International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:268
Vol.2, No.1, January 2013 DOI:10.15693/ijaist/2013.v2i1.26-31

Non-Duplicate Data Extraction in Web Databases by Combining Tag and Value Similarity

Deepika.J
M.E. CSE
Sethu Institute of Technology

ABSTRACT Web databases generate query result pages based on a user's query. Automatically extracting the data from these query result pages is very important for many applications, such as data integration, which need to cooperate with multiple web databases. The novel data extraction and alignment method called CTVS that combines both tag and value similarity is enhanced by using Unsupervised duplicate detection algorithm(UDD). CTVS automatically extracts data from query result pages by first identifying and segmenting the query result records (QRRs) in the query result pages and then aligning the segmented QRRs into a table, in which the data values from the same attribute are put into the same column. Specifically, new techniques are proposed to handle the case when the QRRs are not contiguous, which may be due to the presence of auxiliary information, such as a comment, recommendation or advertisement, and for handling any nested structure that may exist in the QRRs. Also a new record alignment algorithm that aligns the attributes in a record, first pairwise and then holistically, by combining the tag and data value similarity information is designed. Experimental results show that enhanced CTVS achieves high precision with duplicate detection and outperforms existing state-of-the-art data extraction methods.

KEYWORDS
Data extraction, Automatic wrapper generation, Duplicate detection, Data record alignment, Information retrieval, Data alignment, Web databases.

1. INTRODUCTION

ONLINE databases, called web databases, comprise the deep web [6] and [9]. Compared with webpages in the surface web, which can be accessed by a unique URL, pages in the deep web are dynamically generated in response to a user query submitted through the query interface of a web database. Upon receiving a user's query, a web database returns the relevant data, either structured [10] or semistructured [8], encoded in HTML pages. Many web applications, such as metaquerying, data integration and comparison shopping, need the data from multiple web databases. For these applications to further utilize the data embedded in HTML pages, automatic data extraction is necessary. Only when the data are extracted and organized in a structured [1] manner, such as tables, can they be compared and aggregated. Hence, accurate data extraction is vital for these applications to perform correctly. This paper focuses on the problem of automatically extracting data records that are encoded in the query result pages generated by web databases. In general, a query result page contains not only the actual data, but also other information, such as navigational panels, advertisements, comments, information about hosting sites, and so on. The goal of web database data extraction is to remove any irrelevant information from the query result page, extract the query result records (referred to as QRRs in this paper) from the page, and align the extracted QRRs into a table such that the data values belonging to the same attribute are placed into the same table column.

The following two-step method, called Combining T ag and V alue Similarity (CTVS), to extract the QRRs from a query result page p is employed.

1 Record extraction identifies the QRRs in p and involves two substeps: data region identification and the actual segmentation step.
2 Record alignment aligns the data values of the QRRs in p into a table so that data values for the same attribute are aligned into the same table column.

CTVS [12] accurately extracts and aligns the QRRs in query result pages if there are at least two records in the page. Compared with existing data extraction methods, CTVS improves data extraction accuracy in three ways.

1. New techniques are proposed to handle the case when the QRRs are not contiguous in p, which may be due to the presence of an auxiliary information, such as a comment, recommendation, or advertise-ment.
    a. An adapted data region identification method is proposed to identify the

noncontiguous QRRs that have the same parents according to their tag similarities.

b. A merge method is proposed to combine different data regions that contain the QRRs into a single data region. Our experimental results show that the two techniques are effective for addressing the noncontiguous data region problem.

2. A novel method is proposed to align the data values in the identified QRRs, first pairwise then holistically, so that they can be put into a table with the data values belonging to the same attribute arranged into the same table column. Both tag structure similarity and data value similarity are used in the pairwise alignment.
3. A new nested-structure processing algorithm is proposed to handle any nested structure in the QRRs after the holistic alignment. Unlike existing nested-structure processing algorithms that rely on only tag information, CTVS uses both tag and data value similarity information to improve nested-structure processing accuracy.

2. QRR EXTRACTION

Fig. 1 shows the framework for QRR extraction. Given a query result page, the Tag Tree Construction module first constructs a tag tree for the page rooted in the <HTML> tag. Each node represents a tag in the HTML page [3] and its children are tags enclosed inside it. Each internal node n of the tag tree has a tag string ts_n, which includes the tags of n and all tags of n's descendants, and a tag path tp_n, which includes the tags from the root to n.

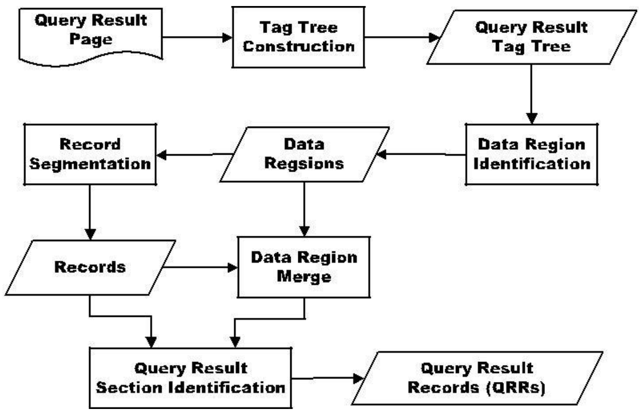

Fig. 1. QRR extraction framework.

Next, the Data Region Identification module identifies all possible data regions, which usually contain dynamically generated data, top down starting from the root node. The Record Segmentation module then segments the identified data regions into data records according to the tag patterns in the data regions. Given the segmented data records, the Data Region Merge module merges the data regions containing similar records. Finally, the Query Result Section Identification module selects one of the merged data regions as the one that contains the QRRs.

3. QRR ALIGNMENT

QRR alignment is performed by a novel three-step data alignment method that combines tag and value similarity.

1. Pairwise QRR alignment aligns the data values in a pair of QRRs to provide the evidence for how the data values should be aligned among all QRRs.
2. Holistic alignment aligns the data values in all the QRRs.
3. Nested structure processing identifies the nested structures that exist in the QRRs.

3.1 Pairwise QRR Alignment

During the pairwise alignment, it is required that the data value alignments must satisfy the following three constraints:

1. Same record path constraint. The record path of a data value comprises the tag from the root of the record to the node that contains the data value in the tag tree of the query result page. Each pair of matched values should have the same tag path.
2. Unique constraint. Each data value can be aligned to at most one data value from the other QRR.
3. No cross alignment constraint.

3.2 Holistic Alignment

Given the pairwise data value alignments between every pair of QRRs, the step of holistic alignment performs the alignment globally among all QRRs to construct a table in which all data values of the same attribute are aligned in the same table column. Intuitively, if we view each data value in the QRRs as a vertex and each pairwise alignment between two data values as an edge, the pairwise alignment set can be viewed as an undirected graph. Thus, our holistic alignment problem is equivalent to that of finding connected components in an undirected graph. Each connected component of the graph represents a table column inside which the connected data values from different records are aligned vertically. While there are many efficient algorithms for finding connected components in the Graph Theory literature, we need to consider two application constraints that are specific to our holistic alignment problem.

1. Vertices from the same record are not allowed to be included in the same connected component as they are considered to come from two different attributes of the record. If two vertices from the same record breach this constraint, a breach path exists between the two.
2. Connected components are not allowed to intersect each other. If C_1 and C_2 are two connected components, then vertices in C_1 should be either all on the left side of C_2 or all on the right side of C_2, and vice versa (i.e., no edge in C_1 cuts across C_2, and no edge in C_2 cuts across C_1).

Accordingly, a 3-step algorithm is designed for the holistic alignment problem. First, traverse the graph once by a depth-first search to discover the preliminary connected components. At the same time mark those components containing breach paths. Next, traverse the components containing breach paths to remove some edges so as to break the breach paths (i.e., enforcing the first constraint). Finally, use a divide-and-conquer method to identify and split up the intersecting components to enforce the second constraint. The function HolisticAlign enforces the no-intersecting-components constraint by a divide and conquer strategy that divides the connected component list into two sublists. Given the graph representing the QRRs and their pairwise alignments, first traverse the graph by a depth-first search algorithm and discover all the connected compo-nents in the graph. During the traversal, a color array is used to indicate whether each vertex has been visited or not. In the Visit function, when a new vertex is encountered, add it into the current connected component. Also check whether other vertices in the same record have been visited or not. If so, a flag is set to true to signify that this component contains at least one breach path.

3.3 Nested Structure Processing

Holistic data value alignment constrains a data value in a QRR to be aligned to at most one data value from another QRR. If a QRR contains a nested structure such that an attribute has multiple values, then some of the values may not be aligned to any other values. Therefore, nested structure processing identifies the data values of a QRR that are in the generated by nested structures (i.e., the repetitive parts of a generating template). Relying only on HTML tags to identify nested structures, as is done by almost all existing methods, may incorrectly identify a plain structure as a nested one. To overcome this problem, CTVS uses both the HTML tags and the data values to identify the nested structures. Given an aligned table, a nested column comprises at least two ordered sets representing the data values that are generated by repetitive parts in the template. A nested column set C is comprised of a set of nested columns. The nested structure identification algorithm shown in Fig. 2 first identifies the nested column set C and then creates a new row for each combination of a repetitive subpart. Given all QRRs, the tag tree T for the query result page p and the QRR's holistic alignment columns as input, the procedure nest_processing tries to find and process any nested structure in p. The nested column set C is initialized to be an empty set (line 1). For each QRR with record root node t in T , the procedure nest_column_identify is invoked to identify any nested columns in the QRR (lines 2 and 3). After all the nested columns are identified, a new row is generated (lines 4 and 5) by copying the remaining parts as well as the repetitive data values. The copy of the repetitive data values is removed from the original row.

Given a node t in tag tree T , the holistic alignment columns and the nested column set C as input, the procedure nest_column_identify identifies all repetitive parts under t in T . This procedure is called recursively until it reaches a node that contains only one data value (lines 6-8). Hence, nested column identification is performed from leaf nodes of T to the root. For each node, we identify the repetitive tag pattern in its children (line 9). Suppose there is a repetitive tag pattern found in t's children, each of which contains a data value of the record. For each tag repetition p that contains data value $f_1; . . . ; f_n$, c_p is defined to be the columns in the holistic alignment that contain $f_1; . . . ; f_n$. We now need to decide, according to the data value similarity in the columns c_p, whether the repetitive tag is generated from a nested structure or it is actually a flat structure (lines 10-12). If it is generated by a nested structure, c_p is added to the nested column set C.

Given columns cp in a holistic alignment and a similarity threshold Snest as input, the procedure nested decides, using the similarities of the data values in cp, whether cp contains a repetitive tag pattern that is formed by a nested structure. It is assumed that two columns are generated by the same attribute if there is a large data value similarity between these two columns. Given a column c1, which contains m data values, define the intracolumn similarity simintra to be the average data value similarity within each column in c1. For cp, its intracolumn similarity is the average of the intracolumn similarity of all columns in cp (line 13). For two columns c1 and c2, which have m and n data values, respectively, the intercolumn similarity siminter is defined to be the average data value similarity of every pair of data values in c1 and c2 (line 14).

After siminter and simintra are calculated for identified columns cp, if siminter=simintra > Snest, where Snest is a threshold that is set to 0.5, cp is assumed to be a nested column set, which means that the data values in it are generated from a nested structure. Given data columns c_p and the nested column set C as input, the procedure add_nested_column adds the nested columns c_p to C. If there is a nested column c_i in C that has overlap with c_p (line 19), which means that the repetitive part has been identified in a previous QRR, then c_i in C is replaced with c_p [c_i (line 20). Otherwise, c_p is simply added as a new element into C (line 22). Given n records with a maximum of m data values and a maximum tag string length of l, the time complexity of the nested structure processing algorithm is $O(nl^2m^2)$.

```
Procedure nest_processing (QRRs, T, holistic_align)
    1. C ← Ø
    2. for each QRR with record root t
    3. nest_column_identify(t, T, holistic_align, C)
    4. for each column pattern $c_p$ in C do
    5. create a new row for each repeated subpart
Procedure nest_processing (QRRs, T, holistic_align)
    6. If(t contains more than one data value) then
    7. for each child $t_i$ of t do
    8. nest_columun_identify($t_i$, T, holistic_align, C)
    9. for each repetition p of any consecutive maximum
repetitive tag pattern found in t's children
    10. $C_p$ = data columns for p in holistic_align
    11. If $c_p \not\subset$ C and nested( $c_p$, $S_{nest}$) then
    12. add_nested_column($c_p$, C)
Function boolean nested($c_p$, $S_{nest}$)
    13. $sim_{intra}$ ← intra-column similarity within $c_p$
    14. $Sim_{inter}$ ← inter-column similarity within $c_p$
    15. if($Sim_{inter}$ / $Sim_{intra}$ > $S_{nest}$) then
    16. return true
    17. else return false
Procedure add_nested_column( $c_p$, C)
    18. for each element $c_i$ in C do
    19. if( $c_p \cap c_i \neq Ø$ ) then
    20. C ← C - $c_i$ + $c_p$ U $c_i$
    21. break
    22. If no element in C shares a common column with
$c_p$ then C ← C + $c_p$
```

Fig. 2. Nested structure identification algorithm

Compared with the nested structure processing methods in DeLa [5] and NET [2] , the nested structure processing method in CTVS has the following advantages.

1. CTVS processes the nested structures after the data records are aligned rather than before as is the case in DeLa and NET. Processing the nested structure before the records are aligned makes them vulner-able to optional attributes since the optional attri-butes make the tag structure irregular. This problem does not occur in CTVS.
2. In CTVS the data value similarity information effectively prevents a flat structure from being identified as a nested structure. Because it shares similar tag structures, a flat structure with several columns having the same tag structure, might be mistakenly identified as a nested structure in DeLa and NET.

4. DUPLICATE DETECTION

In this section, the assumptions on which UDD algorithm shown in Fig. 3 (Unsupervised Duplicate Detection) is based are made as follows:

1. A global schema for the specific type of result records is predefined and each database's individual query result schema has been matched to the global schema.
2. Record extractors, i.e., wrappers, are available for each source to extract the result data from HTML pages and insert them into a relational database according to the global schema.

In UDD the weights are adjusted dynamically and it uses two classifiers [11], WCSS (Weighted Component Similarity Summing) and SVM (Support Vector Machine), that cooperatively can prevent the problem of classifying the results of previous iteration to the next iteration which are vulnerable to false data's. UDD'S performance is not very sensitive to the false negative cases, i.e., actual duplicates from the same data source. UDD is faster than PEBL and Christen's method which require more iterations than UDD to identify all duplicates. UDD tackles a slightly different classification problem, online duplicate record detection for multiple Web databases. In this scenario, the assumption that most records from the same data source are non-duplicates usually holds, i.e., negative examples are assumed without human labeling, which helps UDD overcome from giving only the positive set of training examples.

5. EXPERIMENTS

The experimental results for CTVS over five data sets is presented here and then compare CTVS with ViNTs, DeLa , and ViPER. ViNTs and DeLa are chosen to compare with CTVS because both have been shown to perform very accurate data extraction and implementations of both are available to us.

5.1 Data Sets

Five data sets are used to compare the performance of CTVS, ViNTs, and DeLa. Data set 1 (PROFUSION) is obtained from ViNTs'testbed, which contains 100 websites collected from profusion.com. Twenty of the 100 websites return relational records, such as jobs and entertainment records, and 80 return documents. For each of the 100 websites, 10 queries are submitted and the first 10 result pages are manually collected. A no-result page is also collected for each website by submitting a nonexistent term as a query to the website. For each website, its no-result page and five randomly selected result pages from the 10 result pages are used to build a wrapper, which is used to extract the QRRs from the remaining five result pages. Data set 2 (E-COMM) contains 100 E-commerce deep websites in six popular domains: book, hotel, job, movie, musicRecord, and automobile. Data set 3 is the TestBed for information extraction from Deep web (TBDW) version 1.02, which is available at http:// daisen.cc.kyushu-u.ac.jp/TBDW/.

```
Input:

Potential duplicate vector set P

Non-duplicate  vector set N

Output:

Duplicate vector set D

Algorithm:

    1.  D = φ
    2.  Set the parameters W of $C_1$ according to N
    3.  Use $C_1$ to get a set of duplicate vector pairs $d_1$ from P
    4.  Use $C_1$ to get a set of duplicate vector pairs $f$ from N
    5.  P= P- $d_1$
    6.  While | $d_1$ | ≠ 0
    7.  $N^{'} = N - f$
    8.  $D = D + d_1 + f$
    9.  Train $C_2$ using  D and $N^{'}$
    10. Classify P using $C_2$ and get a set of newly identified
        duplicate  vector  pairs  $d_2$
    11. P =  P - $d_2$
    12. D = D + $d_2$
    13. Adjust the parameters W of  $C_1$ according to $N^{'}$ and D
    14. Use $C_1$ to get a new set of duplicate vector pairs $d_1$
        from p
    15. Use $C_1$ to get a new set of duplicate vector pairs $f$
        from N
    16. N = $N^{'}$
    17. Return D
```

Fig.3. Unsupervised duplicate detection algorithm

It includes 51 online databases from which five query result pages are created for each database. For CTVS, we directly use the first page in each database as the test page. Data set 4 (AUXI) focuses on webpages in which the QRRs are segmented into different data regions due to auxiliary information. Query result pages from 80 websites were collected, whose query result records have at least one auxiliary node. Data set 5 (NESTED) focuses on the query result pages that include nested structure.

5.2 Results and Discussions

The performance of the data extraction methods is compared in three different ways.

Table 1. Data extraction methods comparison

	Nested Structure Processing	Single Result Page	Non-contiguous Data Regions
CTVS	√	√	√
DeLa	√	√	×
ViPER	×	√	×
ViNTs	×	×	×

General data set evaluation presents the performance on the first three data sets, which exhibit a variety of properties and have been used in previous work by others. The other two evaluations focus on specific properties of the query result pages. Noncontiguous [13] QRR evaluation compares the performance for query result pages in which the QRRs are contiguous and noncontiguous. Previous works on Non-contiguous data extraction are based on MDR(Mining Data Records in web pages) algorithm. Existing automatic techniques are not satisfactory because of their poor accuracies. Nested-structure evaluation compares the performance for query result pages with and without a nested structure.

Table. 1 summarizes some characteristics of the data extraction methods [4],[5],[7] and [12] compared in this paper . CTVS has better performance than ViNTs, ViPER and DeLa in both nonnested and nested pages. CTVS achieves high record-level precision and record-level recall than ViNTs and DeLa in the data sets used for information extraction. The single page result column indicates whether a single query result page from a data source is sufficient to extract data.

6. CONCLUSION

A novel data extraction method, CTVS, to automatically extract QRRs from a query result page is presented in this paper. CTVS employs two steps for this task. The first step identifies and segments the QRRs. Existing techniques are improved by allowing the QRRs in a data region to be noncontiguous. The second step aligns the data values among the QRRs. A novel alignment method is proposed in which the alignment is performed in three consecutive steps: pairwise alignment, holistic alignment, and nested structure processing. Experi-ments on five data sets show that CTVS is generally more accurate than current state-of-the-art methods.

7. ACKNOWLEGEMENT

Thanks to the experts who have contributed towards the development of this material.

8. REFERENCES

[1] Arasu and H. Garcia-Molina, “Extracting Structured Data from Web Pages,” Proc. ACM SIGMOD Int’l Conf. Management of Data, pp. 337-348, 2003.

[2] B. Liu and Y. Zhai, “NET - A System for Extracting Web Data from Flat and Nested Data Records,” Proc. Sixth Int’l Conf. Web Information Systems Eng., pp. 487-495, 2005.

[3] D. Buttler, L. Liu, and C. Pu, “A Fully Automated Object Extraction System for the World Wide Web,” Proc. 21st Int’l Conf. Distributed Computing Systems, pp. 361-370, 2001.

[4] H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu, “Fully Automatic Wrapper Generation for Search Engines,” Proc. 14th World Wide Web Conf., pp. 66-75, 2005.

[5] J. Wang and F.H. Lochovsky, “Data Extraction and Label Assignment for Web Databases,” Proc. 12th World Wide Web Conf., pp. 187-196, 2003.

[6] K.C.-C. Chang, B. He, C. Li, M. Patel, and Z. Zhang, “Structured Databases on the Web: Observations and Implications,” SIGMOD Record, vol. 33, no. 3, pp. 61-70, 2004.

[7] K. Simon and G. Lausen, “ViPER: Augmenting Automatic Information Extraction with Visual Perceptions,” Proc. 14th ACM Int’l Conf. Information and Knowledge Management, pp. 381-388, 2005.

[8] L. Chen, H.M. Jamil, and N. Wang, “Automatic Composite Wrapper Generation for Semi-Structured Biological Data Based on Table Structure Identification,” SIGMOD Record, vol. 33, no. 2, 58-64, 2004.

[9] M.K. Bergman, “The Deep Web: Surfacing Hidden Value,” White Paper, BrightPlanet Corporation, http://www.brightplanet.com/resources/details/deepweb.html, 2001.

[10] W. Cohen and L. Jensen, “A Structured Wrapper Induction System for Extracting Information from Semi-Structured Documents,”Proc. IJCAI Workshop Adaptive Text Extraction and Mining, 2001.

[11] Weifeng Su, Jiying Wang, and Fredrick H.Lochovsky, “Record Matching Over Query Results from Multiple Web Databases”, IEEE Trans. Knowledge and Data Eng., vol. 22, no. 4, April 2010.

[12] Weifeng Su, Jiying Wang, and Fredrick H.Lochovsky, “Combining Tag and Value Similarity for Data extraction and Alignment”, IEEE Trans. Knowledge and Data Eng., vol. 24, no. 7, July 2012.

[13] Y. Zhai, R. Grossman and B. Liu, “ Mining Web Pages for Data Records”, IEEE, Intelligent Systems, vol. 19, no. 6, 2005.

Author Profile

J. Deepika received her B.E degree in computer science and engineering from K.L.N. College of Engineering, Sivagangai, India, in 2011. She is currently pursuing her M.E. in computer science and engineering from Sethu Institute of Technology, Virudhunagar, India. Her research interests include web search, web information extraction, the deep web, data Integration, and information retrieval.