www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 5 Issues 6 June 2016, Page No. 17116-17122

Automatic Query Formulation for Extracting Hidden Web: A Review

Manvi Siwach, Swati Gogia

Department of Computer Engineering
YMCA University of Science & Technology
Faridabad, India manvi.siwach@gmail.com
Department of Computer Engineering
YMCA University of Science & Technology
Faridabad, India swatcat72@gmail.com

Abstract:

There is lot of data on the internet which is not indexed by our conventional search engines. This web content is what we call as Hidden web or Deep web. Users have to fill various forms to access this hidden content. So, there is a need to make an interface which help us to automatically fill the forms to access the hidden data. This can be done by extracting attributes from html pages and compare those attributes with users query and resulting attributes would be used to fill those forms automatically. For best results, ontology, specifically domain ontology can be used for filling the forms of hidden web.

Introduction:

A significant amount of information on the web cannot be accessed directly through search engines. Massive amount of data that our search engines don't capture, buried in our databases or other research resources. More than 80% of web content not accessed by our search engines, we call it as hidden web or deep web and it is not indexed by conventional search engines.Users have to manually feed a set of keywords to access these hidden web pages. To access hidden web pages, user have to fill the query forms of web data sources. In order to gain access, users need to fill the attributes in the query forms like for example, if we need to search a book form any website, we have to fill the attributes such as author name, book title and so on. And to fill these fields manually is tedious task. So there is a need to access these hidden web pages without filling the lengthy query web forms.

Ontology is a formal explicit description of concepts, properties of each concepts describing various features and attributes of the concepts. Domain ontology is a description of concepts in a domain of discourse. The common ontology used is WordNet which contain noun, pronoun, verb, all meanings of the word in a single dictionary.

To fill the forms of deep web automatically using ontology can be done by extracting attributes from html pages. Attributes are in the form of programmer viewpoint attributes or in the form of user viewpoint attributes. There are various papers on Attribute extraction, which are using different approaches to extract these attributes from web pages such as html.

Basic steps which can be performed to fill the forms in the hidden web automatically to first extract the attributes from web pages, then use some ontology to retrieve basic words related to attributes extracted and then at last compare the retrieved attributes with the query interface attributes and fill the forms automatically using these attributes. Various

papers used various approaches to automatically fill the hidden forms so that users need not fill it manually and helps to reduce the task of user.

Literature Survey:

1. **Automatic Attribute Extraction from deep web data sources**(Yoo Jung An,JamesGeller, Yi-Ta Wu,Soon Ae Chun)

The approach used in this paper for extracting proper attributes from web data sources is to find the overlapping area between user viewpoint attributes and programmer viewpoint attributes. This overlapping area is what we call as final attributes. We have query interface in web data sources, and finding final attributes are used to fill the forms automatically.

Programmer Viewpoint Attributes are used by query web page designers whereas User Viewpoint Attributes are attributes presented to users as labels.

First step is to find PVAs by separating inner identifiers of web data sources. Inner identifiers cannot be used directly because they contain several words and symbols. Remove all the special symbols from inner identifiers and separate the string into substrings as a camel case arrives. For each keyword, break the substring into several substrings. Remove duplicated substrings. The result is programmer viewpoint attributes (PVA).

Next step is to find UVAs by extracting free text between two html tags. This free text is call as candidate strings. If any special symbol arises in candidate string, separate the string into substrings. Resulting attributes are user viewpoint attributes.

Ontology, WordNet is used to find the synonyms of these extracted attributes. Only noun synonym is considered, i.e. if word have noun meaning in ontology, word will be kept otherwise discarded.

At last, overlapping attributes are taken as final attributes and used later to fill the forms automatically.

Advantages of AAE algorithm:

1. Existing ontology i.e. WordNet is utilized, no need to construct new ontology.
2. Accuracy of extracted attributes are quite high, for example, 35% of PVAs are removed using WordNet.
3. Improper words are removed from the extracted attributes.

Limitations:

1. Considering only noun form of the word from the WordNet ignores all the other important attributes which are in other forms such as verbs, adjectives, adverbs and many more.
2. WordNet used here lacks domain specific knowledge. More semantic to deep web processing must be added.

2.Automatic Filling Forms of Deep Web Entries based on Ontology (Ying Wang, Tao Peng, Wanli Zuo, Ran Li):

In this paper, authors have presented a feasible and effective approach for filling forms automatically using ontology.

First step is to construct a domain ontology, starting by constructing the core ontology using some ontology tool, then remove stop words and other extended concepts from this core ontology. Then at last use some general ontology, such as WordNet to enrich the concepts in our core ontology.

Next, for schema extraction, fix query interface region by selecting the forms according to features of words in query interfaces and parse this query interface region. Parsing helps us to divide html form information into various tags such as start tag, end tag, simple tag and text tag. To filter useless information and obtain meaningful elements, we should use layout features and appearance features during schema extraction.

Ontology mapping will be done as next step after schema extraction. Ontology mapping is used to obtain mapping relationships between local interfaces or integration interfaces and ontology. Integration interface is where user can fill his/her requirements. The output of ontology mapping is local-ontology-integration mapping tables .When user fill his/her requirements in integration interface, values will be converted into local interfaces using mapping tables.

At last, query translation is performed to achieve semantically closest so that redundancy can be reduced. Similarity of words cab be calculated by either taking numerical similarity or taking text similarity.

Two definitions used in this paper to calculate numerical similarity.

1. Sim (C1,C2) = 1 - | m-n | / max(m, n)
   Where C1 is query value in integration interface and C2 is query value in local interface. m and n are two numerical values.

2. Sim (C1 , C2) = |S1∩S2| / |S1∪S2|
   Where S1= {m1, m2, m3 …} and S2= {n1, n2, n3…}

Text Similarity can be calculated by semantic similarity of phrases and this can be defined in domain ontology as:

1. Sim (P1, P2) = $\sum_{i=1}^{m} S_i + \sum_{i=1}^{n} S_i^l$ / L_1 + L_2
   Where P1 and P2 are phrases, L1 is number of terms for P1 and L2 is number of terms for P2. S is similarity vector.

2. To find the edit distance between two set of characters which can be calculated by minimum number of edit operations required to transform one string s1 into other s2.

Advantages:

1. Ontology Mapping used in the algorithm improves the accuracy.
2. Experiment shows that method is feasible and effective.
3. Submit button will be triggered automatically, which reduce the users work.

Limitations:

1. Calculating Similarity between words is tedious task.
2. Schema extraction arithmetic can be improved which helps user to achieve better results.

3.An Improved Extraction Algorithm from Domain Specific Hidden Web (Juhi Sharma, Mukesh Rawat)

Domain Specific Hidden Web Data Extractor is proposed in this paper, various modules and databases are used for extracting hidden web data.

First module is Page Fetcher, whose work is to fetch all the html pages from the page repository. Html pages are extracted by checking files and directories. Then, Form analyzer is used to extract only those pages which contains forms. Forms can be found by

checking html tag <form> and output i.e. html form pages are stored in a file. This file with all form pages is fetched in Search Interface Extractor which is third module of system architecture and helps to search whether the form is a Query Interface or Input Form. All the forms with query interface i.e. which contains button such as search, go, find are extracted and all the other forms are discarded.

Then main module, heart of architecture is Label extractor which actually fills the forms. Each field of the form is taken and matched with a table which contain Label-Value set pairs and using this table, form fields are filled.

Finally the filled form is submitted automatically and output of this last module is Hidden Web-data which is stored in a repository and can be accessed by user directly.

Advantages:

1. Easy to understand.
2. Hidden data stored in a repository which helps user to directly access hidden web data.
3. Various databases are used which helps user extract information from these databases easily.

Limitations:

1. Implementation of this algorithm is complex.
2. Efficiency of achieving the result is less.

4.On the Automatic Extraction of Data from the Hidden Web(Stephen W. Liddle , Sai Ho Yau , David W. Embley)

In this paper, author describes hidden or deep web in a very different way from other researchers. According to author, dynamically generated pages i.e. our html forms are only accessible through Common Gateway Interaction(CGI) and the information available through such CGI requests are called deep web. The models used in this paper for filling form automatically are:

1. Automated Form Filling
2. Query Submission Plan
3. Processing a Query Response Page
4. Filtering Duplicate Records

In first module, html form is extracted using parse tree and the presence of form is indicated by start and end tags, <form> and </form> respectively. The extracted form is submitted for CGI processing. Submission can be done by two ways,

HTTP POST, form submitted in body of the request.

HTTP GET, form submitting by supplying pairs in the URL.

For example, a question mark (?) separates the base URL and action path from the encoded names and values. In the name/value list, an equality operator (=) separates a name from its assigned value, and an ampersand (&) separates one (name, value) pair from the next.

http://www.usedmusicgear.com/cgi-bin/umg/search.cgi?category=&manuf=&model=&year=&condition=&sort by=1&submit search=SEARCH
This query is submitted automatically and is same as user clicking search button without selecting or typing anything on the web form.

Next module is Query Submission Plan whose main goal is to fill in the form in all possible ways. It is not necessary that one query will obtain all the information. To check whether all the available data is obtained or not, sampling of queries is done.

Then Processing a Query Response Page is done i.e. information from target site is retrieved. Various possibilities of result are checked such as Response page may contain all data or Response page includes many links and many more. When web pages are retrieved within a time period then it is saved in a file otherwise discarded.

Last module is to filter duplicate records and for this task, copy detection system is used. In this system a sentence boundary separator tag <s.> is used at the end of each record. When copy detection system is invoked, it computes hash values for every record separated by <s.>. These hash values compared with previously stored hash values and after eliminating duplicated records, result is stored in repository.

Advantages:

1. Domain Independent approach, applicable on any general web forms.
2. Experiment results give great responses and are encouraging.
3. Detailed description of every module makes implementation easy.

Limitations:

1. Text values can automatically filled if use ontology.
2. Not a task-specific approach, and makes processing difficult for general web data.

5.Automatic Filling of Web Forms (Gustavo Zanini Kantorski and Carlos Alberto Heuser)

For filling forms automatically, this paper explores two strategies, Filling Text Fields(FTF) and Instance Template Pruning(ITP).

FTF is filling text fields which do not have predetermined values and is based on feedback loop, loop continues until the last element produces feedback on first element. For filling the text fields or selecting values from selecting list, value selection module is used. Values selected from previous submissions are used to fill the forms.

ITP is to select queries to submit a particular form so that more data is retrieved with fewer submissions. All possible templates are generated by Candidate template generation module. ITP is used to prune the wasteful instances of template.

Values extracted by option tag are queries which are submitted and stored in a database. Queries submitted first by finite domain fields and then by infinite domain fields.

Advantages:

1. Number of queries submitted on the form are minimized.
2. For any text fields, values are automatically filled.
3. Although the strategy is domain independent, values for text fields adapt the domain.

Limitations:

1. Other than text fields, values are not efficiently automatically selected.
2. Quality of selected values are not good and can be improved by adapting other methods.

Comparison Of Different Works:

Technique Used	Html Attributes Extra	Ontology Used	Accuracy of algori	Remarks

	cted (tags)		**thm used**	
Programmer and user viewpoint attributes are extracted.	yes	General Ontology used (Word Net)	More	Attributes are extracted and ontology applied for better efficiency
Ontology Created, Schema extracted, mapping and translation is done.	yes	Core ontology is created and enriched with general ontology such as Word Net	More	Feasible and effective method.
Hidden Web Data accessed by user directly without filling the forms.	No	No Ontology is used.	Less	A repository is created enriched with attributes used to fill forms automatically.
Dynamically generated pages accessed only by CGI filled automatically	No	No Ontology is used.	More	Processing Difficult for general web data.
FTF and ITP is used for automation of filling forms.	yes	No Ontology is used.	Less	Text field values are filled automatically if this strategy adopted.

Conclusion:

Different researchers applied different methodologies to fill dynamic forms automatically without the intervention of the user. This makes users task easier as if for every query user need not fill the form again and again. Various Ontologies can be used to improve the efficiency and to find the more accurate result. Hence from all these papers it can be concluded that automatic form filling is necessary as it helps users to access hidden web data more efficiently and automation is done to accessing hidden web.

References:

Automatic Attribute Extraction from deep web data sources(Yoo Jung An, James Geller, Yi-Ta Wu, Soon Ae Chun)

Automatic Filling Forms of Deep Web Entries based on Ontology (Ying Wang, Tao Peng, Wanli Zuo, Ran Li):

An Improved Extraction Algorithm from Domain Specific Hidden Web (Juhi Sharma, Mukesh Rawat)

On the Automatic Extraction of Data from the Hidden Web(Stephen W. Liddle, Sai Ho Yau , David W. Embley)

Automatic Filling of Web Forms (Gustavo Zanini Kantorski and Carlos Alberto Heuser)