AUTOMATIC EXTRACTION OF DATA FROM DEEP WEB PAGE

Nripendra Narayan Das  
Associate Professor  
RIET, Faridabad  
nripendraddas@gmail.com,

Ela Kumar  
Dean and Associate Professor  
Gautam Buddha University, Greater Noida  
ela_kumar@gbu.ac.in

ABSTRACT

There is large volume of information available to be mined from the World Wide Web. The information on the Web is contained in the form of structured and unstructured objects, which is known as data records. Such data records are important because essential information are available in these host pages, e.g., lists of products and there detail information. It is necessary to extract such data records to provide relevance information to user as per their requirements. Some of the approaches used to solve this problem are manual approach, supervised learning, and automatic techniques. The manual method is not suitable for large number of pages. It is a challenging work to retrieve appropriate and useful information from Web pages. Currently, many web retrieval systems called web wrappers, web crawler have been designed. In this paper, some existing techniques are examined, then our current work on web information extraction is presented. It is the fact that most of the search engines are not in a position to extract the data from deep web. In this paper an algorithm has been designed and practically implemented which will not only extract the data in efficient manner but will also display as a single page output. Experimental evaluation on a large number of real input web url address collections indicates that our algorithm correctly extracts data in most cases automatically, many algorithms have been designed in the area of information extraction (IE).

The information source can be classified into three main types free text, structured text and semi-structured text. Originally, the extraction system focuses on free text extraction. Natural Language Processing (NLP) techniques are developed to extract this type of unstructured, unregulated information, which employs the syntactic and semantic characteristics of the language to generate the extraction rules. The structured information usually comes from databases, which provide rigid or well defined formats of information, therefore, it is easy to extract through some query language such as Structured Query Language (SQL). The other type is the semistructured information, which falls between free text and structured information. A good example of semi-structured information are Web pages.

According to the statistical results by Miniwatts Marking Group [19], the growth of web users during this decade is over 200% and there are more than 2 billion Internet users from over 278 countries and world regions. At the same time, public information and virtual places are increasing accordingly, which almost covers any kind of information needs. Thus this attracts much attention on how to extract the useful information from the Web. Currently, the targeted web documents can easily be obtained by inputting some keywords with a web search engine. But the drawback is that the system may not necessarily provide relevant data rich pages and it is not easy for the computer to automatically extract or fully understand the information contained. The reason is due to the fact that web pages are designed for human browsing, rather than machine interpretation. Most of the pages are in Hypertext Markup Language (HTML) format, which is a semi-structured language, and the data are not given in a particular format and change frequently [1].

There are several challenges in extracting information from a semi-structured web page e.g. (i) Lack of a schema. (ii) ILl formatting. (iii) High update frequency and semantic heterogeneity of the information.

In order to overcome these challenges, our system design transforms the page into a format called Extensible Hypertext Mark-up Language (XHTML) [20]. Then, we make use of the DOM tree hierarchy of a web page and regular expressions are extracted out using the Extensible Style sheet Language (XSL) [21, 22] technique, with a

1. INTRODUCTION

Before giving introduction of our paper we must know what is Deep Web. The Deep Web (or Invisible web) is the set of information resources on the World Wide Web not reported by normal search engines.

The World Wide Web contains different types of relevant information in the form of free text. The unpredictable growth and popularity of the world-wide web and availability of huge amount of information on the Internet makes extraction of data from web pages difficult. However, due to the heterogeneous and unstructured web page design, it is very difficult to retrieve information sources. Some of the Web mining applications, such as comparison shopping robots, require expensive maintenance to deal with different data formats. For translation of input pages into structured data
human training process. The relevant information is extracted and transformed into another structured format—Extensible Mark-up Language (XML) [23].

The remainder of the paper is organized as follows: some related works are illustrated in section 2, which involve a brief overview of the current web information extraction systems; then detail techniques in our approach are addressed in section 3; experimental results are explained in section 4; finally, the conclusion and future work are mentioned in the last section.

In this paper, we will focus mainly on extracting text information from web pages. After browsing more than 100 web pages, it was observed that almost all of the web pages are displayed in particular format using a particular language i.e. HTML. It was also observed that to display the output page in better format, Table and list HTML tags are used by most of the programmer.

2 RELATED WORK

In last 10-13 years, many extraction systems have been developed. In the very beginning, a wrapper is constructed to manually extract a particular format of information. However, the wrappers were not very much successful, it should be redesigned and reprogrammed accordingly to different types of information. In addition, it is complicated and knowledge intensive to construct the extraction rules used in a wrapper for a specific domain. Therefore only experts may have knowledge to do that. Due to the extensive work in manually constructing a wrapper, many wrapper generation techniques have been developed. Those techniques could be classified into several classes, including language development based, HTML tree processing based, natural language processing based, wrapper induction based, modelling based and ontology based [2].

Researchers have developed several approaches for mining data records from Web pages. David Embley, Yuan Jiang, and Yiui-Kai Ng use a set of heuristics and a manually constructed domain ontology[3]. David Buttler, Ling Liu, and Calton Pu extend this approach in Omini (Object Mining and Extraction System, http://disl.cc.gatech.edu/Omini), which uses additional heuristics based on HTML tags but no domain knowledge[4]. Chia-Hui Chang and Shao-Chen Lui proposed IEPAD (Information Extraction Based on Pattern Discovery), an automatic method that uses sequence alignment to find patterns representing a set of data records[5]. Because this method’s sequence matching is limited, it also performs poorly. Kristina Lerman, Craig Knoblock, and Steven Minton use clustering and grammar induction of regular languages but report unsatisfactory results [6]. Valter Crescenzi, Giannsalvatore Mecca, and Paolo Merialdo compare two pages to find patterns for data extraction [7]. However, their method needs to start with a set of similar pages. By working on individual pages, our technique avoids this limitation. Other related research is mainly in wrapper induction [8]. A wrapper is a program that extracts data from a Web site and puts it in a database. However, rather than mining data records, wrapper induction only extracts certain pieces of information on the basis of user labelled items.

3. PROPOSED ALGORITHM TO EXTRACT DATA FROM WEB PAGES:

To cater the challenge, in this paper we proposed an algorithm to extract the structure of lists by using the regularities both in the format of the pages and the data contained in them. The algorithm below describes the approach:

- Extract all data from lists
  - Compute the page pattern and identify the list on each page
  - Compute a set of features (separators and content) for each data extract
- Identify columns and Classification of data
- Identify rows

3.1 System design and architecture

3.2 Algorithm of web data extraction

**Input:**
- $k =$ key word(s)

**Output:**
- Extracted data from web pages

1. begin
2. while(not end of data records)
3. If(keyword found)
4. Open the web page internally
5. Parse the web page in equivalent HTML code
6. Search for data and extract from web page
7. Store the data in temporary file
8. Move to next record
9. Else
10. Move to next records
11. Display all the relevant extracted data

Figure 2: Pseudo code of the data extraction algorithm

When algorithm runs it starts by tokenizing Web pages, that is, splitting the text of the Web pages into individual words or tokens. Pages are analyze to find common structure. Many types of Web sources, especially those that return lists, generate the page from a template and fill it with results of a database query. It is expected all data are available in the same format and same type, car price for instance (as shown in figure 1). In addition to content, layout features, such as separators, may be useful hints for arranging extracts by columns. However, we cannot rely solely on separators — the table may have missing columns, separators before the first row and after the last one may be different from those separating rows within the list, etc.

The final step of the analysis is to partition the list into rows. Ideally, it should be easy to identify rows from the class assignment, because a row corresponds to a tuple of data, which is repeated for every row. An algorithm will be used this task. Each list, or rather the sequence of column labels for the extracts in the list, can be thought of as a string generated by a regular language. The language captures the repeated structure in the sequences that corresponds to rows. We use this information to partition the list into tuples. The end result of the application of the suite of algorithms is a complete assignment of data in the list to rows and columns.

3.3 Equivalent HTML Code of Figure-1

Due to constraints of page limit equivalent HTML code of 1st car only i.e. Mercedes-Benz S-Class has been shown below.

```html
<figure><img src="http://img2.aeplcdn.com/AutoBiz/Certifications/cargiant.png" class="lazy" title="Certified Car" style="margin-top:10px;" /></figure><br />
</a></dt><dd>
in Ashok Vihar, New Delhi
</dd>
<figure><img src="http://img2.aeplcdn.com/AutoBiz/Certifications/cargiant.png" class="lazy" title="Certified Car" style="margin-top:10px;" /></figure><br />
</a></dt><dd>
in Ashok Vihar, New Delhi
</dd>
```
Observation :
It can be seen from above HTML code (shown in border) that most of the web pages have been designed in a particular format, where data are put inside the Table tag format. e.g. `<TABLE>`, `<TR>`, `<TD>`.

3.4 Finding the page template

During the extraction step, the text of each Web page is split into individual words, or more accurately tokens, and each token is assigned one or more syntactic types [10], based on the characters appearing in it. Thus, a token can be an HTML token, an alphanumeric, or a punctuation token. If it’s an alphanumeric token, it may also belong to one of two categories: numeric or alphabetic, which is further divided into capitalized or lowercase types, and so on. The syntactic token hierarchy is described in [10]. Many Web sources use templates to automatically generate pages and fill them with results of a database query.

The algorithm works in the following way:

(a) When query is submitted in search box, it matches in the database where only web addresses are stored of different domain with key word.

(b) Internally open the web page and searches for key word in the generated HTML code.

(c) If keyword matches with the keyword available in database it opens the web page internally and searches for result. 

(d) Display the result in a particular format.

(e) The process is applied for all the web address and finally result are displayed in single web page.

Once section of the page is identified that contains the relevant data, all data are extracted from it. If the HTML table is carefully formatted, this step would amount to extracting all visible text. However, in addition to HTML tags, punctuation characters are often used to separate data fields. Many programmer use different format as separator. Sometimes a dash (-) is a good separator, but for many frequently encountered data types, such as phone numbers and zip codes, dash is part of data and not a separator. Likewise, comma (,) is sometimes a separator (e.g., “House No - 123, Gali no - 2, South Extension”) and sometimes part of data (e.g., in “1,000,000”), though in our concept it has been chosen to treat it as a separator.

4 RESULTS

Above mentioned algorithm were applied to extract data from some Web information sources containing a wide variety of data types. Three or four pages were randomly selected from each source. The extraction algorithm was applied to each set of pages and manually checked whether the data from the list was extracted correctly or not. The table in Fig. 2 summarizes the results. In some of the web pages data were not extracted as we want to retrieve the data for a particular domain. In this paper used card domain has been discussed and algorithm has been applied to retrieve the data for used car only.

4.1 Prototype Search form for the practical implementation
The web pages from different domain which were taken for practical implementation has been shown below:

<table>
<thead>
<tr>
<th>Name of web sites</th>
<th>Data retrieved or not</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.gaadi.com">www.gaadi.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.cartrade.com">www.cartrade.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.quikr.com">www.quikr.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.indiabookstore.com">www.indiabookstore.com</a></td>
<td>No (on line book purchasing web page)</td>
</tr>
<tr>
<td>in.bookmyshow.com</td>
<td>No(on line Movie ticket booking web page)</td>
</tr>
<tr>
<td><a href="http://www.zigwheels.com">www.zigwheels.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.cars.com">www.cars.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.oncars.in">www.oncars.in</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.pvrcinemas.com">www.pvrcinemas.com</a></td>
<td>No(on line Movie ticket booking web page)</td>
</tr>
<tr>
<td><a href="http://www.autotrader.com">www.autotrader.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.olx.in">www.olx.in</a></td>
<td>Yes</td>
</tr>
<tr>
<td>autos.yahoo.com</td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.rightcar.com">www.rightcar.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.gaadi.com">www.gaadi.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.carazoo.com">www.carazoo.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.amazon.in">www.amazon.in</a></td>
<td>No(on line book purchasing web page)</td>
</tr>
<tr>
<td><a href="http://www.ticketplease.com">www.ticketplease.com</a></td>
<td>No(on line Movie ticket booking web page)</td>
</tr>
<tr>
<td><a href="http://www.autotrader.com">www.autotrader.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.bookmyevent.com">www.bookmyevent.com</a></td>
<td>No(on line Movie ticket booking web page)</td>
</tr>
<tr>
<td><a href="http://www.bookmyseats.in">www.bookmyseats.in</a></td>
<td>No(on line Movie ticket booking web page)</td>
</tr>
<tr>
<td><a href="http://www.cardeko.com">www.cardeko.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.flipkart.com">www.flipkart.com</a></td>
<td>No(on line purchasing web page)</td>
</tr>
<tr>
<td><a href="http://www.justeat.in">www.justeat.in</a></td>
<td>No(On line food order in hotel)</td>
</tr>
<tr>
<td><a href="http://www.carwale.com">www.carwale.com</a></td>
<td>Yes</td>
</tr>
<tr>
<td><a href="http://www.snapdeal.com">www.snapdeal.com</a></td>
<td>No(on line purchasing web page)</td>
</tr>
</tbody>
</table>

Table-1 Sample web pages for retrieval of data

5. CONCLUSIONS

In this paper, a novel and effective technique was proposed to retrieve data records in a Web page. Our algorithm is based on two important observations about data records on the Web and a string matching algorithm. It is automatic and thus requires no manual effort. In addition, our algorithm is able to discover non-contiguous data records, which cannot be handled by existing techniques. Experimental results show that our new
method is extremely accurate. In our future work, it has been planned to find data records that are not formed by HTML table related tags.

6. REFERENCES


[22] http://www.w3.org/TR/xslt.html XSL Transformations, W3C Recommendation

