Identification and Classification of Deep Web Query Interfaces via Ontology

Baohua Qiang, Guoyong Cai, Yimin Wen, Chunming Wu, Chenghua Tang

School of Computer Science and Engineering, Guilin University of Electronic Technology
Guilin, P. R. China, 541004, qiangbh@yahoo.com.cn

College of Computer and Information Science, Southwest University
Chongqing, P. R. China, 400715, chunmingnone@163.com

Abstract

In order to obtain the large quantities of valuable information on Deep Web, it is required to discover the related individual query interface and design the integrated query interface on which user query request can be submitted. The key challenges are to identify and classify the Deep Web query interface accurately. In view of the regular data of Deep Web, we consider to construct the Deep Web domain ontology to help identify and classify the Deep Web query interfaces. Correspondingly, one domain ontology construction approach by referring to the hierarchal schema of query interface is proposed. Based on the constructed domain ontology, the modified interface expression algorithm and the identification and classification algorithm for Deep Web query interfaces are also presented, respectively. Experimental results show the effectiveness of our proposed algorithms.

Keywords: Deep Web, Schema extraction, Query interface identification, Query interface classification, Ontology

1. Introduction

With the rapid development of Web databases, Deep Web turns into an important data resource on the Internet, and Deep Web data integration is becoming an emerging research area. The primary task for Deep Web data integration is to dynamically identify the query interfaces and classify them according to domains [1, 2]. However, the Deep Web cannot be indexed by the present main search engines, which restricts the acquisition of Deep Web resource. Due to the large quantities of Web pages on the internet, identifying and classifying the query interfaces become the main challenges for Deep Web data integration and application.

At present, pre-query and post-query are two main approaches to identifying and classifying Deep Web query interface. In the pre-query approach, Wei Liu [3] proposed a vision-based approach for Deep Web data extraction, which provides a new channel for accurate query interface schema extraction. Hatem A. Mahmoud [4] proposed a schema clustering and retrieval approach for multi-domain data integration systems with higher precision and lower misjudgment rate. Jared Cope [5] and Juliano Palmieri Lage [6] proposed a C4.5 decision tree approach and a set of generalized rules based on practical applications to determine the Web query interfaces, respectively. Although these two approaches can quickly identify the query interfaces, their self-learning abilities and domain independences are unsatisfactory. In the post-query approach, one or more queries should be submitted to a form so as to identify the domain of deep web according to queried results [7-10]. This approach required intensive works with inefficiency. Some Deep Web classified directory search engines such as Completeplanet and InvisibleWeb can classify the Deep Web query interfaces automatically, but the detailed technology keeps secret.

Generally, the present approaches excessively rely on information technology itself to identify and classify the Deep Web query interfaces. These approaches cannot obtain the satisfactory results due to the existences such as homonym, synonymy and other semantic heterogeneity phenomenon. With the advent and development of ontology, ontology plays an important role in knowledge representation and semantic matching [11-14]. The construction and application of domain ontology will provide an effectively alternative method for identifying and classifying Deep Web query interfaces. In this paper, we will focus on solving this problem by analyzing the HTML structures, constructing domain ontology and proposing effective schema expression algorithm.
The main body of this paper is organized as follows. In Section 2, we propose the approach of Deep Web domain ontology construction. The modified interface expression algorithm and the identification and classification algorithm for Deep Web query interfaces are proposed in Section 3 and Section 4, respectively. The experiments of our proposed approaches are given in Section 5. Section 6 concludes the paper.

2. Construction of Deep Web Domain Ontology

Deep Web query interfaces have well-formed structures and their schema are very useful to identify and classify the Deep Web data resource. Generally, the schema of query interface can be represented as an ordered tree of elements, which is consistent with the hierarchical structure of Ontology. Figure 1 and Figure 2 indicate one typical airfare query interface and its hierarchical tree structure, respectively. So automatic domain ontology construction on the basis of extracting the hierarchical schema of query interface become possible, which will help resolve the problem of semantic heterogeneity and thus improve the classification accuracy of Deep Web query interfaces. Once the domain ontology is constructed by referring to the hierarchical schema of query interface, it can be used to assist in identifying and classifying the Deep Web query interfaces on a large scale.

![Figure 1. One example query interface Q of airfare domain](image1)

![Figure 2. The hierarchical structure of Q](image2)

In order to capture the schema of query interface, Wu Wensheng [15] proposed a hierarchical modeling which mainly employed the spatial relationships of attributes to obtain the hierarchical structure. Due to the neglect of semantics, some attributes can not be grouped correctly. By analyzing the characteristics of query interface and our previous research [16], we propose a set of new grouping...
patterns as follows to extract the hierarchical structure of query interface and lay the foundation for Deep Web domain ontology construction.

1) Separator-based patterns, which utilize separators such as section titles and horizontal lines to divide the attributes into groups. Separator-based patterns can be used to construct the top concepts;

2) Alignment-based patterns, which identify groups of attributes which are highly aligned to one another. For instance, attributes f3-f5 in Figure 1 can be classified as the same hierarchies due to their alignments;

3) Indentation-based patterns, which identify groups of attributes based on their indentation relative to a label. For instance, attributes f11 can be grouped together for their indentation relative to Class of Service;

4) Semantics-based patterns, which utilize semantic information such as labels and names of attributes to cluster attributes into groups. By studying a large amount of query interfaces on the Deep Web, we found that the labels or names of attributes with synonym or antonym are always grouped together. For instance, subsets of fields consisting of DepartureDay, DepartureMonth and DepartureTime in the airline domain, which are synonymous each other, are grouped together semantically. Another example is that the attributes of Adults and Children, which are antonymous each other, are also grouped together. Due to the semantic relationships of attributes on query interface relatively simple and regular comparing with natural language processing, the semantics-based patterns can be employed to classify the attributes effectively;

5) Control button-based patterns, which identify groups of attributes based on their control button types;

6) Attribute value-based patterns, which identify groups of attributes according to their instances. These patterns are very useful to divide the attributes when they have different instances but with similar names and labels.

By applying the above grouping patterns, the hierarchical schema of query interface similar to Figure 2 can be obtained, which actually reflects the rough structure of domain ontology. Figure 3 shows the conceptual hierarchical structure of airfare ontology, which is consistent with the hierarchal tree of query interface obviously. Our proposed domain ontology construction approach based on hierarchal schema of query interface can well reflect the concept abstract hierarchies and relations, and will help identify and classify the Deep Web query interfaces accurately.

Figure 3. Concept hierarchical structure of airfare ontology

3. Modified Query Interface Expression Algorithm

Effective schema extraction and expression of query interfaces is the foundation for identifying and classifying the Deep Web data resource. Based on the domain ontology constructed in Section 2, we proposed a formal query interface expression approach. Generally, Deep Web search interface typically
contains some HTML form control elements such as textbox, radio button, checkbox and selection list that allow a user to enter search information. Each element usually has a descriptive label associated with it. The label and element of the same attribute have a certain layout and are usually close to each other, and in most cases they share some similar information. The elements of Web query interface are arranged regularly and reflect the semantic relationships of search interface. In order to express the relationships of the elements of Web query interface formally, Hai He [17] proposed Interface Expression (IEXP) to capture the elements of Web query interface. IEXP is a string consisting of three elements as ‘t’, ‘e’ and ‘|’, where ‘t’ denotes a label/text, ‘e’ denotes an element, and ‘|’ denotes a row delimiter which represents a physical row border in the search interface.

We argue that the IEXP cannot effectively capture the semantic relationships of query interface because of its poor distinguishment about each element and its type of query interface by just using ‘e’ to express the control element. We suggest by using “type:name” instead of ‘e’ to capture the information of elements, where “type” and “name” denote the type and the name of each element, respectively. Besides, we argue that the row border ‘|’, which represents a physical row border in the search interface, is not effective to capture the relationships due to some complicated interfaces are always divided into a few regions by region delimiter, each of which can be expressed by IEXP. So the region separator should be employed to complement IEXP.

In order to identify the Web search query interfaces effectively, the following characteristics of Web Form should be extracted according to our above analysis: 1. Tag names of control elements; 2. Types of control elements; 3. Names of control elements; 4. Delimiters, including row delimiters and region delimiters, of control elements such as </tr>、<br>、<p> and etc. These characteristics will be employed to express the query interfaces. The detailed modified IEXP (MIEXP) algorithm is described as follows.

**Algorithm 1** MIEXP Algorithm for Query Interface  
**Input:** Q, i.e. HTML expression of Web including query interface  
**Output:** E, i.e. Modified query interface expression

1: p=htmltxt.start(Q); /*p denotes the word separated by blank */
2: i=0;
3: while (p≠htmltxt.end()) do
4:   if (p is a new <form>)
5:     i++;
6:     CStringArray Ei←null; /*Define new dynamic array and initialize null */
7:     while (p is not </form>) do
8:       a. When encounter string ‘t’, discard it if Length(t)>L (L=40). If not, append ‘t’ to Ei;
9:       b. When encounter element ‘e’, determine its type and name. if type∈{“text”, “select”, “checkbox”, “radio”}, formalize C as “type: name” and append ‘t’ to Ei;
10:      c. When encounter delimiter elements such as </tr>、<br>、<p>, distinguish the type of it and append ‘|’ to Ei;
11:     p++;
12:   endif
13: end
14: loop
15: loop
16: Return E0, E1, …, Ei.

By algorithm 1, the Forms of Web pages can be discovered and expressed as MIEXP. The length threshold of string ‘t’ on line 8 can filter the comment information effectively. Given any Web page, we will determine whether it belongs to Deep Web page or not and classify it by analyzing the MIEXP and semantic relationships according to domain ontology in section 4.

**4. Identification and Classification Algorithm for Query Interfaces via Ontology**

Upon the modified IEXP algorithm and domain ontology, we firstly analyze the characteristics of Deep Web query interface and then design the corresponding identification and classification algorithm.
The algorithm starts at a Web directory which includes our interest URLs obtained by using typical search engines such as Google, Yahoo! and etc, and search all URLs in the Web directory by breadth-first. After that, the hyperlinks embedded in the Web will be searched and the search depth is restricted. Finally, the constructed domain ontology is referred to classify the Deep Web query interfaces. The detailed algorithm is described as follows:

**Symbols and their descriptions:**

- **AllWebLinkList**—The interest sites for specific domain collected by Web crawler;
- **Oi**—Constructed Domain ontology by using the approach in Section 2;
- **LinksToVisit**—Deque to be accessed;
- **CurrentLinkPage**—The current processed page;
- **DeQueue(LinksToVisit)**—Take out the first page in the LinksToVisit;
- **Depth(Link)**—The depth of current page;
- **VisitedList**—Deque that has been accessed;
- **Discard(CurrentLinkPage)**—Discard the current page and its embedded hyper links;
- **ExtractAllForms(CurrentLinkPage)**—Extract the schemas of query interface from current page;
- **SchemaSize (Form)**—Size of query interface;
- **ExtractLinks(CurrentLinkPage)**—Extract all hyperlinks of current page;
- **DeepWebInterfaceList**—Deep Web query interfaces;
- **SelectDomainOntology(Oi)**—Select domain ontology;
- **ClassifyingInterfacesByDomainOntology**—Classify query interfaces via domain ontology.

**Algorithm 2** Identifying and Classifying Deep Web Query Interfaces

**Input:** AllWebLinkList, Oi

**Output:** DeepWebInterfaceList; F’, i.e. classified query interfaces via domain ontology

```
1: p=AllWebLinkList.first(); /*Search from the first site in the AllWebLinkList*/
2: while (p≠Ø) do
3:    Add the current site in AllWebLinkList to LinksToVisit;
4:    while (LinksToVisit is not empty) do
5:        CurrentLinkPage ← DeQueue (LinksToVisit);
6:        if (Depth(CurrentLinkPage) > 3) /*heuristic rule to restrict web search depth*/
7:            Discard (CurrentLinkPage);
8:            continue;
9:        endif
10:       /*If Depth(CurrentLinkPage)<3, extract the schema of query interfaces and express as MIEXP by algorithm 1*/
11:       F ← MIEXP(ExtractAllForms(CurrentLinkPage));
12:       if (F = Ø || SchemaSize (all Forms in F) < 3) /*heuristic rule to restrict the size of schema*/
13:            LinksToVisit← ExtractLinks (CurrentLinkPage);
14:            Discard (CurrentLinkPage);
15:        else
16:            DeepWebInterfaceList ← CurrentLinkPage;
17:        endif
18:        SelectDomainOntology(Oi);
19:        F’ ← ClassifyingInterfacesByDomainOntology(F);
20:        p++; loop
21: return DeepWebInterfaceList; F’.
```

This algorithm mainly adopts MIEXP algorithm, domain ontology knowledge and some heuristic rules such as the restrictions of web search depth on line 6 and the size of query interface schema on
line 11, which can improve the accuracy and efficiency greatly. We will verify this algorithm in the following section.

5. Experiments

Our experimental data sets include 150 Web pages. One is typical ICQ data set from University of Illinois at Urbana-Champaign (UIUC) [18], which consists of 100 query interfaces pages pertaining to five domains (airfare, auto-mobile, book, job and real estate), each with 20 query interfaces (see Table 1). Another data set consists of 50 Surface Web pages without query interfaces, obtained by Google random search. We conduct experiments on these 150 Web pages to identify the Deep Web pages and classify them using our proposed algorithms in Section 3 and 4. The experimental environments are: CPU is 2.93G, RAM is 1G, and operating system is Windows XP.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Interfaces</th>
<th>Attributes</th>
<th>Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min</td>
<td>Max</td>
<td>Avg.</td>
</tr>
<tr>
<td>airfare</td>
<td>20</td>
<td>10.7</td>
<td>3.6</td>
</tr>
<tr>
<td>auto-mobile</td>
<td>20</td>
<td>5.1</td>
<td>2.4</td>
</tr>
<tr>
<td>book</td>
<td>20</td>
<td>5.4</td>
<td>2.3</td>
</tr>
<tr>
<td>job</td>
<td>20</td>
<td>4.6</td>
<td>2.1</td>
</tr>
<tr>
<td>real estate</td>
<td>20</td>
<td>6.5</td>
<td>2.7</td>
</tr>
</tbody>
</table>

Two main evaluating criterion, P (precision) and R(recall), commonly used in information retrieval domain, are adopted. Suppose IC denotes the number of interfaces belong to category Ci that are identified by classification algorithm, TC denotes the number of interfaces correctly classified to category Ci, and MC denotes the real number of interfaces belong to category Ci. Thus,

\[ P = \frac{TC}{IC}, \quad R = \frac{TC}{MC} \]

5.1. Experimental results of MIEXP algorithm

The MIEXP Algorithm in Section 3 is applied to extract and express the schema of query interfaces. The detailed experimental results are listed in Table 2. Our modified IEXP algorithm can obtain both the semantic information belonging to the same region and those belonging to the different regions, which avoids the inconsistent schema extraction and expression, and then improves the accuracy of Precision and Recall.

<table>
<thead>
<tr>
<th>Domains</th>
<th>IEXP Algorithm</th>
<th>MIEXP Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>airfare</td>
<td>92.1</td>
<td>95.6</td>
</tr>
<tr>
<td>auto-mobile</td>
<td>91.3</td>
<td>96.4</td>
</tr>
<tr>
<td>book</td>
<td>90.2</td>
<td>93.8</td>
</tr>
<tr>
<td>job</td>
<td>89.6</td>
<td>91.1</td>
</tr>
<tr>
<td>real estate</td>
<td>90.1</td>
<td>92.6</td>
</tr>
<tr>
<td>Average</td>
<td>90.7</td>
<td>93.9</td>
</tr>
</tbody>
</table>

Table 2 shows that the MIEXP Algorithm can obtain higher precision and recall in contrast to the IEXP approach. We note that the precisions range from 91.1% in the job domain to 96.4% in the auto-mobile domain, with an average of 93.9%, and the average of recall is 96.5%. The average of precision and recall can be improved 3.2% and 3.3%, respectively by mainly using “type:name” instead of ‘e’ and region separator to capture the information of elements.
5.2 Experimental results of identification and classification algorithm for query interfaces

As described above, the dataset is composed of 150 Web pages with 100 Deep Web pages from ICQ data set including query interfaces and 50 Surface Web pages from Google random search. To verify the effectiveness of our algorithm, we compare our algorithm with the similar algorithm in literature [19]. The detailed experimental results are listed in Table 3.

<table>
<thead>
<tr>
<th>Web Pages</th>
<th>Our Algorithm</th>
<th>Algorithm in [19]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P(%)</td>
<td>R(%)</td>
</tr>
<tr>
<td>Query interfaces</td>
<td>100</td>
<td>98</td>
</tr>
<tr>
<td>Non-query interfaces</td>
<td>96.2</td>
<td>100</td>
</tr>
<tr>
<td>Average</td>
<td>98.1</td>
<td>99</td>
</tr>
</tbody>
</table>

By analyzing the experimental results, we note that two Web pages including query interfaces are wrongly judged as Surface Web pages with non-query interface. The reason for wrong judgment is due to our heuristic rules, i.e. the size of query interface schema less than 3 is not regarded as Deep Web query interface. And the size of the two wrongly judged query interface is just equal to 2. So we should balance the efficient heuristic algorithm against the wrongly judgment. Comparing with the results by using algorithm in literature [19], we found that the classification accuracy of Deep Web query interfaces can be improved accordingly by using our proposed algorithm via domain ontology due to ontology can resolve the problem of semantic heterogeneity effectively.

6. Conclusion

In order to improve the accuracy, we firstly consider constructing and referring to Deep Web domain ontology to help identify and classify the Deep Web query interface. Then the domain ontology construction approach, modified interface expression algorithm and the identification and classification algorithm for Deep Web query interfaces are proposed, respectively. The experimental results show that our proposed approaches not only have better classification precision, but also can maintain higher recall. In the future work, we will construct more detailed and effective domain ontology to assist deep web identification, classification and integrated query interface design, and improve the accuracy further.

7. Acknowledgment

This work is supported by National Natural Science Foundation of China (61163057, 61063039), Open Fund of Guangxi Key Laboratory of Trusted Software (20110803), Chongqing Nature Science Foundation (CSTC2009BB2126). The author would like to thank Guoyong Cai, Yimin Wen, Chenghua Tang from Guilin University of Electronic Technology, and Chunming Wu from Southwest University for their suggestions and system evaluation.

8. References


