Data Record Extraction using Tag Tree Comparison

Aleem Ansari
Shri Venkateshwara University
Gajraula
India

Hemlata Vasishtha, PhD
Shri Venkateshwara University
Gajraula
India

ABSTRACT
This paper presents a robust unsupervised approach for extraction of data records from dynamic web pages using tag tree comparison. Extracting data records from the web pages involves following sequences. We first download the related web pages of interest on our system. Next we construct DOM trees for those pages using a parser. We then compare two or more web pages to eliminate the noisy unwanted data such as header, menu bar, navigation bar, advertisements, etc and find the region of interest called Data region or Object region. We then traverse subtrees of data region to detect individual data record and pull them in the XML file. The main contribution of this paper is in developing a fully unsupervised approach for extracting structured as well as semi-structured data records from the web pages. Our proposed system can extract data records from many commercial websites more precisely. Hence it can serve as a source for integrating information from various web sources which can be used for providing value added services such as comparative shopping, market intelligence, meta-querying and search.

Keywords
Data Record Detection, Information Extraction, Automatic Extraction, Web Mining, Semi-Structured data, Wrapper Generation.

1. INTRODUCTION
The World Wide Web is perhaps the largest source of information. Today there are large numbers of web sites providing access to data records contained in the underlying database. These web sites typically implement some kind of html form that facilitates end user issuing queries against the underlying database. The query result is then embedded in HTML pages conforming to a certain fixed template and returned to the end user. Though human users can easily interpret the results returned by the query, they are not suitable for automatic processing. This is because of the fact that end result contains large amount of unrelated and noisy information such as header, menu bar, navigation bar, advertisements, copyright information, etc. Furthermore, the semantic meaning of different parts of an HTML document may be encoded in ways that do not correspond in a simple way to a structured representation of data [1].

Figure.1 gives an example of related web pages displaying list of books. The description of each book is a data record also referred to as object data or ODATA. Since related pages are generated by the same dynamic program, we can see that both the pages are completely similar, they differ only in the data values appearing in the data record. These collections of data records together form the data region also referred to as Object region or OREG

Fig 1: Sample web pages displaying books information

Due to the huge amount of web pages it is unreasonable to extract data records using manual or semi-automatic approach. Allowing software programs to access these structured data is useful for a variety of purposes [2]. For instance, it allows data integration applications to access the web information in a manner similar to database. It also allows information gathering applications to store the retrieved information maintaining its structure and, therefore, allowing more sophisticated processing.

In this paper we present a novel technique for automatic extraction of data records from the related web pages. Since related pages from the any web site are normally generated by the same dynamic program or carefully maintained template, they tend to share similar structure [3]. It is thus possible to uncover the common structure containing the relevant object information in OREG by analyzing the pages in conjunction. Hence we can detect the OREG by comparing two or more related web pages. After detecting the OREG we mine individual data records and output them in the XML file. Our approach does not make any assumption about the structure of the web pages. We have also validated our technique on large number of commercial websites, obtaining very good results.

The rest of this work is organized as follows: section 2 introduces related work about extracting data record from the web pages. Section 3 presents the overall procedure for detection of OREG through the similarity calculations and extraction of ODATA from the OREG. We present and analyze the results of our experiments in section 4. Finally we conclude our work in Section 5.

2. RELATED WORK
There are many existing techniques for data record extraction in the form of wrappers. A wrapper is a program that extracts data from the web page and put them in the database. There are two main approaches for wrapper generation - wrapper induction and automatic extraction. The main problem with wrapper induction techniques as mentioned in [4], [5], [6] and [7] is that they require human labeling and cannot scale to

large web sites. Automatic extraction methods as mentioned in [8] and [9] are based on a set of heuristic rules, e.g., highest-count tags, repeating-tags and ontology-matching. However [10] shows that these methods produce poor result.

Another problem with the existing automatic approaches is their assumption about the structure and layout of the web pages which does not always holds true. For instance, [11] assumes that the visual space between two data records in a page is always greater than any gap inside a data record. Also some approach [12] assumes that OREG appears in the center of the page which may not always hold true. Another approach [13] assumes that all the records have the same number of child nodes and hence data nodes. It also assumes that all the required attributes are listed in the first row of the table in the web page .The LBDRF technique [14] selects the data region with the specific html tags as its root which again is not applicable to all the web pages. Another novel approach [3] makes no assumption and scale well to large number of websites, but this approach is more complex and computationally more intensive.

The above methods are inefficient and are based on many assumptions which do not always hold true for all the web pages. The proposed method does not make any such assumption and can scale well for most of the web pages. We take advantage of the structural information (the document object model, DOM, tree [15]) contained in web pages to locate relevant information. Also our procedure is automated and identifies data records from the collection of web pages easily. Furthermore, it does not require separate training, validation, and application phases, but simply operates on a collection of pages.

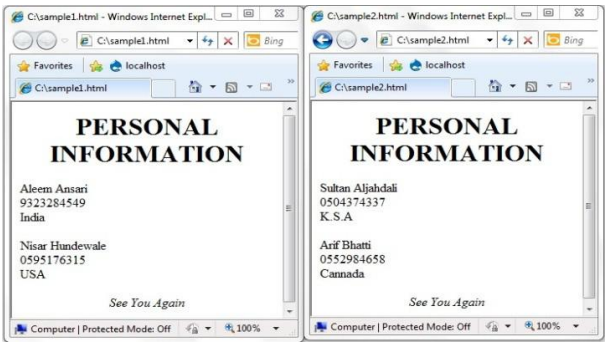
3. DATA RECORD EXTRACTION

This work presents an unsupervised approach for extracting data records from web pages. Extracting data records from the web pages involves the following phrases: First we identify the OREG containing the relevant data records. Second we identify individual ODATA and output them in XML file. The main contribution of this study is in developing a fully automated approach for extracting data records from web pages. Our approach can automatically extract data records from complicated web pages, such as the technical descriptions of digital cameras and personal computers downloaded from manufacturers' and vendors' sites. We also performed extensive experimental analysis to demonstrate that our framework is effective in extracting data records from web documents.

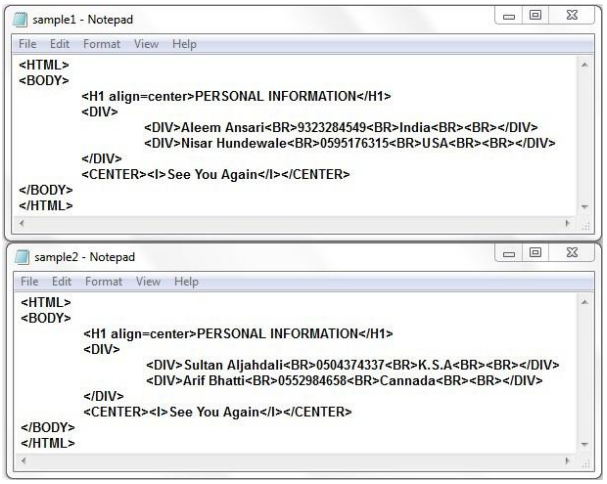
3.1 Outline of our Approach

One key function involved in our approach is how to detect irrelevant components among similar web pages, and similar component within the page itself (if they contain multiple existing objects). We seek inspiration from the following observations:

1. We observe that all pages about the same object such as product description from a single website uses same template for displaying informative content [3]. These pages also contains similar irrelevant components i.e. header, navigation bar, advertisement, etc. For example in figure 2 we can see that the two web pages displays personal information about two people, which are relevant content. However the header and footer are irrelevant contents which are common in both the web pages. This is because in most such cases, they are generated by the same dynamic programs or templates that are carefully maintained by the developers. Thus, we can make use of the stable representation structure of the pages within a web site to identify useful contents [3].



(a)



(b)

Fig 2: (a) Sample web pages displaying personal information (b) HTML source of the web pages

2. After removing irrelevant content from the pages, we can expect OREG to appear in a contiguous area on the screen as well as in the raw source of web pages (e.g. tag tree).

3. If we compare the pages using html elements as unit we can see that they differ in OREG which contains descriptions of individual ODATA. Thus by devising suitable set of similarity measures and content features we can easily differentiate between irrelevant component on web pages as having similar content and OREG containing product description as having distinct content [3].

Based on the above observations, we devise the following workflow to detect OREG and extract the corresponding ODATA from complex structured or semi-structured web pages.

1. We first download similar pages from a specific website, we call this PSET.

2. We the compare each webpage pi with a set of similar web pages PREL such that PREL ⊂ PSET. We use DOM tree of web pages as the feature representation for similarity calculation. HTML tag and texts are considered as content of the node. Because pages similar to pi are generated using the same template, they should have similar structures and many common terms as a whole.

.

3. From DOM tree of the web pages, we can expect the subtrees corresponding to irrelevant content to be similar while those corresponding to OREG to have significant difference as they represent different ODATA. In order to identify OREG we calculate the novelty value for each node as explained in the next section. The subtree that has maximum novelties belongs to OREG [3].

4. Next we identify different ODATA within OREG. This is done by traversing the subtrees within OREG. After extracting each ODATA we output them in the XML file.

3.2 Detecting Object Region Using Similar Web Pages

We first make use of Cobra parser (the one available at http://sourceforge.net/projects/xamj) to generate the DOM trees from the Web pages. Each node in this tree is an HTML element. The OREG is a subtree containing individual ODATA. The unique feature of OREG is that it is usually the largest contiguous region in the web page having distinct content which are used to represent different ODATA. Thus for identifying OREG we define the concept of repeatability for nodes in the DOM trees [3].

Definition 1: To define repeatability, we first define node similarity. Two nodes n1 (with tag tag1 and text text1) and n2 (with tag tag2 and text text2) in parse tress T1 and T2 respectively are similar if they have same tags and text content. Thus we can define similarity as:

$$sim(n1, n2) = s(tag1, tag2) + sa(text1, text2) \quad (\mathbf{1})$$

$$s(tag1, tag2) = \begin{cases} 1 & if\ tag1 = tag2\ are \\ & identical\ in\ string\ matching \\ 0 & otherwise \end{cases}$$

$$sa(text1, text2) = \begin{cases} 1 & if\ text1 = text2\ are \\ & identical\ in\ string\ matching \\ 0.5 & if\ text1, text2\ belongs\ to \\ & same\ data\ type(int, float, etc) \\ 0 & otherwise \end{cases}$$

Definition 2: Suppose nodes n1 and n2 are in parse trees T1 and T2 respectively, we define the repeatability of node n1 with respect to T2 as:

$$R(n1, T2) = \max_{\forall n \in nodes(T2)} \big(sim(n1, n) + sim\big(parent(n1), parent(n)\big)\big) \quad (\mathbf{2})$$

where parent(.) denotes the parent node of n; From Definition 2, two nodes will have high repeatability if they have similar tag and text content.

Definition 3: The novelty of node n1 is defined as:

$$N(n1) = 4 - R(n1, T2) \quad (\mathbf{3})$$

Definition 4: If tree(n) is a subtree rooted at node n1 in DOM tree T1, the novelty of tree(n1) is the weighted sum of the novelties of its child nodes, namely,

$$N(tree(n1)) = N(n1) + \sum N\Big(tree\big(child(n1)\big)\Big) \quad (\mathbf{4})$$

where child(n1) is the child of node n1. Thus tree(child(n1)) is the subtree rooted at the child(ren) of n1. N(tree(n1)) is calculated recursively based on formula (4). Thus N(tree(n1)) is the sum of the novelties of all the involved nodes. Typically, the subtree corresponding to OREG will have the highest tree novelty value. Based on Definition (4), we expect the parent and higher level nodes of subtree to have the largest tree novelty value among its siblings. For example we can see that in figure 3 the novelty of tree body node is highest because it comes from the sum of its child nodes.

Definition 5: OREG for parse tree tree(n0) is the subtree tree(ni) existing in the path from n0 to the leaf with maximal novelty [3].

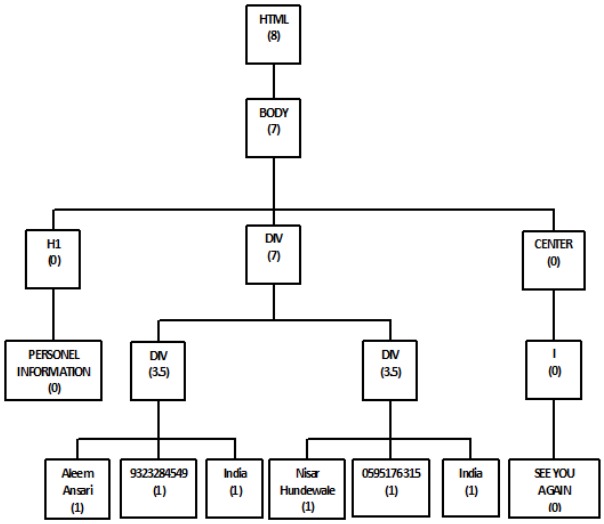


Fig 3: A sketch of tree novelty distribution.

We design an algorithm to detect OREG based on the above discussion.

DetectOREG(ParseTree T0, ParseTree T1)

```
{
  For each node n1 in parse tree T0
    {
        Initialize maximum similarity, maxsim=0;
        For each node n2 in parse tree T1
        {
           Calculate Similarity
           simval = (sim(n1,n2)+ ( (sim(pn1,pn2))) );
           if (simval>maxsim)
           {
              (maxsim = simval);
           }
        }
        Set novelty of node n1, nov = (4 - maxsim);
    }
    Recursively calculate tree novelty for each node n1
inparse tree T0.
    Traverse the parse tree T0 till we get only one subtree
(OREG) having many child subtrees with non-zero novelties.
}
```

3.3 Detecting Object Data from the Object Region

We know that ODATA occurs in the subtrees of the root of OREG. The root of each ODATA has non-zero novelty because it is the summation of novelties from child node(s) which are mostly non-zero as they represent distinct values of individual data record. Hence we traverse each subtree with non-zero novelty to extract the data record. We also know that noisy data items will have zero novelties and hence they can

be excluded. Following is the algorithm for extracting ODATA from an OREG.

ExtractODATA(DataRegionParseTree T0)
{
For each node n1 in parse tree T0
{
Detect immediate child nodes CNodes of node n1 with non-zero novelties.
For each child node cn in CNodes
{
Traverse all child nodes of cn using
Depth First Search algorithm.
Extract each text node with non-zero novelty.
}
}
}

We can also make the algorithm computationally faster for subsequent pages by noting the position of the root of the OREG. Thus for the remaining pages we can make all the computation only for the nodes that falls below the root of the OREG.

4. EXPERIMENT

The proposed work is has been implemented and tested using java. To analyze the validity and performance of our algorithm we made it to pass through various set of web pages downloaded from different websites.

Figure .4 shows sample input web pages from Amazon website and the extracted datasets.

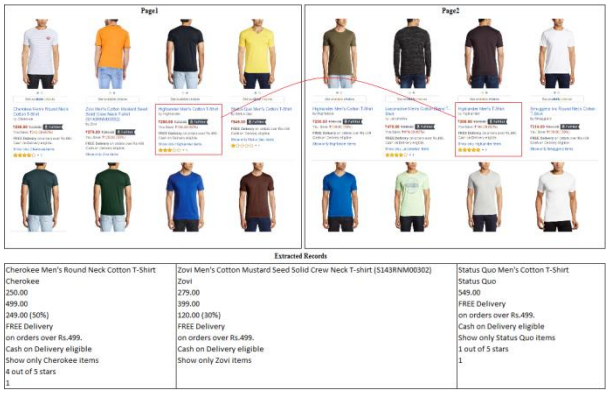


Fig 4: Extracted records from Amazon website.

We can see in the figure .4 that the algorithm extracted only three records and not four. Reason is the third record (marked in red) is common in both the pages. However algorithm will extract the record in next page comparisons. The positive side we can see in the above output is that the algorithm extracts all the records correctly.

We use the standard metrics recall and precision. Recall is computed as the ratio between the number of relevant records extracted by the system and the total number of records that should have been extracted. Precision is computed as the ratio between the number of relevant records extracted and the total number of records extracted by the system. These are the most important metrics in what refers to web data extraction applications because they measure the system performance at the end of the whole process.

Figure 5 shows precision and recall achieved by our technique on different websites.

Website	#Input records	#Extracted records	#Incorrect records	Precision	Recall
http://www.amazon.com	48	22	0	100%	46%
http://www.bestbuy.com	15	15	0	100%	100%
http://www.bookshopofindia.com	15	15	0	100%	100%
http://www.cnet.com	10	10	0	100%	100%
http://www.ebay.in	50	50	0	100%	100%
http://www.gadgetsguru.in	15	0	15	0%	0%
http://www.imdb.com	100	100	0	100%	100%
http://www.randomhouse.co.in	16	0	16	0%	0%
http://shopping.rediff.com	30	0	30	0%	0%

Fig 5: Precision and Recalls for Pages from different websites.

As seen the case with Amazon website in figure .4, the reason for low recall is that the same record is repeated on other pages. The system however extracts the records with high precision.

The tests performed with these pages were used to adjust our algorithm. Our algorithm is able to extract the data record from all above web pages efficiently. However we found that the algorithm gives wrong output if the data region contains only single data record. The reason is that the algorithm assumes that data region always has two or more subtrees with non-zero tree novelties and hence two or more data records. The algorithm also gives wrong output when each data record is displayed in single cell of rows of table. In this case it consider entire table row as a single data record and hence displays multiple records as a single record.

5. CONCLUSION

In this paper we proposed a robust unsupervised approach for extracting data records from the web pages. The proposed method does not make any strict assumption about the structure or layout of the web page and hence can scale well for almost all websites. Also the number of comparisons made in proposed approach is significantly lesser than the other approaches. The practical implementation of the proposed work has few limitations. Its working is dependent on the output of the parser. We know that under some situation the parser is unable to generate the correct node list which is used for further processing.

The future work would be to overcome the limitations of the proposed work's implementation by adapting it to better parser(s). We need to find or develop correct parsers for parsing the webpage so that we can get more accuracy in detecting OREG and hence ODATA. Proper identification of individual data records displayed in individual cells of table rows also needs to be handled properly. Extracting data fields from the data records contained in the data region will be the next step in our future work. We also need to integrate the extracted data from different websites into a single collection.

This collection of data can then be used for various Knowledge Discovery Applications such as making comparative study of products or services from various companies, smart shopping, etc.

6. REFERENCES

[1] Breuel, Thomas. 2003. Information extraction from HTML documents by structural matching. U.S. Patent Application 10/248,681.

[2] Álvarez, M., Pan, A., Raposo, J., Bellas, F., & Cacheda, F. 2010. Finding and extracting data records from web pages. Journal of Signal Processing Systems, 59(1), 123-137.

[3] Ye, S., & Chua, T. S. 2006. Learning object models from semistructured web documents. Knowledge and Data Engineering, IEEE Transactions on, 18(3), 334-349.

[4] Hsu, C. N., & Dung, M. T. 1998. Generating finite-state transducers for semi-structured data extraction from the web. Information systems, 23(8), 521-538.

[5] Kushmerick, N. 2000. Wrapper induction: Efficiency and expressiveness. Artificial Intelligence, 118(1), 15-68.

[6] Muslea, I., Minton, S., & Knoblock, C. 1999. A hierarchical approach to wrapper induction. In Proceedings of the third annual conference on Autonomous Agents (pp. 190-197). ACM.

[7] Pinto, D., McCallum, A., Wei, X., & Croft, W. B. 2003. Table extraction using conditional random fields. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval (pp. 235-242). ACM.

[8] Embley, D. W., Jiang, Y., & Ng, Y. K. 1999. Record-boundary discovery in Web documents. In ACM SIGMOD Record (Vol. 28, No. 2, pp. 467-478). ACM.

[9] Buttler, D., Liu, L., & Pu, C. 2001. A fully automated object extraction system for the World Wide Web. In Distributed Computing Systems, 2001. 21st International Conference on. (pp. 361-370). IEEE.

[10] Liu, B., Grossman, R., & Zhai, Y. 2003. Mining data records in Web pages. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 601-606). ACM.

[11] Zhai, Y., & Liu, B. 2006. Structured data extraction from the web based on partial tree alignment. Knowledge and Data Engineering, IEEE Transactions on, 18(12), 1614-1628.

[12] Dong, Y., & Li, Q. 2009. A Robust Approach of Automatic Web Data Record Extraction. Journal of Computer Information Systems, 5(6), 1757-1766.

[13] Sharma, A. K. 2011. Hidden Web Data Extraction Using Dynamic Rule Generation. International Journal on Computer Science & Engineering, 3(8).

[14] Hong-ping, C., Wei, F., Zhou, Y., Lin, Z., & Zhi-Ming, C. 2009. Automatic Data Records Extraction from List Page in Deep Web Sources. In Information Processing, 2009. APCIP 2009. Asia-Pacific Conference on (Vol. 1, pp. 370-373). IEEE.

[15] Marini, J. 2002. Document Object Model. McGraw-Hill, Inc.