IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 4, Ver. IV (Jul – Aug. 2014), PP 09-22
www.iosrjournals.org

I-ViDE: An Improved Vision-Based Approach for Deep Web Data Extraction

Mrudula Varade[1], Vimla Jethani[2]
[1](Computers, RAIT D.Y.Patil/ University of Mumbai, India)
[2](Computers, RAIT D.Y.Patil/ University of Mumbai, India)

Abstract: *Deep Web contents are accessed by queries submitted to Web databases and the returned data records are enwrapped in dynamically generated Web pages (they will be called deep Web pages in this paper). Extracting structured data from deep Web pages is a challenging problem due to the underlying intricate structures of such pages. Until now, a large number of techniques have been proposed to address this problem, but all of them have inherent limitations because they are HTML language dependent .Visual features are not taken into consideration. All previous methods are mostly dependent on table tags. A Vision based approach for web data extraction has overcome the limitations of previous work by utilizing some interesting common visual features on the web page. But still this approach has one drawback that it can process web page containing only one data region. Due to processing of one data region it reduces the precision and recall rate. As precision give us the rate that how many correct data records are extracted from relevant data records and recall give us the rate that how many relevant data records are extracted from overall data records. The proposed Improved-ViDE approach handles multi data-region in deep web pages which can improve the precision rate and recall rate.*

Keywords: *Web mining, Web data extraction, visual features of deep Web pages.*

I. Introduction

The World Wide Web has more and more online web databases which can be searched through their web query interfaces. The number of Web databases has reached 25 million according to a recent survey. A web database is an organized listing of web pages. It's like the card catalogue that you might find in the library. The database holds a "surrogate" (or selected pieces like the title, the headings, etc.) for each web page. The creation of these surrogates is called "indexing", and each web database does it in a different way. Web databases hold surrogates for anywhere from 1 million to several billion web pages. The program also has a search interface, which is the box you type words into (like in Alta Vista or Google) or the lists of directories you pick from (like in Yahoo). Thus, each web database has a different indexing method and a different search interface. Web data Extraction is a type of information retrieval whose goal is to automatically extract structured information, i.e. categorized and contextually and semantically well-defined data from a certain domain, from unstructured machine-readable documents. All the Web databases make up the deep Web (hidden Web or invisible Web). Often the retrieved information (query results) is enwrapped in Web pages in the form of data records. These special Web pages are generated dynamically and are hard to index by traditional crawler based search engines, such as Google and Yahoo. In this , we call this kind of special Web pages deep Web pages. Each data record on the deep Web pages corresponds to an object. For instance, Fig. 1 shows a typical deep Web page from Amazon.com. On this page, the books are presented in the form of data records, and each data record contains some data items such as title, author, etc. In order to ease the consumption by human users, most Web databases display data records and data items regularly on Web browsers. The World Wide Web will provide the information in the form structured. To provide the structured information it will need to generate the deep web pages. The deep web pages are generated based on the user query while extracting the data from web database and building the web page is a very important rule. Deep Web, as a rich and largely unexplored data source, is becoming nowadays an important research topic. In previous years, data extraction from Web pages has received a lot of attention. Much experience has been also already accumulated in the area of traditional, relational databases integration. Today, these research areas converge, leading to development of systems for Deep Web data extraction and integration. The importance of Deep Web (DW) has grown substantially in recent years not only because its size, but also because Deep Web sources arguably contain the most valuable data.

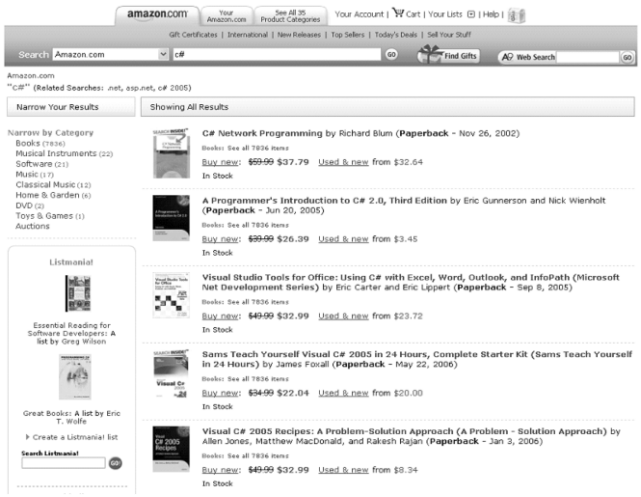


Figure 1: Deep Web page example from Amazon.com

Automatic accessing Web sites that do not provide these facilities is achieved by using Web Wrapper. Web Wrappers have some drawbacks: first, they require developers with strong knowledge on the accessed Web site because Web Wrappers are site specific solutions that have dependencies with Web structure; and second, Web Wrappers require constant maintenance in order to support new changes on the Web sites they are accessing. Web Wrappers development tools evolved trying to solve these drawbacks and also to make possible an easier integration of Web data from heterogeneous sources. The web data extraction is done by several techniques are available, they are Manual Approach, Semi Automatic Approach, and Automatic Approach. The Manual Approach in which languages were designed to assist programmer in constructing wrappers to identify and extract all the desired data items/fields. Some of the best known tools that adopt manual approaches are Minerva, TSIMMIS, and Web-OQL. Obviously, they have low efficiency and are not scalable. The Semi Automatic Approach mainly classified two types, they are sequenced based and tree based. The former, such as WIEN, Soft-Mealy, and Stalker, represents documents as sequences of tokens or characters, and generates delimiter based extraction rules through a set of training examples. The latter, such as W4F and XWrap, parses the document into a hierarchical tree (DOM tree), based on which they perform the extraction process. The user first manually labels a set of trained pages. A learning system then generates rules from the training pages. The resulting rules are then applied to extract target items from web pages. These methods either require prior syntactic knowledge or substantial manual efforts. These approaches require manual efforts, for example, labeling some sample pages, which is labor-intensive and time-consuming. In order to improve the efficiency and reduce manual efforts, most recent researches focus on automatic approaches instead of manual or semiautomatic ones. Some representative automatic approaches are Omini [2], RoadRunner [4], IEPAD [6], MDR [7], and DEPTA [9]. Some of these approaches perform only data record extraction but not data item extraction such as MDR.

To evaluate the experimental results of our improved ViDE approach, we used Precision and recall rates. Precision is the actual retrieval set may not perfectly match the set of relevant records. Recall is the ratio of the number of relevant records retrieved to the total number of relevant records in the database.

II. Related works

A number of approaches have been reported in the literature for extracting information from Web pages. There are many approaches in Web data extraction some are manual approaches, semi-automatic approaches and rest are fully automatic. In this literature we are going to survey previous automatic approaches.

2.1 MDR

In this paper, Bhing Liu and Robert Grossman had proposed a method to mine data records in Web page automatically. The algorithm is called MDR [7] i.e. Mining Data Records in web pages. It currently finds all data records formed by table and form related tags, i.e., table, form, tr, td, etc.

This algorithm is based on two observations:

A group of data records are presented in a particular region of a page and formatted using similar HTML tags.

- A group of similar data records being placed in a specific region are under the same parent in a tag tree.

The MDR algorithm works as follows:
1) The DOM tree of the input document is built.
2) It then uses a combinatorial algorithm to find so-called generalised nodes, which are subsets of nodes that fulfil the following conditions: they are siblings, they are adjacent, they have the same number of children, and the edit distance amongst them does not exceed a predefined threshold. The idea is to detect regions that contain repetitive similar structures. The edit distance is calculated on the strings that result from serialising the nodes to be compared as strings; this serialisation does not take text nodes into account, only HTML tags.
3) The subsets of generalised nodes that result from the previous step are considered as data regions since each data region is supposed to contain two or more data records that have similar structures.
4) The algorithm then separates the data records inside the previous data regions using the following heuristics: i) if the region consists of only one generalised node, it then checks if this node is not a table row, but all of its children are similar; if the condition is met, then the children are returned as independent data records; otherwise the generalised node itself is returned as a data record. ii) If the generalised node contains two or more nodes with the same number of children and these children are similar to each other, then it means that they are non-contiguous data records, i.e., the data region is an HTML table in which each data record is formatted in columns and not in rows; otherwise, the whole generalised node is returned.

2.1.1 Limitations of MDR:

This paper proposed a technique to mine the data records in a web page. The limitation of this technique is that it does not take the visual features from the page .It is html tag dependent and it does not align the data item. Rate of Precision and Recall rate is low as compared to the existing technique. MDR fails in documents that have large menus, long listings of user comments, or documents in which the relevant information is rendered using lists, divisors, or other HTML constructs.

2.2 DEPTA

Data Extraction based on Partial Tree Alignment is a two step approach. First step is to identify data record. Second step extracts data items using partial tree alignment method. DEPTA[7] can be only applicable to Web pages that contain two or more data records in a data region. However, instead of discovering repeat substring based on suffix trees, which compares all suffixes of the HTML tag strings, it compares only adjacent substrings with starting tags having the same parent in the HTML tag tree (similar to HTML DOM tree but only tags are considered). The insight is that data records of the same data region are reflected in the tag tree of a Web page under the same parent node. Thus, irrelevant substrings do not need to together as those in suffix-based approaches. Furthermore, the substring comparison can be computed by string edit distance instead of exact string match when using suffix trees where only completely similar substrings are identified. The described algorithm, called MDR [7], works in three steps. First, it builds an HTML tag tree for the Web page where text strings are disregarded. Second, it compares substrings for all children under the same parent. Third, it extracts the data records. MDR does extract the data items. To overcome this limitation DEPTA uses partial alignment algorithm for data item extraction. DEPTA [9] utilizes only one visual feature that is it just gives the boundaries of the rectangle of each HTML element.

2.2.1 Limitations of DEPTA:

Highly dependent on html tags especially only on table tag. Method of constructing a tag tree has the limitation that, the tag tree can be built correctly only as long as the browser is able to render the page correctly. Rate of precision and recall is better than MDR but low compared to existing technique.

The precision rate and recall rate of the DEPTA is poor compare to ViDE because in documents region that have large menus, long listings of user comments, or documents in which the relevant information is rendered using lists, division will not be consider in DEPTA. As precision and recall is nothing but total number of correctly extracted data records from total number of data record extracted and total number of correctly extracted from total number of data records. As ViDE overcome this drawback by utilizing visual features so precision and recall rate is better than DEPTA.

III. Vide

The previous techniques have certain limitations that they do not utilize the visual features and they are html table tag dependent. As DEPTA tried to utilize the visual feature by considering the position feature but still it was table tag dependent as it is extension to MDR technique. This ViDE [11] technique overcomes the limitations of the previous techniques by utilizing the visual features of deep Web pages.
The ViDE Approach general process steps

- Given a sample deep Web page from a Web database, obtain its visual representation and transform it into a Visual Block tree.

- Extract data records from the Visual Block tree.
- Partition extracted data records into data items and align the data items of the same semantic together.

3.1 VIPS [8]

To obtain the visual representation and to transform it into a visual block tree we use VIPS approach. Vision-based Page Segmentation, or VIPS [8] for short, is intended to find all of the regions of which a document is composed. It builds on the hypothesis that web designers provide visual cues that help people recognize the different regions of which a document is composed, e.g., horizontal or vertical rules, boxes, colored panels, special fonts, or background images. **Figure 3.1** shows the visual page segmentation of www.shop.airtel.com and **Figure 3.2** shows the visual block tree of this Airtel website.

The VIPS algorithm works as follows:

- The DOM tree of the input document is built, and it is enriched with information about visual features, e.g., position, background color, foreground color, font information, or background image.
- Initially, the algorithm assumes that the whole document is a big region; it then traverses the DOM tree level after level and analyses each node to determine if it can be considered a sub-region. To do so, the authors devised a collection of twelve heuristics, including the following: if a parent node has a child node of type hr, then that node must be divided into two sub-regions; if the background color or a node is different from the background color of one of its children, then that child is a sub-region; if a table cell does not have any sub-regions, then the next table cell should not have any sub-regions, either; and so on.
- After discovering the sub-regions in each level of the DOM tree, the algorithm calculates a set of separators, which are visual boxes that do not intersect any of the sub-regions. In other words, a separator is an empty region within a document. Each separator is assigned a weight that is related to the visual difference between the regions that it separates. The authors devised a number of heuristic to calculate the weight of a separator building on how similar the blocks it separates are, what colors they have, if a horizontal rule overlaps the separator and so on. Adjacent regions that have a separator whose weight is smaller than a predefined threshold are merged.
- Once all of the regions have been identified, they are organized into a tree that represents containment relationships, i.e., a region is the child of another region as long as the rendering box of the former is contained within the rendering box of the latter.

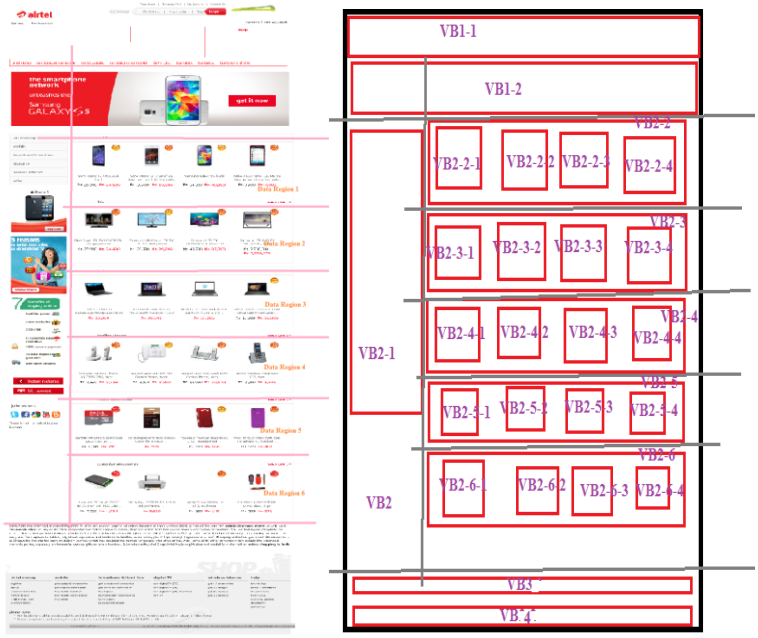


Figure 3.1: Example of web page segmentation

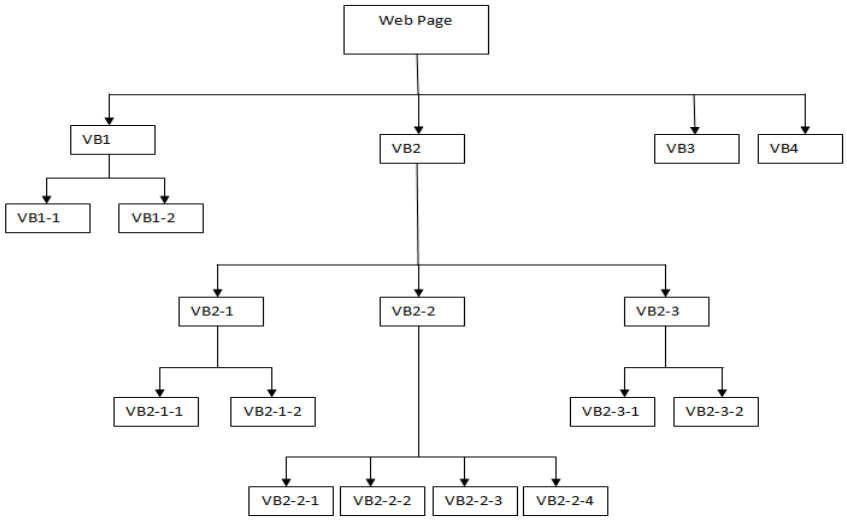


Figure 3.2: Visual Block Tree of Figure 3.1

3.2 Visual Features [13]

Position features (PFs):

-PF1: Data regions are always centered horizontally.

-PF2: The size of the data region is usually large relative to the area size of the whole page.

Layout features (LFs): These features indicate how the data records in the data region are typically arranged.

-LF1: The data records are usually aligned flush left in the data region.

-LF2: All data records are adjoining.

-LF3: Adjoining data records do not overlap, and the space between any two adjoining records is the same.

Appearance features (AFs): These features capture the visual features within data records.

-AF1: Data records are very similar in their appearances, and the similarity includes the sizes of the images they contain and the fonts they use.

-AF2: The data items of the same semantic in different data records have similar presentations with respect to position, size (image data item), and font (text data item).

-AF3: The neighboring text data items of different semantics often (not always) use distinguishable fonts.

Content feature (CF). These features hint the regularity of the contents in data records.

-CF1: The first data item in each data record is always of a mandatory type.

-CF2: The presentation of data items in data records follows a fixed order.

-CF3: There are often some fixed static texts in data records, which are not from the underlying Web database.

3.3 Data Record Extraction

Data record extraction is to discover the boundary of data records based on the LF and AF features. That is, we attempt to determine which blocks belong to the same data record. We achieve this in the following three phases:

Phase 1: Filter out some noise blocks. Because noise blocks are always at the top or bottom, we check the blocks located at the two positions according to LF1. If a block at these positions is not aligned flush left, it will be removed as a noise block. This step does not guarantee the removal of all noise blocks.

Phase 2: Cluster the remaining blocks by computing their appearance similarity.

The formula for computing the appearance similarity between two blocks b1 and b2 is given below:

$$sim(b_1, b_2) = w_i * simIMG(b_1, b_2) + w_{pt} * simPT(b_1, b_2) + w_{lt} * simLT(b_1, b_2), \quad \text{..... (1)}$$

where simIMG(b1,b2), simPT(b1,b2), simLT(b1,b2) are the similarities based on image size, plain text font, and link text font, respectively. And wi, wpt, and wlt are the weights of these similarities, respectively. **Table 3.1** gives the formulas to compute the component similarities and the weights in different cases. The weight of one type of contents is proportional to their total size relative to the total size of the two blocks.

Phase 3: Discover data record boundary by regrouping blocks.

The algorithm of regrouping blocks consists of three steps. Step 1 rearranges the blocks in each cluster based on their appearance order on the Web page, i.e., from left to right and from top to bottom (lines 1-7). In addition, a minimum bounding rectangle is formed for each cluster on the page (line 8). In Step 2, n groups are initialized with a seed block in each group as discussed earlier, where n is the number of blocks in a maximum cluster, denoted as Cmax. According to CF1, we always choose the cluster that contains the first mandatory data item of each record as Cmax. Let bmax,k denote the seed block in each initial group Gk. Step 3 determines to

which group each block belongs. If block bi,j (in Ci, Ci is not Cmax) and block bmax,k (in Cmax) are in the same data record, then bi;j should be put into the same group bmax,k belongs to. According to LF3, no two adjoining data records overlap. So, for bmax,k in Cmax, the blocks that belong to the same data record with bmax,k must be below bmax,k-1 and above bmax,k+1. For each Ci, if data record Ri is ahead of Rmax, then the block on top of Ri is ahead of (behind) the block on top of Rmax. Here, "ahead of" means "on the left of" or "above," and "behind" means "on the right of" or "below."

Table 3.1: Formulas for computing similarity between blocks [13]

formulas	remarks
$simIMG(b_1,b_2)=\frac{Min\{sa_i(b_1),sa_i(b_2)\}}{Max\{sa_i(b_1),sa_i(b_2)\}}$	$sa_i(b)$ is the total area of images in block b. $sa_b(b)$ is the total area of block b. $fn_{pt}(b)$ is the total number of fonts of the plain texts in block b. $sa_{pt}(b)$ is the total area of the plain texts in block b. $fn_{lt}(b)$ is the total number of fonts of the link texts in block b. $sa_{lt}(b)$ is the total area of the link texts in block b.
$w_i=\frac{sa_i(b_1)+sa_i(b_2)}{sa_b(b_1)+sa_b(b_2)}$	
$simPT(b_1,b_2)=\frac{Min\{fn_{pt}(b_1),fn_{pt}(b_2)\}}{Max\{fn_{pt}(b_1),fn_{pt}(b_2)\}}$	
$w_{pt}=\frac{sa_{pt}(b_1)+sa_{pt}(b_2)}{sa_b(b_1)+sa_b(b_2)}$	
$simLT(b_1,b_2)=\frac{Min\{fn_{lt}(b_1),fn_{lt}(b_2)\}}{Max\{fn_{lti}(b_1),fn_{lt}(b_2)\}}$	
$w_{lt}=\frac{sa_{lt}(b_1)+sa_{lt}(b_2)}{sa_b(b_1)+sa_b(b_2)}$	

Algorithm of block regrouping [13]

Input: $C_1, C_2, \ldots\ldots\ldots, C_m$: a group of clusters by blocks clustering from a given sample deep web page P

Output: $G_1, G_2, \ldots\ldots\ldots, G_n$: each of them corresponds to a data record on P

Begin

// Step 1: sort the blocks in C_i according to their positions in the page from top to bottom and then left to right.

For-each cluster C_i do

for any two blocks $b_{i,j}$ and $b_{j,k}$ in C_i// $1 \le j < k \le |C_i|$

if $b_{i,j}$ and $b_{j,k}$ are in different lines to P, and $b_{j,k}$ is above $b_{i,j}$

$b_{i,j} \leftrightarrow b_{j,k}$; // exchange their orders in C_i

else if $b_{i,j}$ and $b_{j,k}$ are in same line to P, and $b_{j,k}$ is in front of $b_{i,j}$

$b_{i,j} \leftrightarrow b_{j,k}$

end until no exchange occurs;

from the minimum bounding rectangle Rec_i for C_i;

// Step 2. Initialize n groups, and n is the number of data records on P

$C_{max} = \{C_i \mid |C_i| = \max\{|C_1|, |C_2|, \ldots. |C_m|\}\}$; // $n = |C_{max}|$

for each blocks $b_{max,i}$ in C_{max}

Initialize group G_i;

put $b_{max,i}$ into G_i

// Step 3: put the blocks into the right groups, and each group corresponds to a data record

for each cluster c;

if Rec_i over laps with Rec_{max} on P

if Rec_i is ahead of (behind) Rec_{max}

for each block $b_{i,j}$ in C_i

Find the nearest block $b_{max,k}$ in C_{max} that is behind (ahead of) $b_{i,j}$ on the web page;

Place $b_{i,j}$ into group G_k;

End

3.4 Data Item Extraction

These processes will, process on the results of data records extractions. This will obtain the following process those are: Data Item matching and Data Item Alignment.

Data Item Matching:

This process wills match the information of items with the actual webpage data. The data item will getting correctly or not check and give to Data Item Alignment.

```
Input: item1, item2: two data items
Output: matched or unmatched: the match result (Boolean)
Begin
1 if (font(item1) ≠ font(item2))
2     Return unmatched;
3 if (position(item1) = position(item2))
4     return matched;
5 if (itemp1 and itemp2 are matched)     // itemp1 and itemp2 are the data
items immediately in front of item1 and item2 respectively
6         return matched;
7 else
      return unmatched;
End
```

Figure 3.4: The algorithm of data item matching [13]

To process the data item matching will use the Data Item Matching algorithm. The algorithm general steps are following:

To process the operation first check two data items the absolute position in addition to having the same font. Here, absolute position is the distance between the left side of the data region and the left side of a data item.

When two data items do not have the same absolute position, they can still be matched if they have the same relative position.

For match on relative position, the data items immediately before the two input data items should be matched.

Data Item Alignment:

This process will align the data item in the web page with the specific location based on the layout process given pixel rages. The data item alignment will process based on the Data Item Alignment Algorithm. The figure 3.5 shows the algorithm of data item alignment.

The general steps process will follow:

- Put the first unaligned data items of each data records into the current Item set.
- Use Data Item Matching Algorithm top cluster the current item sets and from the group.
- Group process will give the positions that or absolute position or relevant position based on the positions aligns the current item set unique positions.
- Align the unique position data items and process the current data items, until to generate sample page.

3.5 Visual Wrapper Generation

Our wrappers include data record wrapper and data item wrapper. They are the programs that do data record extraction and data item extraction with a set of parameter obtained from sample pages. For each Web database, we use a normal deep Web page containing the maximum number of data records to generate the wrappers. The wrappers of previous works mainly depend on the structures or the locations of the data records and data items in the tag tree, such as tag path. In contrast, we mainly use the visual information to generate our wrappers.

3.6 Analysis of Previous Work

The limitations of MDR [7] and DEPTA [9] are overcome by the ViDE approach with the observation that it utilizes the visual features of the web page. In below table we can see comparison of MDR, DEPTA, and ViDE [13]. MDR [7] and DEPTA are almost same technique the only difference is data item alignment

algorithm is used in DEPTA. Due to the proper alignment of the data item precision and recall rates of DEPTA is better compare to MDR.

```
Input: a set of extracted data records {ri | 1≤i≤n}
Output: a set of data records {ri | 1≤i≤n} with all the data items aligned
Begin
1  currentItemSet=ϕ;
2  currentCluster=ϕ;
//put the first unaligned data item of each ri into currentItemSet:
// Item_i^U(i) refers to the first unaligned item of the ith data record
3   currentItemSet ↔ Item_i^U(i) (1≤i≤n);
4   while currentItemSet≠ϕ
5       use the data item matching algorithm to group the data items
in currentItemSet into k clusters {Ci | 1≤i≤k} (k≤n);
6       for each cluster Ci
7          for each rj that does not have a data item in Ci
8            if Item_j^U(j)+k is matched with data items in Ci
9                   Log position k;
10          else
11                  Log position 0;
12  Pi = max value of these logged positions for Ci;
      /*Till now, each cluster Ci has a position Pi */
13    if any PL==0
14      currentCluster=CL;
15    else
16      currentCluster=CL whose PL is max {P1, P2,…,PK};
17    for each rj whose Item_j^U(j) is in currentCluster CL
18      remove Item_j^U(j) from currentItemSet;
19      if Item_j^U(j)+1 exists in rj
20            put Item_j^U(j)+1 into currentItemSet;
21    for each rj that has no item in currentCluster CL
22      insert a blank item ahead of Item_j^U(j) in rj ;
23    U(j)++;
End
```

Figure 3.5: The algorithm of data item alignment [13]

The precision rate and recall rate of the DEPTA is poor compare to ViDE because in documents region that have large menus, long listings of user comments, or documents in which the relevant information is rendered using lists, division will not be consider in DEPTA. As precision and recall is nothing but total number of correctly extracted data records from total number of data record extracted and total number of correctly extracted from total number of data records. As ViDE overcome this drawback by utilizing visual features due to which precision and recall rate is better than DEPTA.

Table 3.2: Comparison of MDR, DEPTA and ViDE

	MDR	DEPTA	ViDE
Input	HTML document (DOM Tree)	HTML document (DOM Tree)	Region Tree
Applicability	Semi-structured web documents that contain one or more data regions formatted using HTML table tags, such that each data region contains several similar flat data records.	Semi-structured web documents that contain one or more data regions formatted using HTML table tags, such that each data region contains several similar flat data records.	Web documents in which the data region is the largest one, centred, and data records inside the data region are aligned to the left, adjacent, do not overlap, are separated homogeneously, and are similar from a visual point of view.
Precision	Poor	Compare to MDR better	Compare to DEPTA and MDR Better
Recall	Poor	Compare to MDR better	Compare to DEPTA and MDR Better

Limitation of ViDE:

The ViDE can only process deep Web pages containing one data region, while there is significant number of multi data-region deep Web pages. We propose an Improved-ViDE approach to handle multi data-region in deep web pages. It fails in cases in which a document contains several data regions that are separated by banners, for instance. If a page contain multi-data region than the precision and recall rate will reduce in ViDE as it process only one data region. As precision give us the rate that how many correct data records are extracted from relevant data records and recall give us the rate that how many relevant data records are extracted from overall data records. Because of the reason that it process only one data region ViDE fails to extract the

data record from other region which reduces the precision and recall rate. In some cases precision rate of ViDE can be better because it takes only one data region so chances of extracting irrelevant data may be less.

IV. Proposed method

The ViDE approach does not read the multi data region. It just considers the largest region of the web page. In Improved- ViDE it read reads more than one data region of the web page. For this various range for the data region is need to set. Ranges are set by analyzing the area of data region from various pages by using the metric $(area_b/area_{page}) > T_{region}$, where $area_b$ is area of block b, $area_{page}$ is the area of whole page and T_{region}.

4.1 System Architecture of I-ViDE

The **figure 4.1** will specific to System Architecture. The System architecture will gives the description about the module of the project and functionality of the each project. The following system architecture will give the following modules:

4.1.1 Multi region Visual Block Tree Generation

This block will paring the webpage and generating the visual block tree considering the html tags of web page with the define as the root as the page name and node elements are tags and sub tags as the child nodes.

Although proposed approach uses the VIPS algorithm to obtain a deep Web page's Multi region Visual Block tree and VIPS needs to analyze the HTML source code of the page, the proposed solution is independent of any specific method used to obtain the Visual Block tree.

The **figure 4.2 (a) and 4.2 (b)** show the example of www.shop.airtel.com webpage with visual blocks. And **figure 4.3** showing visual block tree.

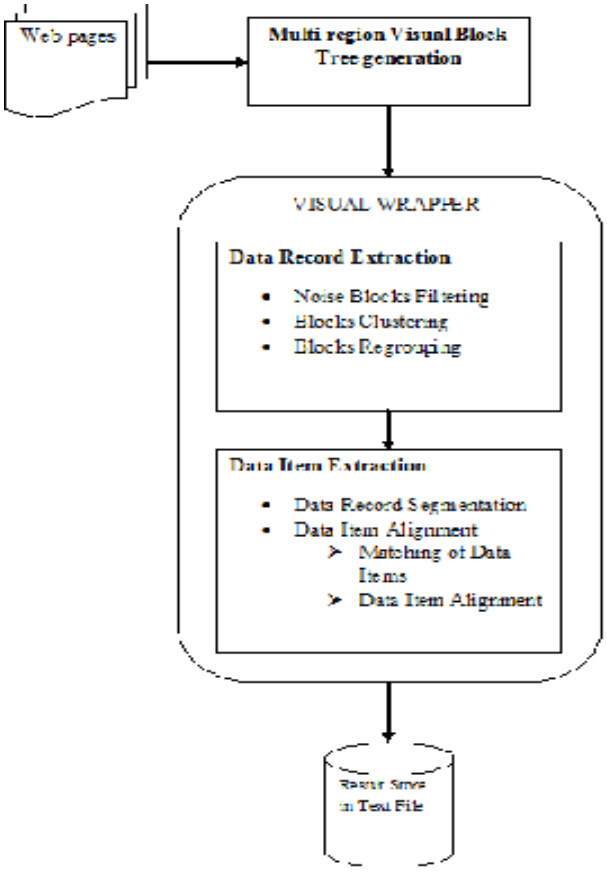


Figure 4.1: System Architecture of I-ViDE

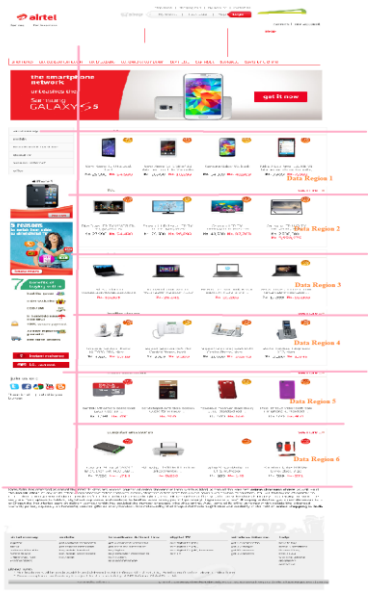


Figure 4.2 (a): Web Page Showing Data Region

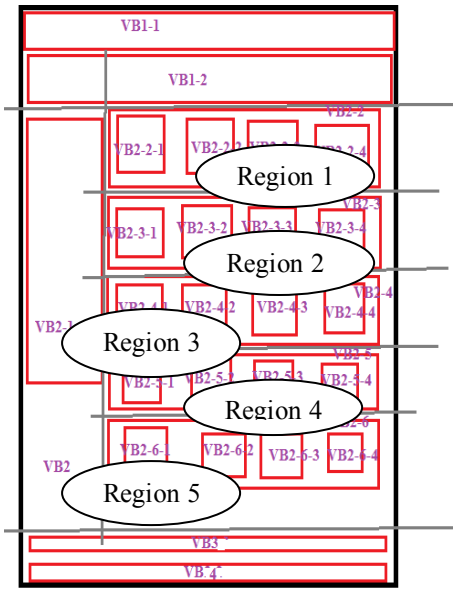


Figure 4.2 (b): Showing visual blocks

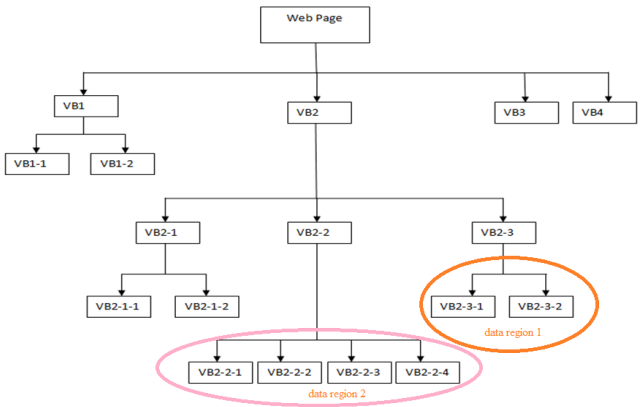


Figure 4.3: Visual Block Tree

After getting the visual block tree area is calculated for multi-data region rest all steps are same as that of ViDE where after calculating multi-data region data records are extracted and after that data items are extracted. After extracting data items we get the final output which contains information of multi data region.

V. Results & Analysis

We have implemented an operational deep web data extraction system for I-ViDE based on techniques we have mentioned. I-ViDE algorithm runs on various websites. If the website is created using the concept of CSS box model and if the products are aligned properly then only VIPS will work. As per rules of ViDE about visual features we get the result in the other case it can also fail if the website structure is not according to the rules of VIPS and ViDE. Out of those results of 4 websites i.e. www.shop.airtel.com**, www.healthkart.com, www.shopping.rediff.com, www.themobilestore.in** which has multi-data regions has been taken in our experiment. These deep web pages will process using ViDE and I-ViDE approaches.

5.1 Analysis

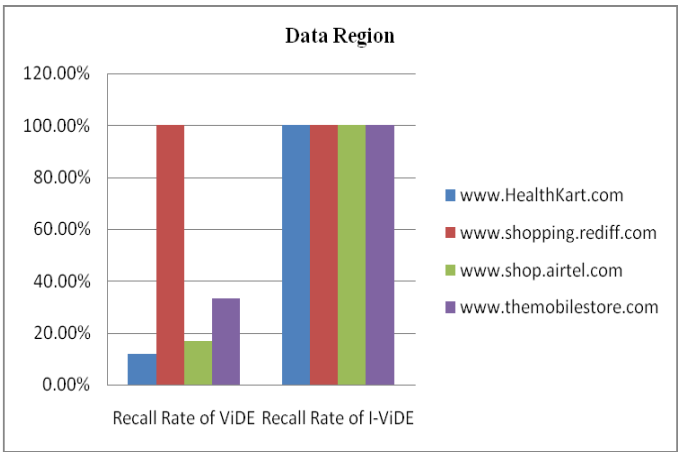
For analysis here we compare the results of ViDE and I-ViDE approaches by calculating the precision and recall rates. We have done comparison on precision and recall values of data region, data record and data item. Recall rate is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage.

Precision rate is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. It is usually expressed as a percentage.

Table 5.1 shows the recall rate with respect to data regions identified by ViDE and I-ViDE approach.

Table 5.1: Recall rates with respect to Data Region

	Recall Rate of ViDE	Recall Rate of I-ViDE
www.healthkart.com	12.00%	100%
www.shopping.rediff.com	100%	100%
www.shop.airtel.com	16.66%	100%
www.themobilestore.in	33.33%	100%

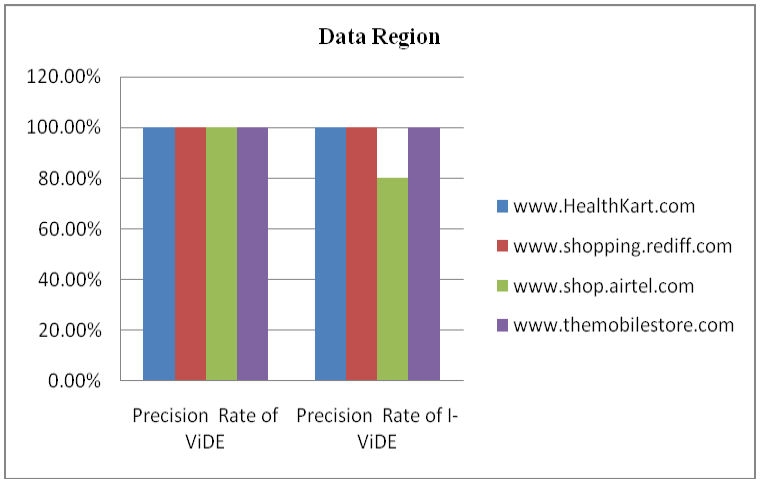


Graph 5.1: Shows the recall rate with respect to data region identified

The graph 5.1 shows the recall rate of ViDE and I-ViDE with respect to data region identified. In this graph I-ViDE results are good than ViDE because ViDE has taken only one data region and it fail to take other data region where as I-ViDE taken all data region. We get all relevant results in I-ViDE.

Table 5.2: Precision rates with respect to Data Region

	Precision Rate of ViDE	Precision Rate of I-ViDE
www.healthkart.com	100%	100%
www.shopping.rediff.com	100%	100%
www.shop.airtel.com	100%	80%
www.themobilestore.in	100%	100%



Graph 5.2: Shows the precision rate with respect to data region identified

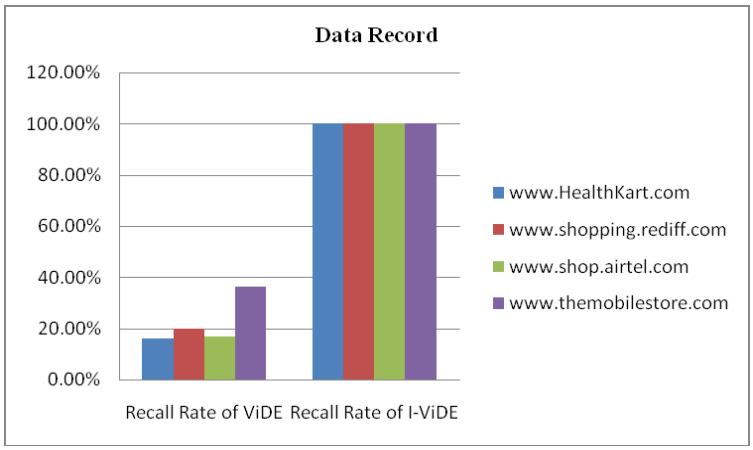
The table 5.2 and graph shows the precision rate with respect to data region. Precision rate of ViDE is good because it has not extracted irrelevant data region where as precision rate of I-ViDE is also good because I-ViDE has not extracted irrelevant data regions.

Table 5.3 and graph shows the recall rates with respect to data records identified by ViDE and I-ViDE approach.

Table 5.3: Recall rates with respect to Data Records

	Recall Rate of ViDE	Recall Rate of I-ViDE
www.healthkart.com	16.00%	100%
www.shopping.rediff.com	20%	100%
www.shop.airtel.com	16.66%	100%
www.themobilestore.in	36.36%	100%

In this below graph 5.3 I-ViDE results are excellent then ViDE because ViDE fails when there is more than one region on a webpage. As I-ViDE processes on all data region it succeeded in extracting all data records from all data region.

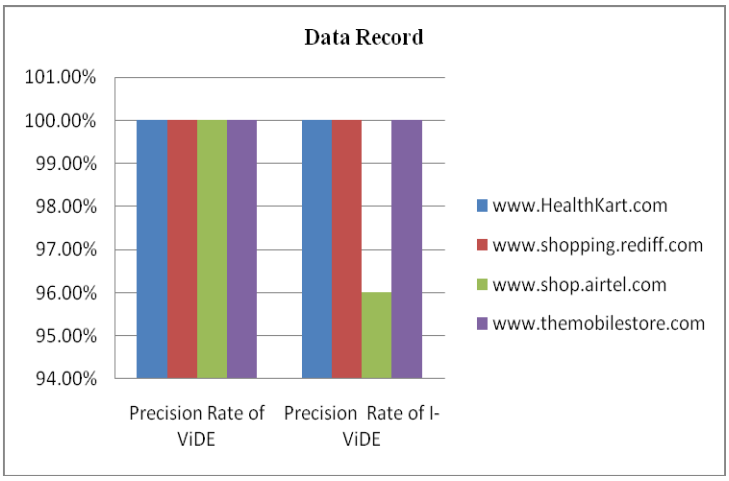


Graph 5.3: Shows the recall rate with respect to data records identified

The table 5.4 and graph 5.4 shows the precision rate with respect to data records. Here ViDE results are excellent because it has not extracted any irrelevant data records. I-ViDE results are also excellent for HealthKart, rediff and mobile store website but not for Airtel page because for Airtel page it has extracted some irrelevant data region as some non-useful blocks has same area as that of useful blocks. This is possible in some cases only in I-ViDE.

Table 5.4: Precision rates with respect to Data Records

	Precision Rate of ViDE	Precision Rate of I-ViDE
www.healthkart.com	100%	100%
www.shopping.rediff.com	100%	100%
www.shop.airtel.com	100%	96%
www.themobilestore.in	100%	100%

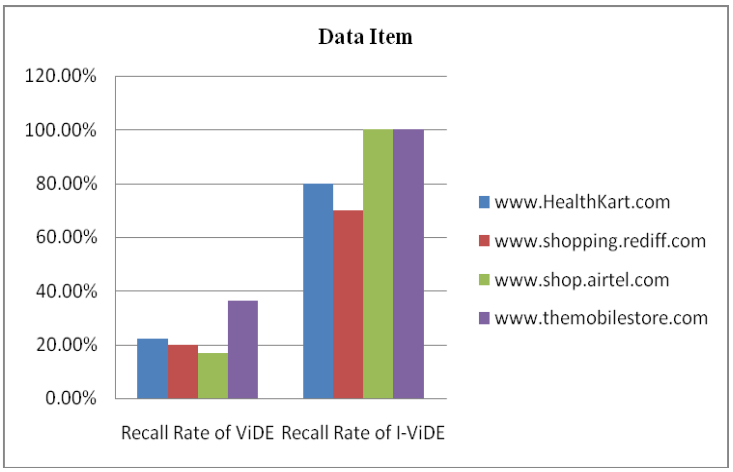


Graph 5.4: Shows the precision rate with respect to data records identified

Table 5.5 and graph 5.5 shows the recall rates with respect to data items extracted by ViDE and I-ViDE approach.

Table 5.5: Recall rate with respect to Data Items Extracted

	Recall Rate of ViDE	Recall Rate of I-ViDE
www.healthkart.com	22.35%	80%
www.shopping.rediff.com	20%	70%
www.shop.airtel.com	16.66%	100%
www.themobilestore.in	36.36%	100%

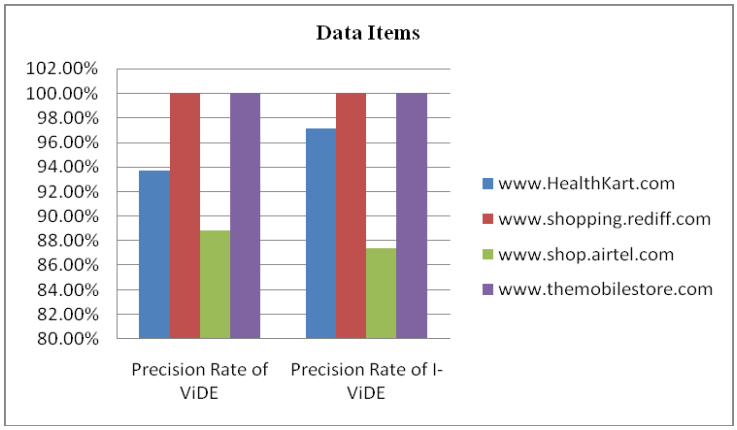


Graph 5.5: Shows the recall rate with respect to data items extracted

I-ViDE results of data items are excellent as compare to ViDE because ViDE do not process on all data region it misses some relevant information.

Table 5.6: Precision rate with respect to Data Items Extracted

	Precision Rate of ViDE	Precision Rate of I-ViDE
www.healthkart.com	93.70%	97%
www.shopping.rediff.com	100%	100%
www.shop.airtel.com	88.80%	87%
www.themobilestore.in	100%	100%



Graph 5.6: Shows the precision rate with respect to data items extracted

Table 5.6 and graph 5.6 shows the precision rates with respest to data items. The precision rate of healthkart page is not good in ViDE because of extraction of irrelevant data item though it has not extracted irrelevant region but it has extracted some irrelevant data item. Rediff page precision rate is not good in I-ViDE because of extracting some irrelevant data items.

VI. Conclusion

With the flourish of the deep Web, users have a great opportunity to benefit from such abundant information in it. In general, the desired information is embedded in the deep Web pages in the form of data records returned by Web databases when they respond to users' queries. Therefore, it is an important task to extract the structured data from the deep Web pages for later processing.

As precision give us the rate that how many correct data records are extracted from relevant data records and recall give us the rate that how many relevant data records are extracted from overall data records ViDE will fail to extract the data records from other region which reduces the precision and recall rate to resolve the problems on ViDE approach we proposed the I-ViDE approach. In some cases we find that ViDE results are good because in I-ViDE the irrelevant blocks may have same area as that of relevant data blocks. As in I-ViDE we set two threshold values in which we can get irrelevant data block. Over all we can say that Improved- ViDE the precision and recall rate is better compare to ViDE as it process on multi-data region. Our Improved-ViDE method works on various websites still not on every websites because some website like Amazon website has high version of HTML language which is not compatible to our algorithm and some websites have unstructured data which are not properly arrange and not properly align on a page. To overcome that in future we have to first modify the VIPS algorithm and then I-ViDE by proposing some rules.

References

[1] V. Crescenzi and G. Mecca, "Grammars Have Exceptions," Information Systems, vol. 23, no. 8, pp. 539-565, 1998.
[2] G.O. Arocena and A.O. Mendelzon, "WebOQL: Restructuring Documents, Databases, and Webs," Proc. Int'l Conf. Data Eng. (ICDE), pp. 24-33, 1998.
[3] J. Hammer, J. McHugh, and H. Garcia-Molina, "Semi-structured Data: The TSIMMIS Experience," Proc. East-European Workshop Advances in Databases and Information Systems (ADBIS), pp. 1-8, 1997.
[4] C.-N. Hsu and M.-T. Dung, "Generating Finite-State Transducers for Semi-Structured Data Extraction from the Web," Information Systems, vol. 23, no. 8, pp. 521-538, 1998.
[5] N. Kushmerick, "Wrapper Induction: Efficiency and Expressiveness," Artificial Intelligence, vol. 118, nos. 1/2, pp. 15-68, 2000.
[6] A. Laender, B. Ribeiro-Neto, A. da Silva, and J. Teixeira, "A Brief Survey of Web Data Extraction Tools," SIGMOD Record, vol. 31, no. 2, pp. 84-93, 2002.
[7] B. Liu, R.L. Grossman, and Y. Zhai, "Mining Data Records in Web Pages," Proc. Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 601-606, 2003.
[8] D. Cai, S. Yu, J. Wen, and W. Ma, "Extracting Content Structure for Web Pages Based on Visual Representation," Proc. Asia Pacific Web Conf. (APWeb), pp. 406-417, 2003.
[9] Y. Zhai and B. Liu, "Web Data Extraction Based on Partial Tree Alignment," Proc. Int'l World Wide Web Conf. (WWW), pp. 76-85, 2005.
[10] W. Liu, X. Meng, and W. Meng. Vision-based web data records extraction. In WebDB, pages 14–19, 2006.
[11] C.-H. Chang, M. Kayed, M.R. Girgis, and K.F. Shaalan, "A Survey of Web Information Extraction Systems," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 10, pp. 1411-1428, Oct. 2006.
[12] L. Li, Y. Liu, A. Obregon, and M. Weatherston. Visual segmentation-based data record extraction from web documents. In IRI, pages 502–507, 2007.
[13] W. Liu, X. Meng, and W. Meng. ViDE: A vision-based approach for deep web data extraction. IEEE Trans. Knowl.Data Eng., 22(3):447–460, 2010.