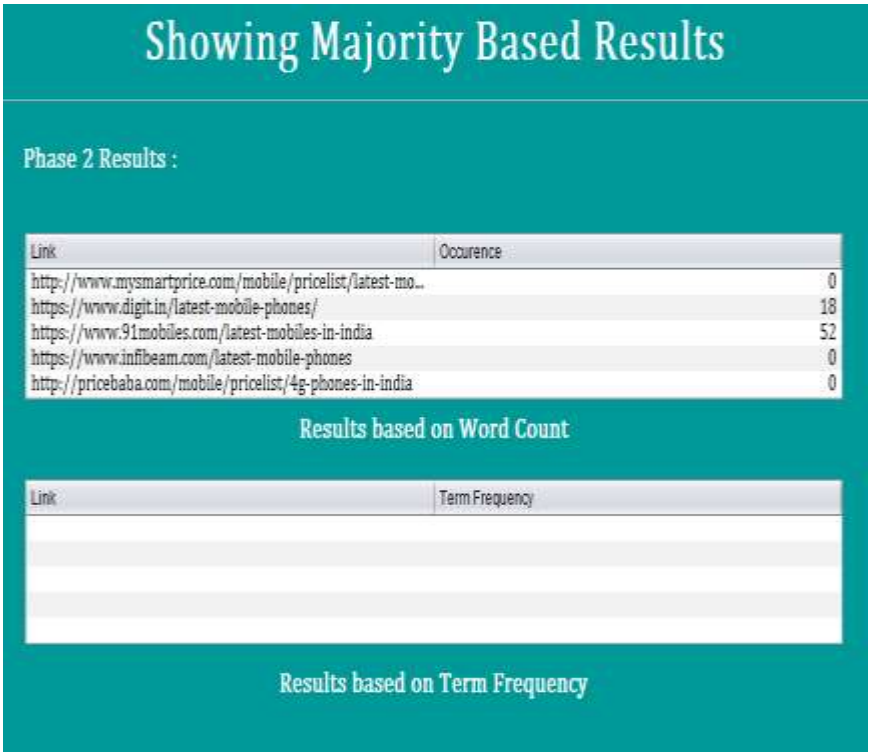


Fig 4: word count on top results

The above fig.4: word count on top results, shows the URL and word count perform on URL.
When the the Query “Latest mobile” get fire using JSON to Google server, top five result of tittle and URL are stored in a arraylist . Using Jsoup and URL connection class open each URL and store it in string Data[] array. After that Jsoup filter extract only open text data from string Data[]. By Using Iterator count or occurrences of each URL for query “latest mobile” is display.

IV. CONCLUSION AND FUTURE WORK

Past frameworks have numerous issues and difficulties. To achieve more accurate results Crawler ranks websites to prioritize highly relevant ones for a given topic. Proposed crawler opens the web pages internally and preprocess it. It performs the word count of query in web pages. To improve the efficiency and to overcome the problem of existing system, KNN perform on top results with pre-processing.

In third stage, we will performs frequency analysis based on TF and IDF. It also uses a combination of TF*IDF for ranking web pages.This framework actualizing new classifier Naïve Bayes. Project experimental results on a set of representative domains show the agility and accuracy of our proposed crawler framework, which efficiently retrieves deep-web interfaces from large-scale sites and achieves higher harvest rates than other crawlers using Naïve Bayes algorithm.

V. REFERENCES

[1] Feng Zhao, Jingyu Zhou, Chang Nie, Heqing Huang, Hai Jin "Smart Crawler: A Two-stage Crawler for Efficiently Harvesting Deep-Web Interfaces" in IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 9, NO. 4, JULY/AUGUST 2016.

[2] Jianxiao Liu, Zonglin Tian, Panbiao Liu, Jiawei Jiang, "An Approach of Semantic Web Service Classification Based on Naive Bayes" in 2016 IEEE International Conference on Services Computing, SEPTEMBER 2016.

[3] Bo Tang, Student Member, IEEE, Steven Kay, Fellow, IEEE, and Haibo He, Senior Member, IEEE "Toward Optimal Feature Selection in Naive Bayes for Text Categorization" in IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 9 Feb 2016.

[4] Amruta Pandit ,Prof. Manisha Naoghare, "Efficiently Harvesting Deep Web Interface with Reranking and Clustering", in International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 1, January 2016.

[5] Anand Kumar , Rahul Kumar, Sachin Nigle, Minal Shahakar, "Review on Extracting the Web Data through Deep Web Interfaces, Mechanism", in International Journal of Innovative Research in Computer and Communication Engineering, Vol. 4, Issue 1, January 2016

[6] Sayali D. Jadhav, H. P. Channe "Comparative Study of K-NN, Naive Bayes and Decision Tree Classification Techniques" in International Journal of Science and Research, Volume 5 Issue 1, January 2016.

[7] Akshaya Kubba, "Web Crawlers for Semantic Web" in International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 5, May 2015.

[8] Monika Bhide, M. A. Shaikh, Amruta Patil, Sunita Kerure,"Extracting the Web Data Through Deep Web Interfaces" in INCIEST-2015.

[9] Y. He, D. Xin, V. Ganti, S. Rajaraman, and N. Shah, "Crawling deep web entity pages," in Proc. 6th ACM Int. Conf. Web Search Data Mining, 2013, pp. 355–364.

[10] Raju Balakrishnan, Subbarao Kambhampati, "SourceRank: Relevance and Trust Assessment for Deep Web Sources Based on Inter-Source Agreement" in WWW 2011, March 28–April 1, 2011.

[11] D. Shestakov, "Databases on the web: National web domain survey," in Proc. 15th Symp. Int. Database Eng. Appl., 2011, pp. 179–184. [12] D. Shestakov and T. Salakoski, "Host-ip clustering technique for deep web characterization," in Proc. 12th Int. Asia-Pacific Web Conf., 2010, pp. 378–380.

[12] S. Denis, "On building a search interface discovery system," in Proc. 2nd Int. Conf. Resource Discovery, 2010, pp. 81–93.

