Automatic Extraction of Query Results from Deep Web Interfaces

Roshana Bangar

ABSTRACT

The content unseen behind HTML forms, has shorty been documented as a significant gap in search engine coverage. It represents vital contents of the data on the Web; retrieving Deep-Web content is not an easy task for the database community. Indexing of the searched data is major problem tackled by web crawlers that has deeply effect on search engine efficiency. Latest study about searching contents on the web illustrate that nearly 96% of data over internet is encapsulated as well as hidden i.e. hidden from search engines. The main task faced by the search engines is to retrieve and access hidden web data (web interfaces) or contents at low cost. The proposed system uses a machine learning approach that is very scalable, totally automatic, and identically efficient to use, that helps to expand data retrieval functionality at lower cost. The proposed system uses focused crawling strategy for accessing perfect searched results related to query and pick out only relevant information or data according to their similarity with respect to query. The proposed algorithm can selects only possible candidates rather than searching whole document for addition in to your web search index. The automatic attribute building is used for form classification that helps to reduce manual training time and data set building.

Keyword: *Attribute Extraction, Deep Web, Parallel Crawler, Focused Web Crawler, Web Crawler*.

INTRODUCTION:

Present-day web search engines do not capable to index and search a main portion of the Web therefore, the web users not capable to discover a large quantity of information from the non-indexable part of the Web. Generally, dynamic pages produced based on parameters provided by a user through web interfaces are non-indexed by search engines. This is the main task to recognize the resulting web pages without submitting parameters to the web form. Old web search engines are capable to index only a selected portion of the Web. The web, which is poorly indexed by search engines, is not only the part of deep web. Deep web [16], also contains the web pages which is never get register into the World Wide Web and the web pages that contains form on it which do not able to index by the traditional crawling approach.

The unknown web also include the dynamic data provided by web applications which returns real-time information after accepting specific user request for example ticket booking systems or online shopping. Depending on the delivered request at each time the different result will be generated. Although, these websites may provide a link structure to the items in database to accomplish crawling by the crawlers designed for the surface web. But there is no surety that those search engines will have the updated and current information about prices and remaining items in stock. Obviously this significant portion of the Web having information in the form of electronic web is badly accessible by conventional web crawlers designed for general purpose search engines.

Web Crawler constantly downloads web pages and indexed them, after indexing kept in database [2]. Search engines use an programmed tool called web crawler to collect web pages to be indexed. The web crawler primarily starts with a list of URLs to visit called the seeds set and as the crawler progresses these URLs, it extracts all hyperlinks from visited web pages and kept them to the crawler frontier holds the list of URLs to visit. According to the crawlers rules set, the URL's from the crawl frontier are recursively visited. The web pages which URL's are not inserted into search engines indices are neither in the seeds set nor in the crawl frontier.

The crawler which crawl only web pages correlated to specific domain then it is called as focused crawler [10]. The web pages which are not related to the specific domain are not measured. A crawler which fetches all searchable form without focusing on a specific topic is called a generic crawler [1]. Form focused crawler can spontaneously discovers the searchable web interfaces on a specific topic [1].

MOTIVATION:

Variety of users uses variety of search terms according to their knowledge and awareness to discover relevant Web pages that they are looking for. The amounts of relevant Web pages resumed to users differ dramatically, mainly dependent on the terms entered into conservatively available Web search engines. Many Web pages resumed to users may be entirely irrelevant, and it's a time-consuming for users to identify the relevant Web pages by going through too many results. It is essential to develop a new approach such that the number of returned Web pages becomes lesser while the overall number of relevant Web pages becomes higher.

RELATED WORK:

In day-to-day life Search engines are the most substantial part to visit in internet worldwide. In the whole WWW for searching some content by using search engine, web crawler leads a vital role. Typical approach of crawler for negotiating the web is long- lasting in terms of resource usage on both client and server. Thus, to assemble the most relevant pages, the most of the researchers concentrate on the structural design of the algorithms that are related with topic of interest. The topic specific web page crawler indicates the focused crawling was introduced by [11]. To determine the links which are most relevant by reducing the irrelevant search of the web the focused web crawler approach [10] is used.

Liu et al [5][6], discovers focused web crawler built on Hidden Markov Model (HMM) crawler for categorizing relevant pages paths. The another approach introduced by Liu et al [5] to overcome the problem of sequential pattern detection, they uses Maximum Entropy Markov Model (MEMM) to expand the features of focused web crawler. In this they use grouping of link structure and content analysis approach [9], to detect sequential patterns leading to targets. The prediction overhead of hop distance is the problematic issue with this system. To address the previous problems they proposed [4], the probabilistic approach for catching sequential patterns of pages related to their domain areas by using HMM, MEMM and CRF (conditional random fields) based models. It expands the relevant set of pages but the computational overheads are high.

Batsakis et al [2], presents a strategies to improve the performance of focused crawler. They work on the several different approach to focused web crawlers estimated using HMM crawler having the page content similarity and anchor text similarity. It is necessary that the input query must be well framed.

Rungsawang et al [3], addresses a learnable topic specific web crawler for discovering efficient result of web pages. For accomplishing this objective they keep the log of prior crawling to build some knowledge bases: seed URL's, topic keywords, URL prediction. The capability of learning tendency of crawler between the successive crawling is the most challenging job.

David et al [8], defines the system for surfacing deep web content by using the incremental search for informative query templates (ISIT) algorithm. The vital issue is to index the content behind the millions of HTML forms by recognizing the input values of a specific kind. This system is capable of handling form powered by HTML language only.

Abdul Nabi et al [7] announced domain based information system which crawls associated to specific domain. It expands the performance of the system by decreasing the searching space and searching time. It use pattern matching algorithm to rank the web page and then total rank is designed. The presented system is requires the huge number of collection from the precise domain.

Feng Zhao et al [1], presents a Smart Crawler for efficiently harvesting deep web interfaces having a two stage approach. Even they use adaptive link ranking, reverse search techniques, the manual training for each classifier necessary.

PROPOSED APPROACH:

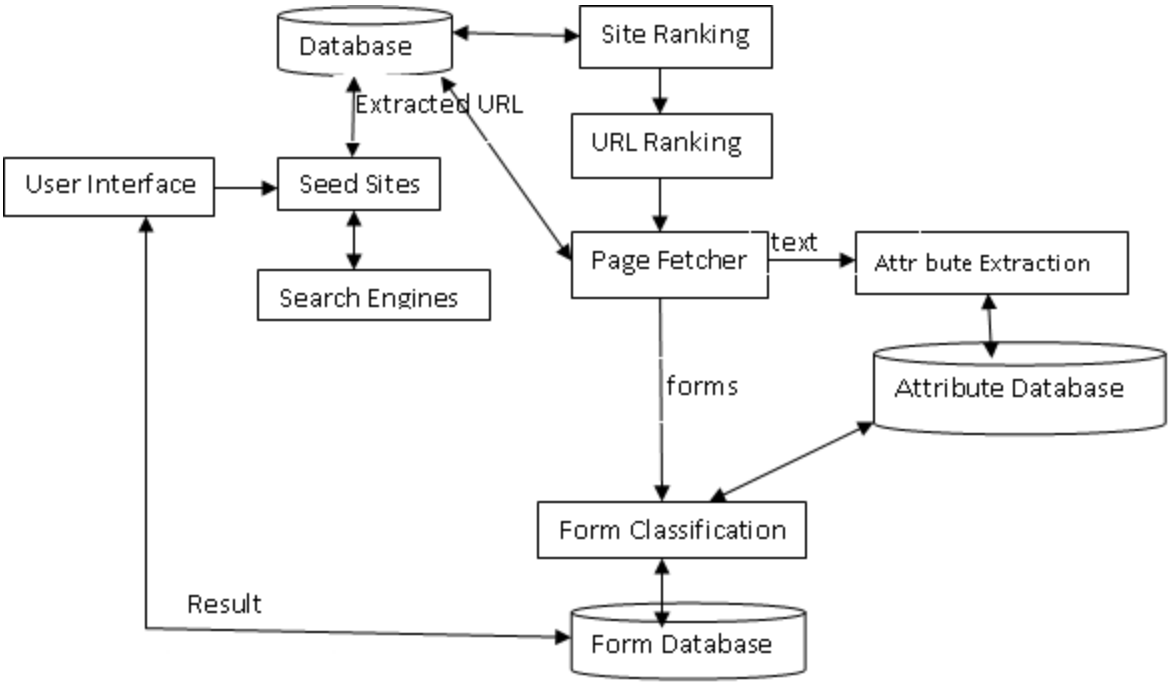

Fig 1. Proposed System Architecture

The proposed system is designed with the four major parts which are site classifier, site ranking, URL ranking and form attribute builder to efficiently discover the deep web interfaces. The operator can submit their query to the user interface (UI) and come to be result back with relevant pages on the same which is design by using the HTML. Specifically, the crawling process starts with seed sites [7]. The seed sites are a set of candidate sites for crawler to start crawling. The main motive is to diminish the number of visited URL's and at the same time maximize the deep web sites in seed sites. To accomplish these goals system can takes help from altered search engines to get website URL's on given topic. After gathering seed sites in site database the site ranker fetches the URL from it and ranks them by using site frequency and site similarity measures. The similarity of related features between new website s and the kwon deep web sites is measured is called site similarity. The number of times a site appears in the kwon deep web sites than the other sites is called site frequency measures. Relevant sites then continue for the URL ranking.

For ranking the promising link URL crawler automatically learn pattern of URL and recognize their focus as crawl evolutions. The page fetcher (PF) downloads the page after URL ranking and forwards them to attribute builder. In this system, the attributes can be extracting automatically by using the viewpoint of the user (UVA) and viewpoint of the programmer (PVA). PVA means the web application. The final attribute set can be built by reconciling the results obtained by classifying both set of attributes with the help of WordNet ontology.

If the extracted web pages comprise a form then it can be cross checked by attributes extracted from the attribute extraction module for specified topic. If the attributes can be found in the extracted form then such form is categorized as relevant form and kept in database for future use.

PROPOSED ALGORITHM:

Algorithm 1: Automatic Attribute Extraction

Input: Function *AAE(DS$_i$)* Web Data Source

Output: *FA$_i$*

begin

for each *DS$_i$* **do begin**

Obtain *HF$_i$*

For each *HF$_i$* **do begin**

Obtain *II$_k$*

end

end

Obtain KW_i
Obtain PVA_i and $SOPVA_i$
for each DS_i **do begin**
Obtain UVA_i and *SOUVAi*
Compare $SOPVA_i$ *and* $SOUVA_i$
Obtain FA_i
end
end

MATHEMATICAL MODEL:
Let S be a system defined as,
S = {I, f(x)}
I: Input Dataset or URL.
f(x): It provides a set of functions that performs on the input URL or Dataset, defined as,
f(x) = { SSII, OPVA, OUVA, PSA, FAE}

Step1:
Firstly we have separating a set of inner identifiers of a Web data source DS_i and obtain the set of IICA.
Function: SSII = {DS, HF, KW}
Output: O_1= {II, IICA}
Where,
SSII = Separating the Set of Inner Identifier
DS = Web Data Source set
II = Inner Identifier set
KW = Pre Defined Keyword set
IICA =Inner Identifier Candidate Attribute
HF = HTML Forms

Step 2:
After obtaining the inner identifier based candidate attributes (*IICAi*) of each Web data source (*DSi*), the set of PVAs is obtained from all sets *IICAi*.
Function: OPVA = {II, SSII (II)}
Output: O_2= {PVA}
Where,
OPVA = Obtaining Programmers Viewpoint Attribute
PVA = Programmers Viewpoint Attribute

Step 3:
For obtaining the UVAs for each Web data source *(DSi)*, requires that the free text between two HTML tags.
Function: OUVA= {HF}
Output: O_3 = {UVA}
Where,
OUVA = Obtaining User Viewpoint Attribute
UVA = User Viewpoint Attribute

Step 4:
For obtaining the synonym for each candidate attribute of *PVA* or *UVAi* we use WordNet Ontology.
Function: PSA = {PVA, UVA}
Output: O_4 = {SOPVA, SOUVA}
Where,
PSA = Programmers Set of Attribute
SOPVA = Synonyms of PVA
SOUVA = Synonyms of UVA

Step 5:
By comparing, *SOPVAi* and *SOUVAi* generate a final attribute (*FAi*) set

Function: FAE = {SOPVA, SOUVA}
Output: O_5 = {FA}
Where,
FAE = Final Attribute Extraction
FA = Final Attribute

CONCLUSION:

ACKNOWLEDGMENT:
To prepare this paper, I would like to be very thankful to my project guide and P.G. Coordinator Prof. Kahate S.A., in Computer Department of Sharadchandra Pawar Collage of Engineering Affiliated to SavitribaiPhule Pune University. Because of their support only I am able to complete my work.

REFERENCES:
[1] Feng Zhao, Jingyu Zhou, Chang Nie, Heqing Huang, Hai Jin. "SmartCrawler: A Two-stage Crawler for Efficiently Harvesting Deep-Web Interfaces" IEEE Transactions on Services Computing, 2015
[2] Batsakis, Sotiris, Euripides Petrakis, and Evangelos Milios. "Improving the performance of focused web crawlers." ELSEVIER, 2009.
[3] Rungsawang, Arnon, and Niran Angkawattanawit. "Learnable topic-specific web crawler." Science Direct, 2005: 97–114.
[4] Liu, Hongyu, and EvangelosMilios. "ProbabilisticModels for Focused Web Crawling." An International journal on Computational Intelligence, Volume 28, Number 3,289-328,2012.

[5] Liu, Hongyu, and EvangelosMilios. "ProbabilisticModels for Focused Web Crawling."Computational Intelligence, 2010.

[6] Liu, Hongyu, EvangelosMilios, and Larry Korba. "Exploiting Multiple Features with MEMMs for Focused Web Crawling."NRC, 2008.

[7] Sk.Abdul Nabi, Dr. P.Premchand, "Effective Performance of Information Retrieval by using Domain Based Crawler", Vol. 4, No.7, 2013.

[8] Jayant Madhavan, David Ko, Lucja Kot, Vignesh Ganapathy, Alex Rasmussen, Alon Halevy. "Google's Deep-Web crawl", VLDB, 2008.

[9] Anshika Pal, Deepak Singh Tomar, S.C. Shrivastava, "Effective Focused Crawling Based on content and Link Structure Analysis" Vol. 2, No. 1, June 2009.

[10] Vruksha Shah, Riya Patni , Vivek Patani, Rhythm Shah,"Understanding focused crawler." International Journal of Computer Science and Information Technologies, Vol. 5 (5) , 2014, 6849-6852.

[11] Chakrabarti, Soumen, Martin van den Berg, and Byron Dom. "Focused crawling: a new approach to topic-specific Web resource discovery." Elsevier, 1999.