A Pure Visual Approach for Automatically Extracting and Aligning Structured Web Data

by

Fadwa M A Estuka

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

Database-driven Websites and the amount of data stored in their databases are enormously growing. Web databases retrieve relevant information in response to users' queries; the retrieved information is encoded in dynamically generated Web pages in the form of structured data records. Identifying and extracting retrieved data records is a fundamental task for many applications, such as competitive intelligence and comparison shopping. This task is challenging due to the complex underlying structure of such Web pages and the existence of irrelevant information. Numerous approaches have been introduced to address this problem, but most of them are HTML-dependent solutions which may no longer be functional with the continuous development of HTML. Although, few vision-based techniques have been built upon the visual presentation of Web page objects, various issues exist that inhibit their performance. To overcome this, we propose a novel visual approach, i.e., programming-language-independent, for automatically extracting structured Web data. The proposed approach makes full use of the natural human tendency of visual object perception and the Gestalt laws of grouping. The extraction system consists of two tasks: (1) data record extraction where we apply three of the Gestalt laws (i.e., laws of continuity, proximity, and similarity) which are used to group the adjacently aligned visually similar data records on a Web page; and (2) data item extraction and alignment where we employ the Gestalt law of similarity which is utilized to group the visually identical data items. Our experiments upon large-scale test sets show that the proposed system is highly effective, and outperforms the two state-of-art vision-based approaches, ViDE and rExtractor. The experiments produce an average F-1 score of 86.68%, which is approximately 57% and 37% better than that of ViDE and rExtractor, respectively; and an average F-1 score of 86.21%, which is approximately 38% better than that of ViDE, for data record extraction and data item extraction, respectively.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. James Miller for his support and effort in guiding me through my graduate program. I would also like to thank my graduate course professors, including Petr Musilek, Denilson Barbosa, and Okan Bulut.

I would also like to thank the Libyan Ministry of Higher Education & Scientific Research for their financial support.

I would also like to express my profound appreciation to my husband for his unfailing support, and to my parents for taking care of my daughter and providing me with the continuous encouragement throughout my years of study and research. Most of all, I would like to thank Pinder Bains for her continued guidance during my academic endeavors. Thank you.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

DOM	- Document Object Model
XML	- eXtensible Markup Language
HTML	- Hyper Text Markup Language
CSV	- Comma-Separated Values
DBMS	- Database Management System
EST	- Extended Subtree
TBDW	- TestBed for information extraction from Deep Web
API	- Application Programming Interface
VIPS	- Visual Page Segmentation
GLM	- Gestalt Layer Merging
CSS	- Cascading Style Sheets
HD	- Hausdorff distance
NHD	- Normalized Hausdorff Distance
SSIM	- Structural Similarity Index
GUI	- Graphical User Interface
DRE	- Data Record Extraction
DIE	- Data Item Extraction

1. Introduction

Nowadays, the massive growth of the Internet requires a considerable amount of web data to be stored in huge networked databases. The prevalence of these databases online has significantly deepened the web and their stored data are referred to as the *hidden web* (or the *deep web*) (He, 2007).

In surface web, static Uniform Resource Locator (URL) links are used to access static web pages (Su, 2012). These pages are characterized by having a fixed content structure that does not change upon multiple user requests. On the other hand, dynamic URL links are used to invoke deep web pages. A dynamic URL is a combination of: the web page’s static URL, and user-selected parameters. Deep web pages are mostly accessible through the query interface of the underlying database (Saissi, 2014). A web database dynamically generates a deep web page when it receives a parameterized query from a user who submits a Hyper Text Markup Language (HTML) search form (Su, 2012); the retrieved data (i.e., query results) are assembled and displayed in the generated query result page. HTML tags are used to encode the retrieved information into semi-structured data objects called *data records* (Shi, 2015), each of which is a collection of data items (i.e., attributes) describing an entity such as a car or a book.

While billions of static web pages are linked in the surface web, the number of database-driven websites is rapidly increasing, and it is believed that the amount of available information in the hidden web is several hundred times larger than what users can access through the surface web (He, 2007). In addition, the structured databases that are contained in the deep web are three times more than the unstructured databases, and they are of greater interest to the database and data mining community (Saissi, 2014). These structured databases store relational records with attribute-value pairs. A high level of organization is presented in the structured databases which raises their significance and the demand for exploring the deep web (Liakos, 2016).

Identifying and extracting the retrieved data, which are wrapped in deep web pages as data records, is an essential task known as web information extraction system (Sleiman, 2013). Organizing the extracted data in a structured format, such as tables, XML, or CSV, is also a part of a web information extractor (Sleiman, 2013), where all data values of the same attribute are aligned together, such as arranging them in a single table column (Su, 2012).

Mining data records is fundamental for a wide range of applications, particularly in the business domain (Ferrara, 2014). Database building is one of the procedures that greatly benefit from web data extraction, where the extracted data are used to automatically populate the DBMS (Ferrara, 2014). Many applications in the web marketing sector, such as real estate and e-commerce companies, utilize web data extraction techniques in *competitive intelligence* (Ferrara, 2014). The extracted marketing information of competitors is used to improve decision-making activities of business managers.

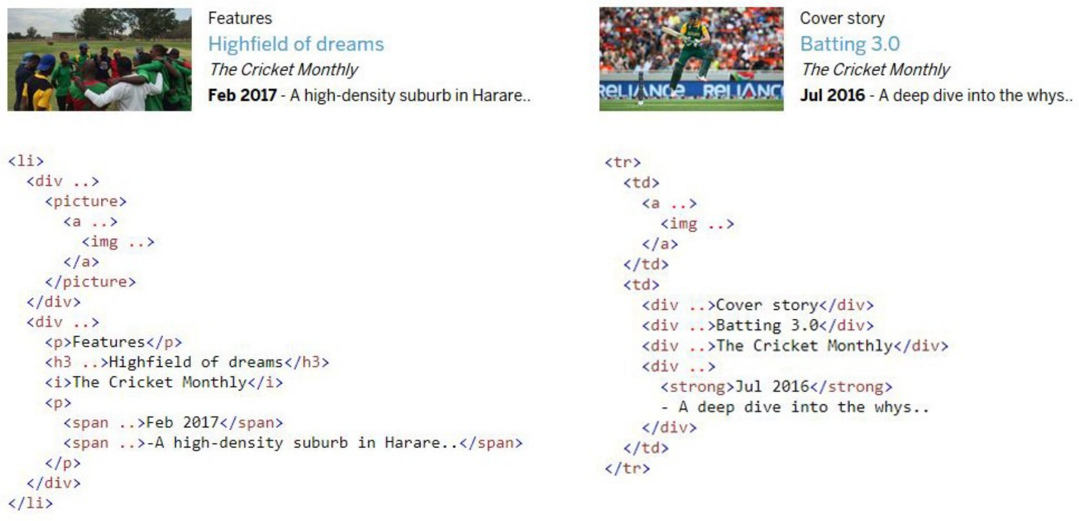
From a customer-centric perspective, web data extraction techniques can be applied in *Comparison shopping* (Su, 2012). A customer can compare features of similar items from different online stores.

Based on customer shopping activities, companies aim to enhance their marketing and sales capabilities via *context-aware advertising* (Ferrara, 2014). Web data extraction techniques identify relevant shopping data, from which useful shopping-related ads can be matched to customer preferences and placed on the web page. Other examples of web applications that take advantage of web data extraction are data integration, meta-querying (Su, 2012), news aggregation services, music and movie recommendations (Varlamov, 2016), mashup applications, user opinion mining, and citation database building (Ferrara, 2014).

Extracting data records from query result web pages has become a research area that attracts a lot of interest recently in which new methodologies are still emerging (Sleiman, 2013). The problem of extracting data records is challenging due to the existence of irrelevant information such as; navigation panels, menu buttons, search forms, advertisements, information about the website, etc. (Su, 2012). The automation degree of the extraction method is another challenge; manual and semi-automatic approaches that require human interference are labor-intensive and less effective than the automatic methods. Analyzing the structure of HTML web pages (i.e., DOM tree) is the main technique that most existing proposals depend on (Fan, 2014). Identifying the repeated patterns in a web page's DOM tree as data records is not sufficient, because the visual layout of the page and the corresponding DOM tree may have different representation hierarchies (Xu, 2016). This inconsistency is due to the usage of different runtime tools (e.g., JavaScript) that can dynamically modify the displayed web page (Anderson, 2013). In other words, DOM nodes adjacency does not necessarily indicate the adjacency of corresponding elements on the web page. Additionally, a web page's DOM tree may contain many invisible elements that users can't read, which make them useless (Xu, 2016). Directly accessing the tag tree to get the tag names of its nodes for the purpose of similarity calculation (Fan, 2014) or for obtaining the data types (Su, 2012) makes the solution fully dependent on a specific programming language (e.g., HTML). HTML-dependent solutions may no longer be functional with the continuous developing of the HTML language and the emergence of new tags (Liu, 2010). Moreover, segmenting the tag tree into similar data records based on the observation that the subtrees of these records share one parent node (Bing, 2011), fails when the retrieved data records are displayed on the web page as groups, where each group of records is encoded in a separate block.

As HTML is still evolving, superior approaches must be independent of using the HTML tag names to identify data types or to calculate the similarity between web page elements. Tag trees are utilized by web browsers to display web pages; visitors are only interested in seeing what is rendered by the browsers and they are not aware of the underlying HTML code. The same visual presentation can be expressed by two different pieces of HTML code (Alpuente, 2009). Consider the two web page fragments in Figure 1 (a) and Figure 1 (b). Although, the two fragments are visually similar, their corresponding HTML codes are very different.

Figure 1: Example of similar visual presentation of two different HTML codes



Features
Highfield of dreams
The Cricket Monthly
Feb 2017 - A high-density suburb in Harare..

```
<li>
  <div ..>
    <picture>
      <a ..>
        <img ..>
      </a>
    </picture>
  </div>
  <div ..>
    <p>Features</p>
    <h3 ..>Highfield of dreams</h3>
    <i>The Cricket Monthly</i>
    <p>
      <span ..>Feb 2017</span>
      <span ..>-A high-density suburb in Harare..</span>
    </p>
  </div>
</li>
```

(a)

Cover story
Batting 3.0
The Cricket Monthly
Jul 2016 - A deep dive into the whys..

```
<tr>
  <td>
    <a ..>
      <img ..>
    </a>
  </td>
  <td>
    <div ..>Cover story</div>
    <div ..>Batting 3.0</div>
    <div ..>The Cricket Monthly</div>
    <div ..>
      <strong>Jul 2016</strong>
      - A deep dive into the whys..
    </div>
  </td>
</tr>
```

(b)

The solution is to utilize the fact that a website's layout is built around the natural human tendencies of visual object perception. According to the Gestalt laws of grouping (Xu, 2016), human beings tend to perceive web page objects as a single group if they are: aligned (i.e., law of continuity), close to each other (i.e., law of proximity), or similar in their appearance (i.e., law of similarity). Thus, data records displayed on a query result web page should follow these rules to be perceived as one group. Furthermore, the main content of a typical web page is placed on the center of the page (Ahmadi, 2012); hence, the retrieved data records are centrally located since they are considered the essential subject of a query result web page.

Accordingly, we propose a matching algorithm that traverses a web page's tree searching for nodes that satisfy the above-mentioned specifications regarding: alignment, proximity, visual similarity, and central layout.

Instead of analyzing the DOM tree, our matching algorithm runs on a simpler and accurate representation of a web page's tree which is the visual semantic block tree that simulates human perception and whose hierarchy is identical to the page's visual layout (Xu, 2016). Also, we employ the extended sub tree similarity model (EST) (Shahbazi, 2014) to match the visual features of the subtrees of the data records blocks.

According to the mentioned principle of similarity, which suggests that similar elements tend to be grouped together (Xu, 2016); the data items of the same attribute (e.g., price) within different data records should have identical presentation style. So, we utilize this feature to align the data items of the extracted data records. Finally, the result is constructed in the form of a table holding all the extracted data records; whose each row contains all the data fields of a single record, and columns hold the semantically aligned data items.

In addition, we notice that the majority of the web data extraction proposals use the popular *TestBed for information extraction from Deep Web*[1] (TBDW) version 1.02 (Yamada, 2004) in their experimental work. TBDW is out-of-date because it has not been maintained since 2004. Hence, we propose a new testbed dataset for deep web data record extraction that surpasses TBDW in two aspects; (1) the categorization of the included websites, and (2) the number of the corresponding web pages.

The proposed system provides a novel programming-language-independent solution to the problem of automatic data records extraction and alignment. We outperform the state-of-art vision-based algorithms, ViDE (Liu, 2010) and rExtractor (Anderson, 2013) on all of the benchmark datasets available, and prove that our algorithm:

- Is HTML-independent, i.e., analyzing HTML tags is not required.
- Is able to identify data records arranged in one column (i.e., list view). It identifies 85% and 79% records in our dataset and TBDW, respectively; compared to

[1] http://daisen.cc.kyushu-u.ac.jp/TBDW/

rExtractor that identifies 63% and 38% records, and ViDE that identifies 38% and 21% records, respectively.

- Can identify data records arranged in multiple columns (i.e., grid view). It identifies 88% and 99% records in our dataset and TBDW, respectively; compared to rExtractor that identifies 50% and 97% records, and ViDE that identifies 23% records in our dataset and cannot identify any of the records arranged as a grid in TBDW.
- Has the ability to extract data records displayed as groups on a web page. It identifies 83% and 77% records in our dataset and TBDW, respectively; compared to rExtractor that identifies 82% and 11% records, and ViDE that identifies 68% and 10% records, respectively.
- Can obtain a higher F1-score for data record extraction than ViDE and rExtractor on the introduced dataset (89.43% compared to 31.99% and 55.41%, respectively); and on TBDW (83.92% compared to 27.57% and 44.55, respectively).
- Can achieve a higher F1-score for data item extraction than ViDE on the introduced dataset (89.70% compared to 54.99%); and on TBDW (82.72% compared to 42.07%).

The remainder of the thesis is organized as follows: Chapter 2 covers background concepts and relevant definitions; Chapter 3 categorizes the existing data record extraction techniques; Chapter 4 discusses the related vision-based extraction approaches; Chapter 5 demonstrates our solution for data record extraction and alignment; Chapter 6 conducts experiments on the proposed algorithm, evaluates the results of the Data Record Extraction (DRE) and Data Item Extraction (DIE), and compares their performance with ViDE and rExtractor; and finally, Chapter 7 draws a conclusion from the experimental results and provides pointers to potential future work.

2. Background

2.1 Tree Representation of Web Pages

Each web document has an embedded tree structure whose elements are described as parents, children, siblings, ancestors, and descendants. A web page’s tree representation is primarily revealed in its HTML tag tree. A tag tree is simply constructed by arranging the nested tag structure of the HTML code (Zhai, 2005). However, the HTML code may not be well-organized because of some missing and ill-formatted tags.

The Document Object Model (DOM) is an application programming interface (API) that specifies the logical tree structure for HTML, XHTML, and XML documents (Le Hégaret, 2000). Nodes of each document are represented as a group of objects organized in a tree structure called the DOM tree (Siddharthjn, 2017). The DOM tree hierarchy allows developers to navigate through a web document searching for particular information. Moreover, DOM tree objects are characterized by having properties and methods. A DOM property is a value that can be set, whereas a DOM method is an action that can be performed. DOM properties and methods establish a programming interface that allows scripting languages (e.g., JavaScript) to dynamically access documents and modify their structure, style, and content (Le Hégaret P. a., 2005). The processing results are reflected on the displayed web page.

However, the DOM tree may contain invisible elements that are not drawn by the web browser. These elements are considered as noise since they are not seen by the reader (Xu, 2016). Accordingly, the DOM tree cannot be considered the ideal representative of a web page despite the fact that it is a fast, traversable, and dynamic technique to access the visual and structural information of the web page.

To better represent a web page, an alternative data structure must be employed. The semantic block tree (Xu, 2016) solves the DOM hierarchy issues and groups nodes that are semantically related.

2.1.1 Block Tree Representation of Web Pages

Web page segmentation is an essential pre-processing task for web data mining applications; it aims to divide a web page into smaller segments. Some of these segments have immaterial content from a data mining perspective and must be disregarded. Other sections contain valuable information that can be utilized for different objectives by different data mining methods (Zeleny, 2017). The accuracy and performance of data mining can be improved when the web page is divided into distinct blocks (Xu, 2016).

Many proposals have been introduced to address the problem of Web page block identification. Some of them are DOM-based techniques that utilize DOM-level features with trained classifiers (Uzun, 2013). Although these techniques are generally fast, their accuracy relies on the employed heuristics since the DOM tree does not accurately reflect the visual features of the page elements (Zeleny, 2017). Other approaches are text-based which rely on textual content attributes (e.g., density) neglecting the page layout properties. However, the effectiveness of these approaches is very low when dealing with rich-format modern Web pages (Xu, 2016).

In addition to the DOM-based and text-based approaches, vision-based methods are proposed to consider the visual clues of rendered web page elements as they are perceived by human readers (Zeleny, 2017). They examine various visual attributes such as layout and location on the page, styling, images sizes, background colors, and font size and type (Eldirdiery, 2015). Computing these features of the displayed elements requires accessing the browser’s rendering engine. In spite of the fact that vision-based approaches are believed to be independent of the HTML-related heuristics or any implementation language specifics (Zeleny, 2017), some of these approaches are not purely vision-based as they use different combinations of code and visual features (Cormier, 2016).

Visual Page Segmentation (VIPS) algorithm (Cai, 2003) is one of the earliest and influential vision-based segmentation algorithms which have been used as a starting step by other projects, including template detection algorithms, web adaptation, and information extraction and retrieval. Some data records extraction proposals, such as ViDE (Liu, 2010), and VSDR (Li, 2007), are built on VIPS which is used to obtain a deep web page’s visual block tree. VIPS simulates users' visual perception of understanding web layout structure. It segments a web page into visual blocks by utilizing the page’s layout features, and detects visual separators among these blocks by following a repetitive top-down manner (Cai, 2003). The visual separators are the horizontal and vertical lines that do not intersect with the detected blocks on the page. Although VIPS has been a dominant work in this area, it still relies on limited segmenting heuristics that are manually inferred by the researchers who analyze the page’s rendering style (Xu, 2016). Additionally, VIPS mostly depends on identifying visual separators to extract web page’s blocks; these separators are much less apparent in modern web pages that have more advanced layout. Thus, VIPS performance efficiency on modern web pages is lower than traditional web pages.

To solve the issues of the previous methodologies, the *Gestalt Layer Merging (GLM)* model (Xu, 2016) was proposed. It simulates human perception by utilizing Gestalt laws of grouping to detect a web page’s visual semantic block tree, where each node (i.e., block) of the block tree represents a group of semantically related elements. The GLM model involves the following three main tasks:

1. *Layer tree constructor*: a web page's DOM tree is utilized as a prototype to construct its layer tree. The visual features of all the rendered DOM elements are retrieved. These features include both explicit CSS properties (e.g., text and background styles) and geometric attributes (i.e., location and size). Each node (i.e., layer) in the layer tree corresponds to a visible node in the DOM tree, and all the layer tree nodes follow the visual hierarchy of the web page layout; the hierarchy indicates the overlapping relationship of web page elements. Thus, the constructor includes two procedures:
   - Eliminating the invisible DOM elements: a DOM element is considered to be invisible if: (1) its area (i.e., height or width) equals 0; (2) its HTML tag is hidden; (3) it is empty (i.e., it has no displayed content); or (4) it is transparent (i.e., it is hidden by its visible child elements).
   - Adjusting the tree hierarchy: during building up the layer tree, each DOM element is initially added up to the tree with the same hierarchical position. Consequently, a modification may be applied according to the calculated geometrical features of the node. For instance, node *A* may be moved downward to become a child of node *B* if *A* is totally placed inside *B* on the page.
2. *Gestalt laws translator*: web page's tree structure details are simplified by applying Gestalt laws of grouping. These laws are interpreted into machine process-able rules to resemble the human manner of recognizing objects. Two of the interpreted Gestalt laws (i.e., law of simplicity and law of closure) are applied in the layer tree building step, while the other four are used for web page blocks identification.
   - *Gestalt law of simplicity*: this law states that people tend to perceive web page contents in their simplest form. Accordingly, the GLM model defines DOM elements to be the smallest entities and that each layer represents an entire DOM node without additional division.
   - *Gestalt law of closure*: this law indicates that an uncompleted web page object (i.e., an object that is partially hidden by the upper content) is seen as complete.

People are likely to ignore gaps and complete shape lines. Thus, each layer tree node is considered as a complete rectangle in the GLM model.

- *Gestalt law of proximity*: this law states that objects that are close to each other on a web page are likely to be perceived as one group, while distant objects are assigned into separate groups. Accordingly, the GLM model groups the adjacent elements on a web page in a single block if their proximities are the same, and places them in distinct blocks otherwise. To calculate the distance between two layers, the GLM model employs the Hausdorff distance (HD) with a normalization factor to remove any inconsistency generated by the size of the web page objects which are represented as rectangles. The calculation of the normalized Hausdorff distance (NHD) between two layers L_1 and L_2, is shown in Equation (1):

$$NHD(L1, L2) = \max\left\{\frac{hd_{1,2}}{Re_{L_1}}, \frac{hd_{2,1}}{Re_{L_2}}\right\} \qquad (1)$$

where $hd_{1,2}$ is the HD from L_1 to L_2; $hd_{2,1}$ is the HD from L_2 to L_1; $Re_{L_1}$and Re_{L_2} are the relevant lengths of L_1 and L_2, respectively; the relevant length of the layer can be either its height, width, or both (the diagonal length), depending on the relative location of the two layers.

The calculations of $hd_{1,2}$ and $hd_{2,1}$ are shown in Equation (2) and (3), respectively:

$$hd_{1,2} = \sup_{l_1 \in L_1} \inf_{l_2 \in L_2} \|l_1 - l_2\| \qquad (2)$$

$$hd_{2,1} = \sup_{l_2 \in L_2} \inf_{l_1 \in L_1} \|l_2 - l_1\| \qquad (3)$$

Where $\|l_1 - l_2\|$ and $\|l_2 - l_1\|$ calculate the Euclidean distance between two points l_1 in L_1 and l_2 in L_2; *sup* and *inf* are the supremum (i.e., maximum) and infimum (i.e., minimum) of a given set of distances, respectively.

- *Gestalt law of similarity*: this law reveals that objects that appear similarly on a web page are more likely to be organized together. Thus, the GLM model compares the similarity of layer tree nodes by evaluating their visual features which are represented by a series of CSS attributes. The three main aspects that GLM takes into account for comparing the similarity are: size similarity, foreground similarity, and background similarity.
  - For size similarity, GLM checks if two web page objects have the same width and height.
  - For foreground similarity, GLM examines the textual style attributes of the layer tree nodes. These attributes are directly obtained from the related CSS properties.
  - For background similarity, GLM inspects both the background color and background image. Color evaluation is carried out in CIE-Lab color space rather than RGB color space because CIE-Lab has a larger range that includes all RGB colors, and it is also designed to imitate human visual perception. Hence, GLM converts RGB colors that are retrieved from CSS into CIE-Lab colors, and ΔE_{00}^{12} is utilized as the color difference metric. The calculation of ΔE_{00}^{12} is shown in Equation (4) :

$$\Delta E_{00}^{12} = \sqrt{\left(\frac{\Delta L'}{k_L S_L}\right)^2 + \left(\frac{\Delta C'}{k_C S_C}\right)^2 + \left(\frac{\Delta H'}{k_H S_H}\right)^2 + R_T \left(\frac{\Delta C'}{k_C S_C}\right)\left(\frac{\Delta H'}{k_H S_H}\right)} \tag{4}$$

    The parameters of (4) are not listed in (Xu, 2016); they are explained in details in (Luo, 2001). The specified threshold for ΔE_{00}^{12} is set to 3.30 by GLM. Two colors are considered to be different if ΔE_{00}^{12} is greater than this value. On the other hand, GLM evaluates the similarity between two images using the Structural Similarity (SSIM) index which is designed to measure the perceptual similarity between two digital images based on the visible structures of the images. The calculation of SSIM is shown in Equation (5):

$$SSIM(I_1, I_2) = \frac{(2\mu_1\mu_2 + c)(2\sigma_{1,2} + d)}{({\mu_1}^2 + {\mu_2}^2 + c)({\sigma_1}^2 + {\sigma_2}^2 + d)} \qquad (5)$$

where I_1 and I_2 are the two images, each represented as an array of color pixels; μ_1 and μ_2 are the means of I_1 and I_2, respectively; ${\sigma_1}^2$ and ${\sigma_2}^2$ are the variances of I_1 and I_2, respectively; $\sigma_{1,2}$ is the covariance of I_1 and I_2; and, c and d are two factors to stabilize the division with weak denominator. The value of SSIM is 1.0 for two identical images, and 0 for two completely different images. GLM considers two images are the same if their SSIM is above the empirical threshold which is 0.48 (Xu, 2016).

- *Gestalt law of continuity*: this law expresses that elements on a web page are visually grouped by a human reader if they are aligned with each other. Accordingly, GLM groups continuous web page objects in a single block. Two objects are continuous on a web page if one of the four coordinates (i.e., left, top, right, and bottom) is the same in both objects.
- *Gestalt law of common fate*: this law indicates that web page elements that move together in the same direction are likely to be observed as related. Therefore, GLM places two web page objects in one block if their motion trends are the same. In particular, GLM checks the “static” trend attribute of the objects.

3. *Web page blocks identifier*: the translated Gestalt laws from the previous step are merged using a classifier. The trained classifier inspects the sibling layers of the constructed layer tree, applies the integrated laws, and assigns the layers into one of the two classes: “merge” or “not merge”. The merged layers are placed in one group which represents a semantic block. Ultimately, all the blocks are arranged hierarchically to create the web page’s block tree.

2.2 Tree Distance and Tree Similarity

Tree similarity (or distance) functions are applied to compare trees and obtain the degree of how similar (or dissimilar) the trees are (Shahbazi, 2014). The larger the similarity value, the more similar the two trees are. On the other hand, the larger the distance value, the less similar the two trees are.

For any rooted, ordered, and labeled tree T, let $|T|$ be the number of nodes of T, $V(T)$ the set of nodes of T, v_i the ith node of T (in post order formatting), and T_i the subtree of T whose root is v_i. For any two trees T^a, T^b, let $S(T^a, T^b)$ be the similarity between the trees, $S^*(T^a, T^b)$ the normalized similarity between the trees ranging between 0 and 1 (i.e., 1 means that the trees are identical; while 0 means that the trees are completely different), $D(T^a, T^b)$ the distance between the trees, and $D^*(T^a, T^b)$ the normalized distance between the trees ranging between 0 and 1 (i.e., 1 means that the trees are totally distinct; while 0 means that the trees are identical). As previously mentioned, tree similarity and tree distance are opposite of each other; thus, the relation between normalized similarity and normalized distance is shown in Equation (6) (Shahbazi, 2014):

$$S^*(T^a, T^b) = 1 - D^*(T^a, T^b) \qquad (6)$$

A tree with a single root node is called a rooted tree. Also, when the horizontal order between the siblings in a tree is essential, the tree is referred to as ordered. Finally, a tree is called a labeled tree if each node of the tree has a specified label.

The numerous applications of tree comparison lead to introducing a variety of approaches to address the problem of tree distance, where each approach has particular advantages and limitations in terms of the calculated distance score and the cost of calculation (Cohen, 2014). A recent proposed similarity function, called Extended Subtree (*EST*) (Shahbazi, 2014), introduces new rules for subtree mapping to overcome the issues of the previous methods.

2.2.1 Extended Subtree

Extended Subtree (*EST*) (Shahbazi, 2014) measures the distance/similarity between two trees. *EST* considers that subtrees are more significant than single nodes because similar subtrees of two trees have a larger weight in determining the similarity between the two trees in contrast with their disjoint mapped nodes. Moreover, a similar substructure between two trees can be recognized when there are identical mapped subtrees among the two trees, while it cannot be represented by the disjoint mapped nodes of the trees. This observation can be applied in web mining because the descendants of each web page block are arranged on top of the block. Thus, a user reads the subtree of each block instead of the details of the single block.

Mapping in *EST* is carried out according to a set of defined mapping rules:

1. Subtree mapping rule: both subtrees and single nodes can be mapped by *EST*.
2. One-time mapping rule: common subtrees of any two mapped subtrees are not allowed to be mapped together.
3. One-to-many mapping rule: a subtree of one tree is allowed to be mapped to more than a subtree of another tree.
4. Mapping weight assignment: the mean value of the two mapped subtrees' weights is considered as the weight of the mapping. For any two mapped subtrees T^{av} and T^{bv} in T^{a} and T^{b}, the weight of their mapping m_v is calculated as shown in Equation (7):

$$W(m_v) = \frac{W(T^{av}) + W(T^{bv})}{2} \qquad (7)$$

A subtree weight is the sum of its nodes' weights as shown in Equation (8):

$$W(T^{av}) = \sum_{t_i^{av} \in T^{av}} W(t_i^{av}) \qquad (8)$$

where a node's weight, $W(t_i^{av})$, is equal to one for the largest subtree that the node is part of, and zero otherwise.

Ultimately, *EST* utilizes the weights of all the potential valid mappings to calculate the similarity between T^a and T^b as shown in Equation (9):

$$S(T^a, T^b) = \sqrt[\alpha]{\sum_{m_v \in M} \beta_v \times \theta_v \times W(m_v)^\alpha} \qquad (9)$$

Where $W(m_v)$ is the weight of a mapping m_v. α, $\alpha \geq 1$, is a factor that magnifies the weight of large subtrees. More significance is put on larger subtrees as the value of α gets higher. β_v, $0 \leq \beta_v \leq 1$, is a factor that increases the weight of the same depth subtrees. As β_v gets lower, a less emphasis is placed on subtrees with different structural position (level). θ_v, $0 \leq \theta_v \leq 1$, is a factor that amplifies the weight of subtrees with smaller depth in their original trees. As θ_v grows larger, a more weight is assigned to subtrees closer to root nodes.

EST normalizes the calculated similarity as shown in Equation (10):

$$S^*(T^a, T^b) = \frac{S(T^a, T^b)}{Max(|T^a|, |T^b|)} \qquad (10)$$

Where $|T^a|$ and $|T^b|$ represent the number of nodes in T^a and T^b, respectively. The normalized similarity ranges from zero to one regardless of the size of the two trees. Specifically, one and zero represent two identical or two completely different trees, respectively.

2.3 Data Records Extraction and Alignment

The source of the greatest part of *deep web* data is the back-end structured databases which are referred to as *web databases* (Lu, 2013). By submitting HTML query forms, the hidden information stored in the web databases is accessible to users (Anderson, 2013). Web databases respond to users' queries and return the requested information organized in dynamically generated web pages called *query result web pages* (or *deep web pages*) (Liu, 2010).

The retrieved data (i.e., query results) are enwrapped in the query result pages as semi-structured data objects called *data records* (or *search result records*) (Shi, 2015). A data record is "an element of the class of data that is of interest for the user" (Varlamov, 2016) where each data record represents an object of a specific class (e.g., book or car). Data records displayed on a query result page usually belong to the same class since they are produced from the same query.

Every data record consists of a group of attributes which called *data items* (or *data units*) (Lu, 2013), where each data item describes a characteristic of the object it belongs to.

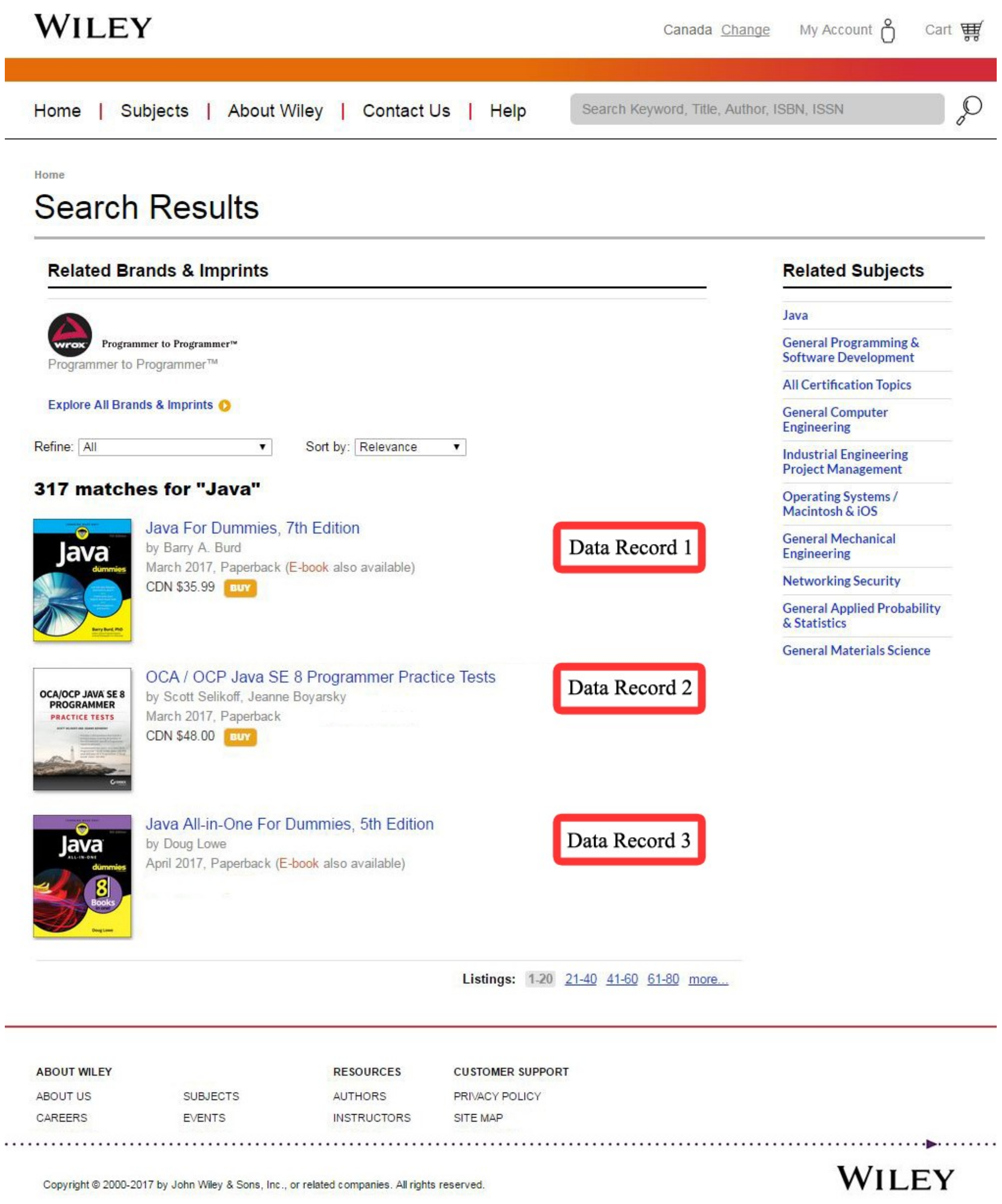
For example, Figure 2 shows a sample query result web page from Wiley.com (note that only the first three records from the results are shown as an example). On this page, multiple books are retrieved as a result of searching for *Java* books. The books are displayed in the form of data records where each displayed data record represents a book. A set of data items is included in each data record describing some attributes of the book that it corresponds to. As shown in the figure, these attributes include: *book title*, *authors*, *publication date*, *type*, *E-book availability,* and *price*.

However, some data attributes may not have corresponding values in a number of data records. These attributes are called optional attributes compared with the mandatory attributes that have corresponding values in all the data records; for instance, the *E-book availability* attribute, in the example shown in Figure 2, is an optional attribute.

In addition, data records are displayed on query result web pages in one of two forms (Liu, 2010):

1. List View: the data records are organized in one column as in the page fragment shown in Figure 3 (a). Although, they might have dissimilar width and height, they are left aligned. This observation is found to be true in 98% of the web pages of the two utilized datasets.
2. Grid View: the data records are organized in multiple columns as in the page fragment shown in Figure 3 (b). The data records in each column are considered to be a list which follows the same observation mentioned above.

Figure 2: Example of query result page from Wiley.com



WILEY

Canada Change My Account Cart

Home | Subjects | About Wiley | Contact Us | Help

Search Keyword, Title, Author, ISBN, ISSN

Home

Search Results

Related Brands & Imprints

Programmer to Programmer™

Programmer to Programmer™

Explore All Brands & Imprints

Refine: All

Sort by: Relevance

317 matches for "Java"

Java For Dummies, 7th Edition
by Barry A. Burd
March 2017, Paperback (E-book also available)
CDN $35.99 BUY

Data Record 1

OCA / OCP Java SE 8 Programmer Practice Tests
by Scott Selikoff, Jeanne Boyarsky
March 2017, Paperback
CDN $48.00 BUY

Data Record 2

Java All-in-One For Dummies, 5th Edition
by Doug Lowe
April 2017, Paperback (E-book also available)

Data Record 3

Listings: 1-20 21-40 41-60 61-80 more...

Related Subjects

- Java
- General Programming & Software Development
- All Certification Topics
- General Computer Engineering
- Industrial Engineering Project Management
- Operating Systems / Macintosh & iOS
- General Mechanical Engineering
- Networking Security
- General Applied Probability & Statistics
- General Materials Science

ABOUT WILEY
ABOUT US SUBJECTS
CAREERS EVENTS

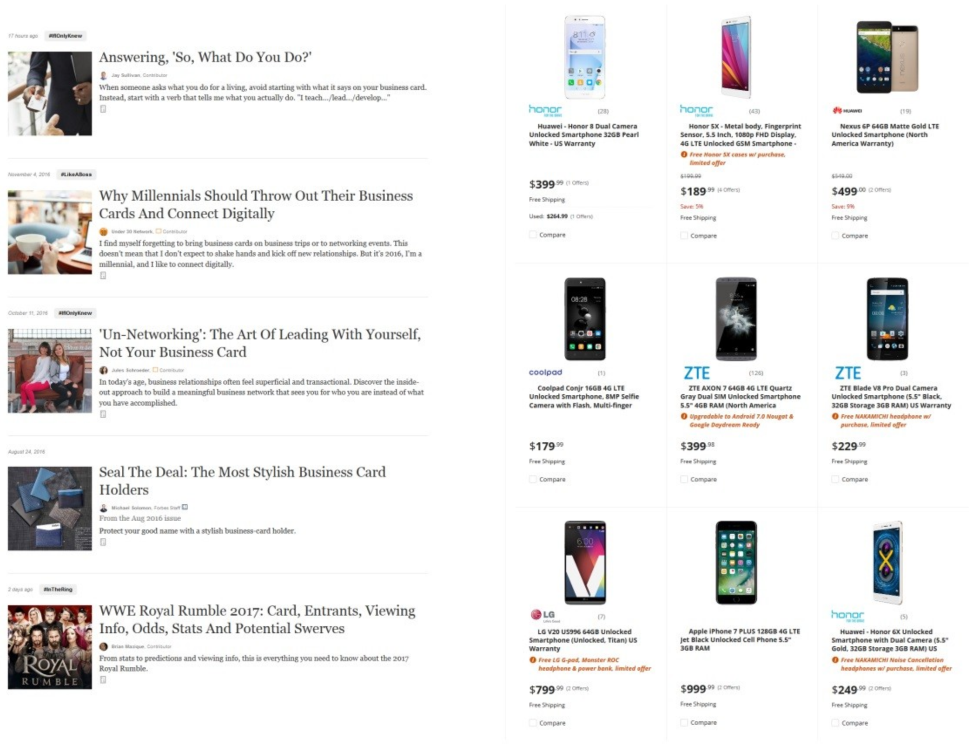
RESOURCES
AUTHORS
INSTRUCTORS

CUSTOMER SUPPORT
PRIVACY POLICY
SITE MAP

Copyright © 2000-2017 by John Wiley & Sons, Inc., or related companies. All rights reserved.

WILEY

Figure 3: Example of the two layout forms of data records on deep web pages



17 hours ago #IfIOnlyKnew

Answering, 'So, What Do You Do?'

Jay Sullivan, Contributor

When someone asks what you do for a living, avoid starting with what it says on your business card. Instead, start with a verb that tells me what you actually do. "I teach.../lead.../develop..."

November 4, 2016 #LikeABoss

Why Millennials Should Throw Out Their Business Cards And Connect Digitally

Under 30 Network, Contributor

I find myself forgetting to bring business cards on business trips or to networking events. This doesn't mean that I don't expect to shake hands and kick off new relationships. But it's 2016, I'm a millennial, and I like to connect digitally.

October 11, 2016 #IfIOnlyKnew

'Un-Networking': The Art Of Leading With Yourself, Not Your Business Card

Jules Schroeder, Contributor

In today's age, business relationships often feel superficial and transactional. Discover the inside-out approach to build a meaningful business network that sees you for who you are instead of what you have accomplished.

August 24, 2016

Seal The Deal: The Most Stylish Business Card Holders

Michael Solomon, Forbes Staff

From the Aug 2016 issue

Protect your good name with a stylish business-card holder.

2 days ago #InTheRing

WWE Royal Rumble 2017: Card, Entrants, Viewing Info, Odds, Stats And Potential Swerves

Brian Mazique, Contributor

From stats to predictions and viewing info, this is everything you need to know about the 2017 Royal Rumble.

honor (28)
Huawei - Honor 8 Dual Camera Unlocked Smartphone 32GB Pearl White - US Warranty
$399.99 (1 Offers)
Free Shipping
Used: $264.99 (1 Offers)
Compare

honor (43)
Honor 5X - Metal body, Fingerprint Sensor, 5.5 Inch, 1080p FHD Display, 4G LTE Unlocked GSM Smartphone -
Free Honor 5X cases w/ purchase, limited offer
~~$199.99~~
$189.99 (4 Offers)
Save: 5%
Free Shipping
Compare

HUAWEI (19)
Nexus 6P 64GB Matte Gold LTE Unlocked Smartphone (North America Warranty)
~~$549.00~~
$499.00 (2 Offers)
Save: 9%
Free Shipping
Compare

coolpad (1)
Coolpad Conjr 16GB 4G LTE Unlocked Smartphone, 8MP Selfie Camera with Flash, Multi-finger
$179.99
Free Shipping
Compare

ZTE (126)
ZTE AXON 7 64GB 4G LTE Quartz Gray Dual SIM Unlocked Smartphone 5.5" 4GB RAM (North America
Upgradable to Android 7.0 Nougat & Google Daydream Ready
$399.98
Free Shipping
Compare

ZTE (3)
ZTE Blade V8 Pro Dual Camera Unlocked Smartphone (5.5" Black, 32GB Storage 3GB RAM) US Warranty
Free NAKAMICHI headphone w/ purchase, limited offer
$229.99
Free Shipping
Compare

LG (7)
LG V20 US996 64GB Unlocked Smartphone (Unlocked, Titan) US Warranty
Free LG G-pad, Monster ROC headphone & power bank, limited offer
$799.99 (2 Offers)
Free Shipping
Compare

Apple iPhone 7 PLUS 128GB 4G LTE Jet Black Unlocked Cell Phone 5.5" 3GB RAM
$999.99 (2 Offers)
Free Shipping
Compare

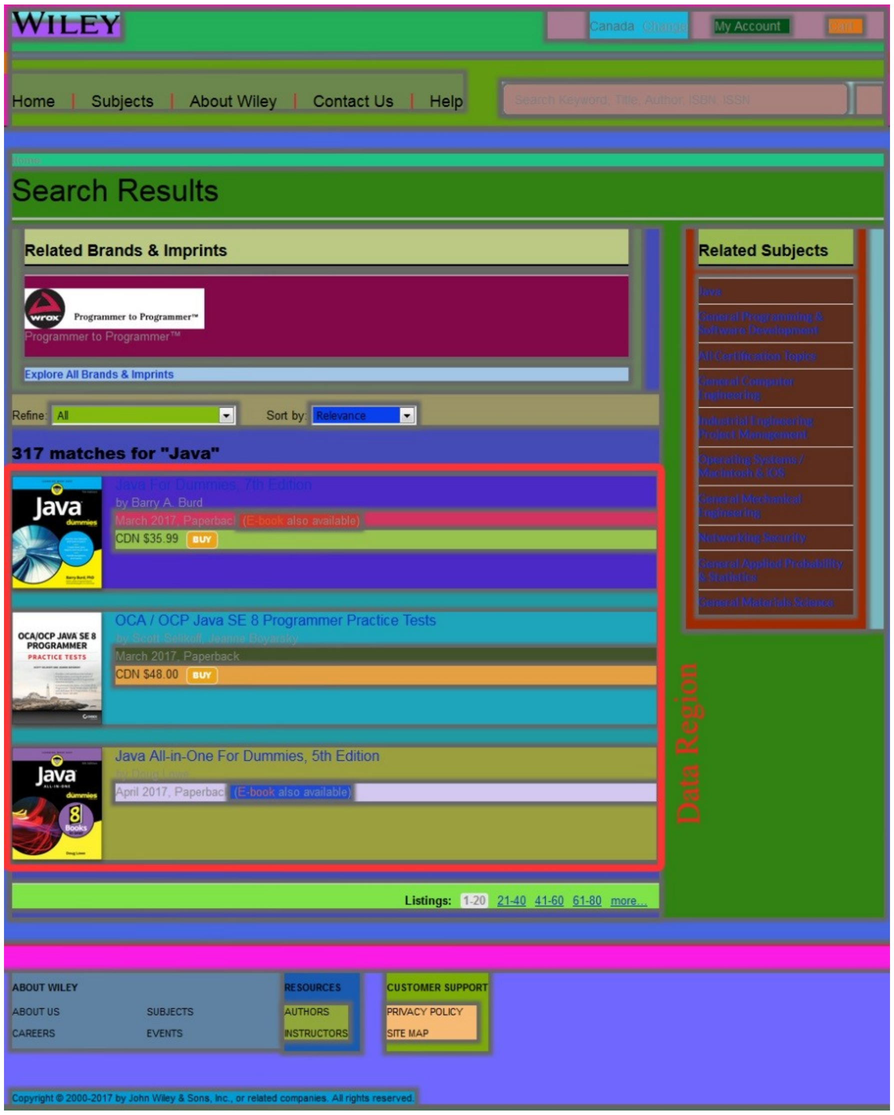
honor (5)
Huawei - Honor 6X Unlocked Smartphone with Dual Camera (5.5" Gold, 32GB Storage 3GB RAM) US
Free NAKAMICHI Noise Cancellation headphones w/ purchase, limited offer
$249.99 (2 Offers)
Free Shipping
Compare

(a) A page fragment showing data records arranged as a list (b) A page fragment showing data records arranged as a grid

In general, query result data records are arranged in one or more dynamically generated data regions (Shi, 2015). A data region is defined as the minimal area of the web page surrounding a set of retrieved data records (Varlamov, 2016). Based on the utilized approach, a data region can be a portion of the HTML code, one or multiple DOM subtrees, or a visual block. However, a search result page contains ancillary regions in addition to the data regions, including headers, footers, sidebars, and other navigation menus (Sleiman, 2013). These ancillary regions contain similar elements which are not data records. Differentiating between these similar elements and the actual data records is not straightforward for automatic single-page-based approaches (Shi, 2015).

For example, Figure 4 illustrates the visual blocks identified by the GLM model in the running example from Wiley.com. A total of 99 blocks have been identified. Only one block that contains the three retrieved data records is considered the data region, while the remaining blocks are either part of the records or other ancillary regions. Distinguishing between the regions of a web page is a primary procedure of a web information extraction system which becomes more complex with the dynamic layout of modern web pages (Varlamov, 2016).

Figure 4: Blocks identified by GLM in a query result page from Wiley.com


WILEY
Canada
My Account
Home | Subjects | About Wiley | Contact Us | Help
Search Results
Related Brands & Imprints
Programmer to Programmer™
Explore All Brands & Imprints
Refine: All
Sort by: Relevance
317 matches for "Java"
Java For Dummies, 7th Edition
by Barry A. Burd
March 2017, Paperback (E-book also available)
CDN $35.99 BUY
OCA / OCP Java SE 8 Programmer Practice Tests
March 2017, Paperback
CDN $48.00 BUY
Java All-in-One For Dummies, 5th Edition
April 2017, Paperback (E-book also available)
Related Subjects
Data Region
Listings: 1-20 21-40 41-60 61-80 more...
ABOUT WILEY
ABOUT US
CAREERS
SUBJECTS
EVENTS
RESOURCES
AUTHORS
INSTRUCTORS
CUSTOMER SUPPORT
PRIVACY POLICY
SITE MAP
Copyright © 2000-2017 by John Wiley & Sons, Inc., or related companies. All rights reserved.

The problem of automatically identifying, extracting, and aligning data records and data items they include, has received a lot of attention and many proposals have been introduced to address this problem (Sleiman, 2013). Two main steps are included in the data extraction system:

1. Data record detection: this step involves analyzing the tree representation of a query result web page searching for segments (i.e., DOM subtrees or visual blocks) that satisfy specific conditions based on the utilized approach. The identified data records are then pinpointed and saved for the next step.
2. Data items alignment: this step organizes the extracted data records in a structured format, such as table, XML, or CSV. Data items of all the identified data records are aligned by placing the data values of the same attribute in one group (e.g., arranging the data values of the price attribute in a single table column).

For example, the three books records of the previous example which are illustrated in Figure 2 are extracted and aligned as shown in Table 1.

Table 1: Extracted and aligned data from Wiley.com example

Java For Dummies, 7th Edition	Barry A. Burd	March 2017	Paperback	(E-book also available)	CDN $35.99
OCA / OCP Java SE 8 Programmer Practice Tests	Scott Selikoff, Jeanne Boyarsky	March 2017	Paperback	-	CDN $48.00
Java All-in-One For Dummies, 5th Edition	Doug Lowe	April 2017	Paperback	(E-book also available)	-

Other works propose approaches for a complementary phase, called *annotation* or *attribute labelling*. This phase seeks to assign a semantic label for each group (i.e., column) of data items of all the extracted data records (Lu, 2013). For instance, the semantic labels of the aligned data items from Wiley.com example are: *Book Title*, *Authors*, *Publication Date*, *Type*, *E-book availability*, and *Price*. These labels can be respectively employed as the column headings for Table 1. Indeed, assigning semantic labels for the aligned data items is usually carried out in a supervised manner, where a user has to provide some annotated examples (Murolo, 2016).

3. Categorization of Extraction Systems

The problem of extracting data records from deep web pages has been studied extensively; numerous approaches have been proposed to address the problem in the last two decades. Some approaches, including the algorithms proposed in (Shi, 2015), (Bing, 2011), (Anderson, 2013), (Thamviset, 2014), (Jimenez, 2015), (Bing L. a.-L., 2013), and (Pouramini, 2015), work on just identifying data records. Other approaches, such as the algorithms proposed in (Liu, 2010), (Murolo, 2016), (Grigalis, 2014), (Chang, 2016), (Fan, 2014), and (Su, 2012), work further to align the data items of the identified data records and present them to the user in an appropriate format.

Comparisons between the existing proposals are provided in a number of surveys that classify these proposals from different aspects (e.g., the main features used, the automation degree, the technique, and the extent of application) (Chang, 2016). Some of these surveys can be found in (Sleiman, 2013), (Ferrara, 2014), (Varlamov, 2016) (Devika, 2013). In the following, we present more recent examples for each categorization framework.

3.1 The Degree of Automation Criterion

The main aspect that has been used to classify web data extraction techniques is their degree of automation. Based on this criterion, the existing techniques can be mainly divided into two classes: (1) semi-automatic (i.e., supervised) approaches, and (2) fully-automatic (i.e., unsupervised) approaches.

The Semi-automatic approaches require human intervention to deduce the extraction rules. A user has to manually label data on some sample web pages which are utilized as training examples to learn the extraction rules (Grigalis, 2014). Some of these systems provide users with a Graphical User Interface (GUI) to facilitate the labelling procedure (Varlamov, 2016). For example, the DeepDesign system (Murolo, 2016) is a supervised method that provides users with a control panel, allowing them to annotate fields of example data records on a web page. The annotated fields are then used to learn rules for matching the other data records. The method proposed in (Jimenez, 2015) is also supervised; it requires a training set that consists of positive and negative examples annotated by the user. The system learns the extraction rules based on an open catalogue of features along with the features of the annotated examples. Although, some supervised learning approaches try to minimize the amount of human effort needed for the system to work, they still have limitations on extensive web applications. These approaches are time consuming, and their extraction rules have to be distinctively learned for each specific website (Grigalis, 2014).

The fully automatic approaches have been developed to overcome the above-mentioned limitations. The manual labour is entirely reduced in these systems, and they can be efficiently utilized in large-scale web applications (Shi, 2015). Examples of the unsupervised approaches are the methods proposed in (Grigalis, 2014), (Chang, 2016), (Su, 2012), (Thamviset, 2014), (Fan, 2014), (Shi, 2015), (Liu, 2010), and (Anderson, 2013), The input to these systems can be either a single web page or multiple web pages; hence, they are further classified based on their input. Indeed, the automatic approaches are the dominant techniques in the area of web data extraction (Varlamov, 2016).

3.2 The Level of Extraction Criterion

Based on the level at which the extraction process is carried out, the unsupervised systems are classified into two categories: (1) page-level (i.e., multi-pages based) approaches, and (2) record-level (i.e., single-page based) approaches.

The page-level approaches examine several search result web pages generated from the same template, and attempt to derive a shared pattern that can be used to extract the data records from these pages (Murolo, 2016). The UWIDE system (Chang, 2016) is an example of a multi-pages based approach. It identifies repeated leaf nodes patterns in all the input pages where every leaf node is encoded by its tag path and content type. A possible data region is detected by searching for the youngest common ancestor (YCA) of all the occurrences of a leaf node in a repeated pattern. The data records subtrees are then formed by clustering the subtrees under the YCA based on the edit distance of the leaf nodes' code strings, and assembling the subtrees from different clusters based on the order of these clusters in the input pages. Although, page-level approaches are capable of distinguishing between the template and the dynamically generated data, the major drawback of these approaches is that their input could be enormous for large-scale applications since they require multiple pages as input.

On the other hand, the record-level approaches inspect individual web pages separately in order to identify frequent patterns in each web page; the identified patterns are then utilized to extract data records from that page (Murolo, 2016). Examples of these methods are presented in (Grigalis, 2014), (Su, 2012), (Thamviset, 2014), (Fan, 2014), (Shi, 2015), (Liu, 2010), and (Anderson, 2013),

In general, the page-level and the record-level methods employ various features of the HTML source code, the DOM tree, or the block tree of a web page to identify repetitive patterns. Hence, they are further classified based on the utilized approach.

3.3 The Utilized Approach Criterion

Based on the applied technique and the utilized features, the automatic approaches are classified into five categories (Shi, 2015): (1) String-based approaches, (2) DOM-tree-based approaches, (3) Vision-based approaches, (4) Ontology-based approaches, and (5) Hybrid approaches.

A string-based approach works by mining repetitive patterns from constructed tag strings. The ClustVX algorithm (Grigalis, 2014) is an example of a string-based approach. It first translates the rendered HTML code into XML. The rendered visual features are embedded into the HTML elements as attributes. An XString is then created for each visible web page element; the XString is composed of the element XPath and its font features. All the indexes are removed from the XPaths and only the tag names are left. The ClustVX system clusters the created XStrings, so that the visually similar elements are grouped in one cluster. The data region is then identified by finding the longest common tag path of all the elements in each cluster, and the data records are then segmented based on two heuristics depending on whether each record has its own parent or all the records have the same parent. Ultimately, the ClustVX system assumes that the data items of the same semantics have the same tag path in all the data records. A key limitation of this approach is that it considers that all data records have the same number of data items neglecting the existence of optional attributes.

The DOM-tree-based methods have defined diverse similarity measurements that are applied on a web page’s DOM tree (Fang, 2017). A recent DOM-tree-based approach is the AutoRM algorithm (Shi, 2015). AutoRM first constructs the DOM tree of the web page. Starting from the root node of the DOM tree, the algorithm inspects each set of adjacent sibling nodes to obtain groups of similar elements; the trees rooted at these nodes are inspected to acquire their similarity. Tree similarity is determined based on the similarity of the trees’ leaf nodes which is in turn computed according to various features (i.e., structural, visual, content, and data type features). AutoRM merges the identified sets of similar nodes that have intersected index intervals to define a possible data region. Sets of adjacent similar Candidate Records (C-Records) are mined from each data region; a C-Record set is chosen if it covers a large area and has a high cohesion (i.e., a highly cohesive C-Record set has a high similarity between its C-Records and a low similarity between the nodes of each C-Record). Based on the existence of separator nodes between the C-Records, AutoRM proposes two different methods (i.e., separator-based and head-order-based algorithms) for C-Records mining. Ultimately, the system recursively mines actual data records from each set of identified similar C-Records. However, AutoRM is an HTML-dependent approach which mainly relies on comparing the HTML tag names of the nodes to judge their similarity. Another limitation of AutoRM that it always considers data records as adjacent sibling nodes. Thus, it fails when there are multiple groups of data records where each group has its own parent (19.60% of the data records in our testbed datasets are organized as groups).

As mentioned earlier, data records that are retrieved from the same query are similarly presented on the web page. Consequently, visual features of data records have been exploited by several extraction systems (Fang, 2017). The visual features have been utilized in the above referred techniques (i.e., ClustVX and AutoRM) as auxiliary information to identify repetitive patterns in a web page; however, the vision pattern is not the basic conception that these approaches are built upon. To the best of our knowledge, ViDE (Liu, 2010) and rExtractor (Anderson, 2013) are the most two recent vision-based techniques that are mainly built upon the visual presentation of web page objects. They will be discussed in details in the Related Work chapter. Indeed, our algorithm falls in this category; precisely, it is a purely-visual automatic single-page approach.

Some approaches employ the domain ontology as part of their extraction and alignment process; an automatically generated ontology or a predefined ontology of the domain of interest is used to extract data from the same domain web pages (Bing L. a.-L., 2013). For instance, WordNet[2], which is a lexical database for the English language, has been used in (Hong, 2011) as a predefined ontology to check the similarity of data records. Another recent ontology-based approach is the OntoExtract system (Hong, 2013). OntoExtract first inspects the web page's DOM tree searching for specific Block Tags (e.g., Table and Div). A detected Block Tag is then considered a possible data region if its area is above a specified threshold. Three ontological tools (WordNet, CYC, and Wikitology) are then used to select the relevant region (i.e., the region which contains the search results) among all the identified ones. WordNet matches the keywords within a region according to their semantic similarity. Then, CYC forms relations between these keywords. Finally, Wikitology determines how much the provided information is related to each other. A data region is selected as the relevant region if its semantic similarity is above a predefined threshold. However, requiring the availability of additional sources to access these ontologies is a limitation of the ontology-based approaches (Su, 2012). Also, the extraction process mainly relies on the completeness and accuracy of the utilized ontology database. Indeed, integrating many ontological tools as it has been introduced in the OntoExtract system leads to a slow speed performance (Hong, 2013).

Lastly, the hybrid approaches combine the HTML tag features and the visual features to build a data record extractor (Fang, 2017). An example of these approaches is the algorithm presented in (Fan, 2014). The algorithm accesses the rendering engine and considers the production of the web page's tag tree and the CSS attributes as the web page's block tree. The data region block is first identified based on the proportion of its area to the area of the whole page. All the child blocks of the data region are then clustered using *Jaccard* similarity where two blocks are considered to be similar only if their tags and all of their visual properties are the same. The algorithm then regroups the blocks from different clusters to form data records. The regrouping step is carried out

[2] https://wordnet.princeton.edu/

based on the assumption that the first mandatory block of each data record is contained in the cluster that has the maximum number of blocks. Ultimately, the data item alignment is performed using simple tree matching and partial tree alignment. The nodes of each record's tree are mapped against the nodes of the seed tree (i.e., a seed tree is the tree of the record that has the maximum number of data items). However, the algorithm considers that the blocks of each cluster always belong to distinct data records; this leads to incorrect regrouping when the data records themselves consist of similar sub-blocks which will be clustered together. Indeed, 43% of the data records in our testbed datasets contain visually similar sub-blocks.

4. Related Work

The ViDE algorithm (Liu, 2010) is built upon the assumption that all the retrieved data records are contained in a unique data region, and that all the data records are left-aligned, contiguous, consistently separated, and visually similar. It first employs the VIPS algorithm (Cai, 2003) to create the web page's visual block tree. It then locates the data region by searching for the block that is horizontally centered and relatively large compared with the size of the whole page. ViDE defines noise blocks (e.g., statistical information) as the blocks that are located at the top or bottom of the selected data region and are not aligned to the left. The noise blocks have to be discarded from the data region. Each of the remaining blocks in the data region is first classified into one of three data types (i.e., image, plain text, and link text), ViDE then uses the size of the images and the shared fonts of the plain texts and link texts to cluster the remaining blocks based on their appearance similarity. A bounding rectangle is created for each cluster to surround all the blocks in the cluster; the positional arrangements of these rectangles are then utilized to regroup the blocks and form data records. The cluster that has the maximum number of blocks is selected as the base of the regrouping step; the blocks in the maximum cluster are considered to be the initial data records for the regrouping. The initial data records are grown with the blocks from the other clusters whose bounding rectangles overlap the bounding rectangle of the maximum cluster. ViDE then segments each data record into data items by finding the sequence of the leaf nodes that each record subtree is composed of; the left-to-right order of the data items in each record is preserved. ViDE assumes that data items of the same semantic in different data records are presented similarly. Thus, ViDE matches the data items from different data records based on their font and position attributes. Ultimately, a multi-alignment algorithm is carried out to align all the data items according to their order in their corresponding data records; the data items of the same semantic are arranged under a single column and the optional data items in some records are assigned predefined blank items. However, ViDE takes into account data records that are arranged in one column (i.e., list view) only, and disregards data records that are organized in multiple columns (i.e., grid view) on the web page. Furthermore, when ViDE fails to locate a data region block among all the blocks generated by VIPS, then there is no data record that can be extracted; in fact, ViDE fails to identify data regions in 18% of the web pages in our testbed datasets. Also, ViDE considers that the similar sub-blocks in a data region are parts of different data records; however, it may cluster sub-blocks that belong to a single record which then leads to an incorrect regrouping. In spite of the fact that ViDE mainly relies on the visual features of the web page elements, a portion of it is still HTML-dependent; ViDE inspects the HTML elements' tag names to determine their data types (e.g., <*img*> tags for images). Lastly, the assumption that the noise blocks are not aligned to the left as the other data records fails in a large number of web pages (the noise blocks in 62% of the web pages in our testbed datasets are aligned similarly with the data records on the page).

The rExtractor system (Anderson, 2013) is built on the human intuition of understanding the visual presentation of objects on query result web pages. A human reader utilizes the visual regularity of the displayed data records to derive semantic relations among the data items that can be grouped into data records. rExtractor accesses the layout engine to obtain the web page's visual block tree which is defined as the outcome of the tag tree and the CSS of the page. The visual block tree consists of two types of visual blocks (i.e., container blocks and basic blocks). The rExtractor algorithm assumes that the highest priority region (i.e., the region that contains the main content) of a web page is located between the center and the top left of the page. Thus, the algorithm employs a clockwise *Ulam Spiral*[3] to select a seed block which is a single basic block that belongs to one of the data records. The spiral starts from the center of the page and proceeds towards the top left corner to scan the largest possible area of the page's main region. The container blocks that contain the seed block are considered candidate record blocks. Hence, rExtractor selects all the container blocks on the page whose widths are the same as the widths of the candidate record blocks. The selected container blocks are then clustered based on their widths. The algorithm then groups the child blocks of each container block where each group contains visually similar child blocks (i.e., two child blocks are visually similar if all of their visual properties are the same). The content similarity between the candidate record blocks in each cluster is then calculated (i.e., container blocks have similar content if the similarity index between their corresponding multi-sets of child blocks is above a predefined threshold). Ultimately, the candidate record block that has content similarity to the maximum number of container blocks is selected to represent the actual container blocks of all the data records. However, rExtractor assumes that all the data record blocks on a query result page have similar width. By investigating the web pages in our testbed datasets, we found that almost 21% of the pages contain data records with dissimilar widths. Moreover, rExtractor only identifies data records without proceeding to the next process which is extracting and aligning their data items.

[3] http://mathworld.wolfram.com/PrimeSpiral.html

As HTML is still evolving, superior approaches must be independent of using the HTML tag names to identify data types. Tag trees are used by web browsers to display web pages; visitors are only interested in seeing what is rendered by the browsers. A superior approach must also take into account the structural inconsistency that may occur between the web page's underlying tag tree and the dynamically displayed elements on the page. Furthermore, web sites' designers try to improve their users' experience by imitating human mechanisms of perceiving objects. Utilizing this feature can be helpful in identifying sections that contain information of interest on web pages.

The proposed algorithm in this thesis is a purely visual approach. The none-visual approaches are beyond the scope of the thesis. Therefore, the extraction efficiency of the proposed algorithm is compared against the efficiency of the two current state-of-the-art vision-based systems (i.e., ViDE and rExtractor). Indeed, the whole extraction system consists of two procedures (i.e., data records detection and data items alignment). ViDE is composed of two components (i.e., Vision-based Data Record extractor (ViDRE) and Vision-based Data Item extractor (ViDIE)) to perform each procedure. Thus, the comparison against ViDE is carried out for each of the two components, separately. On the other hand, the rExtractor system implements the first step only (i.e., data records detection). Hence, the comparison against rExtractor is carried out just for the data records detection step.

5. The proposed Extraction and Alignment System

In the following, we demonstrate the proposed strategy for data record extraction and alignment. A prerequisite procedure (i.e., obtaining the web page's block tree) and a definition (i.e., visual similarity of the semantic blocks) are first introduced.

5.1 Acquiring the Web Page's Block Tree

Our extraction system is a single-page-based approach; the system processes the displayed web page which is of interest to the user. Specifically, the input to the system is the block tree of the target web page. We utilize the GLM (Xu, 2016) model to obtain the web page's block tree. In the following, we summarize the constructing procedure as the details can be found in 2.1.1 and (Xu, 2016). We also show some statistical information regarding to the utilization of block trees.

The web page's layer tree is utilized as a prototype to construct its block tree. To obtain the layer tree of the web page, the visible DOM nodes are first extracted into layer nodes, and the hierarchical inconsistency that may exist in the DOM tree is fixed while building the layer tree with nodes.

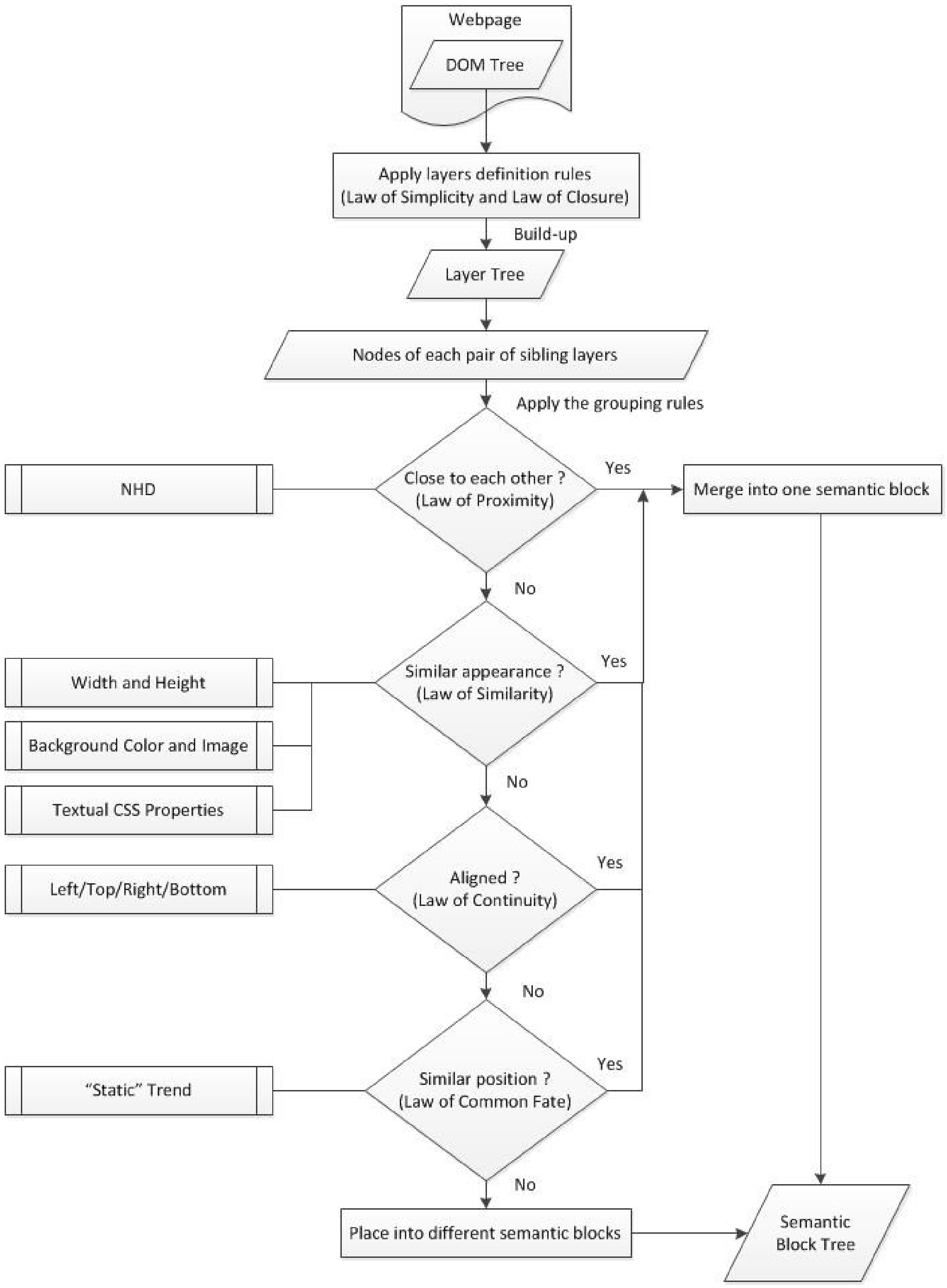
Each of the six Gestalt laws of grouping is translated into a machine process-able rule. The Gestalt laws of simplicity and closure are translated into layer definition rules and are taken into account while constructing the layer tree. The other four Gestalt laws (i.e., proximity, similarity, continuity, and common fate) are translated into evaluation metrics to assess the layer tree nodes and merge them into semantic blocks.

The semantic blocks identifier traverses the web page's layer tree starting from the root node, and applies the four Gestalt laws of grouping on each sibling layer. Instead of combining the four metrics into a unified rule as it is implemented in (Xu, 2016), each metric is applied separately on each pair of sibling layers. The two layers are merged into one semantic block if one of the following conditions is met:

1. The proximity of the two sibling layers is the same as the proximity of all pairs in the same series of sibling layers (law of proximity). The proximity between two layers is calculated using the Normalized Hausdorff Distance (NHD) shown in Equation (1).
2. The two sibling layers have similar appearance in terms of the size (i.e., width and height), background content (i.e., background color and background image), and foreground content (i.e., textual styles) (law of similarity). Color similarity is calculated using Equation (4), while image similarity is calculated using the Structural Similarity (SSIM) index shown in Equation (5). The other visual features are assessed by directly comparing the related text and paragraph CSS properties.
3. The two sibling layers are left, top, right, or bottom aligned (law of continuity).
4. The two sibling layers share the same position property (law of common fate).

If none of the four previous conditions is met, the two layers are placed in distinct semantic blocks. The entire block tree building-up procedure is illustrated in Figure 5.

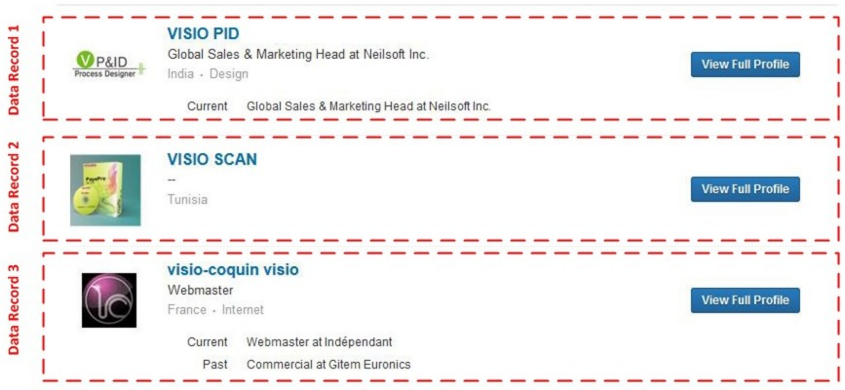
Figure 5: Working Procedure of Block Tree Building-up


Webpage
DOM Tree
Apply layers definition rules (Law of Simplicity and Law of Closure)
Build-up
Layer Tree
Nodes of each pair of sibling layers
Apply the grouping rules
NHD
Close to each other ? (Law of Proximity)
Yes
Merge into one semantic block
No
Width and Height
Background Color and Image
Textual CSS Properties
Similar appearance ? (Law of Similarity)
Yes
No
Left/Top/Right/Bottom
Aligned ? (Law of Continuity)
Yes
No
"Static" Trend
Similar position ? (Law of Common Fate)
Yes
No
Place into different semantic blocks
Semantic Block Tree

In addition to fixing the visual and hierarchical inconsistency, utilizing the semantic block trees to represent search result web pages in the extraction process is important for the two following main reasons:

1. The displayed data records are more accessible and comparable when each single data record is represented by a distinct subtree (i.e., all the subtrees that constitute the data record are arranged under a separate root node). For example, Figure 6 (b) shows the DOM tree representation of the HTML page fragment shown in Figure 6 (a); the subtrees of the displayed data records are not well-separated (i.e., the data records do not have discrete head nodes).

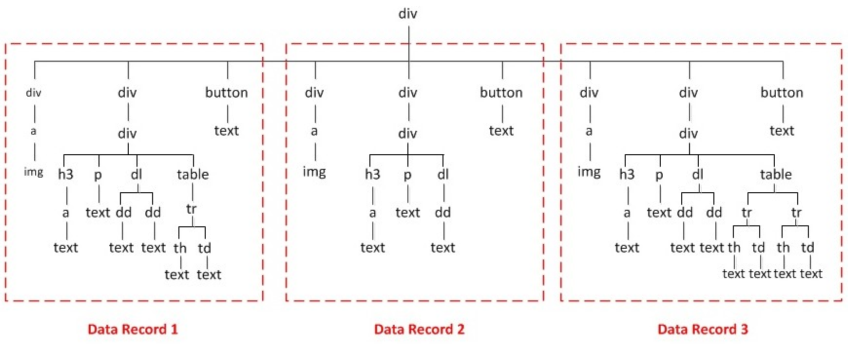
Figure 6: Example of data record trees that are not well-separated


Data Record 1
VISIO PID
Global Sales & Marketing Head at Neilsoft Inc.
India · Design
Current Global Sales & Marketing Head at Neilsoft Inc.
View Full Profile

Data Record 2
VISIO SCAN
--
Tunisia
View Full Profile

Data Record 3
visio-coquin visio
Webmaster
France · Internet
Current Webmaster at Indépendant
Past Commercial at Gitem Euronics
View Full Profile

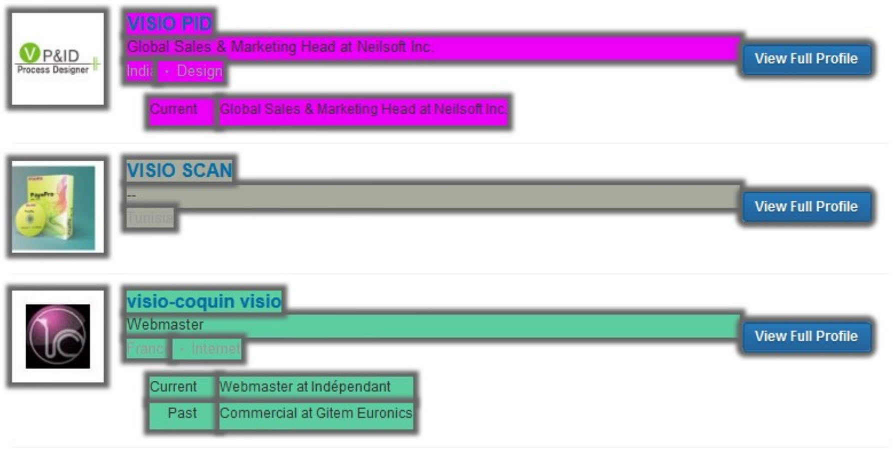
(a) HTML page fragment showing three data records



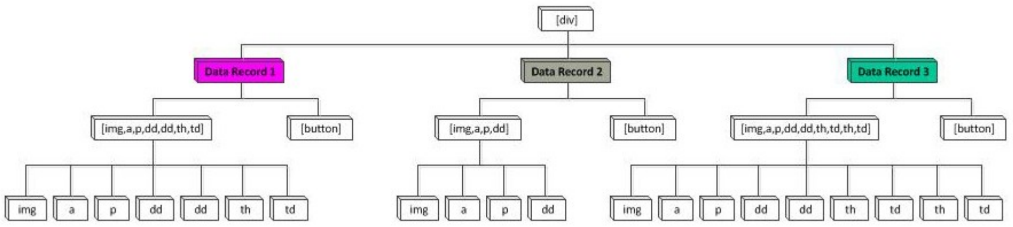
(b) DOM tree representation of the three data records

In general, an excellent web page layout simulates the human natural way of observing objects. Humans observe objects that are close to each other as one group, while distant objects are perceived as different groups (i.e., law of proximity). Thus, designers tend to leave an obvious space between every two successive data records and smaller spaces among the data items that constitute each individual record. Therefore, GLM fixes the issue of the underlying tree structure that exist in (some) web pages by grouping all the sub-parts of a data record under a distinct block. Figure 7 shows the result of applying the GLM model to get the semantic blocks of the previous example shown in Figure 6.

Figure 7: Semantic blocks of the three data records from the previous example



(a) Illustration of the semantic blocks of the three data records



(a) Block tree representation of the three data records

By analyzing the web pages in our testbed datasets, we found that the GLM model successfully identifies a single semantic block for each data record in 86% of the total data records.

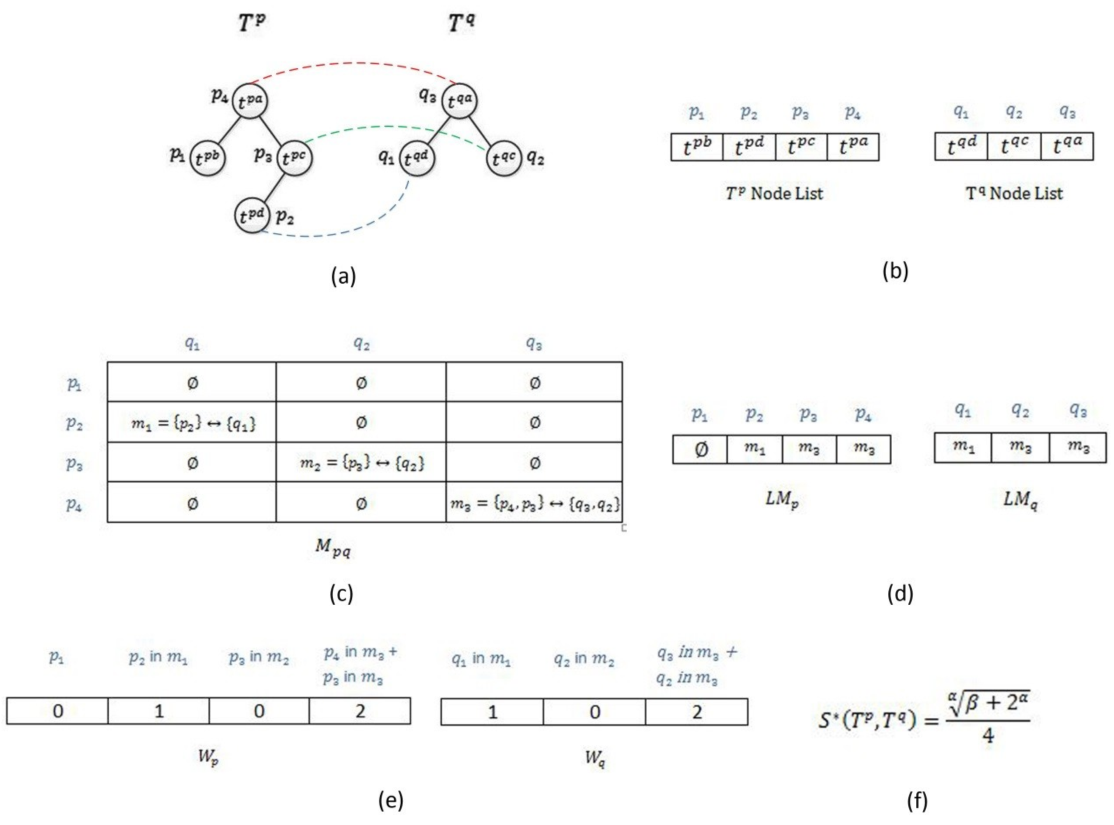
2. The semantic block tree of a web page is a simplified representation of the web page. The DOM tree may contain a large number of invisible nodes which makes traversing web page a complex and a time-consuming task. These invisible nodes are discarded in the block tree. Moreover, the semantic block tree groups the semantically related elements; thus, the number of nodes in a web page's tree is considerably reduced. In the previous example, the number of nodes in the DOM tree (Figure 6 (b)) is minimized by almost 40% in the semantic block tree (Figure 7 (b)). By inspecting the web pages in our testbed datasets, we found that the number of nodes in the DOM trees is minimized by an average of 79% in the semantic block trees of the web pages.

5.2 Defining Visual Similarity of Two Blocks

Each block of a web page's block tree, except the leaf blocks, comprises a tree of sub-blocks. Hence, calculating the visual similarity among the blocks is carried out using a tree similarity function. We employ the EST (Shahbazi, 2014) model to obtain the similarity of a pair of blocks. The implementation details of EST can be found in 2.2.1 and (Shahbazi, 2014).

EST determines the sub tree mapping relationship first before assigning weights to each mapping and a mapping is created between two nodes (i.e., sub-blocks) if they are similar. Figure 8 demonstrates an example of the similarity estimation between two trees T^p and T^q using the EST method:

Figure 8: Example of the similarity estimation using the EST method


T^p
T^q
p_4 t^{pa}
p_1 t^{pb}
p_3 t^{pc}
t^{pd} p_2
q_3 t^{qa}
q_1 t^{qd}
t^{qc} q_2
(a)
p_1 p_2 p_3 p_4
t^{pb} t^{pd} t^{pc} t^{pa}
T^p Node List
q_1 q_2 q_3
t^{qd} t^{qc} t^{qa}
T^q Node List
(b)
$m_1 = \{p_2\} \leftrightarrow \{q_1\}$
$m_2 = \{p_3\} \leftrightarrow \{q_2\}$
$m_3 = \{p_4, p_3\} \leftrightarrow \{q_3, q_2\}$
Ø
M_{pq}
(c)
Ø m_1 m_3 m_3
LM_p
m_1 m_3 m_3
LM_q
(d)
p_1 | p_2 in m_1 | p_3 in m_2 | p_4 in m_3 + p_3 in m_3
0 1 0 2
W_p
q_1 in m_1 | q_2 in m_2 | q_3 in m_3 + q_2 in m_3
1 0 2
W_q
(e)
$S^*(T^p, T^q) = \frac{\sqrt[\alpha]{\beta + 2^\alpha}}{4}$
(f)

- The nodes of the two trees T^p and T^q are shown in Figure 8(a) stored and numbered in a post-order formatting as shown in Figure 8(b). The nodes t^{pb}, t^{pd}, t^{pc}, t^{pa} in tree T^p are denoted as p_1, p_2, p_3 and p_4; and the nodes t^{qd}, t^{qc}, t^{qa} in tree T^q are denoted as q_1, q_2, and q_3, respectively.
- All the possible mapping relationships between the two trees are first identified before further analysis. The following three mappings are first located: $m_1 = \{p_2\} \leftrightarrow \{q_1\}$ (marked by the blue dotted curve), $m_2 = \{p_3\} \leftrightarrow \{q_2\}$ (marked by the green dotted curve), $m_3 = \{p_4\} \leftrightarrow \{q_3\}$ (marked by the red dotted curve).
- Because subtrees are considered more important than single nodes, subtrees are mapped together instead of single nodes by EST. Thus, m_3 is updated to map the two subtrees of a-c as follows: $m_3 = \{p_4, p_3\} \leftrightarrow \{q_3, q_2\}$.

- The mapping results are saved in the $n \times m$ mapping matrix M_{pq} shown in Figure 8(c), where $n = |T^p|$ and $m = |T^q|$. Each cell in the matrix stores the mapping relationship between the corresponding nodes. For instance, the cell in 2nd row and 1st column of M_{pq} stores the mapping m_1 which is between the 2nd and 1st nodes of T^p and T^q, respectively. If there is no mapping between nodes, then an empty set is stored in the corresponding cell.
- To construct the largest subtree mapping, EST applies the one-time mapping rule by avoiding mapping common subtrees of T^p and T^q. Figure 8(d) shows the constructed largest mappings, LM_p and LM_q, for both trees, T^p and T^q, respectively. Of all the mapped subtrees, p_1 does not belong to any mapping relationship, so the first element of LM_p stores an empty set. p_2 and q_1 only belong to m_1, so this mapping relationship is stored in the 2nd and 1st elements of LM_p and LM_q, respectively. Ultimately, both p_3 and p_4 in T^p (also q_2 and q_3 in T^q) belong to m_3, and thus, the 3rd and 4th elements of LM_p (and the 2nd and 3rd elements of LM_q) store this mapping relationship.
- Each node in T^p and T^q is assigned a weight by comparing the mapping matrix M_{pq} with both largest mappings LM_p and LM_q. LM_p has 1 of m_1, 0 of m_2, and 2 of m_3. Hence, the weight matrix W_p has the weights of $\{p_2\}$, $\{p_3\}$, and $\{p_4, p_3\}$ as 1, 0, and 2, respectively. Similarly, the weight matrix W_q has the weights of $\{q_1\}$, $\{q_2\}$, and $\{q_3, q_2\}$ as 1, 0, and 2, respectively. Therefore, the mapping weights can be computed as $W(m_1) = 1$, $W(m_2) = 0$, and $W(m_3) = 2$.
- Following the recommendation in (Shahbazi, 2014), α and β are set to 2 and 0.5, respectively. The depth of the nodes t^{pd} and t^{qd} in T^p and T^q are not the same; thus, β is utilized to reduce the weight of m_1, The calculation of the similarity between T^p and T^q is shown in Figure 8(f) which equals to 0.53.

In this thesis, the similarity is assessed based on the visual properties of the two nodes. We employ 53 visual features which can be directly retrieved from each obtained semantic block. Some of these features are related to the textual content of a block, while the remaining features are general aspects of the block. Table 2 shows the utilized visual properties.

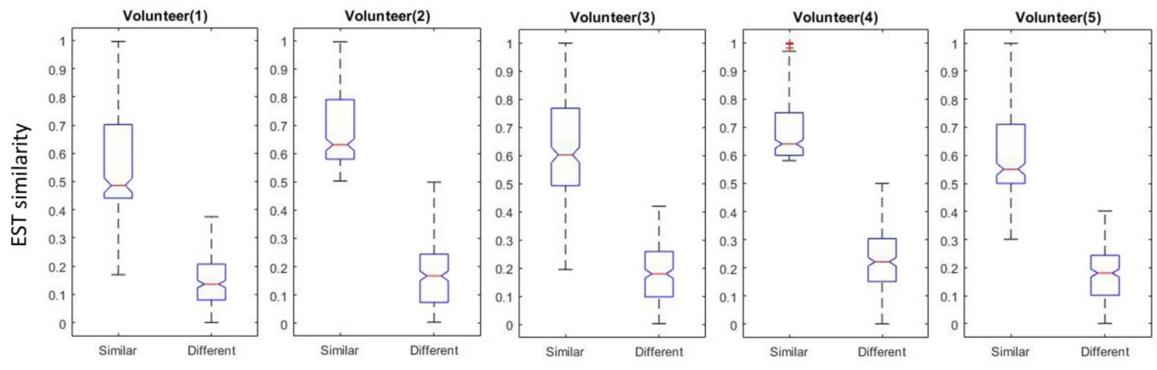
If the values of all of these properties are identical in both of the comparable sub-blocks, then the sub-blocks are considered to be visually similar and a mapping is created between them during the EST similarity calculation process. The weight of each mapping is calculated using Equation (7) and the similarity of the two block trees is calculated using Equation (9). Ultimately, Equation (10) is employed to obtain the normalized similarity (i.e., ranging from zero to one) between the two block trees.

Table 2: The utilized properties to assess the visual similarity

General Properties	Textual Properties	List Properties
background color and image	text (writing) direction	list style-image
left/top/right/bottom borders color	letter spacing	list style- position
left/top/right/bottom borders style	line height	list style-type
left/top/right/bottom borders width	text alignment	
outline color, style, and width	text decoration	
bottom-left/ bottom-right/ top-left/ top-right radius of the border	text indentation	
box shadow	text capitalization	
vertical alignment	white-space and word-spacing	
position	text overflow	
	text shadow	
	Line-breaking and word-wrapping	
	count, gap, and width of text columns	
	rule-color, rule-style, and rule-width of text columns	
	font family, size, weight, style, variant, and color	

The following experiments are conducted in order to evaluate the validity of the EST metric and to find out the optimal similarity threshold. We asked 5 volunteers to manually classify a number of web page portions according to their visual appearance. A total of 2500 different web pages are randomly retrieved from the world's 500 top Websites as defined from statistics produced by Alexa[4], and a pair of blocks is randomly selected from each web page. Every volunteer is given 500 web pages and asked to compare the pair of blocks of each page to assess whether they are similar or different. 500 EST records are produced for the blocks compared by each volunteer. As shown in Figure 9, these records are displayed in two boxplots, where the EST values of visually similar block pairs are illustrated by the first plot, while the EST values of visually different block pairs are denoted by the second plot.

Figure 9: Boxplots of EST values of two groups of similar and different blocks



According to the above displayed boxplots, the boxplots of the first group (i.e., the similar block pairs) demonstrate larger EST values than of the second group (i.e., the different block pairs) for all the 5 volunteers. The Wilcoxon rank sum test is conducted to analyze the EST records and verify the quantitative difference of the EST values between the two groups; the test is conducted with a Significance level of 5% (i.e., alpha=0.05). The null hypothesis of the test states that there is no difference between the ranks of the groups

[4] https://www.alexa.com/topsites. The top sites were retrieved on February, 2017

and that they have equal medians. According to the results in Table 3, all the p-values of the 5 experiments are smaller than 0.05 which indicates that the test rejects the null hypothesis and concludes that there is difference between the two groups of EST records.

Table 3: Results of Wilcoxon rank sum test

Experiment	**Volunteer(1)**	**Volunteer(2)**	**Volunteer(3)**	**Volunteer(4)**	**Volunteer(5)**
p-value	3.19E-81	2.23E-83	1.58E-71	2.23E-83	1.09E-80

Furthermore, we compute the effect size based on two measures: Cohen's d (Cohen J. , 1988) and Cliff's $delta$ (Cliff, 1993). Cohen's d reveals the standardised difference between the means of the two distributions, and Cliff's $delta$ demonstrates how often the values in the first distribution are larger than the values in the second distribution. According to the results in Table 4, all the Cohen's d values are categorized as “huge” (Sawilowsky, 2009) and all the Cliff's $delta$ values are categorized as “large” (Romano, 2006). This verifies that the EST values between visually similar block pairs are higher than those between different pairs; accordingly, we conclude that this metric is able to identify the visual similarity among web page blocks.

Table 4: Results of Cohen's d and Cliff's delta effect size estimation

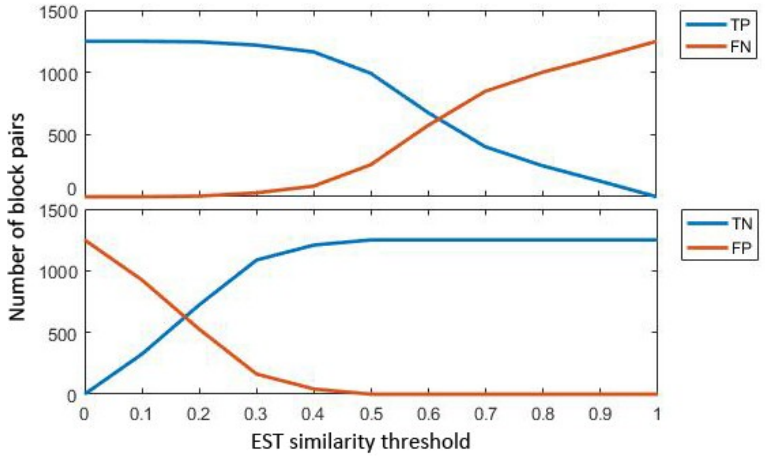
Experiment	**Volunteer(1)**	**Volunteer(2)**	**Volunteer(3)**	**Volunteer(4)**	**Volunteer(5)**
Cohen's d	3.0149	4.4428	2.5546	3.8381	2.9189
Cliff's $delta$	0.9867	0.9979	0.9244	0.9979	0.9834

To identify the optimal similarity threshold of the EST similarity metric, all the EST scores (i.e., 2500 scores) of the 5 experiments are categorized into the following four categories:

- True Positive (TP): two similar blocks are identified as similar correctly;
- True Negative (TN): two different blocks are identified as different correctly;
- False Positive (FP): two different blocks are identified as similar incorrectly; and
- False Negative (FN): two similar blocks are identified as different incorrectly.

Figure 10 displays a plot of these four categories versus the corresponding EST thresholds. It can be seen that EST=0.40 is a turning point of all the four metrics. TN increases and FP decreases noticeably prior to this turning point, while TP decreases and FN increases prominently after this point.

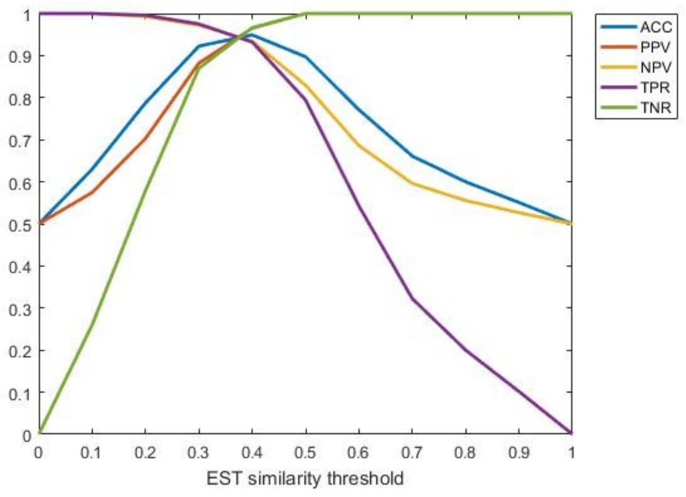
Figure 10: TP, TN, FP, and FN of the empirical metric (EST) on different thresholds



Further investigation is carried out in order to find out the optimal similarity threshold. Five additional metrics are calculated based on the above mentioned categories (i.e., TP, TN, FP, and FN). These five metrics include: precision or positive predictive value (PPV); negative predictive value (NPV); recall or true positive rate (TPR); specificity or true negative rate (TNR); and accuracy (ACC).

Figure 11 shows a plot of these five metrics against the corresponding EST thresholds. A similar pattern can be inferred from the plots where EST=0.40 is the crucial threshold that noticeably turns the values of ACC, PPV, NPV, TPR, and TNR. This indicates that 0.40 is a best point for a threshold.

Figure 11: ACC, PPV, NPV, TPR, and TNR of the EST metric on different thresholds



5.3 Locating the Main Data Region

The data extractor targets a specific block (i.e., the main data region) of the web page or works directly on the whole page. In this thesis, we compare the extraction efficiency of the two cases (i.e., extracting data records from the main region and extracting from the full web page).

The data region of a search result web page is defined as the block that contains all the displayed data records, i.e., the data region holds the main content of a particular search result web page. In general, a web page is divided into a number of main common sections including: the top section, the bottom section, the left (right) menu section, and the main content (Ahmadi, 2012). The main content is located in the center of the web page and covers a relative large area of the whole page. However, more than a block can satisfy these specifications; hence, we identify all candidate data region blocks; among all the candidate blocks, we select the block whose horizontal center is the closest to the horizontal center of the entire web page’s block as the data region. We define a candidate data region as the block that satisfies the following conditions:

1. Its top border is below the top border of the entire web page’s block.
2. Its bottom border is above the bottom border of the entire web page’s block.
3. Its area is larger than a specified threshold. Since web pages areas are not fixed, the threshold is proportional to the target web page’s total area. The ratio of the data region area is set to be greater than or equal to 0.4 (i.e., 40% of the whole web page’s area) by ViDE (Liu, 2010).

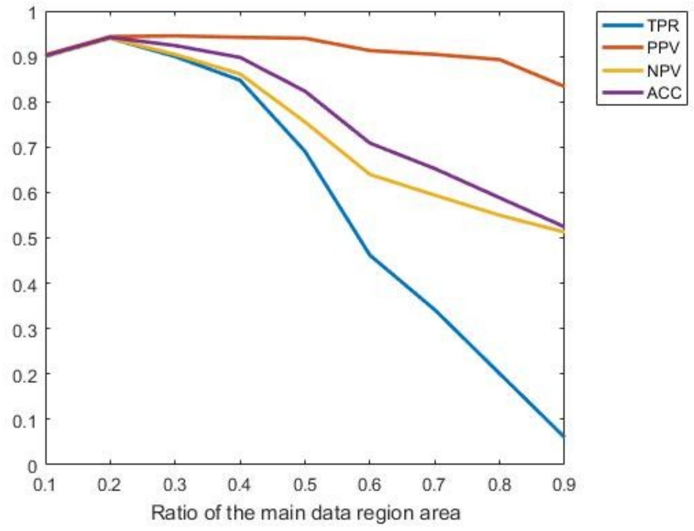
   We conducted an experiment in order to find out the optimal threshold for the area ratio. A total of 1000 different web pages are randomly crawled from the world's 500 top Websites as defined from statistics produced by Alexa (the web pages were retrieved on February, 2017), For each web page, the data region was located based on a set of thresholds ranging from 0.10-0.90 (i.e., 10%-90% of the whole web page’s area). Four types of results are defined:

   - True Positive (TP): data region is identified as data region correctly;
   - True Negative (TN): none-data region is identified as none-data region correctly;
   - False Positive (FP): none-data region is identified as data region incorrectly; and
   - False Negative (FN): data region is identified as none-data region incorrectly.

   Four metrics based on the above four types are examined to find out the optimal threshold. These four metrics include: recall or true positive rate (TPR); precision or positive predictive value (PPV); negative predictive value (NPV); and accuracy (ACC).

   Figure 12 demonstrates how these metrics change according to the alteration of the threshold. It can be noticed that the ratio of 0.20 has the highest values for all of the four metrics. After this point, PPV has a slight decrease, while TPR, NPV, and ACC decrease significantly. This indicates that 0.20 is the best point for a ratio threshold. However, we compare between the extraction results based on this threshold (i.e., 0.20) and the threshold utilized in ViDE (i.e., 0.40); we found that setting the ratio of the main region area to 0.20 improves the results in almost 37% of the web pages that fail when setting the ratio to 0.40.

Figure 12: TPR, PPV, NPV, and ACC of the main region proportional area thresholds



5.4 Extracting the Data Records

Based on the user’s choice, the extractor system inspects the block tree of the main data region or of the whole web page. The targeted block tree is first traversed, and its blocks are sorted in ascending order by their top and left attributes to be utilized as the input for the data record extractor. The proposed data record extractor mainly depends on Gestalt laws of similarity, continuity, and proximity to identify data record blocks on a search result web page; retrieved data records are visually similar and aligned with each other to be recognized as a whole by human readers. The working procedure of the data record extractor is explained as follows.

5.4.1 Matching the Visually Similar Aligned Blocks

A human reader groups elements that are aligned with each other on a web page (i.e., law of continuity). Web page designers arrange data records to be left-aligned as a vertical list or top-aligned as a horizontal list. This observation is empirically found to be true in 98% of the web pages in our testbed data sets. Blocks of adjacent and similar data records are siblings because they are semantically grouped by the GLM model (i.e., laws of proximity and similarity). Accordingly, the data record extractor starts by locating and matching the aligned similar blocks; two blocks are matched to each other if they are:

1. Siblings (i.e., they are grouped in one parent block).
2. Visually similar (i.e., their normalized EST similarity is above the threshold).
3. Vertically aligned (i.e., the distances from the left edge of the page to their left edges are the same) or horizontally aligned (i.e., the distances from the upper edge of the page to their upper edges are the same).
4. Separate (i.e., do not overlap).

Matching only sibling blocks to each other ensures that data item blocks from different data records are not matched. However, this condition will not be true when groups of categorized data records are displayed apart from each other with a separate parent block encompassing each group of adjacent records. Although, a separate set of matched blocks is created for each group of data records, they still can be merged with each other according to the similarity and continuity of their data record blocks, Two created sets of matched blocks are merged together if any of their two blocks are found to be visually similar, and vertically/horizontally aligned. Methods, such as (Shi, 2015) and (Bing L. a., 2011), that consider data records as just adjacent sibling nodes fail in this case. We match these blocks to each other according to their visual appearance regardless of their hierarchical organization in the web pages’ tree.

5.4.2 Discarding the non-Candidate Data Records

Other aligned similar blocks are matched to each other in addition to matching the actual data record blocks. These additional blocks are referred to as non-candidate data records which have to be removed from the created sets of the matched blocks. The non-candidate data records could be one of the following two types:

- Type (1): data item blocks of a single data record that are aligned and visually similar
- Type (2): parent blocks of grouped data records that are aligned and visually similar when they contain similar numbers of data records.

Data record blocks are displayed on a web page with similar visual features to be perceived as a whole, while adjacent data items that describe the same entity have distinct visual features to be distinguished by human readers (according to Gestalt law of similarity). However, some non-contiguous data items inside a data record block may have similar visual features; accordingly, they are matched to each other while they cannot be matched to the dissimilar data items in the same record. On the contrary, all data record blocks that are grouped inside a single parent block are matched to each other according to their visual similarity.

Therefore, to identify blocks of non-candidate data records, parent-child relations are set up between all the blocks that are stored during the matching process; each block must be located inside its parent as they visually appear on the web page. For each identified parent block and its set of child blocks, the parent block is excluded as a non-candidate data record if all of its child blocks are matched to each other (i.e., Type (2)). Otherwise, the child blocks are excluded as non-candidate data records, while their parent is saved as a candidate data record (i.e., Type (1)).

5.4.3 Filtering the Noise Blocks

In addition to the retrieved data records, a web page contains search/filter/sort/navigation elements that are utilized by users to: (1) initiate a new search by entering a new search keyword; (2) refine and sort the search results by some selected options; or (3) navigate between the retrieved data records. A search result web page may also contain blocks that display some statistical information related to the current query (e.g., total number of retrieved/displayed data records). All of these blocks are referred to as noise blocks that have to be excluded. Designers tend to place these blocks above and below the retrieved data records, i.e. the noise blocks enclose the main data region from the top and bottom boundaries. By examining the web pages in our testbed data sets, we found that the noise blocks in 79% of the web pages are located above and below the retrieved data records. These noise blocks will be matched to each other if they are aligned and visually similar as previously explained. In this case, both of the created set of the actual data record blocks and the created set of the noise blocks have similar relative positions on the web page which makes distinguishing between them not straightforward. Thus, they have to be removed from all the identified sets of candidate data record blocks.

ViDE (Liu, 2010) assumes that the noise blocks are the blocks that are not aligned to the left with the actual data records. This assumption is not always true (the noise blocks in 62% of the web pages in our datasets are found to be aligned with the data record blocks). Therefore, we follow a different strategy by calculating the distance between the noise blocks that are located above and below the data records. The set of the noise blocks is removed if the distance between any two of its blocks is large relative to the height of the entire web page; i.e. large enough to encompass the main data region that contains the retrieved data records. Hence, we utilize the threshold discussed in 5.3.

5.4.4 Identifying the Actual Data Record Blocks

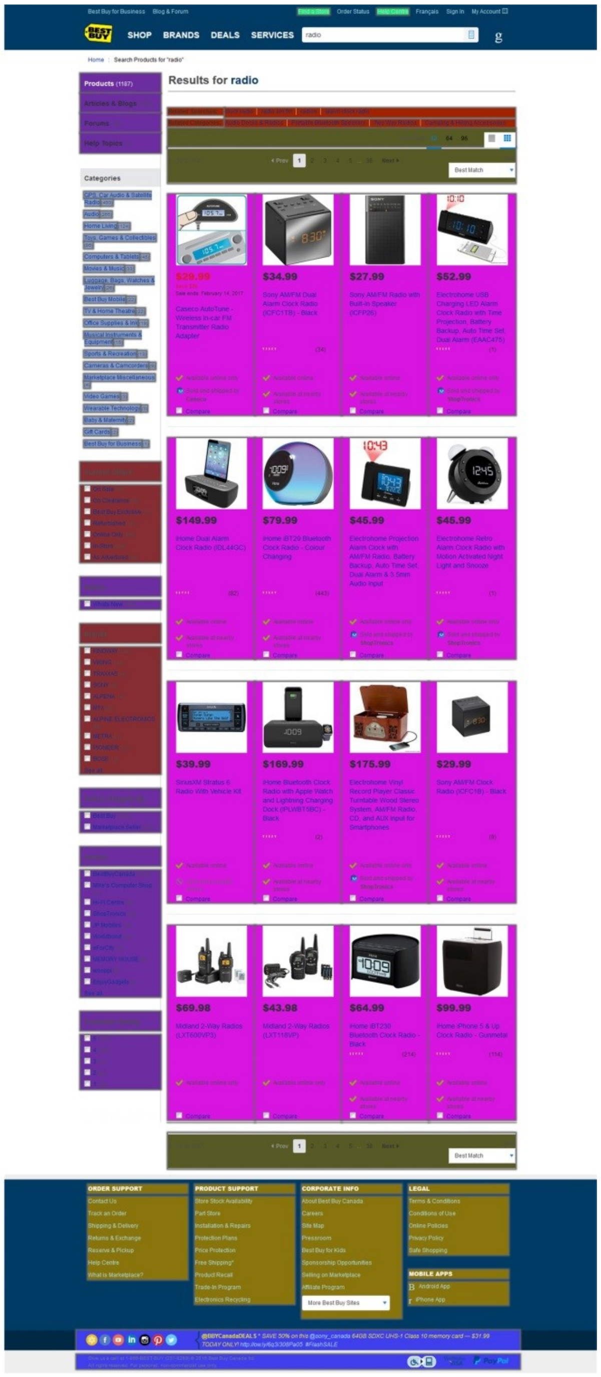
Among all the identified sets of candidate data record blocks, only one set represents the actual retrieved data records. Distinguishing between them is carried out based on the relative position of each set on the web page. Thus, we first create a representative block to enclose all the blocks in each set. The left (right, top, bottom) margin for each created block is equal to the left (right, top, bottom) margin of the block that has the smallest distance from its left (right, top, bottom) edge to the left (right, top, bottom) edge of the page.

The main content of a typical web page is centrally located (Ahmadi, 2012); retrieved data records are placed on the center of the page because they are considered the main subject of the query result. Therefore, for each created representative block, we calculate the Hausdorff distance from its center to the center of the entire web page (or the main data region). Calculation of the normalised Hausdorff distance is carried out using Equation (2). Ultimately, the actual retrieved data record blocks are the closest to the center of the web page.

For instance, Figure 13 shows the identified sets of the candidate data record blocks on a search result web page from Bestbuy.com where a distinct color is selected to illustrate each set of candidate record blocks. Set (3) in the figure represents the noise blocks which are displayed above and below the retrieved data record blocks, while Set (5) represents the actual retrieved data record blocks.

Figure 13: Example of the sets of the candidate data record blocks

- 🟪 Set (1)
- 🟥 Set (2)
- 🟩 Set (3)
- 🟦 Set (4)
- 🟪 Set (5)
- 🟥 Set (6)
- 🟨 Set (7)
- 🟦 Set (8)



5.5 Extracting and Aligning the Data Items

Each identified data record consists of data items that describe specific attributes of a particular object. The goal of the following steps is to extract these data values from all the retrieved data records, and align together those that belong to the same attribute.

According to the law of similarity, elements that appear similarly on a web page are more likely to be organized together by a human reader. Web page designers assign identical visual features to data items that describe the same attribute and distinguish adjacent data items within a single data record with distinct visual features. However, two data items of different attributes may appear visually similar; thus, the arrangement of the data items should be taken into account in addition to their visual similarity. Data items in each data record follow a consistent order within all of the retrieved data records; this consistent order makes these data records appear similarly on a web page. This observation is empirically found to be true in all the web pages in our datasets. However, the number of data items in each record may vary when some of the data items are optional. Our proposed data item extractor considers the data record blocks with the maximum number of data items as the records that have the minimum probability of missing any data item and considers them as the basis for matching the data items to each other. The matching process is carried out using the EST similarity method taking into account both all the available visual features and the order of the data items. The data item extractor works as follows.

5.5.1 Segmenting each Data Record into Data Items

A data record block is a rooted tree which is composed of a root block, internal blocks, and leaf blocks. The leaf blocks represent the minimum semantic units that cannot be further divided up (Liu, 2010); thus, each data item corresponds to a leaf block in the data record block that it belongs to.

However, some leaf blocks have to be merged together to form a complete data item. HTML text formatting tags (e.g., <*strong*>, <*em*>, <*strike*>, <*b*>, <*i*>, etc.), which are used to format text on a web page, lead to segment the block that belongs to a single attribute into sub-blocks. For instance, web page designers tend to highlight the existence of the search keywords in the displayed data records by assigning them a distinct format (e.g., bold or italic). During the segmentation, the block that contains the highlighted text is selected because it is a leaf block, while the remaining unformatted text is missing.

To solve this problem, approaches, such as (Grigalis, 2014), inspect the tag tree of the web page to remove the formatting tags. This manner is HTML-dependent because it is restricted to a limited number of HTML tags. A new set of formatting tags may be introduced as HTML is still evolving and these methods will not be longer effective. Therefore, we choose a different mechanism by checking the existence of text nodes as siblings of the detected leaf blocks. In this case, the textual content of the missing text nodes will be combined with the textual content of the leaf blocks according to their order.

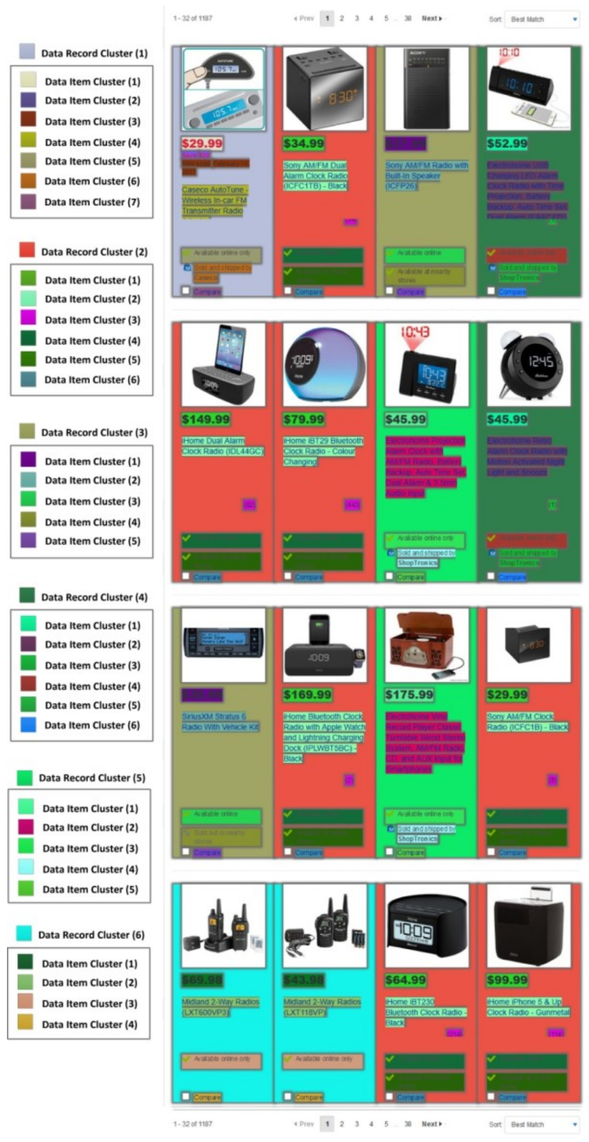
5.5.2 Selecting the Basic Data Items

In this step, we select the set of the visually identical data record blocks with the maximum number of data items as the basic data records. The basic data records have the minimum probability of missing any data item compared to the other retrieved data records, and they are considered as the basis for matching the data items to each other.

All the identified data record blocks are clustered into groups where each group contains visually identical data record blocks. Two data records are visually identical if they have the same number of leaf blocks and the normalized EST similarity between them is equal to 1. For each created cluster, the data item blocks of the stored data records are also clustered. Two data items are clustered together if they belong to different data records and they are visually identical. Among all the formed clusters of data records, the cluster with the maximum number of data item clusters is selected as the basic data records.

The previous example web page from Bestbuy.com has six clusters of data records and each cluster has a set of data item clusters as shown in Figure 14; the distinct colors represent either a data record cluster or a data item cluster. As can be seen from the figure, the first data record cluster has a total of seven data item clusters which are selected to be the base for the matching process.

Figure 14: Example of clustering the data items of each data record cluster



5.5.3 Matching all Data Items with the Base Data Items

In this step, all the created clusters of data items are utilized to extend the basic data item clusters by either merging them with visually matched clusters or inserting them as new clusters. The goal is to group together all the data values of the same attribute taking into account the existence of the optional attributes.

Each cluster of data items is compared against all the basic data item clusters and matched only to the first detected visually identical cluster that have not been matched before to any cluster from the same set. Since each cluster groups visually identical blocks, the first stored data item is chosen as a representative of that cluster. Accordingly, two clusters are matched to each other if their first stored data items are visually identical (i.e., the normalized EST similarity between them is equal to 1). To maintain the consistent order of the data items within all the retrieved data records, we ensure that they are traversed and compared against each other in the same order as they visually appear on the web page. If the cluster could not be matched to any of the basic data item clusters, this indicates the existence of an optional attribute. The cluster is then added as a new cluster to the set of the basic data item clusters. Ultimately, the basic data item clusters are expanded where each cluster is a group of data items that belong to the same attribute.

5.5.4 Aligning the Data Items in a Table

A table is created to align and store all the extracted data items. Data items that belong to the same attribute are placed in one column according to the order of the data records that they belong to. The total number of the identified data records on a web page represents the number of the table rows, while the total number of the identified attributes represents the number of the table columns. Table 5 shows how the extracted data items from the previous example web page from *Bestbuy.com* are aligned.

Table 5: Example of aligning the extracted data items in a table

	$29.99	Save $20	Sale ends: February 14, 2017	Caseco AutoTune - Wireless In-car FM Transmitter Radio Adapter		Available online only		Sold and shipped by Caseco	Compare
$34.99				Sony AM/FM Dual Alarm Clock Radio (ICFC1TB) - Black	(34)	Available online	Available at nearby stores		Compare
$27.99				Sony AM/FM Radio with Built-In Speaker (ICFP26)		Available online	Available at nearby stores		Compare
$52.99				Electrohome USB Charging LED Alarm Clock Radio with Time Projection, Battery Backup, Auto Time Set, Dual Alarm (EAAC475)	(1)	Available online only		Sold and shipped by ShopTronics	Compare
$149.99				iHome Dual Alarm Clock Radio (IDL44GC)	(82)	Available online	Available at nearby stores		Compare
$79.99				iHome iBT29 Bluetooth Clock Radio - Colour Changing	(443)	Available online	Available at nearby stores		Compare
$45.99				Electrohome Projection Alarm Clock with AM/FM Radio, Battery Backup, Auto Time Set, Dual Alarm & 3.5mm Audio Input		Available online only		Sold and shipped by ShopTronics	Compare
$45.99				Electrohome Retro Alarm Clock Radio with Motion Activated Night Light and Snooze	(1)	Available online only		Sold and shipped by ShopTronics	Compare
$39.99				SiriusXM Stratus 6 Radio With Vehicle Kit		Available online	Sold out in nearby stores		Compare
$169.99				iHome Bluetooth Clock Radio with Apple Watch and Lightning Charging Dock (IPLWBT5BC) - Black	(2)	Available online	Available at nearby stores		Compare
$175.99				Electrohome Vinyl Record Player Classic Turntable Wood Stereo System, AM/FM Radio, CD, and AUX Input for Smartphones		Available online only		Sold and shipped by ShopTronics	Compare
$29.99				Sony AM/FM Clock Radio (ICFC1B) - Black	(9)	Available online	Available at nearby stores		Compare
$69.98				Midland 2-Way Radios (LXT600VP3)		Available online only			Compare
$43.98				Midland 2-Way Radios (LXT118VP)		Available online only			Compare
$64.99				iHome iBT230 Bluetooth Clock Radio - Black	(214)	Available online	Available at nearby stores		Compare
$99.99				iHome iPhone 5 & Up Clock Radio - Gunmetal	(114)	Available online	Available at nearby stores		Compare

6. Experimental Work

In this section, we present our experimental results to show the effectiveness of our approach. We first describe the two datasets utilized in our experiments, and then introduce the employed performance measures. The entire extraction system consists of two procedures: Data Record Extraction (DRE) and Data Item Extraction (DIE); thus, we present the experimental results of each part separately. The efficiency of the system is compared against ViDE (Liu, 2010) and rExtractor (Anderson, 2013), which are the two current state-of-the-art vision-based approaches that automatically mine data records from individual Web pages. The proposed system is implemented in JavaScript as a Mozilla Firefox extension; the extracted record blocks are visually reflected on the processed Web page and the extracted data are saved to the user's system. The experiments were conducted on an Intel Core i7-4770, 3.40 GHz PC with 8GB memory. The average running time for the system to process a Web page is about 7 sec including building up the block tree.

6.1 Datasets

We have conducted experiments based on the following two datasets:

1. **TBDW** (TestBed for information extraction from Deep Web) version 1.02: this is a public third-party dataset which is collected by (Yamada, 2004) and is available at http://daisen.cc.kyushu-u.ac.jp/TBDW/. TBDW consists of 253 Web pages that are queried from 51 Websites randomly selected from 114,540 Web pages with search forms of various search engines. Among all the Websites, three same-template Web pages are queried from the 43rd Website, while five same-template Web pages are queried from each of the other 50 Websites. TBDW can be considered as a benchmark dataset that has been utilized for evaluation by many proposed data extraction systems, including (Shi, 2015), (Bing L. a.-L., 2013), (Su, 2012), (Bing L.

a., 2011), (Liu, 2010). The collectors of this dataset have provided annotations of the first data record displayed on each Web page as a sample of the other data records on the same page. All the works that have utilized this dataset, have presented their experimental results based on the identification of the first retrieved data record only; the other extracted data records have no weight on determining the entire extraction result. Therefore, we provide annotations for the remaining data records in all the Web pages in this dataset by following the provided annotation templates, then we evaluate the extraction efficiency based on the identification of all the data records. In our experiments, four Websites (i.e., twenty Web pages) are excluded because of the ambiguous record annotations or the distorted codes of their Web pages. Hence, the remaining 47 Websites, including 233 Web pages and 4297 data records in total, are used in our experiments.

2. **Our dataset**: In addition to the fact that the number of the Websites included in TBDW version 1.02 is limited for experimental comparison, TBDW is out-of-date because it has not been maintained since 2004. Web page design has enormously evolved in the last decade; the design and the layout of the Web pages involved in TBDW are much different than the design and the layout of current Web pages. Consequently, we created our own dataset, which includes 450 Web pages queried from 150 Websites selected from the world's 500 top Websites as defined from statistics produced by Alexa[5]. Specifically, the 150 Websites span the 15 different categories defined in Alexa. For each category, we visited the top 10 Websites, and three Web pages were queried from each visited Website. During the selection of the top Websites, we disregarded "inappropriate" samples of Websites, such as sites that do not have any search form to run a query, duplicate sites (e.g., "amazon.in" and "amazon.ca", etc), temporarily unavailable sites, and sites that contain inappropriate content. The 450 keywords that have been utilized to initiate the queries, are

[5] https://www.alexa.com/topsites. The top sites were retrieved on February, 2017.

selected from the most popular search keywords defined by WordStream[6]. For each of the 15 categories, we randomly select 30 keywords from the list of 10000 popular search keywords provided by WordStream for each category. Indeed, our dataset includes the total of 9865 data records retrieved from all of the generated queries. In the experiments, the data records extracted by the algorithm are compared against the actual retrieved data records. Thus, we manually extracted the actual data records by creating a data table for each Web page in the dataset; the cells of each table row are all the data items of a single data record, and the cells of each table column are the data items that belong to the same attribute from all the data records.

General statistics about the two utilized datasets (i.e., TBDW, and our dataset) are shown in Table 6:

Table 6: General statistics about the two utilized datasets

Feature	TBDW version 1.02	Our Dataset
Number of Websites	47	150
Number of Web pages	233	450
Total number of data records	4297	9865
Minimum number of data records per page	1	2
Maximum number of data records per page	160	147
Average number of data records per page	18	22
Total number of data items	17808	56117
Minimum number of data items per record	1	1
Maximum number of data items per record	17	49
Average number of data items per record	4	5

[6] http://www.wordstream.com/popular-keywords/

6.2 Performance Measures

Most of the existing proposals employ *precision*, *recall*, and *F-measure* to evaluate the performance of their extraction systems. Here, we also use the three measures as the evaluation metrics. Because the whole extraction system is composed of two parts (i.e., data record extraction and data item extraction), the performance measures are defined for them separately as follows:

- For data record extraction, precision P_{DR}, is the fraction of the total number of correctly extracted data records, DR_c, out of the total number of data records extracted, DR_e. This corresponds to the following equation:

$$P_{DR} = \frac{DR_c}{DR_e} \tag{11}$$

Recall R_{DR}, is the fraction of the total number of correctly extracted data records, DR_c, out of the total number of the actual data records in the query result pages, DR_r. This is shown in the following equation:

$$R_{DR} = \frac{DR_c}{DR_r} \tag{12}$$

The third measure, F_1 score or F-measure, is a derived metric that uses both precision and recall to give a single value that acts as a summary of the performance. The F-measure employed to evaluate the performance of the data record extraction can be calculated from above-mentioned results of P_{DR} and R_{DR} as shown in the following equation:

$$F1_{DR} = \frac{2 * P_{DR} * R_{DR}}{P_{DR} + R_{DR}} \tag{13}$$

Also, we employ an additional metric, i.e. *accuracy*, for evaluating the performance of the data record extractor. The calculation of accuracy ACC_{DR}, is shown in Equation (14) .

$$ACC_{DR} = \frac{DR_c + DR_j}{DR_c + DR_r + DR_f} \tag{14}$$

Where DR_c is the total number of correctly extracted data records; DR_j is the total number of correctly rejected none data records; DR_r is the total number of the actual data records on the web page; and DR_f is the total number of incorrectly extracted none data records.

- For data item extraction, precision P_{DI}, is the fraction of the total number of correctly extracted data items, DI_c, out of the total number of data items extracted, DI_e. The precision, P_{DI}, is given by the following equation:

$$P_{DI} = \frac{DI_c}{DI_e} \tag{15}$$

Recall R_{DI}, is the fraction of the total number of correctly extracted data items, DI_c, out of the total number of the actual data items in the query result pages, DI_r. This corresponds to the following equation:

$$R_{DI} = \frac{DI_c}{DI_r} \tag{16}$$

The F_1 score or F-measure used to evaluate the performance of the data item extraction can be calculated as shown in the following equation:

$$F1_{DI} = \frac{2 * P_{DI} * R_{DI}}{P_{DI} + R_{DI}} \qquad (17)$$

In addition to the above mentioned performance measures, we introduce a new metric, *accuracy of alignment*, to measure the accuracy of aligning the correctly extracted data items. Specifically, this measure checks if the extracted data items that belong to the same attribute are aligned together (e.g., are all the extracted 'prices' aligned in one column?). We define this metric, A_{LDI}, as the fraction of the correctly aligned data items, LDI_c, out of the total number of all the aligned data items by the algorithm, LDI_r. This is shown in the following equation:

$$A_{LDI} = \frac{LDI_c}{LDI_r} \qquad (18)$$

6.3 Experimental Design

The first step of the experiment is the data retrieval. We develop a Mozilla Firefox extension to implement the proposed data extractor that enables users to extract search result data records. The APIs provided by Mozilla Firefox for manipulating DOM elements facilitate building up the layer tree which is in turn utilized to build up the block tree. Also, users can see (in real-time) any modifications of the web page's DOM tree which are applied immediately by Mozilla Firefox (Xu, 2016). The result of running the extraction extension on a web page containing query results is a table of data that is saved to the user's system where each row represents the data items of a single data record. Furthermore, the identified data record blocks are visually marked by assigning a special background color to the corresponding DOM element of each data record block; the updates are displayed immediately on the original web page. However, a user can choose between extracting the data from the main data region of the web page or from the whole page by selecting one of the two options available by the extension. Hence, to test the two cases, we apply the extractor on all the web pages of the two datasets described in 6.1 to obtain two tables of data for each web page.

The second step is data comparison. The actual targeted data on each web page in the two utilized datasets are already collected and organized in a table to be compared against the data extracted by the algorithm. Accordingly, the experiment evaluates the extraction efficiency by comparing the data table created by the algorithm for each web page and the already saved data table. Each row (i.e., a data record) from the first table is matched to a single row from the second table with the maximum number of similar cells (i.e., data items). Comparing the data items of two data records is addressed as a string similarity problem. We employ the *Jaccard* index as a string similarity metric; it compares the similarity/distance of two finite sets by computing the size of their intersection to the size of their union (Niwattanakul, 2013). However, any other string similarity metric can be used to assess the similarity between two data items.

As explained in section 6.2, number of correctly extracted data records determines the results of the data record extraction (DRE), while the results of the data item extraction (DIE) depends on the number of correctly extracted data items. Only correct extracted data records from DRE are used to evaluate the efficiency of DIE. In the experiments, we consider a correct extracted data record as an extracted data record that contains at least half of the exact data items in the actual matched data record. However, failure of extracting the remaining data items is reflected in the results of DIE. If only the extracted records that contain the full number of the original data items are considered as correct when evaluating DRE, then both precision and recall of DIE will be always 100% which does not accurately reveal the extraction performance of the algorithm.

The last step of the experimental work is comparing the proposed algorithm against the two current state-of-the-art vision-based systems which are ViDE (Liu, 2010) and rExtractor (Anderson, 2013). The executable versions of the two systems are not publicly available and we could not obtain them from the authors. Consequently, we re-implemented the two algorithms according to the explanation provided in both references. Experiments on ViDE are conducted in a similar manner as our proposed system. However, rExtractor works only on identifying data record blocks without further extracting their data items. Hence, experiments on rExtractor are conducted by counting the visually marked detected data record blocks on each web page.

6.4 Experimental Results of Data Record Extraction (DRE)

In this section, we evaluate the first part of the proposed system (i.e., Data Record Extraction (DRE)) and compare it with the extraction performance of ViDE and rExtractor. Table 7 and Table 8 show the experimental results of the proposed visual system, ViDE and rExtractor on our dataset and TBDW version 1.02, respectively. For each dataset, we provide a comparison between the two cases (i.e., extracting data records from the main data region of a web page, and extracting data records from a full web page).

Table 7: DRE results on our dataset

Method	Target	Precision	Recall	F measure	accuracy
The Visual System	Data Region	92.85%	82.51%	87.38%	95.97%
	Full Page	93.97%	85.30%	89.43%	96.59%
ViDE	Data Region	30.42%	15.61%	20.63%	79.88%
	Full Page	31.48%	32.51%	31.99%	76.77%
rExtractor	Data Region	53.30%	56.14%	54.68%	84.29%
	Full Page	53.41%	57.63%	55.44%	84.36%

Table 8: DRE results on TBDW version 1.02

Method	Target	Precision	Recall	F measure	accuracy
The Visual System	Data Region	88.29%	74.46%	80.79%	88.82%
	Full Page	89.04%	79.36%	83.92%	90.40%
ViDE	Data Region	36.78%	11.53%	17.56%	66.04%
	Full Page	42.56%	20.37%	27.55%	66.54%
rExtractor	Data Region	44.66%	39.61%	41.98%	68.55%
	Full Page	46.37%	42.87%	44.55%	69.36%

From both Table 7 and Table 8, we can make the following observations. First, DRE results are slightly higher when the extractor targets a full web page than the main data region of a web page for both datasets. This is due to the failure in identifying the blocks of the main data regions for some web pages. Specifically, the visual system, ViDE, and rExtractor miss the data region blocks in 5%, 20%, and 7% of the web pages in our dataset, respectively. For TBDW, the visual system, ViDE, and rExtractor fail in locating the data region blocks in 4%, 13%, and 5% of the web pages, respectively. The specifications indicated in 5.3 for locating the data region block are equally employed in the three comparable algorithms; however, the input for each algorithm is different. The proposed visual system utilizes the GLM (Xu, 2016) visual block tree as the input; ViDE utilizes the VIPS (Cai, 2003) visual block tree; rExtractor uses the visual block model (i.e., the product of the tag tree and the CSS of the page). VIPS identifies a limited number of blocks; some big blocks are identified by VIPS while smaller blocks are missed (Xu, 2016). Also, VIPS performance efficiency on modern web pages is extremely low because the visual separators that VIPS uses to segment web pages are much less apparent in modern pages than traditional ones. Accordingly, the failure rate in locating the data region blocks in ViDE is higher than the visual system and rExtractor on both datasets which interprets the low recall result value of ViDE.

The second observation is that the proposed visual system performs significantly better than ViDE and rExtractor on both datasets and the two cases (i.e., data region and full page). As mentioned above, the type of the input tree has an influence on the extraction results. However, the technique that the algorithm applies to analyze a web page's tree to detect data record blocks has the major influence. Our analysis observes that the clustering and regrouping technique of ViDE is the main cause for its low efficiency. ViDE clusters the visually similar data item blocks on a web page and assumes that the blocks of each cluster belong to distinct data records. Consequently, regrouping the clustered blocks by ViDE creates erroneous data record boundaries when each of these records contains visually similar data item blocks. In contrast, the visual system employs the GLM model (Xu, 2016) which simulates human perception by utilizing the Gestalt laws of grouping to segment a web page into visual semantic blocks. Applying the law of proximity groups the adjacent data item blocks of each data record into a single node of the block tree as explained in 5.1. This creates a significantly better representation of data records than the faulty regrouping mechanism of ViDE.

Furthermore, ViDE assumes that all data record blocks have the same distance to the left boundary of the page; thus, it is only able to address data record blocks that are arranged in a single column and discards the other blocks of a displayed grid. Conversely, the visual system follows the Gestalt law of continuity to address both vertically and horizontally aligned blocks to identify all the data records arranged as a grid (ViDE identifies only 22.54% of the data records displayed in multiple columns compared to the visual system that identifies 87.18%).

Our analysis of rExtractor observes that its clustering mechanism is the main cause of its errors. rExtractor clusters blocks based on their visual appearance and widths; it assumes that all the data record blocks on a web pages are visually similar and have the same width. In fact, the size of the data items of a single data record determines its width;

thus, assuming that all the displayed data records have the same width is not always true (we found that almost 21% of the web pages in our testbed datasets contain data records with different width). In addition, rExtractor does not distinguish between the similar sub-blocks inside a single data record block and the similar blocks of the actual data records. For instance, if a web page contains 10 data records and each record is composed of two visually similar sub-blocks with the same width, rExtractor mistakenly identifies these sub-blocks as the actual data records and, thus, the number of extracted data records will be duplicated to 20. On the contrary, we take into account the parent-child relations between the identified similar blocks to detect the two types of non-candidate data records as explained in 5.4.2.

By analyzing the results of the visual system on the two datasets, we could classify the web pages that have low recall values as follows:

- 59% of the web pages have block trees that are inaccurately constructed in accordance to the substandard design of those pages. For instance, an obvious space is usually left between every two successive data records and smaller spaces among the data items that constitute each single record. If the web page's design does not comply with the Gestalt law of proximity, then an inaccurate block representation is created by the GLM model (Xu, 2016) for the displayed data records.
- 17% of the web pages contain other blocks that are closer to center of the page than the retrieved data record blocks; these additional blocks are mistakenly identified as data records.
- 9% of the web pages do not comply with the Gestalt law of continuity; they contain data records that are not displayed as a list or grid (i.e., not vertically/horizontally aligned with each other).

- 8% of the web pages contain only a single data record on each page. Our algorithm works on matching the visually similar blocks on a web page; thus, it fails to detect the correct data record when the query retrieves only one data record. However, we noticed that the web pages that contain one data record all belong to TBDW dataset.
- Lastly, 7% of the web pages contain groups of data records and each group is displayed visually different from the other. For instance, a group contains product data records resulted from the query, while another group has news data records related to the query. In this case, our algorithm is able to identify the data records in one group which is the closest to the center of the page.

The last observation that can be seen from Table 7 and Table 8 is that the performance of the three algorithms on our dataset is better than their performance on TBDW. First, TBDW is out-of-date because it has not been maintained since 2004; web page design and layout have enormously evolved in the last decade. Second, all the web pages of TBDW are just stored as HTML files without any CSS files or images. It can be noticed that the decline in the results on TBDW is more obvious for the visual system and rExtractor (about 6% and 11% respectively). These two recent algorithms are built on simulating the human intuition of understanding the visual presentation of web page objects. This indicates that human mechanisms of perceiving objects are more imitated in the design of modern web pages.

As mentioned in 6.1, our dataset is composed of 15 categories of web pages. In the following, we demonstrate the extraction results for each category and compare the performance of the three algorithms on those categories.

Figure 15: DRE results by category – Our algorithm

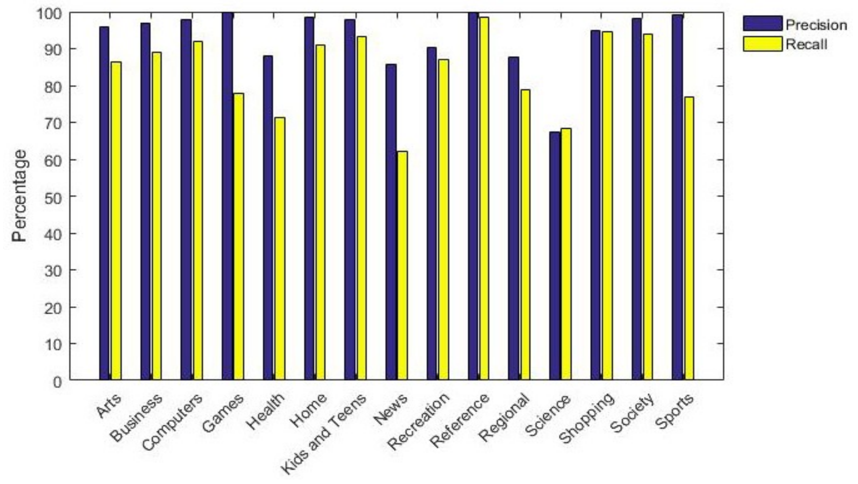


Figure 16: DRE results by category – ViDE

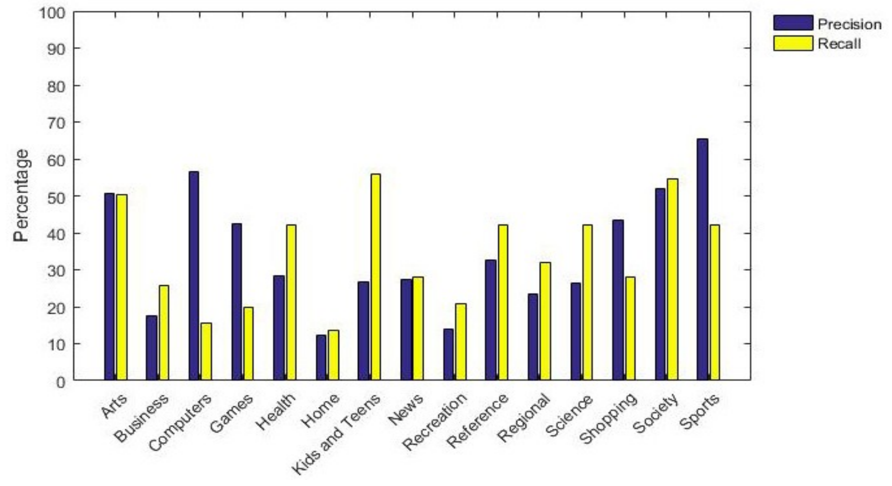


Figure 17: DRE results by category – rExtractor

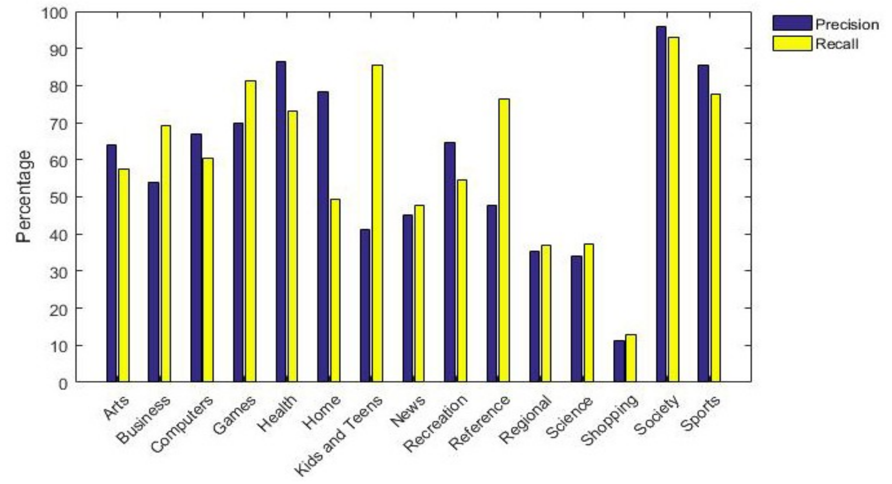


Figure 15 shows the DRE results (precision and recall) of the visual system on each category. According to the chart, web pages of Reference and Games have the highest precision (100% and 99.9%, respectively); the extreme precision values indicate that the false positives on these two categories are very improbable. Reference category has also the highest recall (98.6% of correct data records on Reference web pages have been identified). However, the chart shows that Science and News categories have the lowest precision (67.3% and 85.6%, respectively) and recall (68.5% and 62.1%, respectively). The low precision value on Science web pages is due to the high false positives (i.e., none-data records are mistakenly extracted), while the low recall value on both Science and News pages is resulting from the low true positives (i.e., many of the correct data record blocks have been missed). Overall, the best performance of the visual system can be observed on Reference and Shopping categories where the values of both precision and recall are noticeably high and very close to each other (100%, 98.6% and 94.8%, 94.6%, respectively); this indicates that the Gestalt laws of grouping are more applicable on the web pages of these two categories.

Figure 16 and Figure 17 show the DRE results of ViDE and rExtractor on each category, respectively. The charts show that the extraction efficiency of the two algorithms on Science and News web pages is low while their results on Society web pages are high comparative to the visual system. Although, web pages of Home and Shopping present high results for the visual system, the lowest performance of ViDE and rExtractor can be observed on these two categories, respectively. Generally, the performance of rExtractor is noticeably higher than ViDE on each category and it is more similar to our system.

6.5 Experimental Results of Data Item Extraction (DIE)

In this section, we evaluate the second part of the proposed system (i.e., Data Item Extraction (DIE)) and compare it with the extraction performance of ViDE (rExtractor cannot perform this task.). The experimental results of the visual system and ViDE on our dataset and TBDW are shown in Table 9 and Table 10, respectively. For each dataset, we again compare between the two cases (i.e., extracting data items from main data regions, and extracting data items from full web pages). Only correctly extracted data records by the visual system and ViDE are used to evaluate DIE; for instance, the results of the visual system displayed below for our dataset are on the 82.51% and 85.30% correct data records (as shown in Table 7 and Table 8).

Table 9: DIE results on our dataset

Method	Target	Precision	Recall	F measure	Accuracy of alignment
The Visual System	Data Region	90.69%	89.33%	90.00%	95.23%
	Full Page	90.06%	89.34%	89.70%	95.33%
ViDE	Data Region	46.25%	54.61%	50.09%	74.45%
	Full Page	51.96%	58.40%	54.99%	74.80%

Table 10: DIE results on TBDW version 1.02

Method	Target	Precision	Recall	F measure	Accuracy of alignment
The Visual System	Data Region	86.18%	79.45%	82.68%	97.94%
	Full Page	86.59%	79.18%	82.72%	97.61%
ViDE	Data Region	35.45%	57.98%	43.99%	80.32%
	Full Page	34.51%	53.88%	42.07%	75.42%

The first observation that can be made from Table 9 and Table 10 is that the extraction results of the proposed visual system are significantly higher than ViDE on both datasets and the two cases (i.e., data region and full page). As mentioned in 6.4, the regrouping and clustering technique of ViDE tends to create wrong data record boundaries. Thus, ViDE has a high possibility of mistakenly including additional data items from other blocks to a created data record block which increases the number of the incorrectly extracted data items (i.e., false positives) for some records. In addition, ViDE utilizes VIPS to obtain a web page's block tree; we found that 31% of the missed data items by ViDE are according to the misrepresentation in the created VIPS block trees.

Second, we can observe that the DIE results of the proposed system are fixed among the two cases (i.e., data region and full page) for each dataset. DIE performance is evaluated on the correctly extracted data records obtained from the previous step (i.e., DRE). As explained in 6.4, the number of correctly extracted data records is higher when addressing a full web page than a main data region; hence, the number of relevant data items (i.e., targeted data items that must be extracted) is also higher. However, the number of the extracted data items by an algorithm should be proportional to the number of the relevant data items; this makes the DIE results for both data regions and full pages constant. On the other hand, we can notice that the DIE results of ViDE are irregular among the two cases for each dataset (the results are higher when addressing full web pages of our dataset, while they are higher when addressing data regions of TBDW).

The third observation that can be made is that the extraction results (specifically F1 score) of the proposed system and ViDE on the first dataset are higher than their results on TBDW. Although, the two algorithms apply different alignment strategies, they both consider that adjacent data items of different attributes in a single data record block have different presentation style. By analyzing the web pages in the two datasets, we found that this feature is not valid in only 3% of the web pages of our dataset, while it fails in almost 17% of the web pages of TBDW. Accordingly, the DIE results of the visual system on TBDW slightly declined (by almost 7%) because GLM merges the adjacent visually similar data item blocks and considers them as one data item.

Lastly, we can notice that the alignment results of the visual system and ViDE on TBDW are slightly higher than their results on our dataset. Our analysis of the two datasets found that web page designers tend to highlight the existence of the search keywords in the retrieved data records by assigning them distinct text styles. If the search keyword does not appear in some of the displayed data records, then the highlighted data items will appear visually different from the un-highlighted data items in those records. Accordingly, they will not be aligned in the same column even though they belong to the same semantic concept. This issue found to be slightly more in our dataset than in TBDW.

As mentioned above, the input block tree is different for each of the three algorithms (i.e., the visual system, ViDE, and rExtractor). In addition to the fact that the size of the GLM (Xu, 2016) block tree, the VIPS (Cai, 2003) block tree, and the visual block model is different for the same web page, the time required to build each tree also differs. This, in turn, has a great effect on the total execution time of each algorithm. Figure 18 and Figure 19 demonstrate the execution time of the proposed visual system, ViDE, and rExtractor on our dataset and TBDW version 1.02, respectively, including the time of building up the block trees. The plots show that the visual system has the fastest performance among the three algorithms on both datasets. The average running time for the visual system to process a web page is about 10 sec and 3 sec, respectively, while the average running time for ViDE is about 249 sec and 32 sec. rExtractor has the slowest execution time with an average of 133 sec and 369 sec, respectively.

Figure 18: The execution time of our system, ViDE, and rExtractor on our dataset

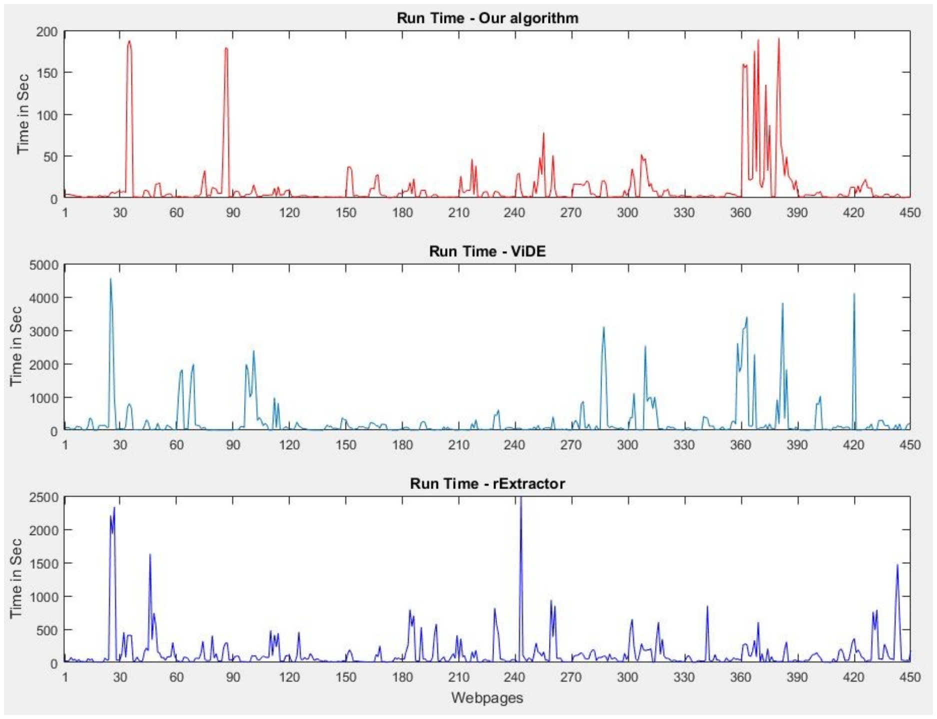
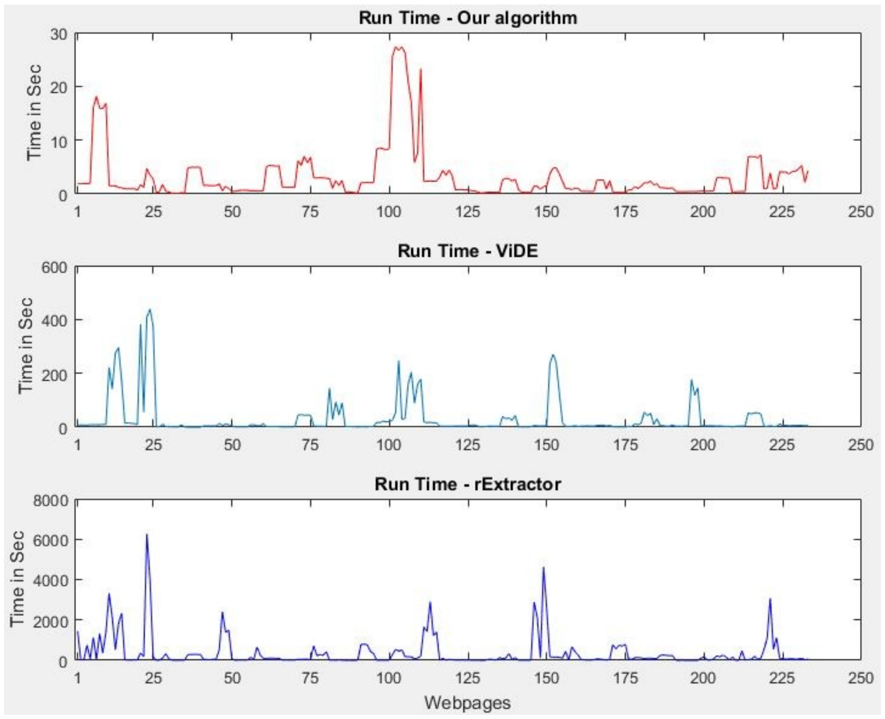


Figure 19: The execution time of our system, ViDE, and rExtractor on TBDW



7. Conclusion and Future Work

In this thesis, we propose a novel purely visual approach that is programming-language-independent for automatically extracting structured web data. Numerous approaches have been introduced to address this problem but most of them are HTML-dependent solutions with limited functionality. The proposed system overcomes the limitation of the previous work by making a full use of the natural human tendencies of visual object perception. Specifically, the system consists of two steps: (1) data record extraction where we apply three of the Gestalt laws of grouping (i.e., laws of continuity, proximity, and similarity) which are used by humans to group data record blocks on a web page; and (2) data item extraction and alignment where we employ the Gestalt law of similarity which is utilized to group the visually identical data item blocks.

Our approach first builds up the web page's visual block tree using the GLM model. It then locates the centrally located main data region and matches the visually similar blocks that are adjacent and aligned with each other; it employs the EST model to obtain the similarity of block pairs based on their visual properties. The extractor sets up parent-child relations between the matched blocks to discard the non-candidate data records and then filters out any existing noise blocks. It finally pinpoints the actual retrieved data records based on their relative position on the page among all the other matched blocks. To carry out data item extraction and alignment, our approach selects visually identical data record blocks, with the maximum number of data items, as the basis for matching the visually similar data items to each other. The matching process is carried out again using the EST similarity method considering the actual order of the data items. Finally, the visually identical data items that belong to the same attribute are aligned together in one column.

Our approach addresses both data records arranged as lists and grids. It is also able to identify visually similar data records displayed apart as groups on a web page despite their underlying tree structure. We have conducted extensive experiments based on human experience to find out the optimal similarity threshold of the EST empirical metric to match data records with optional data items. Furthermore, we propose a new testbed dataset for web data extraction that surpasses the widely used public dataset, TBDW version 1.02. Our experimental results on the introduced dataset and TBDW demonstrate that our approach outperforms the two state-of-art vision-based algorithms, ViDE and rExtractor:

- our approach has significantly higher precision, recall, F-1 score, and accuracy than ViDE and rExtractor for the data record extraction task;
- our approach has notably higher precision, recall, F-1 score, and accuracy of alignment than ViDE for the data item extraction and alignment task;
- our approach provides steadier and higher distributions of precisions and recalls on all of fifteen categories that our dataset is composed of; and
- our approach is able to achieve approximately 96% and 93% better processing time than ViDE and rExtractor on our dataset, respectively; and approximately 91% and 98% better processing time on TBDW version 1.02, respectively.

In our future research, we intend to work on the next phase of the extraction system which is *attribute labelling* or *annotating*. In this stage, we aim to automatically assign a semantic label for each aligned group (i.e., column) of data items using the vocabulary provided by Schema.org (schema.org). Although, few methods have been proposed to address this problem, they follow a supervised manner where users have to provide annotated examples.

References

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, 17*(6), 734-749.

Adomavicius, G., & Tuzhilin, A. (2011). Context-aware recommender systems. In *Recommender systems handbook* (pp. 217-253). Springer.

Ahmadi, H. a. (2012). User-centric adaptation of Web information for small screens. *Journal of visual languages & computing, 23*(1), 13--28.

Airoldi, E. M., & David M. Blei, S. E. (2008). Mixed membership stochastic blockmodels. " *Journal of Machine Learning Research, 9*(Sep), 1981-2014.

Alpuente, M. a. (2009). A visual technique for web pages comparison. *Electronic Notes in Theoretical Computer Science, 235*, 3--18.

Altergeo. (2016, September 7). *Products*. Retrieved from Altergeo: http://platform.altergeo.ru/index.php

Amatriain, X., & Basilico, J. (2012, April 6). *Netflix Recommendations: Beyond the 5 stars (Part 1)*. Retrieved from The Netflix Tech Blog: http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html

Anand, D., & Bharadwaj, K. K. (2011). Utilizing various sparsity measures for enhancing accuracy of collaborative recommender systems based on local and global similarities. *Expert systems with applications, 38*(5), 5101-5109.

Anderson, N. a. (2013). Visually extracting data records from the deep web. *Proceedings of the 22nd International Conference on World Wide Web* (pp. 1233--1238). ACM.

Bawa-Cavia, A. (2011). Sensing the urban: using location-based social network data in urban analysis . *Pervasive PURBA Workshop.*

Berkhin, P. (2006). Bookmark-coloring algorithm for personalized pagerank computing. *Internet Mathematics, 3*(1), 41-62.

Billsus, D., & Pazzani, M. J. (1998). Learning Collaborative Information Filters. *Icml*, *98*, pp. 46-54.

Bing, L. a. (2011). Towards a unified solution: data record region detection and segmentation. *Proceedings of the 20th ACM international conference on Information and knowledge management* (pp. 1265--1274). ACM.

Bing, L. a.-L. (2013). Robust detection of semi-structured web records using a dom structure-knowledge-driven model. *ACM Transactions on the Web (TWEB), 7*(4), 21.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-based systems. 46*, pp. 109-132. Elsevier.

Breese, J. S., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (pp. 43-52). Morgan Kaufmann Publishers Inc.

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5-32.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction, 12*(4), 331-370.

Cai, D. a.-R.-Y. (2003). VIPS: A vision-based page segmentation algorithm.

Chang, C.-H. a.-S.-C.-L. (2016). Efficient page-level data extraction via schema induction and verification. *Proceedings, Part II, of the 20th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining. 9652*, pp. 478--490. Springer-Verlag New York, Inc.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research 16*, 321-357.

Chen, P., Lu, Z., & Gu, J. (2009). Vehicle travel time prediction algorithm based on historical data and shared location. *2009 Fifth International Joint Conference on INC, IMS and IDC* (pp. 1632-1637). IEEE.

Chen, S. (2015). A Review on the Tree Edit Distance Problem and Related Path-Decomposition Algorithms. *arXiv preprint arXiv:1501.00611*.

Cheng, C., Yang, H., King, I., & Lyu, M. R. (2012, July). Fused Matrix Factorization with Geographical and Social Influence in Location-Based Social Networks. *Aaai, 12*, 17-23.

Cheng, C., Yang, H., Lyu, M. R., & King, I. (2013). Where You Like to Go Next: Successive Point-of-Interest Recommendation. *IJCAI*, *13*, pp. 2605-2611.

Cho, E., Myers, S. A., & Leskovec, J. (2011). Friendship and mobility: user movement in location-based social networks. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD ’11)* (pp. 1082-1090). New York, NY: ACM.

Cliff, N. (1993). Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological Bulletin, 114*(3), 494.

Cliff, N. (1996). Answering Ordinal Questions with Ordinal Data Using Ordinal Statistics. In *Multivariate Behavioral Research* (pp. 331-350). Routledge.

Cochran, W. G. (1953). *Sampling techniques.* John Wiley & Sons.

Cohen, J. (1988). Statistical power analysis for the behavioral sciences . Hilsdale. *NJ: Lawrence Earlbaum Associates, 2*.

Cohen, S. a. (2014). A general algorithm for subtree similarity-search. *2014 IEEE 30th International Conference on Data Engineering (ICDE)* (pp. 928--939). IEEE.

Cormier, M. a. (2016). Purely vision-based segmentation of web pages for assistive technology. *Computer Vision and Image Understanding, 148*, 46--66.

Cramer, H., Rost, M., & Holmquist, L. E. (2011). Performing a check-in: emerging practices, norms and'conflicts' in location-sharing using foursquare. *Proceedings of the 13th international conference on human computer interaction with mobile* (pp. 57-66). ACM.

Crandall, D. J., Backstrom, L., Cosley, D., Suri, S., Huttenlocher, D., & Kleinberg, J. (2010). Inferring social ties from geographic coincidences. *Proceedings of the National Academy of Sciences, 107*(52), 22436-22441.

Crunchbase. (2015, September 1). *Organization Brightkite: Overview*. Retrieved from Crunchbase: https://www.crunchbase.com/organization/brightkite#/entity

Crunchbase. (2016, August 11). *Organization Gowalla: Overview*. Retrieved from CrunchBase: https://www.crunchbase.com/organization/gowalla#/entity

Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine Learning Research, 7*, 1-30.

Deshpande, M., & Karypis, G. (2004). Item-based top-n recommendation algorithms.". *ACM Transactions on Information Systems (TOIS), 22*(1), 143-177.

Devika, K. a. (2013). An overview of web data extraction techniques. *International journal of scientific engineering and technology, 2*(4).

Díaz-Uriarte, R., & De Andres, S. A. (2006). Gene selection and classification of microarray data using random forest. *BMC bioinformatics, 7*, 1.

Ding, Y., & Li, X. (2005). Time weight collaborative filtering. *Proceedings of the 14th ACM international conference on Information and knowledge management* (pp. 485-492). ACM.

Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 155-164). ACM.

Drummond, C., & Holte, R. C. (2003). C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. *Workshop on learning from imbalanced datasets II*, *11.*

Duff, I. S. (1977). A survey of sparse matrix research. *Proceedings of the IEEE, 65*(4), 500-535.

Eagle, N., Pentland, A. S., & Lazer, D. (2009). Inferring friendship network structure by using mobile phone data. *Proceedings of the national academy of sciences, 106*(36), 15274-15278.

Eldirdiery, H. F. (2015). Web Document Segmentation for Better Extraction of Information: A Review. *International Journal of Computer Applications, 110*(3).

Facebook, Inc. (2016, July 27). *Facebook Reports Second Quarter 2016 Results.* Retrieved from Facebook Investor Relations: https://investor.fb.com/investor-news/press-release-details/2016/Facebook-Reports-Second-Quarter-2016-Results/default.aspx

Fan, S. a. (2014). Web data extraction based on visual information and partial tree alignment. *Web Information System and Application Conference (WISA), 2014 11th* (pp. 18--23). IEEE.

Fang, Y. a. (2017). STEM: a suffix tree-based method for web data records extraction. *Knowledge and Information Systems*, 1--27.

Ferrara, E. a. (2014). Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems, 70*, 301--323.

Foursquare. (2016a, April 11). *Privacy 101*. Retrieved from foursquare.com: https://foursquare.com/privacy

Foursquare. (2016b, April 16). *Since Foursquare launched in 2009*. Retrieved from The Foursquare Blog: http://blog.foursquare.com/post/142900756695/since-foursquare-launched-in-2009-there-have-been

Foursquare. (2016c, September 6). *What if I forgot to or couldn't check in somewhere?* Retrieved from support.foursquare.com: https://support.foursquare.com/hc/en-us/articles/211542067-What-if-I-forgot-to-or-couldn-t-check-in-somewhere-

Foursquare. (2016d, June 20). *Why are Foursquare and Swarm separate apps?* Retrieved from support.foursquare.com: https://support.foursquare.com/hc/en-us/articles/202630254-Why-are-Foursquare-and-Swarm-separate-apps-

Gao, H., & Liu, H. (2014). *Location-Based Social Network Data Repository*. Retrieved from http://www.public.asu.edu/~hgao16/dataset.html

Gao, H., Tang, J., & Liu, H. (2012). Exploring Social-Historical Ties on Location-Based Social Networks. *ICWSM.*

Gao, H., Tang, J., Hu, X., & Liu, H. (2013). Exploring temporal effects for location recommendation on location-based social networks. *Proceedings of the 7th ACM conference on Recommender systems (RecSys'13)* (pp. 93-100). New York, NY: ACM.

Glas, A. S., Lijmer, J. G., Prins, M. H., Bonsel, G. J., & Bossuyt, P. M. (2003). The diagnostic odds ratio: a single indicator of test performance. *Journal of clinical epidemiology, 56*(11), 1129-1135.

Glueck, J. (2016, January 14). *Foursquare Direct: Foursquare Ushers In A New Era*. Retrieved from Medium: https://medium.com/foursquare-direct/foursquare-ushers-in-a-new-era-f52edb39af6#.7rc896u3n

Gowalla Incorporated. (2011, February 1). *Gowalla*. Retrieved from Gowalla: http://gowalla.com/

Grigalis, T. a. (2014). Unsupervised structured data extraction from template-generated web pages. *Journal of Universal Computer Science (J. UCS), 20*(3), 169--192.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, a. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations, 11*(1).

He, B. a.-C. (2007). Accessing the deep web. *Communications of the ACM, 50*(5), 94--101.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering, 21*(9), 1263-1284.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS), 31*(3), 5-53.

Hernando, A., Bobadilla, J., & Ortega, F. (2016). A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model. *Knowledge-Based Systems, 97*, 188-202.

Holte, R. C., Acker, L., & Porter, B. W. (1989). Concept Learning and the Problem of Small Disjuncts. *IJCAI*, *89*, pp. 813-818.

Hong, J. L. (2011). Data extraction for deep web using wordnet. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 41*(6), 854--868.

Hong, J. L. (2013). OntoExtract-Automated extraction of records using multiple ontologies. *2013 10th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)* (pp. 878--882). IEEE.

Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on* (pp. 263-272). Ieee.

Huang, H., & Gartner, G. (2014). Using trajectories for collaborative filtering-based POI recommendation. *International Journal of Data Mining, Modelling and Management, 6*(4), 333-346.

Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). *Recommender systems: an introduction.* Cambridge University Press.

Jawaheer, G., Weller, P., & Kostkova, P. (2014). Modeling User Preferences in Recommender Systems: A Classification Framework for Explicit and Implicit User Feedback. *ACM Trans. Interact. Intell. Syst., 4*(2), 8:1-8:26.

Jimenez, P. a. (2015). On Extracting Information from Semi-structured Deep Web Documents. *International Conference on Business Information Systems* (pp. 140--151). Springer.

Jo, T., & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter, 6*(1), 40-49.

Kaggle.com. (2015, December 3). *Mean Average Precision*. Retrieved from Kaggle.com: https://www.kaggle.com/wiki/MeanAveragePrecision

Kayed, M. a.-H. (2010). FiVaTech: Page-level web data extraction from template pages. *IEEE Transactions on Knowledge and Data Engineering, 22*(2), 249--263.

Koenigstein, N., & Koren, Y. (2013). Towards scalable and accurate item-oriented recommendations. *Proceedings of the 7th ACM conference on Recommender systems* (pp. 419-422). ACM.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., & Riedl, J. (1997). GroupLens: applying collaborative filtering to Usenet news. *Communications of the ACM, 40*(3), 77-87.

Korb, K. B., & Nicholson, A. E. (2010). *Bayesian artificial intelligence.* CRC press.

Koren, Y. (2010). Collaborative filtering with temporal dynamics. *Communications of the ACM, 53*(4), 89-97.

Kotsiantis, S., Kanellopoulos, D., & Pintelas, P. (2006). Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering , 30*(1), 25-36.

Kubat, M., & Matwin, S. (1997). Addressing the curse of imbalanced training sets: one-sided selection. *ICML*, *97*, pp. 179-186.

Le Hégaret, P. a. (2000, November 13). *What is the Document Object Model?* Retrieved May 5, 2017, from W3C: https://www.w3.org/TR/DOM-Level-2-Core/introduction.html

Le Hégaret, P. a. (2005, January 19). *Document Object Model (DOM)*. Retrieved May 5, 2017, from W3C: https://www.w3.org/DOM/

Leca, C.-L., Nicolaescu, I., & Rîncu, C.-I. (2015). Significant Location Detection & Prediction in Cellular Networks using Artificial Neural Networks. *Computer Science and Information Technology, 3*(3), 81-89.

Li, L. a. (2007). Visual segmentation-based data record extraction from web documents. *Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference* (pp. 502--507). IEEE.

Liakos, P. a. (2016). Focused crawling for the hidden web. *World Wide Web, 19*(4), 605--631.

Lin, Z. a. (2012). A multidimensional sequence approach to measuring tree similarity. *IEEE Transactions on Knowledge and Data Engineering, 24*(2), 197--208.

Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing, 7*(1), 76-80.

Ling, C. X., & Sheng, V. S. (2011). Cost-sensitive learning. In *Encyclopedia of Machine Learning* (pp. 231-235). Springer.

Liu, B., Fu, Y., Yao, Z., & Xiong, H. (2013). Learning geographical preferences for point-of-interest recommendation. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1043-1051). ACM.

Liu, W. a. (2010). Vide: A vision-based approach for deep web data extraction. *IEEE Transactions on Knowledge and Data Engineering, 22*(3), 447--460.

Liu, Y., Liu, C., Liu, B., Qu, M., & Xiong, H. (2016). Unified Point-of-Interest Recommendation with Temporal Interval Assessment. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1015-1024). San Francisco, California, USA: ACM.

Lu, Y. a. (2013). Annotating search results from web databases. *IEEE transactions on knowledge and data engineering, 25*(3), 514--527.

Lu, Z., Wang, H., Mamoulis, N., Tu, W., & Cheung, D. W. (2015). Personalized location recommendation by aggregating multiple recommenders in diversity. *Proceedings of the Workshop on Location-Aware Recommendations (LocalRec2015) co-located with the 9th ACM Conference on Recommender Systems, (RecSys 2015)* (pp. 28-35). New York, NY: ACM.

Luo, M. R. (2001). The development of the CIE 2000 colour-difference formula: CIEDE2000. *Color Research & Application, 26*(5), 340--350.

Mannan, N. B., Sarwar, S. M., & Elahi, N. (2014). A new user similarity computation method for collaborative filtering using artificial neural network. *International Conference on Engineering Applications of Neural Networks* (pp. 145-154). Springer.

Monard, M. C., & Batista, G. E. (2002). Learning with Skewed Class Distrihutions. *Advances in Logic, Artificial Intelligence, and Robotics: LAPTEC 2002 , 85*, 173-182.

Monreale, A., Pinelli, F., Trasarti, R., & Giannotti, F. (2009). Wherenext: a location predictor on trajectory pattern mining. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 637-646). ACM.

Morawski, J., Stepan, T., Dick, S., & Miller, J. (2017). A Fuzzy Recommender System for Public Library Catalogs. *International Journal of Intelligent Systems*.

Mori, T. a. (2015). Similar subtree search using extended tree inclusion. *IEEE Transactions on Knowledge and Data Engineering, 27*(12), 3360--3373.

Moritz, H. (1980). Geodetic reference system 1980. *Journal of Geodesy, 54*(3), 395-405.

Murolo, A. a. (2016). Revisiting Web Data Extraction Using In-Browser Structural Analysis and Visual Cues in Modern Web Designs. *International Conference on Web Engineering* (pp. 114--131). Springer.

Nichols, D. (1998). Implicit rating and filtering. *Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering* (pp. 31-36.). Budapest: ERCIM.

Nilashi, M., Jannach, D., bin Ibrahim, O., & Ithnin, N. (2015). Clustering-and regression-based multi-criteria collaborative filtering with incremental updates. *Information Sciences, 293*, 235-250.

Niwattanakul, S. a. (2013). Using of Jaccard coefficient for keywords similarity. *Proceedings of the International MultiConference of Engineers and Computer Scientists*, *1.*

Noulas, A., Scellato, S., Lathia, N., & Mascolo, C. (2012). Mining user mobility features for next place prediction in location-based services. *2012 IEEE 12th International Conference on Data Mining* (pp. 1038-1043). IEEE.

Oard, D. W., & Kim, J. (1998). Implicit feedback for recommender systems. *Proceedings of the AAAI workshop on recommender system*, (pp. 81-83).

Papagelis, M., Rousidis, I., Plexousakis, D., & Theoharopoulos, E. (2005). Incremental collaborative filtering for highly-scalable recommendation algorithms. *International Symposium on Methodologies for Intelligent Systems* (pp. 553-561). Springer .

Park, M.-H., Hong, J.-H., & Cho, S.-B. (2007). Location-based recommendation system using bayesian user's preference model in mobile devices. *International Conference on Ubiquitous Intelligence and Computing* (pp. 1130-1139). Springer .

Pawlik, M. a. (2015). Efficient computation of the tree edit distance. *ACM Transactions on Database Systems (TODS), 40*(1), 3.

Pham, H., Hu, L., & Shahabi, C. (2011). GEOSO-a geo-social model: from real-world co-occurrences to social connections. *International Workshop on Databases in Networked Information Systems* (pp. 203-222). Springer .

Pouramini, A. a. (2015). Web data extraction using textual anchors. *2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)* (pp. 1124--1129). IEEE.

R Core Team. (2015). R: A language and environment for statistical computing. Vienna, Austria. Retrieved from https://www.R-project.org/

Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010). Factorizing personalized markov chains for next-basket recommendation. *Proceedings of the 19th international conference on World wide web* (pp. 811-820). ACM.

Robusto, C. C. (1957). The cosine-haversine formula. *The American Mathematical Monthly, 64*(1), 38-40.

Rogmann, J. J. (2013). orddom: Ordinal Dominance Statistics. University of Hamburg, Department of Psychology, Germany. Retrieved from https://CRAN.R-project.org/package=orddom

Romano, J. a. (2006). Appropriate statistics for ordinal level data: Should we really be using t-test and Cohen'sd for evaluating group differences on the NSSE and other surveys. In *annual meeting of the Florida Association of Institutional Research* (pp. 1-33).

Saissi, Y. a. (2014). Extraction of relational schema from deep web sources: a form driven approach. *Complex Systems (WCCS), 2014 Second World Conference on* (pp. 178--182). IEEE.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). *Application of dimensionality reduction in recommender system-a case study.* DTIC Document, Minnesota University, Department of Computer Science, Minneapolis.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th international conference on World Wide Web* (pp. 285-295). ACM.

Sawilowsky, S. S. (2009). New effect size rules of thumb.

Scellato, S., Noulas, A., Lambiotte, R., & Mascolo, C. (2011). Socio-Spatial Properties of Online Location-Based Social Networks. *Proceeding of the 5th International AAAI Conference on Weblogs and Social Media.*

Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative Filtering Recommender Systems. In *The Adaptive Web* (pp. 291-324). Springer.

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. *Proceedings of the 25th annual international ACM SIGIR conference on Researchand development in information retrieval* (pp. 253-260). ACM.

schema.org. (n.d.). Retrieved 2017, from http://schema.org/

Shahbazi, A. a. (2014). Extended subtree: a new similarity function for tree structured data. *IEEE Transactions on knowledge and Data Engineering, 26*(4), 864--877.

Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In *Recommender systems handbook* (pp. 257-291). Springer US.

Shen, Y., & Jin, R. (2012). Learning personal+ social latent factor model for social recommendation. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 1303-1311). ACM.

Shi, S. a. (2015). AutoRM: An effective approach for automatic Web data record mining. *Knowledge-Based Systems, 89*, 314--331.

Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., et al. (2013). Real-time human pose recognition in parts from single depth images. *Communications of the ACM, 56*, 116-124.

Siddharthjn. (2017, April 18). *Introduction to the DOM*. Retrieved May 5, 2017, from Mozilla Developer Network MDN: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

Sinha, R. R., & Swearingen, K. (2001). Comparing Recommendations Made by Online Systems and Friends. *DELOS workshop: personalisation and recommender systems in digital libraries*, *106.*

Sleiman, H. A. (2013). A survey on region extractors from web documents. *IEEE Transactions on Knowledge and Data Engineering, 25*(9), 1960--1981.

socialbakers. (2016, August 5). *10 Most Checked-In Facebook Places*. Retrieved from socialbakers: https://www.socialbakers.com/blog/479-10-most-checked-in-facebook-places

Su, W. a. (2012). Combining tag and value similarity for data extraction and alignment. *IEEE Transactions on knowledge and Data Engineering, 24*(7), 1186--1200.

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence 2009*, 4.

Svetnik, V., Liaw, A., Tong, C., Culberson, J. C., Sheridan, R. P., & Feuston, B. P. (2003). Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of chemical information and computer sciences, 43*(6), 1947-1958.

Thamviset, W. a. (2014). Bottom-up region extractor for semi-structured web pages. *Computer Science and Engineering Conference (ICSEC), 2014 International* (pp. 284--289). IEEE.

Thamviset, W. a. (2014). Information extraction for deep web using repetitive subject pattern. *World Wide Web, 17*(5), 1109.

The Nielsen Company. (2015, December 17). *Tops of 2015: Digital*. Retrieved from Nielsen: http://www.nielsen.com/us/en/insights/news/2015/tops-of-2015-digital.html

Tomek, I. (1976). Two modifications of CNN. *IEEE Trans. Systems, Man and Cybernetics, 6*, 769-772.

Twitter, Inc. (2016, August 18). *REST APIs*. Retrieved from Twitter Developer: https://dev.twitter.com/overview/documentation

Ugander, J., Karrer, B., Backstrom, L., & Marlow, C. (2011). The anatomy of the facebook social graph. *arXiv preprint arXiv:1111.4503*.

Uzun, E. a. (2013). A hybrid approach for extracting informative content from web pages. *Information Processing & Management, 49*(4), 928--944.

Varlamov, M. a. (2016). A survey of methods for the extraction of information from Web resources. *Programming and Computer Software, 42*(5), 279--291.

Wang, H., Terrovitis, M., & Mamoulis, N. (2013). Location Recommendation in Location-based Social Networks using User Check-in Data. *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL GIS 2013)* (pp. 374-383). New York, NY: ACM.

Wang, Y., Yuan, N. J., Lian, D., Xu, L., Xie, X., Chen, E., et al. (2015). Regularity and conformity: Location prediction using heterogeneous mobility data. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1275-1284). ACM.

Weiss, G. M. (2004). Mining with rarity: a unifying framework. *ACM SIGKDD Explorations Newsletter, 6*(1), 7-19.

Williams, D. R. (2016, May 19). *Earth Fact Sheet*. Retrieved from NASA Space Science Data Coordinated Archive: http://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html

Williams, J. (2011, December 5). *Going to Facebook*. Retrieved from Gowalla Blog: http://blog.gowalla.com/post/13782997303/gowalla-going-to-facebook

Xu, Z. a. (2015). A New Webpage Classification Model Based on Visual Information Using Gestalt Laws of Grouping. *International Conference on Web Information Systems Engineering* (pp. 225--232). Springer.

Xu, Z. a. (2016). Identifying semantic blocks in Web pages using Gestalt laws of grouping. *World Wide Web, 19*(5), 957--978.

Yamada, Y. a. (2004). Testbed for information extraction from deep web. *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters* (pp. 346--347). ACM.

Yang, D., Zhang, D., Yu, Z., & Wang, Z. (2013). A sentiment-enhanced personalized location recommendation system. *Proceedings of the 24th ACM Conference on Hypertext and Social Media* (pp. 119-128). ACM.

Ye, J., Zhu, Z., & Cheng, H. (2013). What's your next move: User activity prediction in location-based social networks. *Proceedings of the SIAM International Conference on Data Mining.* SIAM.

Ye, M., Yin, P., & Lee, W.-C. (2010). Location recommendation for location-based social networks. *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems* (pp. 458-461). ACM.

Ye, M., Yin, P., Lee, W.-C., & Lee, D.-L. (2011). Exploiting geographical influence for collaborative point-of-interest recommendation. *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval* (pp. 325-334). ACM.

Yu, Y., & Chen, X. (2015). A survey of point-of-interest recommendation in location-based social networks. *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence.*

Yuan, Q., Cong, G., Ma, Z., Sun, A., & Thalmann, N. M. (2013). Time-aware point-of-interest recommendation. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 363-372). ACM.

Zaiontz, C. (2016, August 7). *Wilcoxon Signed-Ranks Table*. Retrieved from Real Statistics Using Excel: http://www.real-statistics.com/statistics-tables/wilcoxon-signed-ranks-table/

Zeleny, J. a. (2017). Box clustering segmentation: A new method for vision-based web page preprocessing. *Information Processing & Management, 53*(3), 735--750.

Zhai, Y. a. (2005). Web data extraction based on partial tree alignment. *Proceedings of the 14th international conference on World Wide Web* (pp. 76--85). ACM.

Zhang, K. a. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing, 18*, 1245--1262.

Zheng, Y., Xie, X., & Ma, W.-Y. (2010). GeoLife: A Collaborative Social Networking Service among User, Location and Trajectory. *IEEE Data Eng. Bull., 33*(2), 32-39.