Information Processing and Management 53 (2017) 1120–1138

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

DERIN: A data extraction method based on rendering information and n-gram

Leandro Neiva Lopes Figueiredo, Guilherme Tavares de Assis, Anderson A. Ferreira[*]

Departamento de Computação, Universidade Federal de Ouro Preto, Campus Universitário Morro do Cruzeiro, CEP 35400-000, Ouro Preto - MG, Brazil

A R T I C L E   I N F O

Article history:
Received 5 April 2016
Revised 6 April 2017
Accepted 26 April 2017
Available online 11 May 2017

Keywords:
Rendering information
Visual information
Wrapper
Main data region
Path expression
N-gram
Data extraction

A B S T R A C T

Extracting data from web pages is an important task for several applications such as comparison shopping and data mining. Ordinarily, the data in web pages represent records from a database and are obtained using a web search. One of the most important steps for extracting records from a web page is identifying out of the different data regions, the one containing the records to be extracted. An incorrect identification of this region may lead to an extraction of incorrect records. This process is followed by the equally important step of detecting and correctly splitting the necessary records and their attributes from the main data region. In this study, we propose a method for data extraction based on rendering information and an n-gram model (DERIN) that aims to improve wrapper performance by automatically selecting the main data region from a search results page and extracting its records and attributes based on rendering information. The proposed DERIN method can detect different record structures using techniques based on an n-gram model. Moreover, DERIN does not require examples to learn how to extract the data, performs a given domain independently and can detect records that are not children of the same parent element in the DOM tree. Experimental results using web pages from several domains show that DERIN is highly effective and performs well when compared with other methods.

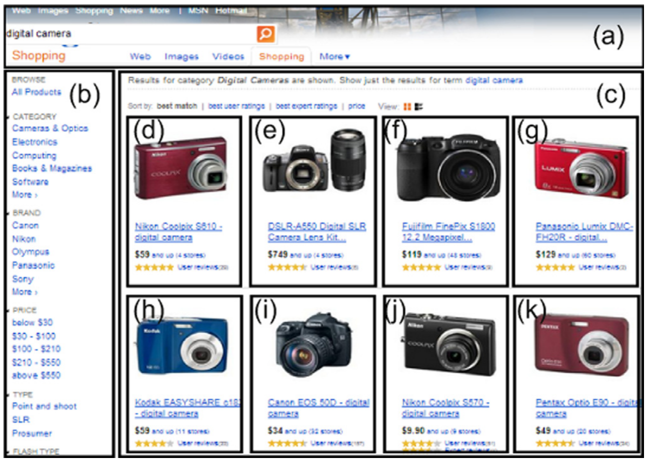
1. Introduction

A large amount of data available on web pages comes from records stored in databases. In this context, if a website produces several web pages, these pages are generally produced using the same script and contain the same structure. These pages, known as search result pages, are accessed by a search from a website and contain several data regions; i.e., data groups located in different positions on the screen, such as menus, advertisements, or search result records (SRR). Each SRR represents an object from a database related to a user search and contains one or more attributes. For example, in Fig. 1, rectangles (a), (b) and (c) are data regions, and rectangles (d) to (k) are SRRs. In this example, each SRR has a name, price, and user rating.

Several applications, such as price comparators, digital libraries, data mining, and data integration require data from these pages. In commercial applications, a company may use the web to gather and analyze information about the activities of

[*] Corresponding author.
E-mail addresses: leomcl@gmail.com (L.N.L. Figueiredo), gtassis@iceb.ufop.br (G.T. de Assis), ferreira@iceb.ufop.br (A.A. Ferreira).

http://dx.doi.org/10.1016/j.ipm.2017.04.007
0306-4573/© 2017 Elsevier Ltd. All rights reserved.



Results for category *Digital Cameras* are shown. Show just the results for term digital camera

Fig. 1. Examples of data regions ((a), (b) and (c)), SRRs ((d) to (k)) and data units (e.g., product name, price, user rating).

their competitors (in a process known as Competitive Intelligence), identify opportunities in the market, and anticipate their competitors' actions (Ferrara, De Meo, Fiumara, & Baumgartner, 2014).

The process of web data extraction is aimed at determining the location of the main data region in a page, i.e. where the records returned by a search are located, understanding the structure and attributes of these records, and extracting them and making them available in a format that can be used by applications.

In this article, we propose DERIN (Data Extraction based on Rendering Information and N-gram) that aims at achieving better performance in the data extraction process from web pages compared with wrappers which use DOM tree. DERIN is an extension of the method proposed in (Figueiredo, Ferreira, and de Assis, 2014), which only identified the main data regions in web pages. DERIN is a completely automated method that does not only identify the main data region in an HTML page, but identifies SRRs and attributes, analyzes the DOM tree, and renders information associated with its elements. In addition, DERIN can detect different record structures.

As the main contribution of this article, the proposed method analyzes different SRRs structures using techniques based on n-gram model by detecting the start and end of each SRR from the attribute positions of all records in the main data region. Using these n-gram techniques, DERIN can also detect SRRs that are not children of the same parent element in the DOM tree. Moreover, DERIN also groups characteristics of other proposals, minimizing their weakness.

The remainder of this article is organized as follows: Section 2 presents the main work related to ours. Section 3 details our proposed method. Section 4 evaluates our method and discusses its application in several application domains. Finally, Section 5 shows our conclusions and discusses future work.

2. Related work

Data extraction from web pages has been studied in previous works. Some studies have analyzed the structures of pages as a tree, representing their elements in a hierarchical manner (Arasu & Garcia-Molina, 2003; Crescenzi, Mecca, & Merialdo, 2001; Liu, Pu, & Han, 2000; Sahuguet & Azavant, 1999; Wang & Lochovsky, 2003; Zhai & Liu, 2005), while others applied machine learning techniques or probabilistic models using training examples to learn how to wrap data (Chang, Hsu, & Lui, 2003; Hsu & Dung, 1998; Kushmerick, 1997; Muslea, Minton, & Knoblock, 1998). In addition, some studies used examples provided by a GUI interface (Graphical User Interface) to find and extract data by analyzing similar objects (Adelberg, 1998; Laender, Ribeiro-Neto, & da Silva, 2002). Finally, some works based on ontologies and languages also exists (Arocena & Mendelzon, 1999; Crescenzi & Mecca, 1998; Hammer, McHugh, & Garcia-Molin, 1997).

Analyzing structures and visual information have also been proposed by the research community (Ansari & Vasishtha, 2015; Cai, Yu, Wen, & Ma, 2003; Chang et al., 2003; Chu, Hsu, Lee, & Tsai, 2015; Don, Chu, & Ling, 2015; Fumarola, Weninger, Barber, Malerba, & Han, 2011; Grigalis & Cenys, 2014; Hiremath & Algur, 2009; Irmak & Suel, 2006; Kadam & Pakle, 2014; Krupl-Sypien, Fayzrakhmanov, Holzinger, Panzenbïck, & Baumgartner, 2011; Liu, Meng, & Meng, 2010; Shi, Liu, Shen, Yuan, & Huang, 2015; Simon & Lausen, 2005; Trieschnigg, Tjin-Kam-Jet, & Hiemstra, 2012; Uzun, Agun, & Yerlikaya, 2013; Velloso & Dorneles, 2013; Zhai & Liu, 2005; Zhao, Meng, Wu, Raghavan, & Yu, 2005). Vision-based page segmentation (VIPS), proposed by Cai et al. (2003), uses the position of the elements in the DOM tree and their visual information (e.g., type and font color) to group similar objects from different segments. VIPS assigns a score to each segment and uses this score to improve search engines, which may use VIPS to detect navigation, interaction, and contact information, excluding them from the results. As discussed by Li, Liu, and Obregon (2007), VIPS has some disadvantages: (1) results decrease when some search result pages have markedly different structures; (2) on some pages, VIPS stops the partitioning processes earlier than expected; (3) VIPS can incorrectly split records with similar visual information into different data regions.

In (Zhai and Liu, 2005), the authors propose DEPTA (Data Extraction Based on Partial Tree Alignment), which is based on the idea that information on web pages is in continuous data regions. First, DEPTA uses rendering information to assemble a tree representing the dispositions of elements on the screen. Next, it combines multiple adjacent elements to delimit data regions, and subsequently applies a partial alignment tree algorithm to obtain the SRRs and their attributes. This method analyzes the entire structure of a page and extracts all data regions. Thus, the user must manually separate the search result records from other information on the page.

In (Trieschnigg et al., 2012), the authors proposed a method that uses elements' positional information in the DOM tree, along with their visual characteristics (e.g., font style, visibility, and rendered area) to find the SRRs and generate candidate XPath expressions to be used in the extraction process. Unlike our proposed method, Trieschnigg et al. (2012) assumes that all SRRs contain at least one HTML link. Thus, it generates XPath expressions only for elements at least one HTML link. These are then ranked by the similarity of their resulting nodes. As noted in the experiments in this work, that method fails when the SRRs do not have the same parent element or do not have similar structures (for instance, elements with different numbers of attributes).

In (Zhao et al., 2005), the authors propose ViNTs, which uses visual information (excluding HTML elements) to identify content regularities (text, images, links, etc.), and then combines them with the HTML structure to generate wrappers. ViNTs requires at least five search result pages from the same website as positive examples and one negative example. ViNTs considers blocks with lines in a specific order as data regions. This method performs well in well-structured search result pages, but excludes SRRs with irregular content patterns.

The method proposed by Velloso and Dorneles (2013) represents each element as a symbol and a set of symbols as an alphabet. It analyzes the string of symbols to identify SRRs and the main data region. Next, it excludes other regions from the page source code and uses MDR (Liu, Grossman, & Zhai, 2003) to read the new structure and extract the SRRs and their attributes. This method was applied in a collection of 23 search result pages and identified main data regions with an accuracy of 86.96%.

In (Liu et al., 2010), the authors proposed ViDE, which uses the blocks generated by VIPS to detect the main data region and extract its records and attributes. ViDE assumes the main data region contains a large area in relation to the whole page. ViDE also views any block within the main data region as a record, excluding possible outliers in this region. ViDE also applies to records containing more than one HTML element (i.e., SRR splits in several subtrees in an HTML page). In addition, ViDE uses the visual information of the blocks in each SRR to extract the attributes. Although ViDE was effective in its experimental evaluation, it assumes that the main data region is horizontally centralized in a page, and the outliers are at the head or foot of, and left aligned in, the main data region; the results are shown as a vertical list.

The method proposed by Fumarola et al. (2011) extracts data from all regions of pages where the elements are displayed as lists (both vertical and horizontal). To detect such lists, the authors use the DOM tree structure and the position of the elements on the screen (detecting neighboring elements); thus, it extracts irrelevant data, since it extracts data from any list from a page.

ViPER, proposed by Simon and Lausen (2005), uses the position of the elements in pages to simulate a printed representation of the page, thereby detecting regions and locating spaces among likely SRRs. Thus, ViPER fails in detecting SRRs from pages whose results are not in lists (e.g., thumbnails).

Chu et al. (2015) proposed a method that compared the paths of elements in the DOM tree to detect and extract records and attributes. It only uses elements inside the center block of the page with similar rendering size by a browser, and this assumption is not true for all search result pages.

Ansari and Vasishtha (2015) gathered several search result pages from the same website and compared two or more pages, thereby eliminating noise and determining the main data region. Next, the authors proposed to extract records and data attributes by comparing the structure of the elements inside the main data region. They consider the largest contiguous region in the web page with distinct contents the main data region. This method has some limitations: it requires several pages from the same website to be effective, it does not extract records whose content repeats in those pages, and it does not align the data attributes.

In (Don et al., 2015), the authors proposed detecting the main data region using the number of characters from the links inside a region divided by the total number of characters of the region, called the link character density method. The main data region corresponds to the region whose link character density is at most equal to a given threshold. To extract the records, the authors propose a training a model using provided examples. In addition, they proposed using a linear method to integrate the classification results, thereby generating the final extraction results.

A system to create a schema to extract data from web pages is proposed in (Chang, Chen, Chen, and Ding, 2016). This system uses part of the input pages to infer a schema without supervision, and extracts data from the rest of the input pages matching the schema. A finite state machine (FSM) is generated based on the inferred schema, and a template is used to extract data from web pages. The system extracts data from pages per the FSM. This system is very sensitive to changes in the structures of the web pages.

Uzun et al. (2013) proposed a hybrid method that use examples of pages and decision trees to generate rules for extracting content from web pages. This method performs only in pages with just one record, such as news articles or book chapters.

Shi et al. (2015) proposed AutoRM, an automated method for extracting data records which looks for similar adjacent objects in the DOM tree. AutoRM may extract data records that are not children of the same parent node, as our method.

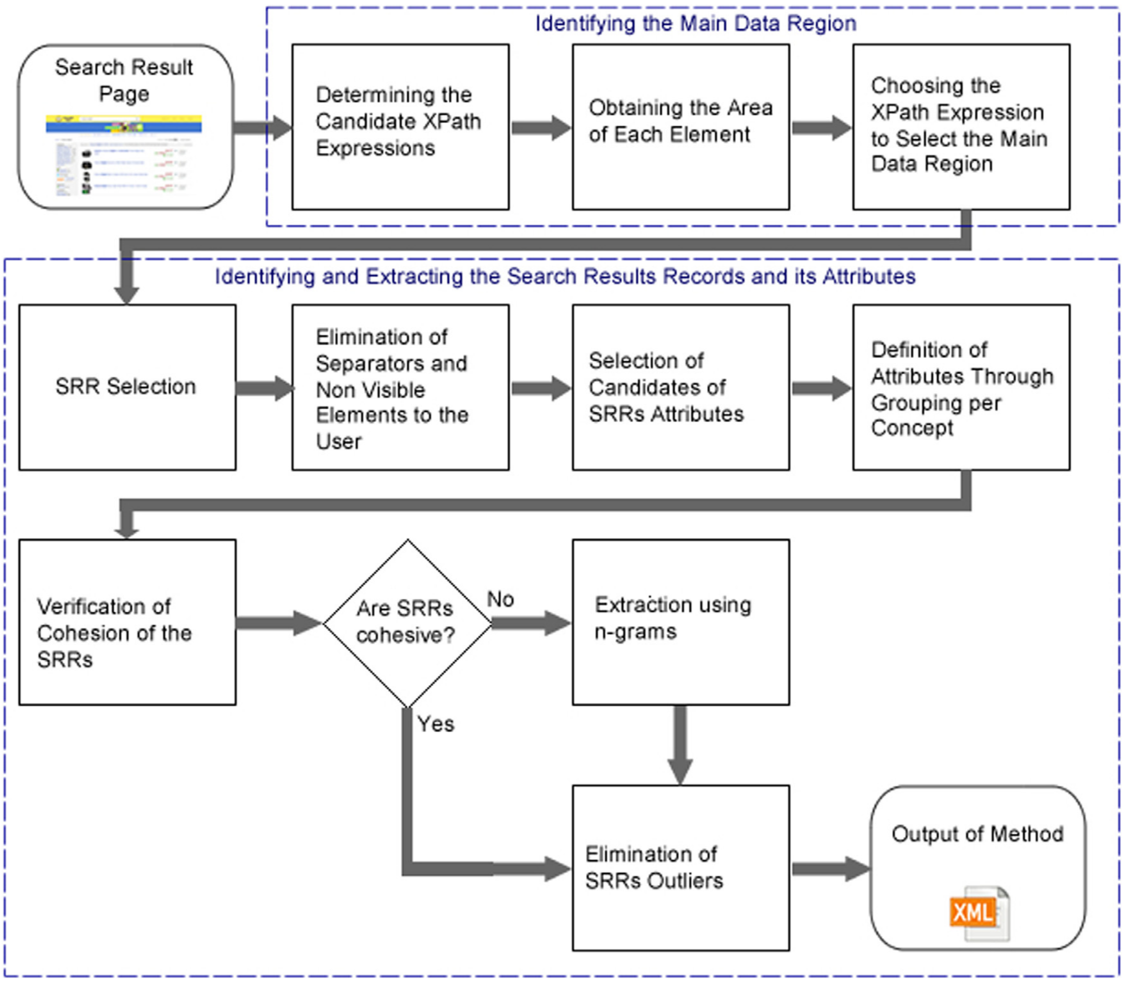

Identifying the Main Data Region
Search Result Page
Determining the Candidate XPath Expressions
Obtaining the Area of Each Element
Choosing the XPath Expression to Select the Main Data Region
Identifying and Extracting the Search Results Records and its Attributes
SRR Selection
Elimination of Separators and Non Visible Elements to the User
Selection of Candidates of SRRs Attributes
Definition of Attributes Through Grouping per Concept
Verification of Cohesion of the SRRs
Are SRRs cohesive?
No
Yes
Extraction using n-grams
Elimination of SRRs Outliers
Output of Method
XML

Fig. 2. DERIN phases.

The records are identified by detecting separators and head elements of the records, however, as it does not detect the main data region in the page, records are extracted from all regions as DEPTA and data attributes are not aligned.

Chang et al. (2003) propose IEPAD, a method that detects SRRs and its attributes analyzing the DOM tree and using a PAT Tree to detect patterns. IEPAD does not use visual information and it is semi-supervised, i.e., it requires user interaction to extract data.

In (Grigalis and Cenys, 2014), the authors propose ClustVX, which combines visual information with DOM tree structural analysis to build XStrings. Then, XStrings are grouped in clusters, that are candicates to data region. In order to find the main data region, the method chooses the largest visual height, based on the average area of a SRR multiplied by the number of attributes inside that region. As other methods, ClustVX assumes that SRRs are children of the same parent element. This assumption may decrease accuracy as in some situations SRRs are children of different parent elements.

In our work, the candidate elements of the main data region are found without analyzing the entire DOM tree of an HTML page. This analysis is performed only on candidate elements, thereby reducing the number of elements to be analyzed. Furthermore, our proposed method works in main data regions where SRRs are not children of the same parent element. Finally, our proposed method uses techniques based on n-grams to identify SRRs, with positive results, even in web pages with more complex record structures.

3. DERIN

In this section, we first describe how DERIN identifies, in three steps, the main data region in a search result page. Next, we detail the extraction of SRRs and their attributes. In Fig. 2 is shown an illustrative schema of DERIN in which are illustrated its phases.

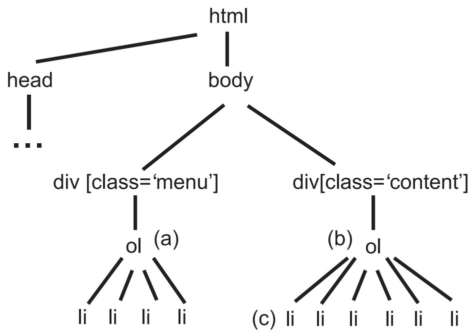


Fig. 3. An example of an HTML page in DOM format with two data regions (a) and (b), and six SRRs (c).

3.1. Identifying the main data region

To identify the main data region from a search result page, DERIN performs a three-step process using the information rendered from the page. In this phase, DERIN receives a search result page as an input and produces an XPath expression as an output to locate the main data region. The three steps of this phase are:

1. Generating candidate XPath expressions for locating the main data region;
2. Obtaining visual data based on rendering information; and
3. Choosing the candidate XPath expression most likely to locate the main data region.

DERIN works on the HTML DOM Tree[1]; i.e., it manipulates a web page as a tree of objects. Next, we describe the three steps of this phase.

3.1.1. Step 1: determining the candidate XPath expressions

As previously mentioned, the main data region of a search result page (HTML page) contains the SRRs of this page. The main data region may have several SRRs and all SRRs are descendants of the "body" element, which is a child of the "html" element (the web page root element). Thus, any other subtree of the "html" element is not considered by our method (e.g., "head" element).

For the first step of this phase, our method produces candidate XPath expressions to locate the main data region. We consider all data regions containing at least a given number of records (i.e., all elements with at least a given number of child elements in DOM format) are candidates for the main data region. We use the $\#_{\text{min}}$ parameter as the minimum number of child elements required in a parent element to be a candidate. The total number of child nodes for a parent node does not count HTML elements which are not used as record delimiters on web pages (i.e., elements that indicate form fields, comments, javascript codes, table columns etc.). Thus, our method does not consider the elements "input", "textarea", "select", "option", "link", "script", "style", "img", "!–", "td", and "noscript" when counting total number of child nodes.

After we have selected all nodes containing at least $\#_{\text{min}}$ child elements, DERIN generates a positional XPath expression for each selected node and adds it to the set of candidate XPath expressions *S*. For example, if our method is applied to a page whose DOM format is as shown in Fig. 3, DERIN adds the expressions (a) "/html[1]/body[1]/div[1]/ol[1]" and (b) "/html[1]/body[1]/div[2]/ol[1]" to *S*.

In this example, expression (a) selects the data region with the page's menu and expression (b) selects the data region with the SRRs. The next two steps will determine the correct XPath expression to select the main data region.

3.1.2. Step 2: obtaining the area of each element

In this step, the visual information used by DERIN corresponds to the height and width attributes. We used these attributes to determine the area of each element whose expressions are in *S* (i.e., each element selected by the expressions in *S*) when it is shown in a web browser. In order to accurately determine this area, DERIN renders the search result page using MSHTML[2] to obtain the height, width, and area of each selected data region. Specifically, the height and width of each element and the area are obtained as follows:

1. MSHTML loads the web page and displays it in the same manner as the Internet Explorer browser. When a browser renders a page on the screen, a visual information is associated with each element (e.g., *x* and *y* coordinates and height and width of the element)
2. DERIN assigns the corresponding height and width to each element in the DOM tree.

[1] http://www.w3.org/DOM/.

[2] MSHTML is an dynamic-link library (DLL) that simulates the Internet Explorer browser and may be used by programs written in several visual programming languages.

3.1.3. Step 3: choosing the XPath expression to select the main data region

The third step of this phase is to choose the expression most likely to locate the main data region from among the candidate expressions in S. To locate the main data region of a page, our method assumes that this region, in which is the most important information to the user, fills the largest portion on the screen when a browser renders the page.

As previously mentioned, the main data region is inside the “body” element of a web page, and a browser renders only descendant elements of the “body” element. Thus, to determine the portion on the screen filled by each element, we calculate the fraction of the area associated to each element regarding the “body” element’s area.

To determine which candidate expression will correctly select the main data region, DERIN performs a multi-stage filtering process to select the correct expression from S. The stages of this process are described below.

Filtering candidate expressions by area. The first stage aims to filter and select the candidate expressions which select the elements with the largest visual area as these are the most likely to contain SRRs. Specifically, let $S = \{exp_1, exp_2, \ldots, exp_n\}$ be the set of n candidate expressions and $E = \{e_1, e_2, \ldots, e_n\}$ be the set of elements selected by each expression in S, where e_i is the element selected by the expression exp_i. The height and width of each element e_i and the “body” element is defined in the second stage; the area of each element, e_i and “body”, is calculated by multiplying its corresponding height and width. We divide the area associated to each element e_i ($area(e_i)$) by the area of the “body” element ($area(body)$), and if this value is larger than the given threshold $\%_{area}$, the corresponding exp_i is inserted into the set of filtered expressions S'. This is expressed as:

$$S' = \{exp_i \in S | area(e_i)/area(body) \geq \%_{area}\} \tag{1}$$

Filtering candidate expressions by height. There are elements showed in a browser which fill a large portion on the screen (i.e., a large area) but do not contain SRRs; e.g., page headers and footers, or other elements whose height is low compared with the “body” element. Thus, we remove from the set S' any expressions whose height is insufficient to contain SRRs. If the ratio between the height of an element e_i ($height(e_i)$) and the height of the “body” element ($height(body)$) is at least equal to a given threshold, $\%_{height}$, its expression exp_i remains in the set S'. Otherwise, the corresponding expression is removed from S'. This is expressed as:

$$S' = S' - \{exp_i \in S' | height(e_i)/height(body) < \%_{height}\} \tag{2}$$

Filtering candidate expressions by width. Similarly, there are also elements with large areas and narrow widths which do not contain SRRs; e.g., page side menus. Thus, we also remove from S' expressions whose corresponding element’s width is too narrow. To determine whether the width of an element e_i is too narrow, we divide its width ($width(e_i)$) by the width of the “body” element ($width(body)$) and compare the result against a given threshold $\%_{width}$. If the result is lower than the threshold, the expression is removed from S'. This is expressed as:

$$S' = S' - \{exp_i \in S' | width(e_i)/width(body) < \%_{width}\} \tag{3}$$

Filtering candidate expressions that select elements’ ancestors. Frequently, a candidate element (i.e., an element selected by a candidate expression) for a main data region contains other candidate elements as children. This occurs, for example, when before a “div”[3] element, that contains SRRs, there is another “div” element with text informing the current page (e.g., “showing 1 of10”), or when a “div” is followed by another “div” element with a panel for changing the page (e.g., “previous page” and “next page”). Thus, both “div” elements containing SRRs and their parent elements may have candidate expressions in S' after the applying the previously defined filters. In addition, we noticed that elements containing SRRs filled a large space within their parent area.

To a remove a candidate expression exp_i that is a prefix of other expression exp_j where the element e_j (selected by exp_j) contains the SRRs from S', we determined whether the ratio of e_j’s area to e_i’s area is larger than a given threshold $\%_{parent}$. Thus, the expression exp_i (the expression that selects the parent element) is removed from S' (the set of candidate expressions), or:

$$\begin{aligned} S' = S' - \quad & \{exp_i \in S' | exp_j \in S' \wedge parent(e_j) = e_i \\ & \wedge area(e_j)/area(e_i) > \%_{parent}\} \end{aligned} \tag{4}$$

Defining the expression that selects the main data regions. Among the candidate expressions in S', the one with the largest area will be used to select the main data region, however this expression may not be able to select the element containing all SRRs. This occurs because, after a browser renders a web page, the SRRs are sometimes shown in the same portion on the screen. Contrarily, in a DOM tree, the elements containing SRRs may have sibling elements without SRRs or may have distinct parents (i.e., the SRRs may be children from different parents). For example, in the DOM tree shown in Fig. 4, we should have two candidate expressions in S', selecting the “ol” elements in “div” elements (b) and (d) when, filters are applied. The “ol” element in the “div” element (b) would be considered the main data region as its corresponding area is

[3] The “div” element defines a division or section in a HTML document.

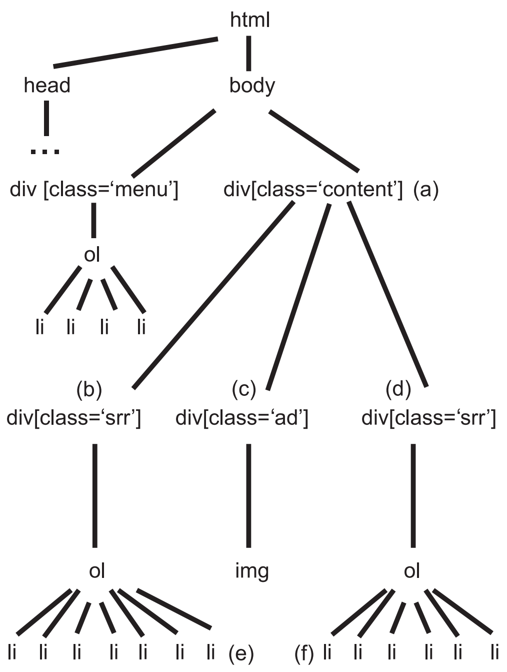


Fig. 4. An example of an HTML page in DOM format with two SRR groups (e) and (f) in different subtrees (b) and (d). (c) is a region containing publicity and (a) is the main data region.

largest. However, this result is incorrect because it does not include all SRRs. As we are using positional expressions, the same expression cannot select both elements. Thus, as an option, we combine the corresponding expressions to select a common ancestor. As shown in Fig. 4, the resulting expression would select the "div" element (a).

To solve this problem, DERIN performs the following after the expression *exp* and its selected element *e* have already been chosen:

1. Let *p* be the parent of *e*;
2. If there are descendants of *p* similar to children of *e, p* is considered the element with the main data region. A positional expression is generated and returned as the expression to select the main data region;
3. If the result of stage 2 is false, *p* receives its parent and the operation is performed again.

This method returns to stage 2 a maximum of three times (empirically evaluated). We use the Robust Algorithm for the Tree Edit Distance (RTED) (Pawlik & Augsten, 2011) to calculate the similarities between elements (subtrees from the DOM tree). RTED receives as input two elements (subtrees) and outputs a value relative to the number of operations required to transform one tree to another. For similar elements, the value is close to 0. For two identical trees, the value is 0. Our method makes some assumptions for using RTED, which are as follows:

- For each operation of renaming (exchange) elements, a weight of 3 was assigned. For other operations weight remained at 1.
- A limited number of tree levels were used when comparing values when a subtree of an SRR with many attributes with that of an SRR with fewer attributes. This was done to avoid RTED returning values which were too large for comparison. We limited the number of levels in the comparison of subtrees to two; specifically, we compared the subtrees considering only the root nodes and the children of the root nodes. Furthermore, as the elements of SRRs are not "br", "h1", "h2", "h3", "h4", "h5", "h6", and "a" elements, which are used to visually separate SRRs on the screen or display related information, these are ignored.
- As final distance between two subtrees that may contain the SRRs, we use the average distance between the possible SRRs in both subtrees. Thus, two subtrees are similar whether the average distance is at most equal to a given threshold $\delta_{\max}$.

In certain situations, the set of candidate expressions is empty after filters are applied. In such cases, we consider the "body" element as the main data region and the positional expression "/html[1]/body[1]" is returned.

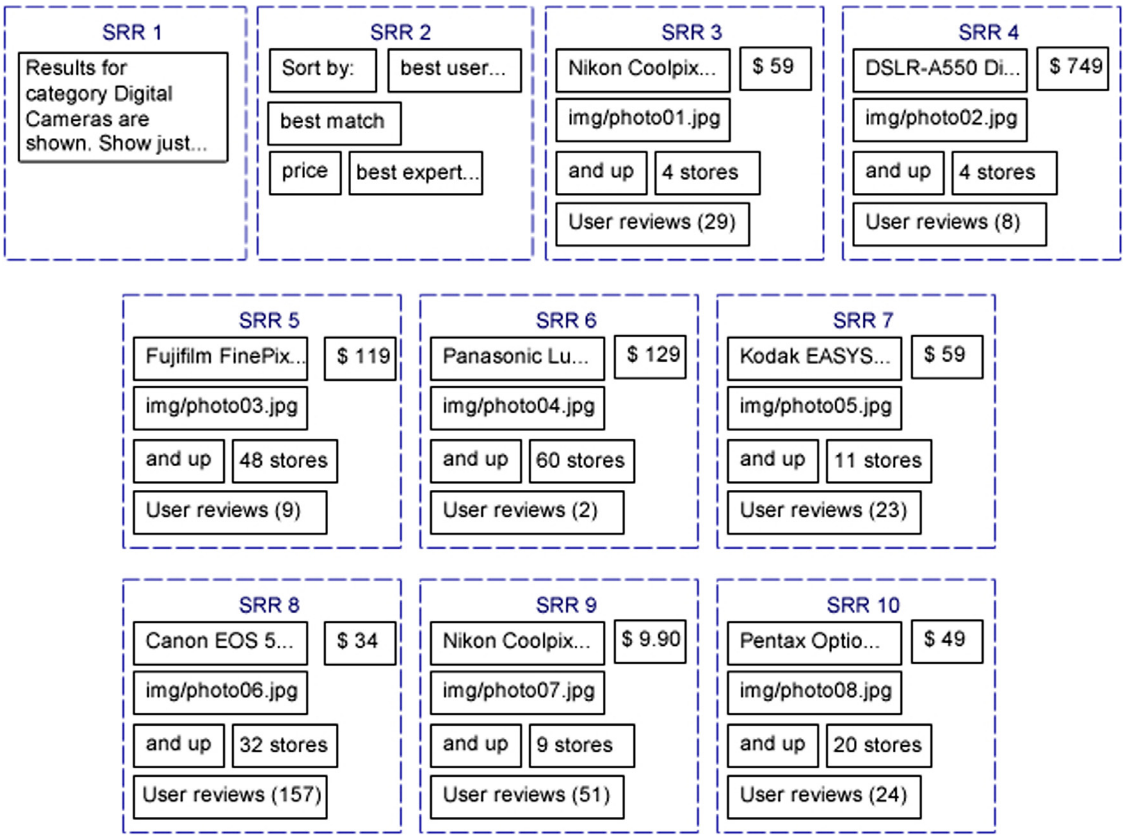


Fig. 5. Example of selection of SRRs and candidate attributes.

3.2. Identifying and extracting the search results records and its attributes

After identifying the main data region, we must extract the SRRs and their respective attributes. This process is defined by the discovery of the SRRs' structure, and eliminates separators, invisible elements, and noise, and then wraps the attributes.

3.2.1. SRR selection

Although DERIN had already determined which region contained the SRRs, the structures of each of these records were unknown. To define these structures, we must determine where each SRR starts and ends, which requires separating each SRR and their attributes. While a structure is not defined, each child element of the main data region is an SRR. Specifically, let $R = \{r_1, r_2, \ldots, r_n\}$ be the set of n elements whose parent is the main data region. Using the example in Fig. 1, where the main data region would be the element (c), the set R would be formed by components (d) to (k) and two elements that are children of (c): the element containing information about what is been displayed and the element containing the ordering panel. The result of the initial definition of the SRRs can be seen in Fig. 5, where they are represented by the dashed rectangles.

3.2.2. Elimination of separators and non visible elements to the user

Some web pages present elements that are only used as separators, such as lines or blank spaces. In addition, some elements are invisible to the user but are necessary for data storage to support certain page features. These elements do not contain information vital to the user, and should therefore be eliminated by the algorithm.

These elements are removed from set R generating R' by:

$$R' = R - \{r_i \in R | content(r_i) = null\} \tag{5}$$

$$R' = R' - \{r_i \in R' | visible(r_i) = false\} \tag{6}$$

Table 1
Example of aligned attributes. Each column is a attribute and each row is a SRR.

AT01	AT02	AT03	AT04	AT05	AT06	AT07	AT08	AT09	AT10	AT11
Results...										
Sort by:	best user...	best match	price	best expert...						
					Nikon Coolpix...	img/photo01.jpg	\$ 59	and up	4 stores	User reviews (29)
					DSLR-A550 Di...	img/photo02.jpg	\$ 749	and up	4 stores	User reviews (8)
					Fujifilm FinePix...	img/photo03.jpg	\$ 119	and up	48 stores	User reviews (9)
					Panasonic Lu...	img/photo04.jpg	\$ 129	and up	60 stores	User reviews (2)
					Kodak EASYS...	img/photo05.jpg	\$ 59	and up	11 stores	User reviews (23)
					Canon EOS 5...	img/photo06.jpg	\$ 34	and up	32 stores	User reviews (157)
					Nikon Coolpix...	img/photo07.jpg	\$ 9.90	and up	9 stores	User reviews (51)
					Pentax Optio...	img/photo08.jpg	\$ 49	and up	20 stores	User reviews (24)

3.2.3. *Selection of candidates of SRRs attributes*

As previously mentioned, each SRR contains one or more units of information, called attributes. The algorithm considers as a probable attribute of an SRR each of its descendant elements. These probable attributes are called the candidate attributes of the SRR r_i and generate the set A_{r_i}, expressed as:

$$A_{r_i} = \{a_j | ancestor(a_j) = r_i \wedge r_i \in R'\} \tag{7}$$

where each attribute a_j is a descendent element from SRR r_i.

The diagram in Fig. 5 shows the results of applying the selection of an SRRs and its candidate attributes from the example in Fig. 1. As mentioned above, each dashed rectangle represents an SRR. The rectangles within the SRRs are their respective candidate attributes; some candidate attributes were ignored to simplify the example.

3.2.4. *Defining attributes through grouping per concept*

After defining all candidate attributes for each SRR, the algorithm must then group these attributes by concept (e.g., product name, price, photo). For grouping attributes by concept, a data alignment process based on their common characteristics, similar to that used by He, Meng, Zhao, and Yu (2007) was implemented, but with the following improvements:

- In the calculus of the similarity of SRR paths (SimR), this work used the tree edit distance, as opposed to string edit distance as tree edit distance has the advantage of accounting for the hierarchy of the nodes;
- when at least one of the compared attributes has no content, the presentation type and data type similarities are considered to be 0 (zero);
- two new data types were added: “image” and “URL”, which were treated in (He et al., 2007) as “plain text”;
- to calculate the cosine similarity between attributes of the URL type, the extraction process considered the various separators that can exist in a web address (such as bars, dots, etc.) differently from the extraction of the term “plain text”, where terms are separated by a blank space;
- a new style feature was created to indicate whether the attribute is an HTML link;
- we eliminated repeated attributes or attributes with no content, as they are considered useless to the user.

Next, we will describe the data alignment process implemented in this work.

Data alignment. Our data alignment process is based on the premise that data units belonging to the same concept share common characteristics. When grouped by concept, candidate attributes become attributes, where each SRR has just one attribute per concept. Applying this process to Fig. 5, produced the results shown in Table 1.

To confirm that attributes which represent the same concept are part of the same group, a method for comparison is required. For this, five features were considered, and are detailed below.

1. Data content: the content (or value) of attributes of the same concept usually share some terms, either because they are terms of the query that generated the search result page, or they contain terms used to identify the attribute.
2. Presentation style: attributes of the same concept are generally displayed with the same presentation style. Seven presentation features were used in this work: font type, font size, decoration (underline, strikethrough, etc.), font weight, font color, italic font, and whether a link is an HTML link.
3. Data type: based on its content, an attribute is defined as being of one of the following data types: common text, date, time, monetary value, integer, decimal, percentage, image, or URL.
4. Tag path: the sequence of HTML “tags” from the root element of the SRR to the attribute. Generally, attributes of the same concept are positioned in the same way in the SRRs.
5. Adjacency: the preceding and succeeding attributes of an attribute. As mentioned before, attributes of the same concept are generally positioned the same way in the SRRs. Thus, the concept of the attribute that precedes and the concept of the attribute the succeeds an attribute usually repeats in other SRRs.

Table 2
Example of aligned attributes after the elimination of repeated values.

AT01	AT02	AT03	AT04	AT05	AT06	AT07	AT08	AT09	AT10
Results... Sort by:	best user...	best match	price	best expert...					
					Nikon Coolpix...	img/photo01.jpg	$ 59	4 stores	User reviews (29)
					DSLR-A550 Di...	img/photo02.jpg	$ 749	4 stores	User reviews (8)
					Fujifilm FinePix...	img/photo03.jpg	$ 119	48 stores	User reviews (9)
					Panasonic Lu...	img/photo04.jpg	$ 129	60 stores	User reviews (2)
					Kodak EASYS...	img/photo05.jpg	$ 59	11 stores	User reviews (23)
					Canon EOS 5...	img/photo06.jpg	$ 34	32 stores	User reviews (157)
					Nikon Coolpix...	img/photo07.jpg	$ 9.90	9 stores	User reviews (51)
					Pentax Optio...	img/photo08.jpg	$ 49	20 stores	User reviews (24)

Similarities between candidate attributes. To determine whether two candidate attributes belong to the same concept, we calculated the similarities between them using the five features explained above. Thus, supposing two candidate attributes a_1 and a_2, these similarities are calculated as follows:

$$\begin{aligned} Sim(a_1,a_2) = {} & p1 * SimC(a_1,a_2) + p2 * SimE(a_1,a_2) + p3 * SimT(a_1,a_2) \\ & + p4 * SimR(a_1,a_2) + p5 * SimA(a_1,a_2) \end{aligned} \tag{8}$$

The similarities between each feature are calculated as follows:

- SimC (data content similarity): the cosine function similarity (Baeza-Yates & Ribeiro-Neto, 1999) is applied to the terms of both candidate attributes being compared.
- SimE (presentation style similarity): is the ratio of the number of equal style features, among the seven used, between the two attributes.
- SimT (data type similarity): if the data types of the two attributes being compared are the same, the similarity of the data type is 1 (completely similar); otherwise it is 0 (totally dissimilar).
- SimR (tag path similarity): the similarities between the paths of two attributes is calculated using the RTED algorithm, as explained in Section 3.1.3.
- SimA (adjacency similarity): is the average of similarity of the attribute which precedes and the attribute which succeeds each attribute that is being compared.

Parameters $p1$, $p2$, $p3$, $p4$ and $p5$ in the formula refer to the weights given to each feature used in data alignment. There is also a T similarity threshold, which is used to verify whether two attributes are similar. We normalized and used the same weights and threshold used in (He et al., 2007), specifically 0.25, 0.62, 1.00, 0.45 and 0.13; and a threshold of 0.86. He et al. (2007) used a genetic algorithm to obtain the best values of such parameters. We also test other values for these parameters, but none of them generated better results.

Grouping of similar candidate attributes. To group candidate attributes per concept, we used a hierarchical agglomerative clustering algorithm (Han & Kamber, 2006). Initially, each candidate attribute represents a cluster of single elements. These clusters are successively fused (or agglomerated) until all clusters are merged in a larger single cluster representing all attributes, or till a condition of stop is reached.

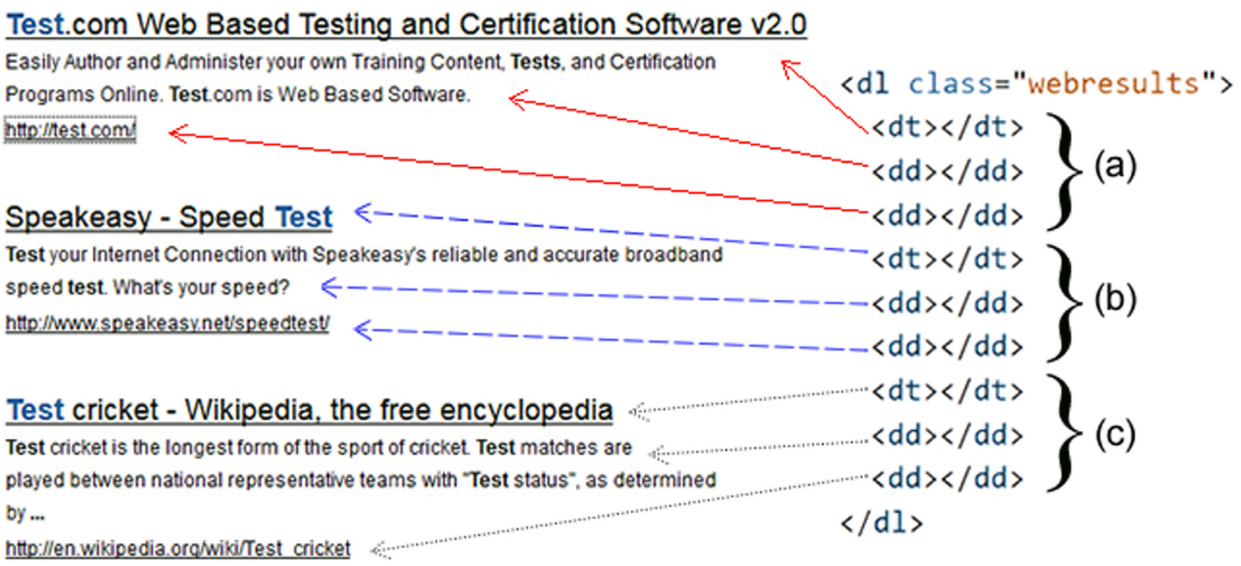
To calculate the distance between clusters we used the average linkage method, where the distance between two clusters is given by determining the average distance between all the elements of both groups; only clusters where the distance is less than or equal to $\theta_{\max}$ were considered. At the end of this process, each cluster is an attribute (i.e., each cluster represents a single concept).

After the defining the attributes via the process mentioned above, it was discovered that attributes with values which occur in all SRRs, such as example AT09 shown in Table 1 may exist. Thus, the example shown in Table 1 would be changed to that shown in Table 2. Currently, the set A of each SRR would contain their real attributes as well as candidate attributes.

3.2.5. Discovery of the structure of SRRs using n-grams

The discovery of SRRs using the proposed method occurs, as mentioned above, when all child elements of the main data region are viewed as SRRs. Therefore it is possible that after the elimination of separator and invisible elements, each remaining element R' would be the root node of an SRR.

However, there are some instances where an SRR is formed by more than one child element of the main data region, as shown in Fig. 6. This example shows the main data region of a page containing three records. In the right corner of the figure is the presentation of the HTML tags that form the displayed region. The element “dl” is the main data region and all its child elements should be SRRs. However, the first record is formed by the first “dt” element and by the second and third “dd” elements (represented by (a) in the figure). The same is true of the second and third SRRs, which are formed by the same sequence of HTML tags (represented by (b) and (c)). The result of the alignment process applied on this example is shown in Table 3. As can be seen, only 3 SRRs should have been defined, however 9 SRRs are shown.


Test.com Web Based Testing and Certification Software v2.0
Easily Author and Administer your own Training Content, Tests, and Certification Programs Online. Test.com is Web Based Software.
http://test.com/
Speakeasy - Speed Test
Test your Internet Connection with Speakeasy's reliable and accurate broadband speed test. What's your speed?
http://www.speakeasy.net/speedtest/
Test cricket - Wikipedia, the free encyclopedia
Test cricket is the longest form of the sport of cricket. Test matches are played between national representative teams with "Test status", as determined by ...
http://en.wikipedia.org/wiki/Test_cricket

```
<dl class="webresults">
<dt></dt>
<dd></dd>
<dd></dd>
<dt></dt>
<dd></dd>
<dd></dd>
<dt></dt>
<dd></dd>
<dd></dd>
</dl>
```

Fig. 6. Example of a structure where a SRR doesn't have a defined root element.

Table 3
Result of data alignment on the example in Fig. 6.

SRR/ATT.	AT01	AT02	AT03	AT04
SRR1	http://test.com/	Test .com Web Based ...		
SRR2			Easily author and admin...	
SRR3	http://test.com/			http://test.com/
SRR4	http://.../speedtest/	Speakeasy - Speed test		
SRR5			Test your internet Connect...	
SRR6	http://.../speedtest/			http://.../speedtest/
SRR7	http://.../Test_cricket	Test cricket - Wikipedia, ...		
SRR8			Test cricket is the longest...	
SRR9	http://.../Test_cricket			http://.../Test_cricket

To deal with such instances, DERIN performs a structure discovery process based on an n-gram model. The main purpose behind the use of n-grams is to try to find standard patterns of attributes and define where each SRR begins and ends. This process involves following steps:

Step 1: Verifying the cohesion of the SRRs. Before applying the structure discovery process, we must first determine whether it is required, which is accomplished by calculating the cohesion of the SRR's contained in R'.

Thus, a similarity matrix is created, which can be understood as a "$n \times n$" table that stores the values of each proximity pair of n objects (Han & Kamber, 2006). These objects are SRRs and the values contained in this array express the similarities between each pair of SRRs. To calculate those similarities, we used the Jaccard index on the matching attributes between the two compared SRRs. Attributes match when they exist in both SRRs independently of their content.

Next, the resulting similarity matrix is compared with an optimal similarity matrix. The optimal similarity matrix is a matrix containing the same pairs of SRRs of the above matrix, but all the similarity values are equal to 1 (i.e., all pairs of SRRs are entirely similar). To compare matrices with one another, the sum of all values of the similarity matrix are divided by the sum of all similarity values of the optimal matrix, giving a percentage representing the ratio of similarities between the SRRs represented therein. This percentage is the cohesion index of the SRRs of the main data region, and is compared with the parameter $\%_{cohesion}$ (the value of $\%_{cohesion}$ is explained in Section 4.4). If the cohesion index is lower than $\%_{cohesion}$, this means that attribute structures of the SRRs are markedly different, and we therefore apply the structure discovery by n-grams method described in the next step. If the contrary is true, we apply the function to remove the SRR outliers that are explained in Section 3.2.6.

Step 2: Extracting n-grams. To extract the n-grams, the names of the attributes that occur in each SRR are concatenated in a string in the order in which they appear, separated by a blank space. For example, the concatenation of the names of the attributes in Table 3 generate the following string: "AT01 AT02 AT03 AT01 AT04 AT01 AT02 AT03 AT01 AT04 AT01 AT02 AT03 AT01 AT04".

Table 4
Result of applying n-grams in the example of Fig. 6.

SRR/ATT.	AT01	AT02	AT03	AT01	AT04
SRR1	http://test.com/	Test .com Web Based ...	Easily author and ...	http://test.com/	http://test.com/
SRR2	http://.../speedtest/	Speakeasy - Speed Test	Test your Internet ...	http://.../speedtest/	http://.../speedtest/
SRR3	http://.../Test_cricket	Test cricket - Wikipedia, ...	Test cricket is the ...	http://.../Test_cricket	http://.../Test_cricket

Table 5
Result of data realignment applied in Table 4.

SRR/ATT.	AT01	AT02	AT03
SRR1	http://test.com/	Test .com Web Based Testing...	Easily Author and Administer...
SRR2	http://.../speedtest/	Speakeasy - Speed Test	Test your Internet Connection...
SRR3	http://.../Test_cricket	Test cricket - Wikipedia, ...	Test cricket is the longest...

From this string, DERIN creates a set of extracted n-grams that has a size limit of $n_{\max}$. This limit is calculated as follows:

$$n_{max} = \frac{|A_s|}{\#_{min}} \tag{9}$$

where A_s is the set of attributes separated by blank space in the string, and the parameter $\#_{\min}$ is defined in Section 3.1.1, which refers to the minimum number of SRRs that the main data region should have to be handled by DERIN. Hence, for the example above, $|A_s|$ has a value equal to 15 and, if we consider $\#_{\min}$ equal to 3, the result would be n equal to 5. Therefore, this method avoids n-grams which would represent structures of less than $\#_{\min}$ SRRs.

Thus, the sets of n-grams generated for the example above would be as follows:

- n=1: {AT01}, {AT02}, {AT03}, {AT04}
- n=2: {AT01,AT02}, {AT02,AT03}, {AT03,AT01}, {AT01,AT04}, {AT04,AT01}
- n=3: {AT01,AT02,AT03}, {AT02,AT03,AT01}, {AT03,AT01,AT04}, {AT01,AT04,AT01}, {AT04,AT01,AT02}
- n=4: {AT01,AT02,AT03,AT01}, {AT02,AT03,AT01,AT04}, {AT03,AT01,AT04,AT01}, {AT01,AT04,AT01,AT02}, {AT04,AT01,AT02, AT03}
- n=5: {AT01,AT02,AT03,AT01,AT04}, {AT02,AT03,AT01,AT04,AT01}, {AT03,AT01, AT04,AT01,AT02}, {AT01,AT04,AT01,AT02, AT03}, {AT04,AT01,AT02,AT03,AT01}

Each of the n-grams generated represents sequences of attributes that could be classified as SRRs. Next, the n-gram or n-grams which correctly represent the beginning and the end of each SRR must be defined, as detailed in the next step.

Step 3: Classifying the n-grams. Each n-gram receives a score, calculated as follows:

$$score = \frac{n * f}{s} \tag{10}$$

where n is the size of the n-gram, f is its frequency (the number of times the n-gram occurs within the string), and s is the number of attributes which remain after removing all occurrences of that n-gram from the string. For example, consider the n-gram {AT01,AT02,AT03,AT01} which has a size of $n = 4$. Its frequency f in the string is equal to 3 and after removing all its occurrences, three attributes remain, i.e., $s = 3$. Hence, its $score = 4$.

After the *score* of each n-gram are calculated, they are added to a stack ranked in decreasing order by this *score*, so they can be used in the next step.

Step 4: Restructuring the SRRs using n-grams. In this step, an n-gram is removed from the stack and used as a definer of an SRR's structure. In the example used, the n-gram, {AT01,AT02,AT03, AT01,AT04} would have the highest score and, therefore, be the first to be used. Each record would have the attributes structure in the same sequence in which they appear in the n-gram. The new SRR structure can be seen in Table 4. It is important to note that, instead of 9 SRRs, there are only 3; this is an actual representation of what shown in Fig. 6.

After we apply this n-gram to restructure the records, the process is repeated using the next n-gram in the stack on the remaining attributes. The *score* is recalculated after each n-gram is popped from the stack. If the correct sequence of the n-gram is not found among the remaining attributes, it is ignored. This process is repeated until all attributes are part of an SRR.

Step 5: Realigning of SRR's data. Once the SRRs are defined with the extracted structures of the n-grams, their attributes must be realigned. It is necessary to ensure that the attributes represent the same concept, and that repeated and empty values are removed. The result of the realignment of the data of Table 4 is shown in Table 5.

3.2.6. Eliminating of SRRs outliers

It is common that elements which do not represent records of the database are found in the main data region in the search result page. For example, in Table 2, the first two rows represent such elements, and must be eliminated from the result.

DERIN eliminates these elements, called SRRs outliers, using the statistic technique of lower and upper quartiles of a set of data. This process is as follows:

- Data ordering: for each SRR, its average similarity is calculated with regards to the other SRRs. The Jaccard index is used as an indicator of similarity. The SRRs are then ordered per this average similarity, thereby forming the set of values to be used.
- Defining the lower and upper quartiles: the lower quartile, called Q_1, delimits 25% of the lowest values of the data; the upper quartile, called Q_3, delimits 25% of the highest values. A breakdown of the calculation of the quartiles can be found in Han and Kamber (2006).
- Calculating the interquartile amplitude: interquartile amplitude is the distance between the lower quartile and the upper quartile, called *IQR*, and is given by:

$$IQR = Q_3 - Q_1 \tag{11}$$

- Defining limiters: the lower (L_1) and upper (L_3) limiters, used in eliminating outliers, are calculated by applying a factor of 1.5 (Han & Kamber, 2006), as follows:

$$L_1 = Q_1 - (IQR * 1.5) \tag{12}$$

$$L_3 = Q_3 + (IQR * 1.5) \tag{13}$$

Therefore, in the sorted set of data, all values below L_1 and above L_3 are considered outliers. However, as the sorted data represents values of average similarity, where the lower similar SRRs (that must be removed) are at the beginning of the list, this method simply eliminate SRRs whose average similarity is below L_1; L_3 is ignored.

4. Experiments

In this section, we present experimental results that demonstrate the effectiveness of our proposed DERIN method. We first describe the collections, baselines, and the evaluation metric. Then, we discuss the effectiveness of DERIN in comparison with the baselines.

4.1. Collections

To evaluate the effectiveness of DERIN, we adopted collections of web pages used in (Trieschnigg et al., 2012), hereinafter referred to as WEBT, and in (Velloso and Dorneles, 2013), hereinafter referred to as WEBV. The collections are described below.

WEBT collection

The WEBT collection of search result pages was assembled by Trieschnigg et al. (2012) by combining two collections, Web1 and Web2, and contains 220 web pages. Web1 contains the search result pages obtained from the top 500 US websites listed by Alexa[4], removing websites that require a user account (LinkedIn, Facebook), contain pornographic material, and torrents. The authors used the search function on the main page of the websites for obtaining the search result pages of each site. The complete result page was downloaded using Mozilla Archive Format[5], which includes all images and CSS (cascading style sheet) files and removes javascript. Web2 contains the search result pages obtained from the top UK websites listed by Alexa in a manner similar to that used for Web1; websites already in Web1 were removed.

WEBV collection

The WEBV collection was assembled by Velloso and Dorneles (2013) and contains 23 search result pages. The search result pages of this collection contain only the source code of the HTML pages, without images or CSS. Thus, its pages are wrongly rendered and present messy layouts, different from when accessing them on the web. In order to solve this problem, we accessed the websites again, searched for the same terms as in (Velloso and Dorneles, 2013), and saved the entire search result pages on a disc, including images and CSS.

Table 6 shows the popularity of each application domain in both collections.

[4] http://www.alexa.com/topsites.

[5] http://maf.mozdev.org/.

Table 6
Distribution of search results pages in application domains.

Domain	WEBV		WEBT	
	# of pages	% of pages	# of pages	% of pages
Academic	3	13.04	3	1.36
Technical articles	0	0.00	13	5.91
Bank	2	8.70	5	2.27
Community	0	0.00	11	5.00
e-Commerce	9	39.13	39	17.73
Government	0	0.00	3	1.36
Images	0	0.00	19	8.64
Company	0	0.00	19	8.64
Games	0	0.00	1	0.45
Books	0	0.00	1	0.45
Search engine	2	8.70	26	11.82
Music	0	0.00	3	1.36
News	5	21.74	50	22.73
Recipes	0	0.00	1	0.45
Job	0	0.00	6	2.73
Vehicle & real state	1	4.35	4	1.82
Video	1	4.35	16	7.27
All domains	23		220	

4.2. Baseline

We used the methods proposed by Velloso and Dorneles (2013) and Trieschnigg et al. (2012) as baselines, from which we obtained the collections, along with Zhai and Liu (2005). The method proposed by Velloso and Dorneles (2013) was used only in the evaluation of the main data region selection, the method proposed by Trieschnigg et al. (2012) was used in the evaluation of main data region selection and SRR extraction, and the method proposed by Zhai and Liu (2005) was used to evaluate the main data region selection, SRR extraction, and the extraction of their attributes. Each of these works are described in Section 2.

When using Velloso and Dorneles (2013) as a baseline in the selection of the main data region, we checked the new pages generated using this method (containing only the main data region for each page) to calculate its effectiveness. It was considered that the main data region was correctly selected if the generated page contained only the main data region of the source page (without any other data regions).

When using Trieschnigg et al. (2012) as a baseline in the selection of the main data region, it was necessary to execute this method on each page of the collections and check whether the XPath expression selects only elements inside the main data region. If at least one element selected was outside the main data region (for example, if a menu was selected), the method was considered to have failed in selecting the main data region. When using this method as a baseline in the extraction of the SRRs, we checked the number of SRRs which were correctly extracted the number that were not. This method has three parameters: *minSimilarityThreshold, avgSimilarityThreshold* and *minimumNodeCount*.

When using Zhai and Liu (2005) as a baseline in the selection of the main data region, we considered only the “table” element of the main data region. If the data of the main data region was split over more than one “table” element, only the element containing the highest number of records was considered, and only data from that “table” element was used to evaluate the extraction of SRRs and their attributes.

4.3. Evaluation metrics

To evaluate the effectiveness of the methods, we used the precision, recall, and F-measure (F1) metrics, calculated as follows:

$$precision = \frac{tp}{tp + fp} \tag{14}$$

$$recall = \frac{tp}{tp + fn} \tag{15}$$

$$F1 = \frac{2 \times precision \times recal}{precision + recall} \tag{16}$$

tp is the total number of objects (main data regions, SRRs or attributes) correctly identified by a method, *fp* is the total number of objects incorrectly identified by a method, and *fn* is the number of correct objects not identified by a method.

Table 7
Parameter values used by DERIN.

Parameter	Description	Value
$\#_{\min}$	Minimum number of children elements	3
$\%_{area}$	Percentage of body's area filled by an element	0.1
$\%_{height}$	Percentage of body's height filled by an element	0.2
$\%_{width}$	Percentage of body's width filled by an element	0.3
$\delta_{\max}$	Average distance between subtrees	2
$\%_{parent}$	Ratio of an element area by its parent	0.2
$\theta_{\max}$	Maximum distance between clusters	0.5
$\%_{cohesion}$	Minimum cohesion threshold to decide whether or not use the structure discovery using n-grams	0.5

Table 8
Effectiveness of the main data region selection in each collection.

Domain	# of main data regions incorrectly selected	
	WEBV	WEBT
Academic	0	0
Technical articles	0	0
Bank	1	1
Community	0	0
e-Commerce	1	3
Government	0	0
Images	0	2
Company	0	2
Games	0	0
Books	0	0
Search engine	0	1
Música	0	0
News	0	7
Recipes	0	0
Job	0	1
Vehicle & real state	0	0
Video	0	1
Precision	91.30	91.81

4.4. Experiments setup

Experiments were performed on each collection. Each method was executed in each web page and manually checked to determine whether the main data region was correctly selected. For the evaluation of the extraction of the SRRs and their attributes, we counted the amount of correctly and incorrectly extracted SRRs and attributes. Based on this, we calculated precision, recall, and F-measure of these tests.

Table 7 shows the values of the parameters used by DERIN in the experiments. It is important to note that these values are the best configuration for WEBT and have not been changed for any specific site, i.e., are the same for all pages and collections.

When using the method proposed in (Trieschnigg et al., 2012), we used the same parameter values defined by the authors; $minSimilarityThreshold = 0.55$, $avgSimilarityThreshold = 0.65$, and $minimumNodeCount = 3$. Regarding the method proposed in (Velloso and Dorneles, 2013), the only parameter is the minimum size percentage of a region compared with the rest of the sequence, so that a given region can be considered the main data region, and this percentage is 0.2. The method proposed in (Zhai and Liu, 2005) has no parameters.

4.5. Results and discussion

Effectiveness of the selection of the main data region

Table 8 shows the number of main data regions incorrectly selected by DERIN in the WEBV and WEBT collections in each domain. The last line of the table shows the precision of DERIN for each collection.

In the WEBV collection, only 2 main data regions were incorrectly selected using our method. We analyzed each case and noticed that the first search result page ("bradesco.com.br"), whose main data region was incorrectly selected, uses the "iframe" element to encompass SRRs. This type of element inserts one page inside another, and thus prevents our method from providing an expression to select the main data region, as the "iframe" element appears empty.

In addition, our method incorrectly selected the main data region for the search result page of the web site "americanas.com". This page contains some elements which are not SRRs, but whose structures are similar to SRRs. These elements

Table 9
Comparison of the precision of the main data region selection with the baselines.

Methods	Collections	
	WEBV	WEBT
DERIN	91.30	91.81
Method proposed by Velloso and Dorneles (2013)	86.96	45.45
Method proposed by Trieschnigg et al. (2012)	86.96	89.55

Table 10
Effectivenes of the extraction of SRRs in WEBV compared with the baselines.

Domain	Trieschnigg et al. (2012)			DEPTA			DERIN		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Academic	74.36	72.50	73.42	100.00	82.50	90.41	97.44	95.00	96.20
Bank	0.00	0.00	0.00	33.33	15.00	20.69	60.00	45.00	51.43
e-Commerce	93.63	83.04	88.01	91.53	38.16	53.87	89.06	83.39	86.13
Search engine	100.00	100.00	100.00	0.00	0.00	0.00	100.00	100.00	100.00
News	100.00	66.67	80.00	100.00	73.33	84.62	100.00	100.00	100.00
Vehicle & real state	100.00	50.00	66.67	100.00	35.00	51.85	100.00	100.00	100.00
Video	100.00	100.00	100.00	100.00	100.00	100.00	100.00	90.00	94.74
Total	92.95	76.73	84.07	92.73	45.64	61.17	91.45	86.13	88.71

induced DERIN in the last strategy of the third step (defining the expression to select the main data region) to incorrectly construct XPath expressions.

In the WEBT collection, amidst 220 search result pages, 20 main data regions were incorrectly selected using our method. We checked the corresponding 20 pages and noticed that:

- In 11 search result pages, the failure occurred because DERIN, in the last strategy of the third step, finds similar subtrees which do not contain SRRs or, due to similarity threshold δ_{min}, subtrees with SRRs are not considered similar;
- In 7 search result pages, the failure occurred because elements without SRRs have larger areas than elements with SRRs;
- When loaded by the browser, a search result page is automatically redirected to another page, which is different from the page in the collection.
- A search result page uses an "iframe" HTML to encompass the main data region. Thus, the main data region comes from another document and, therefore the XPath expression does not select it.

Effectiveness of the main data region selection compared to baselines

Table 9 shows the comparison of DERIN with the baselines in both collections, in terms of precision for selecting the main data region. In the WEBV collection, the baselines obtained the same precision (86.96%). DERIN showed a gain of approximately 5% when compared with both baselines. In the WEBT collection, the gains of our method were approximately 100% and 2.5% when compared with the methods proposed in Velloso and Dorneles (2013) and Trieschnigg et al. (2012), respectively.

Effectiveness of the Eextraction of SRRs and its attributes

In Tables 10 and 11 we compare the effectiveness of DERIN against the baselines in the extraction of SRRs from the collections WEBV and WEBT, respectively. Additionally, in Tables 12 and 13, we evaluate the attribute extraction process of the SRRs under precision, recall, and F1 measures. Each line represents an application domain, with the last line showing overall efficacy. The comparison of our method with the baselines is performed using the $F1$ measure, which represents the harmonic mean of precision and recall.

Hence, after analysis of the extraction of SRRs under $F1$ measure, DERIN showed gains of 5.5% over (Trieschnigg et al., 2012) and 45.0% regarding over DEPTA in WEBV. In WEBT, the gains were of 4.6% with regarding over (Trieschnigg et al., 2012) and 11.8% regarding to DEPTA. It should also be noted that DERIN showed a $F1$ greater than 80% in 6 of 7 domains of WEBV and in 15 of 17 domains of WEBT.

Regarding the extraction of SRR attributes, our method was compared only to DEPTA, as the experiments in (Trieschnigg et al., 2012) did not extract attributes. Thus, using $F1$ as the comparator, DERIN showed gains of 63.0% in relation to DEPTA in WEBV and 42.5% in WEBT. Tables 12 and 13 show that DERIN also obtained a $F1$ greater than DEPTA in all domains in WEBV and in 14 of 17 domains in WEBT.

To measure the influence of the processes which eliminates SRR outliers and determine SRR structure, the proposed method was executed with and without each of these processes. The results are shown in Table 14, where the first line is the full execution of the proposed method.

In WEBV, it was evident that the execution of DERIN without processes for eliminating SRRs outliers and determining SRR structure resulted in a rise of precision, as opposed to an expected drop(as occurred in WEBT). The process which

Table 11
Effectiveness of the extraction of SRRs in WEBT compared with the baselines.

Domain	Trieschnigg et al. (2012)			DEPTA			DERIN		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Academic	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Technical articles	84.33	81.29	82.78	100.00	70.50	82.70	83.10	84.89	83.99
Bank	100.00	100.00	100.00	100.00	92.00	95.83	86.96	80.00	83.33
Community	83.33	82.93	83.13	100.00	93.17	96.46	82.03	86.83	84.36
e-Commerce	86.08	64.50	73.74	100.00	47.56	64.46	95.17	90.43	92.74
Government	100.00	100.00	100.00	100.00	100.00	100.00	93.02	100.00	96.39
Images	88.13	82.62	85.28	93.53	81.84	87.29	95.68	77.93	85.90
Instituicional	58.82	70.18	64.00	100.00	84.50	91.60	82.54	85.67	84.07
Games	100.00	100.00	100.00	100.00	20.00	33.33	100.00	90.00	94.74
Books	50.00	100.00	66.67	100.00	15.00	26.09	100.00	100.00	100.00
Search engine	89.97	87.39	88.66	100.00	64.47	78.40	87.83	84.81	86.30
Música	59.70	77.67	67.51	100.00	84.47	91.58	100.00	100.00	100.00
News	93.39	86.36	89.74	95.99	57.81	72.16	80.19	85.65	82.83
Recipes	90.91	100.00	95.24	100.00	50.00	66.67	0.00	0.00	0.00
Job	97.69	100.00	98.83	100.00	20.47	33.99	68.66	72.44	70.50
Vehicle & real state	100.00	100.00	100.00	100.00	90.59	95.06	100.00	88.24	93.75
Video	100.00	98.39	99.19	99.21	50.81	67.20	87.50	84.68	86.07
Total	85.73	80.54	83.05	98.23	64.25	77.69	87.71	86.01	86.85

Table 12
Effectiveness of the extraction of SRRs attributes in WEBV.

Domain	DEPTA			DERIN		
	Precision	Recall	F1	Precision	Recall	F1
Academic	57.58	89.76	70.15	89.76	89.76	89.76
Bank	17.74	26.83	21.36	65.85	65.85	65.85
e-Commerce	43.97	43.49	43.73	71.56	80.38	75.72
Search engine	0.00	0.00	0.00	100.00	100.00	100.00
News	77.86	85.00	81.27	97.92	97.92	97.92
Vehicle & real state	32.22	80.24	45.98	100.00	100.00	100.00
Video	49.05	100.00	65.81	85.44	85.44	85.44
Total	46.30	53.12	49.48	77.46	84.15	80.67

eliminates SRRs outliers attempts to eliminate elements which are in the main data region but are not database records. This is accomplished by detecting likely SRRs whose structures are markedly different from the others. This process is highly sensitive to SRRs with greater or fewer attributes than others, and can mistakenly eliminate elements that truly represent database records. Thus, we concluded that more web pages with vastly different structures exist in the SRRs of WEBV than in those of WEBT.

5. Conclusion

In this work, we proposed DERIN, an automatic method for detecting the main data regions of search result pages and extracting their records and attributes. The method uses rendered information of the elements from web pages, analyses the DOM tree, and determines SRRs' structure using n-grams, aiming to extract only data concerning the records from the website. Moreover, DERIN can detect different record structures and does not require previous examples to learn to extract the data.

In the experimental evaluation, we first compared DERIN with other automatic methods for detecting main data regions, and DERIN showed gains of between 2.5% and 100% in relation to the baseline methods. In addition, we concluded that the filtering method which had the most influence on detection accuracy was the comparison of the area of an element to that of its parent element. Regarding the detection SRR structure and extraction of SRRs, DERIN showed gains ranging from 4.6% to 45%. Finally, regarding SRR attribute extraction, gains ranged from 42.5% to 63.0%.

Although experiments have shown that our proposed method is effective, it has some limitations. As with any method working with DOM trees of HTML pages, DERIN assumes that the source code of those pages is well structured and does not contain errors. Though many HTML parsers that correct many of these errors exist, they do not ensure complete correction. This characteristic can lead DERIN to derive incorrect height and width values. Another problem detected in some cases was the incorrect rendering of information provided by the browser or MSHTML. This is because in some cases, an element occupies a large area on the screen but its height and width values are incorrectly set to zero.

In future work, we intend to study methods eliminating or automatically configuring the parameters used in the proposed method. Furthermore, the process of eliminating SRR outliers while retaining elements which represent SRRs with

Table 13
Effectiveness of the extraction of SRRs attributes in WEBT.

Domain	DEPTA			DERIN		
	Precision	Recall	F1	Precision	Recall	F1
Academic	89.74	100.00	94.59	99.43	99.43	99.43
Technical articles	51.82	59.79	55.52	60.08	60.21	60.15
Bank	86.06	83.26	84.63	82.07	70.23	75.69
Community	63.42	89.31	74.17	76.44	76.71	76.57
e-Commerce	53.10	50.23	51.63	82.32	85.84	84.04
Government	58.33	87.50	70.00	66.82	91.88	77.37
Images	40.59	50.16	44.87	70.78	78.34	74.37
Instituicional	70.78	84.96	77.23	74.66	72.34	73.48
Games	16.50	17.17	16.83	100.00	90.40	94.96
Books	13.88	31.29	19.23	85.25	85.25	85.25
Search engine	53.83	63.45	58.25	79.81	73.27	76.40
Música	68.51	91.14	78.22	81.65	87.47	84.46
News	53.73	59.68	56.55	71.30	82.73	76.59
Recipes	41.18	58.33	48.28	0.00	0.00	0.00
Job	11.47	13.10	12.23	41.40	50.72	45.58
Vehicle & real state	61.90	78.19	69.10	98.66	89.43	93.82
Video	53.56	45.97	49.47	81.94	75.57	78.63
Total	51.93	56.90	54.30	75.74	79.06	77.37

Table 14
Effectiveness without eliminating SRRs outliers and without finding out structure using n-grams.

Execution type	WEBV		WEBT	
	F1 - SRRs	F1 - Attributes	F1 - SRRs	F1 - Attributes
DERIN	88.71	80.67	86.85	77.37
Without eliminating SRRs outliers	89.68	83.53	84.28	73.80
Without structure discovery using n-grams	85.65	79.38	79.83	73.25
Without both steps above	90.13	85.78	83.36	75.57

greater or fewer attributes than is normally the case must be improved. Regarding attribute extraction, our intention is to develop a method for automatically detecting, separating, and labeling multi-valued attributes (i.e., those which contain various concepts separated by elements or characters).

Acknowledgements

This research is partially funded by the FAPEMIG/PRONEX/MASWeb project (grant number APQ-01400-14) and by the authors' individual research grants from CAPES and UFOP.

References

Adelberg, B. (1998). Nodose: A tool for semi-automatically extracting structured and semistructured data from text documents. *Special Interest Group on Management of Data - SIGMOD Rec., 27*(2), 283–294. doi:10.1145/276305.276330.

Ansari, A., & Vasishtha, H. (2015). Data record extraction using tag tree comparison. *International Journal of Computer Applications, 117*(11), 20–24.

Arasu, A., & Garcia-Molina, H. (2003). Extracting structured data from web pages. In *Proceedings of the 2003 acm sigmod international conference on management of data*. In *SIGMOD '03* (pp. 337–348). San Diego, California: ACM. doi:10.1145/872757.872799.

Arocena, G. O., & Mendelzon, A. O. (1999). Weboql: Restructuring documents, databases, and webs. *Theory and Practice of Object Systems, 5*(3), 127–141.

Baeza-Yates, R. A., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Cai, D., Yu, S., Wen, J.-R., & Ma, W.-Y. (2003). Extracting content structure for web pages based on visual representation. In *Proceedings of the 5th asia-pacific web conference on web technologies and applications. xian, china*. In *APWeb'03* (pp. 406–417). Xian, China: Springer-Verlag.

Chang, C.-H., Chen, T.-S., Chen, M.-C., & Ding, J.-L. (2016). Efficient page-level data extraction via schema induction and verification. *Advances in Knowledge Discovery and Data Mining (PAKDD 2016)*, 478–490.

Chang, C.-H., Hsu, C.-N., & Lui, S.-C. (2003). Automatic information extraction from semi-structured web pages by pattern discovery. *Decision Support Systems, 35*(1), 129–147.

Chu, Y.-C., Hsu, C.-C., Lee, C.-J., & Tsai, Y.-T. (2015). Automatic data extraction of websites using data path matching and alignment. *The Fifth International Conference on Digital Information Processing and Communications (ICDIPC2015)*, 60–64.

Crescenzi, V., & Mecca, G. (1998). Grammars have exceptions. *Information Systems, 23*(9), 539–565.

Crescenzi, V., Mecca, G., & Merialdo, P. (2001). Roadrunner: Towards automatic data extraction from large web sites. In *Proceedings of the 27th international conference on very large data bases*. In *VLDB '01* (pp. 109–118). Roma, Italy: Morgan Kaufmann Publishers Inc.

Don, Y., Chu, Q., & Ling, P. (2015). Web data extraction based on ensemble learning. *International Journal of Database Theory and Application, 8*(3), 311–322.

Ferrara, E., De Meo, P., Fiumara, G., & Baumgartner, R. (2014). Web dat extraction, aplications and techniques: A survey. *Knowledge-Based Systems, 70*, 301–323.

Figueiredo, L. N. L., Ferreira, A. A., & de Assis, G. T. (2014). A rendering-based method for selecting the main data region in web pages. In *Proccedings of the 9th latin american web congress, Ouro Preto, Brazil* (pp. 24–32).

Fumarola, F., Weninger, T., Barber, R., Malerba, D., & Han, J. (2011). Extracting general lists from web documents: A hybrid approach. In *Proceedings of the 24th international conference on industrial engineering and other applications of applied intelligent systems conference on modern approaches in applied intelligence - volume part i*. In *IEA/AIE'11* (pp. 285–294). Syracuse, NY: Springer-Verlag.

Grigalis, T., & Cenys, A. (2014). Unsupervised structured data extraction from template-generated web pages. *Journal of Universal Computer Science, 20*(2), 169–192.

Hammer, J., McHugh, J., & Garcia-Molin, H. (1997). Semistructured data: The tsimmis experience. In *Proceedings of the first east-european conference on advances in databases and information systems*. In *ADBIS'97*. Swinton, UK, UK: British Computer Society. 22–22

Han, J., & Kamber, M. (2006). *Data mining: Concepts and techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc..

He, H., Meng, W., Zhao, H., & Yu, C. (2007). Annotating structured data of the deep web. In *In: Proceedings of the ieee 23rd international conference on data engineering* (pp. 376–385). Istanbul, Turkey: Society Press.

Hiremath, P. S., & Algur, S. P. (2009). Extraction of data from web pages: A vision based approach. *International Journal on Computer Science and Engineering, 1*(3), 50–59.

Hsu, C.-N., & Dung, M.-T. (1998). Generating finite-state transducers for semi-structured data extraction from the web. *Information Systems, 23*(9), 521–538.

Irmak, U., & Suel, T. (2006). Interactive wrapper generation with minimal user effort. In *Proceedings of the 15th international conference on world wide web*. In *WWW '06* (pp. 553–563). Edinburgh, Scotland: ACM. doi:10.1145/1135777.1135859.

Kadam, V. B., & Pakle, G. K. (2014). Deuds: Data extraction using dom tree and selectors. *International Journal of Computer Science and Information Technologies, 5*(2), 1403–1410.

Krupl-Sypien, B., Fayzrakhmanov, R. R., Holzinger, W., Panzenbïck, M., & Baumgartner, R. (2011). A versatile model for web page representation, information extraction and content re-packaging. . In M. R. B. Hardy, & F. W. Tompa (Eds.), *Acm symposium on document engineering* (pp. 129–138). Mountain View, CA, USA: ACM.

Kushmerick, N. (1997). *Wrapper induction for information extraction*. Ph.D. thesis. University of Washington, AAI9819266.

Laender, A. H. F., Ribeiro-Neto, B., & da Silva, A. S. (2002). Debye - data extraction by example. *Data and Knowledge Engineering, 40*(2), 121–154. doi:10.1016/S0169-023X(01)00047-7.

Li, L., Liu, Y., & Obregon, A. (2007). Visual segmentation-based data record extraction from web documents. In *Information reuse and integration, Las Vegas, IL* (pp. 502–507).

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*. In *KDD '03* (pp. 601–606). Washington, D.C.: ACM. doi:10.1145/956750.956826.

Liu, L., Pu, C., & Han, W. (2000). Xwrap: An xml-enabled wrapper construction system for web information sources. *16th International Conference on Data Engineering (ICDE'00), 0*, 611.

Liu, W., Meng, X., & Meng, W. (2010). Vide: A vision-based approach for deep web data extraction. *IEEE Transactions on Knowledge and Data Engineering, 22*(3), 447–460. doi:10.1109/TKDE.2009.109.

Muslea, I., Minton, S., & Knoblock, C. (1998). Stalker: Learning extraction rules for semistructured. *In american association for artificial intelligence (aaai): Workshop on ai and information integration, Madison, Wisconsin, USA*.

Pawlik, M., & Augsten, N. (2011). Rted: A robust algorithm for the tree edit distance. *Very Large Data Bases (VLDB) Endowment, 5*(4), 334–345.

Sahuguet, A., & Azavant, F. (1999). Wysiwyg web wrapper factory (w4f). In *Proceedings of the 8th international conference on world wide web, Toronto, Canada* (pp. 1–22).

Shi, S., Liu, C., Shen, Y., Yuan, C., & Huang, Y. (2015). Autorm: An effective approach for automatic web data record mining. *Knowledge-Based Systems, 89*(C), 314–331.

Simon, K., & Lausen, G. (2005). Viper: Augmenting automatic information extraction with visual perceptions. In *Proceedings of the 14th acm international conference on information and knowledge management*. In *CIKM '05* (pp. 381–388). Bremen, Germany: ACM. doi:10.1145/1099554.1099672.

Trieschnigg, R. B., Tjin-Kam-Jet, K. T. T. E., & Hiemstra, D. (2012). Ranking XPaths for extracting search result records. *Technical Report, TR-CTIT-12-08*. Enschede: Centre for Telematics and Information Technology, University of Twente.

Uzun, E., Agun, H. V., & Yerlikaya, T. (2013). A hybrid approach for extracting informative content from web pages. *Information Processing and Management, 49*(4), 928–944.

Velloso, R. P., & Dorneles, C. F. (2013). Automatic web page segmentation and noise removal for structured extraction using tag path sequences. *Journal of Information and Data Management, 4*(3), 173–187.

Wang, J., & Lochovsky, F. H. (2003). Data extraction and label assignment for web databases. In *Proceedings of the 12th international conference on world wide web*. In *WWW '03* (pp. 187–196). Budapest, Hungary: ACM. doi:10.1145/775152.775179.

Zhai, Y., & Liu, B. (2005). Web data extraction based on partial tree alignment. In *Proceedings of the 14th international conference on world wide web*. In *WWW '05* (pp. 76–85). Chiba, Japan: ACM. doi:10.1145/1060745.1060761.

Zhao, H., Meng, W., Wu, Z., Raghavan, V., & Yu, C. (2005). Fully automatic wrapper generation for search engines. In *Proceedings of the 14th international conference on world wide web*. In *WWW '05* (pp. 66–75). Chiba, Japan: ACM. doi:10.1145/1060745.1060760.