IDENTIFYING WEBPAGE REGIONS AND THEIR ROLES BY COMBINING IMAGE PROCESSING AND MARKUP ANALYSIS

A Thesis

by

SANJEEV KUMAR SINGH

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Richard Furuta
Co-Chair of Committee,	Frank Shipman
Committee Member,	Lynn Burlbaw
Interim Head of Department,	Nancy Amato

August 2014

Major Subject: Computer Science

ABSTRACT

Understanding what are the regions of a webpage and the functions of those regions is important for many services over web pages, including screen readers, web search, and assessing web-page similarity. In this thesis, we present an approach to identify the regions of a webpage based on image processing techniques and to identify the portions of the DOM tree corresponding to these regions. We then present and compare a rule-based approach and a SVM-based approach using the visual and markup information to classify regions based on their roles. A corpus of 150 web pages exhibiting a wide variety of designs was collected. Each page was provided human-assigned regions and their roles to use in training and for evaluating results. The segmentation algorithm accurately identified 77.8% of the 1222 web page regions in the corpus but its performance was not even across different types of regions. Segmentation accuracy was above 80% for headers, footers, body regions, and top navigation bars. The algorithm had more difficulty with left, right, and bottom navigation bars and dynamic content, having lower than 70% accuracy for locating these segments. The correctly segmented web page components were used as a test collection to compare the rule-based and SVM-based approach to assigning the role of each segment. The SVM-based and the rule-based approach both achieved between 74 and 75% accuracy over 951 classifications. The SVM-based approach was better at classifying left and bottom navigation bars while the rule-based approach did better at recognizing dynamic content. Moreover, an accuracy of 81.3% is obtained when we used both the methods to identify regions correctly. In this

case, we considered a region correctly identified if the region is identified correctly either by the rule-based or SVM-based method. Overall, these results are promising for incorporating these segmentation and segment role classifications into web services.

ACKNOWLEDGEMENTS

For this thesis, I have every reason to be thankful to my advisors, Dr. Richard Furuta and Dr. Frank Shipman. I am really grateful to them for their guidance in due course of my research work that led to the completion of this thesis. It's their encouragement that kept me focused and helped me to persevere the research work at hand. I would like to sincerely thank them for the freedom that they gave me to explore various fields while working on my thesis. Undoubtedly, pondering over various possibilities for the given task at hand has broadened my knowledge and strengthened my zeal to pursue research in future.

I am also thankful to Dr. Lynn Burlbaw for being one of my advisory committee members and motivating me to pursue this research.

I take this opportunity to thank my lab members - Luis Meneses and Himanshu Barthwal, whose active participation in research related discussions always helped me to proceed in the right direction.

This gives me a chance to pay my gratitude to my parents, whose sacrifices have made me the person I am today.

Last but not the least, I thank the Almighty for empowering me with knowledge and strength to overcome my limitations all the time.

NOMENCLATURE

OCR	Optical Character Recognition
DOM	Document Object Model
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
SVM	Support Vector Model
CRF	Conditional Random Field
RBF	Radial Basis Function

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

The World Wide Web is today a powerhouse of information. Webpages are an essential constituent of the web. Each of these webpages comprises of various regions, which are visual chunks containing one or more elements. In general, a region of a webpage is visually distinct from the other regions contained in the page. Webpages are broadly divided into the regions as shown in Figure 1.

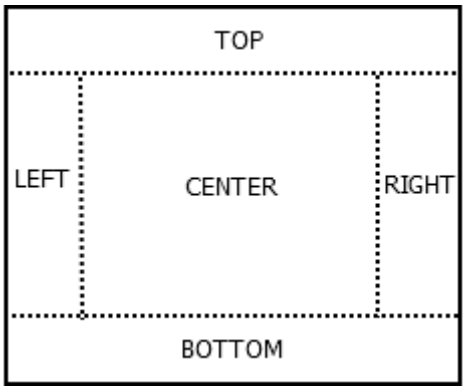


Figure 1. Basic layout of a webpage

Each of these regions has its own uses. The top region is mostly used for site branding and many a times contains a horizontal navigation bar. The bottom region usually has copyright information and often a navigation bar. The center region is often the area where main content is placed. The left region generally contains a vertical navigation bar and the

right region may contain ads or another navigation bar. Some of the common layout patterns of websites are shown in Figure 2.

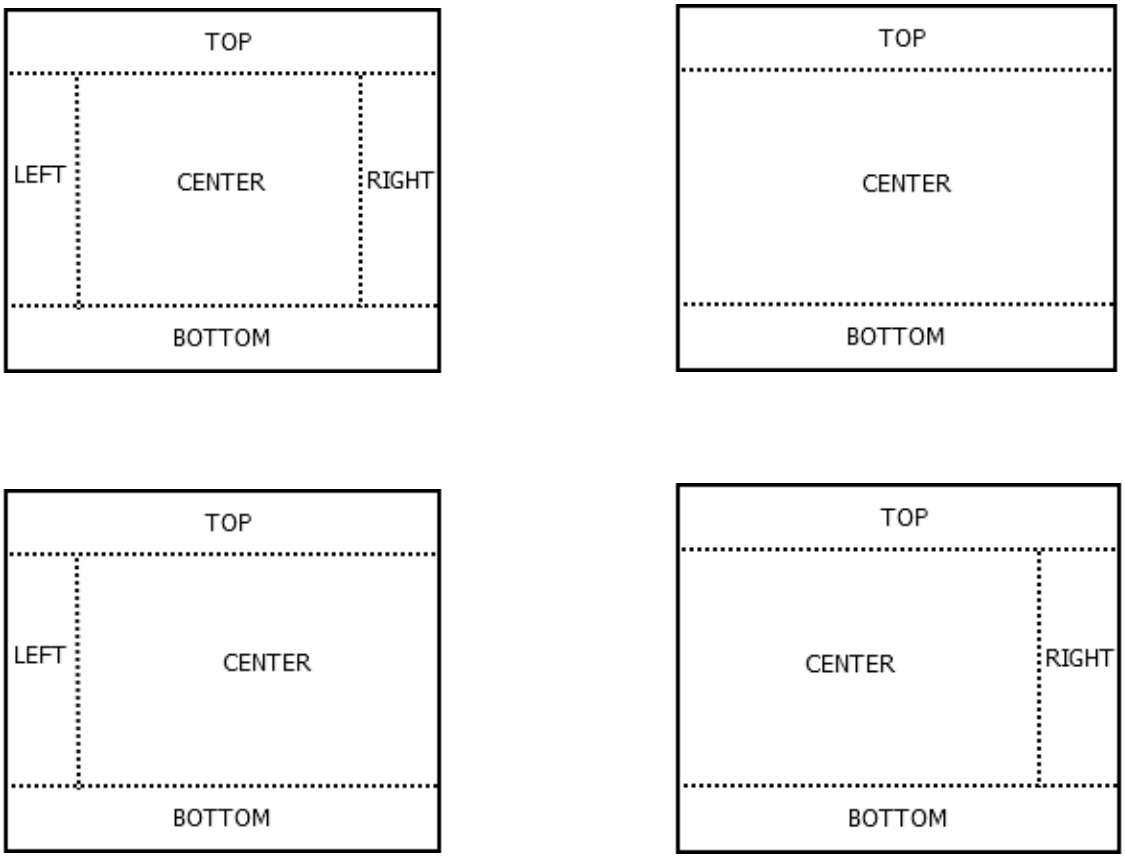


Figure 2. Common layouts of webpages

Digital libraries comprised of web documents encoded in HTML, CSS and similar browser-oriented representations experience more challenges than similar digital libraries where the constituent resources are more traditional in form (e.g., pdf files). A particular challenge for developing library technologies, including content indexing and similarity

assessment, is identifying which portions of a web document are the primary content of a resource. This is because web-based documents are often presented with additional context, including navigation to other collections or resources, advertisements, copyright notices, etc. Identifying the primary and the contextualizing content and understanding the roles of such contextualizing content enable alternative presentation of web resources and more informed assessments of resource similarity.

We have developed techniques for identifying the components of a webpage and their roles that combine image processing with analysis of the resource's source code (e.g., HTML and CSS). As already mentioned, such a capability can aid document indexing and content similarity assessment by identifying the core contents of a web resource. Recognizing the structure of web pages is also important when determining what text in a web page is related to the images on the page [17]. Beyond these applications, the resulting understanding of web page layouts can be used to compare pages based on their look and feel and can be used to improve access to web content via screen readers [27].

There are many approaches to web page segmentation. Some of these approaches are based on the source code of the webpage, i.e. HTML and CSS. Completely relying on such methodologies can be misleading because the same visual structure of a webpage can be generated by very different Document Object Model (DOM) trees. In most hand-coded pages, the visual layout loosely approximates the structural hierarchy of the DOM. But with the development and widespread use of WYSIWYG editors that generate HTML and CSS for the author, webpage layouts have become more complex and less likely to be well

represented by the DOM tree. This results in poor performance of segmentation techniques based purely on the webpage markup.

As an alternative to relying on webpage source code, another approach is to apply image processing techniques to the rendered web resource to enable similarity assessment, image searching, and phishing detection. These approaches first decompose the webpages into visually distinguishable regions and then calculate the visual similarity between two or more webpages. Similarity can be based on a combination of layout similarity and block-level similarity. In practice, many of the vision-based approaches to segmentation attempt to map the segments to elements in the DOM tree.

Identifying the segments of a webpage is a crucial step for many of the applications mentioned earlier. The second step is characterizing the role of the segments. This is crucial for identifying the primary content of a webpage. This paper presents an approach to webpage segmentation and the classification of the roles of webpage segments by using image processing followed by markup analysis to identify the regions of a webpage. Then we compare a rule-based approach and a SVM-based approach to segment classification.

The next section presents additional discussion of related research. Section 3 presents the segmentation and role identification algorithms. Section 4 describes experiments assessing the performance of the algorithms and their results. Finally, Section 5 summarizes our conclusions and presents directions for future work.

2. LITERATURE REVIEW

Understanding the significance and usability of different features of a web page is a major field of research in the area of Computer Vision. A lot of work has been done in the field of web analysis. This work focuses on the identification of webpage regions. A great deal of motivation has been derived from the existing literature for image segmentation and web content analysis.

2.1 Image segmentation

There are numerous contributions by researchers in the field of segmentation and clustering of images. Various approaches have been proposed to segment an image.

- Graph-based methods - Felzenszwalb et al. [12] introduced a method for segmenting an image into regions. The image is represented in the form of a graph and a predicate is defined in order to measure the evidence for a boundary between two regions. Based on this predicate, they proposed a segmentation algorithm. Yi and Moon [31] explained the graph cut based segmentation approach, which uses both boundary and regional information in order to partition an image into different regions. Apart from being applicable to a specific image with known information, it is also effective for an image without any pre-known information.
- Region merging techniques – Peng et al. [23] addressed the automatic image segmentation problem in a region merging style. Initially, the image is over-

segmented into many regions. Then, based on a statistical test, regions are iteratively merged.

- Techniques based on mapping image pixels to some feature space - Comaniciu and Meer [8] used the mean shift algorithm, a simple nonparametric procedure for estimating density gradients, to propose a general technique for the recovery of significant image features. With this approach, a feature space of any nature can be processed that includes color image segmentation.
- Spectral methods – Chen and Li [7] addressed the problem of image co-segmentation through a spectral method. They transformed the co-segmentation problem into a Rayleigh quotient problem, which can be solved by eigen-decomposition. Then, pixels in each image are classified and common objects of the image pair are obtained based on spectral clustering.
- Iterated dividing and shrinking – Jiuxin et al. [19] introduced the Iterated Dividing and Shrinking algorithm for segmenting webpages. A webpage is first transformed into an image and then the image is divided into visually cohesive sub-images based on repeated shrinking and splitting.

2.2 Web content analysis

Previous work on web content extraction is dominated by the construction of wrappers. Wrappers are handcrafted solutions that exploit properties of DOM either at page-level or at site-level. At the page-level, DOM properties are used by applying tag-based heuristics [9, 14]. On the other hand, at the site-level, DOM properties are used by identifying frequent patterns or templates [20, 21, and 29]. Spengler et al. [26] considered

loopy conditional random field (CRF) in the form of graphical model for news content extraction. They used the structural and visual attributes of each region of the webpage to derive a realistic set of local features.

There exist a number of approaches for analyzing web content. Finn et al. [14] considered the HTML document as a list composed of characters and labels. They focused on automatic classification of news articles from WWW for integrating them into digital libraries. Bar-Yossef and Rajagopalan [1] worked on improving the search engine by detecting the template in the web page. Lin and Ho [21] divided the web page into several blocks according to the table labels and devised an approach to discover informative contents from a set of webpages. Feng et al. [13] introduced a framework of web page analysis with coordinate trees, which could divide the pages by space relationships and their locations. Gupta et al. [16] suggested a method of masking off ads by counting the number of linked or non-linked words.

Cai et al. [3, 4] addressed the problem of web content structure analysis using visual characteristics such as background colors, the blank areas, font size, and font type of the web page. Their Vision-based Page Segmentation algorithm [4] is an automatic top-down, tag-tree independent approach for detecting the structure of webpage. It is an efficient algorithm and works well even when the underlying documentation representation is far different from the layout structure.

3. APPROACH

Motivation from the existing work on image segmentation and web content analysis led us propose a method to identify regions in a webpage. First, we segment a webpage into various regions using image processing techniques on the screenshot of the page. Post segmentation, we traverse the DOM tree of the webpage and map its elements and their attributes with respective segmented regions. Once the mapping is in place, based on the relevant features (e.g., text length, hyperlink text density, font size, font weight, etc.) and spatial attributes (e.g., height, width, and position), each of these regions are classified into various categories like header, footer, navigation bar, body etc. The work flow of the proposed method is shown in Figure 3.

3.1 Webpage to image conversion

The first and foremost step is to convert the webpage into an image that can be further processed as per our need. Since most webpages do not fit into a standard screenshot, we used Python QtWebkit to capture a view of the entire webpage. Nowadays, almost all popular browsers provide APIs to convert webpage into image. Other relevant commercial software products can also be used. Experiment with a few webpages indicated that the Portrait orientation is to be preferred over Landscape while capturing the image of a webpage.

3.2 Image preprocessing

The segmentation method that we have used is based on Edge Detection – an Image Processing based technique to detect areas in images with sudden change in

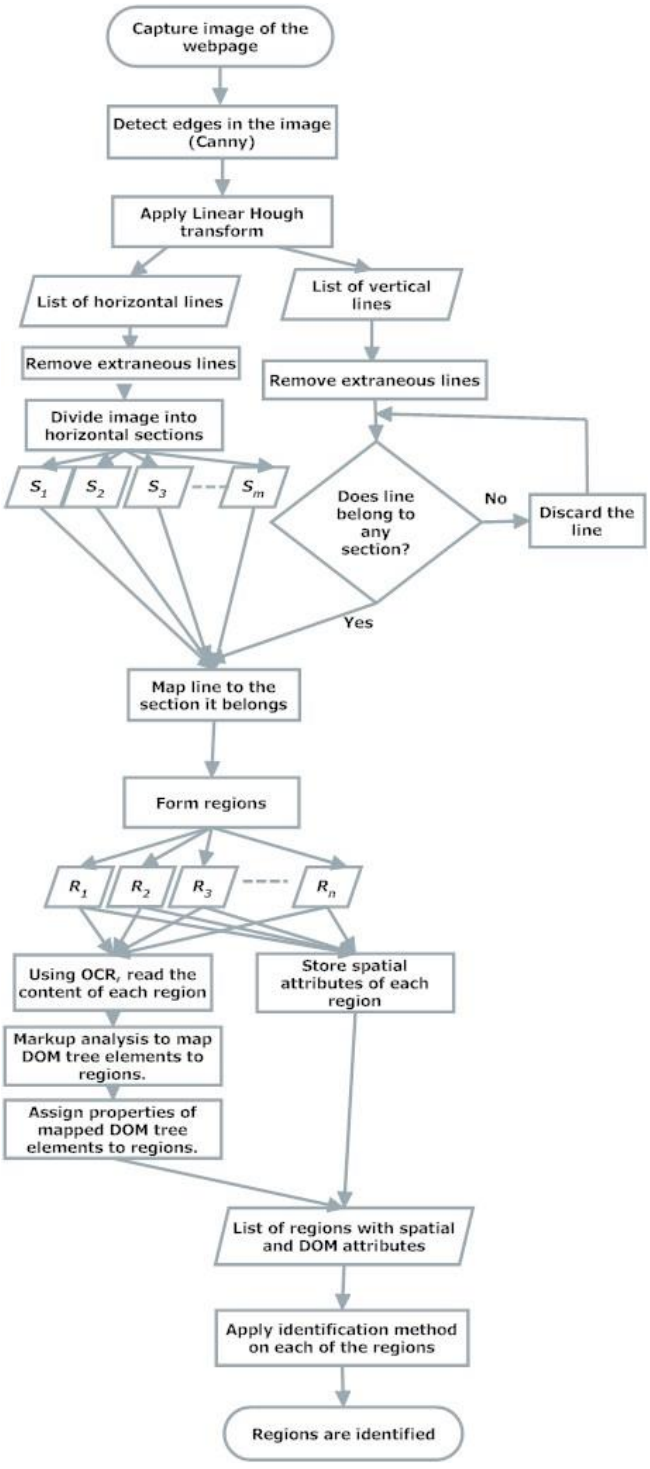


Figure 3. Work flow of the proposed method

brightness. It reduces the amount of data in an image and preserves only the important ones for further processing. A good edge detector should maximize the probability to detect 'real' edges and minimize the probability to detect 'unreal' edges. The captured images are often complex and are usually in RGB format. Using edge detection, we convert the image into grayscale digital image, which reduces the amount of data in an image but preserves the important features needed for further processing. There are several existing techniques for detecting edges [22] in an image like Sobel, Kayyali, Robert Cross, Prewitt, etc., yet Canny [5] yielded the best results in our case.

The image is converted into a grayscale digital image i.e. each pixel of image carries only intensity information varying from black at the weakest intensity to white at the strongest. Based on the intensities of pixels, the image is stored in a homogeneous multidimensional array. Since the Canny edge detector is susceptible to noise present in raw unprocessed image data, a Gaussian filter [10] is used. This results in a slightly blurred version of original image that is noise resistant to a significant level. Gaussian blur is applied on the original image mainly to remove the less prominent edges in the image which are not of much significance. Later, noise removal and smoothening of image is done using morphological transformations like Erosion, Dilation and Morphological Gradient [18].

Edges are places in an image where sudden rise or fall in intensity occurs. Finding the absolute gradient of the image intensity at each point helps us to identify edges in the image. We have used the Sobel operator [30], which returns the first derivative values in

the horizontal direction (S_x) and the vertical direction (S_y). These values help in determining the edge gradient and direction using Equations (1) and (2).

$$|S| = \sqrt{S_x^2 + S_y^2} \quad (1)$$

$$\theta = atan2(S_y, S_x) \quad (2)$$

where $|S|$ is edge gradient and *atan2* is arctangent function. The edge direction angle is further rounded to represent one of the four directions vertical, horizontal, left diagonal and right diagonal.

Following this, we trace the edge in the edge direction and suppress any pixel value that is not considered to be an edge. Finally, two thresholds – a high (T_1) and a low (T_2) are chosen. Any pixel in the image that has a gradient value greater than T_1 is presumably marked as an edge pixel. Starting from these edge pixels and using the directional information derived earlier, edges can be traced through the image. While tracing an edge, we consider T_2 for deciding an edge pixel, which allows us to trace faint sections of the edge as long as we have a starting point. Thus, we obtain a complete binary image where each pixel is marked as either an edge pixel or non-edge pixel. Figure 4 demonstrates the Canny edge detection over the image of a sample webpage [28].

Edge detection results in numerous edges, both linear and non-linear. Since, we want to segment the image into rectangular blocks, we need to consider only linear edges and discard non-linear ones. Linear Hough transform [11] is used for this purpose. Hough transform works on a Boolean matrix image i.e., the color in the image should be either

black or white (no grayscale). This is the reason to undergo Canny edge detection over the original image beforehand.

Considering the case where straight lines exist in an image, we notice that for every point (x_j, y_j) in the image, all the straight lines that pass through that point satisfy Equation (3) for varying values of slope (m) and intercept (c) as shown in the Figure 5.

$$y_j = mx_j + c \tag{3}$$

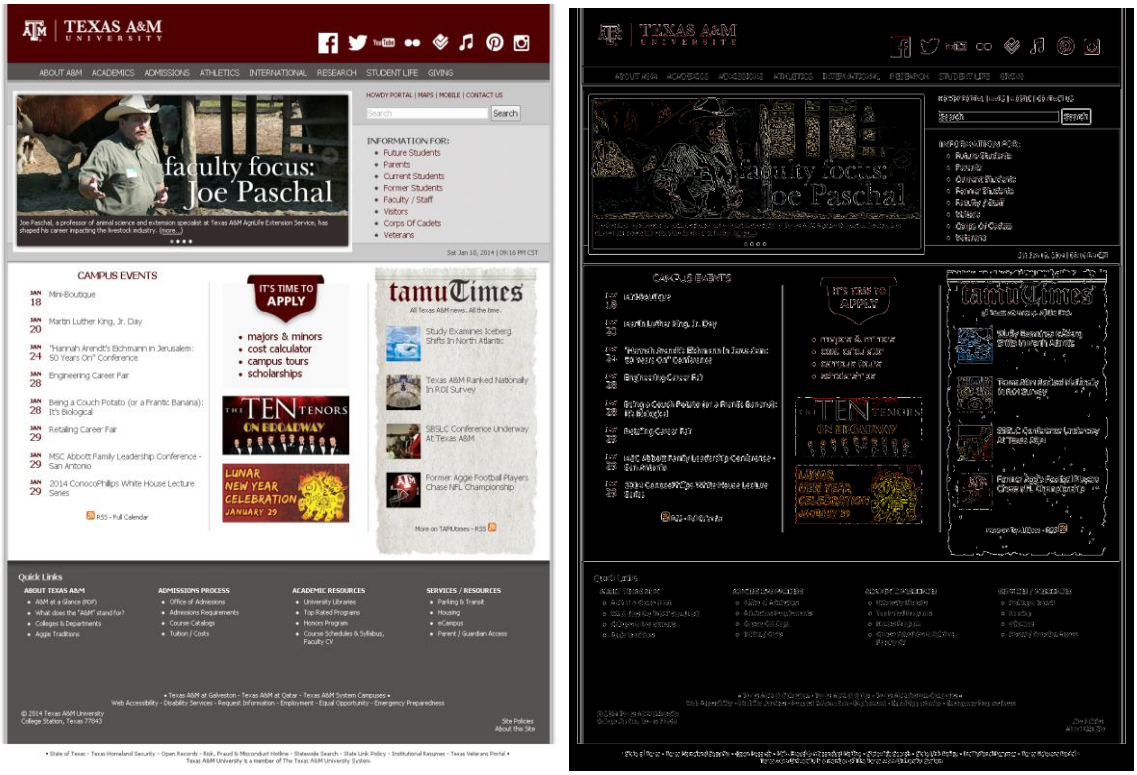


Figure 4. Result of Canny edge detection over image of the sample webpage [28]

Interestingly, if we reverse the variables and consider the values of slope and intercept as a function of the image point coordinates (x_j, y_j), then Equation (3) turns out to be Equation (4), which also represents a straight line as shown in Figure 6.

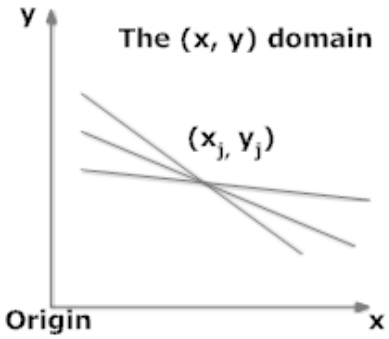


Figure 5. Straight lines passing through a point in the image in (*x*, *y*) domain

$$c = y_j - mx_j \tag{4}$$

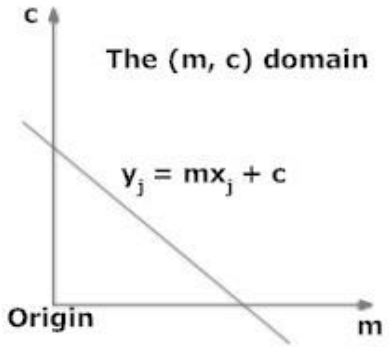


Figure 6. Alternate representation of all straight lines passing via a point in the image in (*m*,*c*) domain

Thus, we can represent all the possible lines through a pixel by a single line in the (*m*, *c*) space. Figure 7 shows all the pixels lying on the same line in the (*x*, *y*) space are represented by lines that pass through a single point in the (*m*, *c*) space.

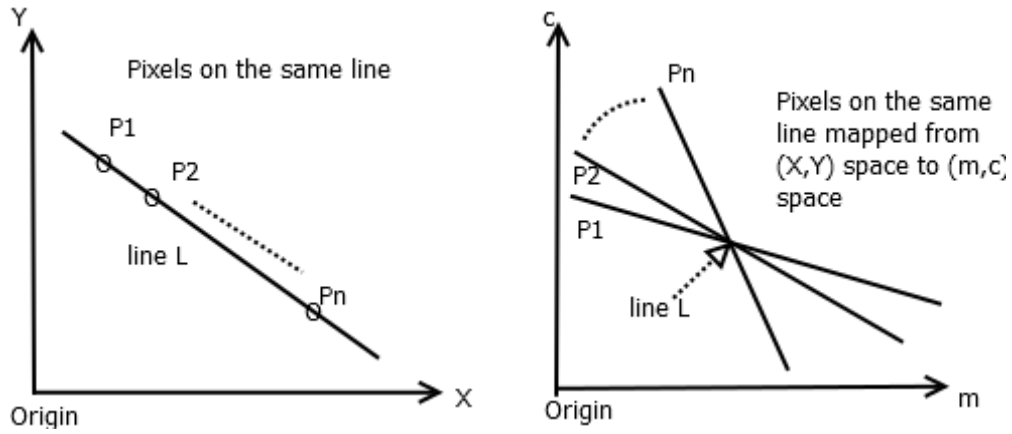


Figure 7. Representation of all pixels lying on the same line in the image

However, for vertical lines in the image, the value of m is infinite. So, in order to avoid this problem, Equation (3) is alternatively formulated as Equation (5) and shown in Figure 8.

$$x\cos\theta + y\sin\theta = c \tag{5}$$

We consider a two-dimensional array called Accumulator in order to store the quantized values of *r* and θ. The whole Boolean matrix image that we obtained after Canny edge detection is scanned. If enough evidence of a straight line is observed at a pixel and its neighborhood, the parameters (r, θ) of that line are calculated. Further the calculated

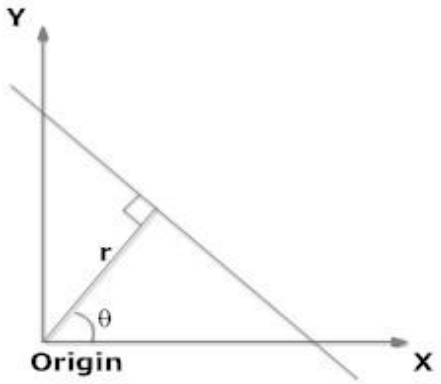


Figure 8. Representation of a line in the image with polar coordinates

parameters are looked up in Accumulator's bin and the values of that bin is incremented. The most likely lines along with their geometric definitions are extracted by looking for local maxima in the accumulator space. Accumulator threshold parameter (σ) is set in order to consider lines corresponding to a bin with value greater than that of the threshold. Since we need only horizontal and vertical lines in order to segment the image into rectangular boxes, so we filter lines with rotation angles 0 and $\pi/2$ as horizontal and vertical lines respectively

3.3 Image segmentation

Edges are places in an image where sudden rise or fall of intensity occurs. Therefore, the part of the image bounded by edges represents a visually cohesive block. However, a block can be too small to be semantically identified in the webpage. This thesis introduces a promising method to segment the webpage into visually and semantically identifiable cohesive blocks. A representation of two-dimensional image *I(x,y)* with width

Wt and height *Ht* is shown in Figure 9. We define terminology that forms the base of the proposed segmentation methodology:-

Definition 1 *Region*. Region is a visually and semantically cohesive block contained within the image. It is represented as $R_i = \{x_i, y_i, w_i, h_i\}$, where (x_i, y_i) is the top-left corner of region, w_i and h_i are width and height of the region respectively. Mathematically, we divide an image I into n regions – $R_1, R_2, R_3, \ldots, R_n$ such that each region R_i is a subset of I. Moreover, these regions should be

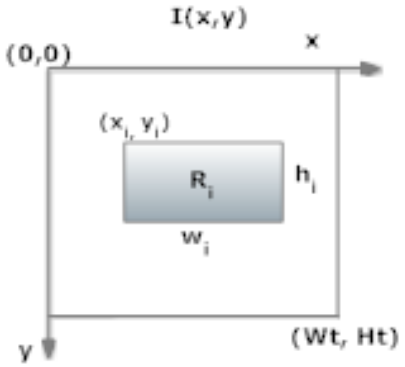


Figure 9. Region R_i contained within image I

mutually exclusive and exhaustive in nature for the given image as demonstrated by Equations (6) and (7).

$$R_1 \cup R_2 \cup R_3 \ldots \cup R_n = I \tag{6}$$

$$R_i \cap R_j = \emptyset, \ \forall\, i,j \ such \ that \ i \neq j \ and \ R_i, R_j \subseteq I \tag{7}$$

Definition 2 *Section*. Section is a part of the image that contains one or more regions of the same height.

Definition 3 *Hyperlink text density*. It is the ratio of hyperlinked text length to the total length of the text contained in a region.

After applying linear Hough transform over the Boolean matrix image (color in the image is either black or white) obtained as a result of Canny edge detection over the original image, we get horizontal and vertical edges present in the image. In order to conveniently work with these edges, we represent them in Cartesian form (x_1, y_1, x_2, y_2), where (x_1, y_1) and (x_2, y_2) are the start and end points of the line respectively. Then the horizontal and vertical edges are stored as H and V respectively.

Generally, we find that webpages are often divided into horizontal sections. Very often we come across webpages where a header section is placed at the top and a footer section at bottom occupying the entire width of the webpage. We observe similar behavior for top navigation bar. So, simply recognizing horizontal lines of the image can divide the entire webpage into some meaningful horizontal sections. In case the webpage layout is designed in a way that entails only vertical division of the page and no horizontal line of sufficient length exists that can significantly divide the page into horizontal segments, we still have the top and bottom borders of the image to keep our hypothesis intact. In order to eliminate the extraneous horizontal lines that cannot contribute to the broad-level horizontal division of the webpage, we define a threshold value η, which is computed in terms of percentage of the horizontal line of maximum length (h_{max}). h_{max} is computed as:

$$h_{max} = \max\{(x_2 - x_1);\ x_1, x_2 \in l_i;\ \forall\ l_i \in H\ \} \quad (8)$$

$$\eta = \beta * h_{max}\ ; where\ \ 0 < \beta < 1 \quad (9)$$

The value of β is adjusted empirically by starting with a random value, observing the result in the dataset, and then tuning lines belonging to *H* whose lengths are smaller than η are removed. However, there is a possibility that a few of the lines can be so close to each other that the existence of a meaningful region between them is improbable. Therefore, we need to substitute such lines with a single line. In order to tackle such a situation, we define δ_1 as the minimum allowable distance between two horizontal lines, which is expressed in terms of percentage of the height of the image. The value of δ_1 is adjusted empirically in accordance with the dataset under consideration. Thus, *H* is updated again based on the value of δ_1 as:

$sort\ lines\ in\ H\ basing\ value\ of\ y_1\ for\ each\ line$

$temp = H[0][1]$

$for\ each\ line\ l\ (except\ the\ first\ one)\ in\ H$:

    $if\ abs(temp - l[1]) <\ \delta_1$:

        $remove\ l\ from\ H$

    $else$:

        $temp = l[1]$

The presence of lines contained in *H* corresponding to the image of a sample webpage is shown in Figure 10. The image is divided into sections using the lines of *H*. Figure 11 illustrates the creation of different sections in the image.

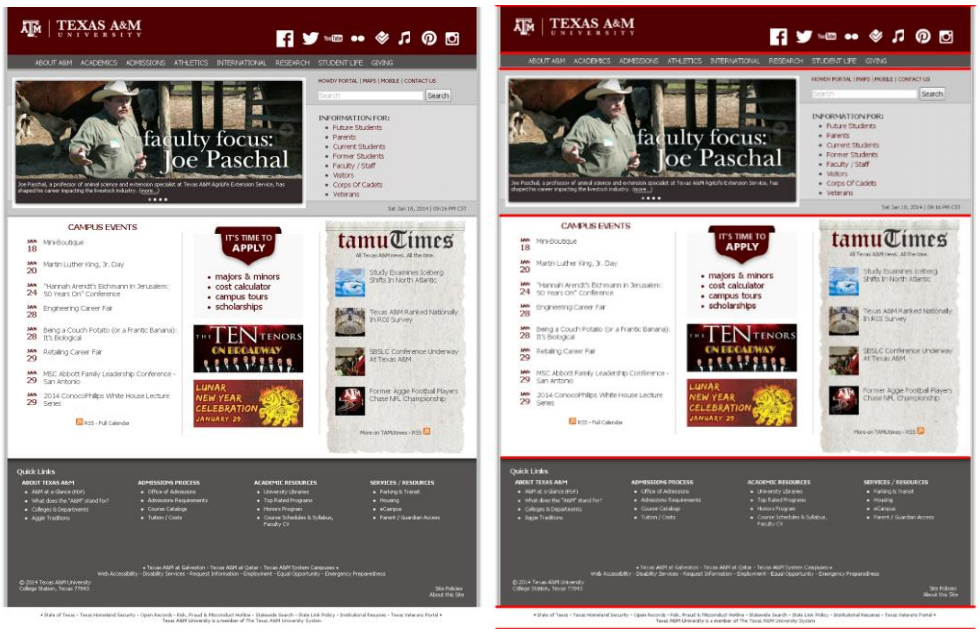


Figure 10. Horizontal lines after removing extraneous lines from image of the sample webpage [28]

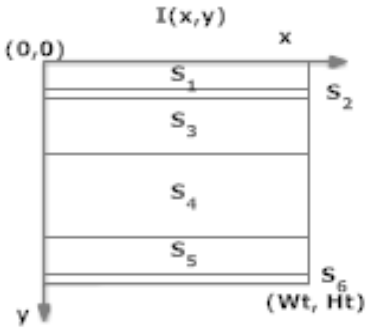


Figure 11. Division of Figure 10's image into sections

While creating sections using horizontal lines, we may need to make slight adjustments in the length of horizontal lines in order to avoid the formation of thin vertical stripes. We consider each of these sections individually. Within each section, we look for the existence of vertical lines. Suppose, V_i represents the vertical lines that belong to section S_i. We define a threshold value μ for vertical lines in terms of the percentage of the height of the section. This is done to void the significance of extraneous vertical lines generated, which is possible because of edge detection issues.

$$\mu = \alpha * height\ of\ S_i\ ; where\ 0 < \alpha < 1 \quad (10)$$

Again, the value of α is to be adjusted according to the dataset. Lines having length less than μ are eliminated from V_i. Also, in order to get rid of cases where two vertical lines are so close to each other that existence of meaningful region between them is improbable, we define δ_2 as the minimum allowable distance between two vertical lines and V_i is updated in the same fashion as H. δ_2 is defined in terms of the percentage of the width of the image and its value is determined empirically based on the dataset under consideration. Vertical lines present in updated V_i are extended to fit the height of the section. A common observation is that there exists a significant difference in the way horizontal and vertical regions of webpages are designed. In general, we observe that two horizontal lines, in spite of being very close to each other may contain a meaningful region like a horizontal navigation bar, a copyright section at the bottom of the webpage, etc. On the other hand, vertical lines need to be further apart in order to contain a meaningful region between them. Therefore, we have introduced separate tolerance limits (δ_1 and δ_2) for horizontal and vertical lines.

Finally, using vertical lines contained in V_i, S_i is further divided into even finer components (i.e., regions). Figure 12 shows the formation of regions using vertical lines present in a section. Figure 13 demonstrates the result of the proposed image segmentation method on the sample webpage.

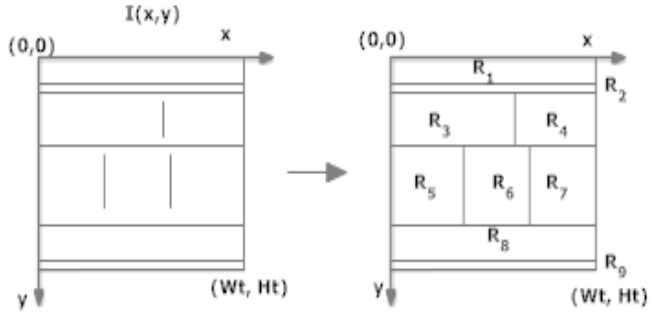


Figure 12. Sections segmented into regions

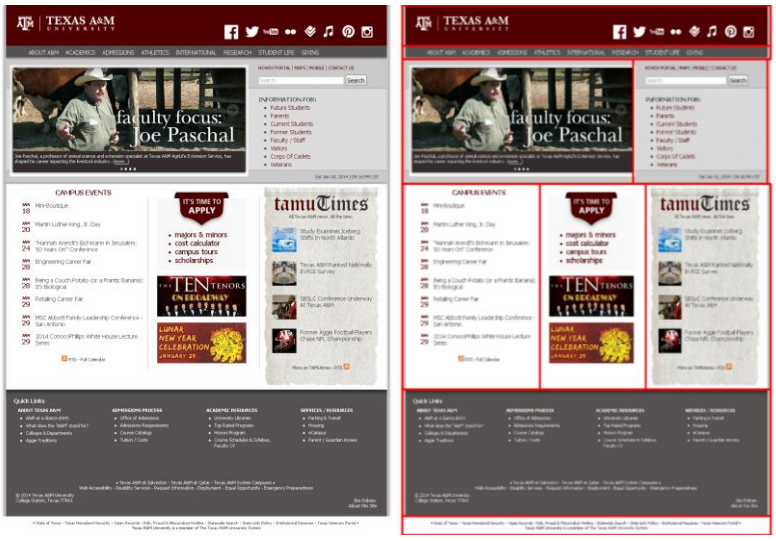


Figure 13.Segmentation result of image of the sample webpage [28]

Since we have coordinates of each of the segmented regions of the image, so we crop each of these regions separately in order to analyze each of these regions independently. Figure 14 shows cropped segmented regions of the sample webpage under consideration.

3.4 Text extraction from image

Each of the segmented regions are cropped out based on their coordinates in the image. Further, considering each of the segmented regions individually, using Optical Character Recognition (OCR), we extract the text embedded in the region. In our case, we have used Tesseract-OCR [15], which is an open-source OCR engine.

I.

II.

III.

IV.

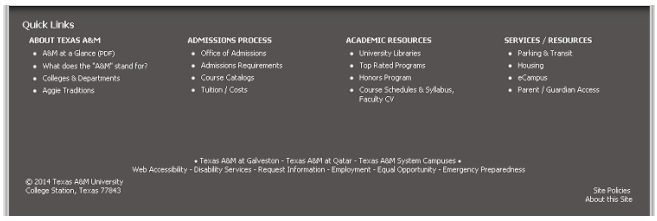
Figure 14. Cropped segmented regions of image of the sample webpage [28]

V.

VI.

VII.

VIII.



IX.

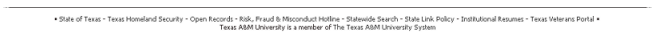


Figure 14. Continued.

3.5 Markup analysis and mapping

Each webpage corresponds to a DOM tree where tags with their attributes are internal nodes and the detailed text and images are leaf nodes. We traverse the DOM tree and visit all its leaf nodes leading to two possibilities:

Case 1. Leaf node contains an image *(img)*: In this case, we iterate over the segmented regions and check which one of these contains *img*. The OpenCV [18] library provides APIs to determine whether a given image is contained within another image. Once the region that contains *img* is determined, the attributes related to *img* is assigned to that region.

Case 2. Leaf node contains text *(txt)*: we iterate over the texts corresponding to each of the regions that we have extracted using OCR. We verify *txt* as the substring of the extracted texts of various regions. In order to cope up with the inaccuracy of OCR tool being used, a text match of 75% or above is considered to be a perfect match. The region containing *txt* is associated with the attributes of *txt*.

However, there is a possibility that *txt* or *img* can be contained within more than one of the regions. In order to tackle such a situation, we use the spatial information of the regions that we have obtained after segmenting the webpage. Corresponding to each of the regions, we have the coordinates of their corner points. Also, we have the relative position of each region with respect to other regions in the image. Analysis of CSS of the webpage helps in finding the spatial attributes of a DOM element. The spatial attributes of the DOM element and segmented regions help us to map the element to a region. Moreover, we have

followed the top-down approach while mapping the DOM elements and segmented regions. This avoids the worst case scenario, in more than 80% of the cases, which is two or more DOM elements containing the same text/image and having the same styling properties.

3.6 Region identification

Complete traversal of the DOM tree leads to an associative mapping between various regions and their relevant properties like font size, font weight, text length, hyperlink text length, etc. These properties along with the spatial attributes of the regions help us to classify each of these regions into various categories like top navigation, side navigation, body, header, footer, and so forth. We have followed two approaches to identify regions: rule-based and classifier-based.

3.6.1 Rule-based approach

Different regions of a webpage are distinguished based on their visual cues and spatial attributes. Based on the analysis of the ground-truth-established webpages of dataset, we empirically came up with a few rules to identify a region in a webpage.

Rule 1. *Header* is the topmost segmented region which covers more than two-third of the entire width of the page. Also, the height of the segment should not be more than one-fifth of the total height of page. Presence of text with larger font-size and "bold" font-weight emphasizes a region to be a Header.

Rule 2. *Footer* is the bottommost segmented region which covers more than two-third of the entire width of the webpage and the height of the segment should not be more than one-fifth of the total height of the page.

Rule 3. *Navigation* is a segmented region that has hyperlink-text-density greater than 0.80.

Rule 4. *Top-nav* is a Navigation region that covers at least two-third of the entire width of the page and is placed in the upper half of the page. The width of such region is greater than the height.

Rule 5. *Bottom-nav* is a Navigation region that covers at least two-third of the entire width of the page and is placed in the lower half of the page. The width of such regions is greater than the height.

Rule 6. *Left-nav* is a Navigation region that is located in the left half of the page. The height of such region is greater than the width.

Rule 7. *Right-nav* is a Navigation region that is located in the right half of the page. The height of such region is greater than the width.

Rule 8. If a region contains one or more images, we have labeled it as "Contains image".

Rule 9. When a region contains neither any text nor any image, we consider that as a blank region and do not consider it for any kind of assessment. Occurrence of such regions in a webpage is less probable. In case, if blank regions exist in a webpage, they are usually located at the boundary of the webpage.

Rule 10. When a non-blank region fails to establish any mapping with any of the DOM tree elements of the webpage, we consider the region as *Dynamic*. A Dynamic region can be anything such as advertisement, video, RSS feed, etc.

Rule 11. All the non-blank regions are considered to be a *Body* if they do not belong to Header, Footer and Navigation classes. Moreover, a Dynamic region is considered as a Body only if it is located in central part of webpage.

Based on the above mentioned rules, a region is classified as a Header, Footer, Top-navigation, Bottom-navigation, Left-navigation, Right-navigation, Body, Dynamic, or a Contains Image class. Moreover, a region may belong to more than one class.

3.6.2 Classifier-based approach

We divided our dataset into two parts - training set and testing set. Based on the ground truth of each of the pages of the training set, we extracted all the features associated with each region and fed that into a classifier. We have used a Support Vector Machine (SVM) classifier [24] for classifying regions into various classes.

SVM [2] is a classification algorithm developed based on statistical theory and is proven to be one of the most successful classifiers in various domains. In the context of high dimensional space, different classes are distinguished by constructing a decision surface known as hyperplane. There is a possibility of the existence of several such hyperplanes. So, the challenge lies in finding out the optimal hyperplane which maximizes the marginal difference between the classes under consideration. In fact, larger margin leads to lower generalization error of the classifier. The optimal hyperplane is obtained using the data points obtained from the training set. During testing phase, distance of the newly introduced data point(s) from the testing set is computed from the optimal hyperplane. Based on the computed distance, classification of data point is done.

We have used Radial Basis Function (RBF) as the kernel function for SVM in our research, which needs two important parameters (C and γ) to be defined. The associated cost with the presence of a data point on the wrong side of hyperplane is reflected by C and the range of RBF kernel is decided by γ.

There are 9 features that we have used for classification—font size, font weight, text length, hyperlink text length, top-offset, bottom-offset, right-offset, left-offset and whether the region contains an image. Top-offset and bottom-offset are defined as the percentage of the total height of the webpage's image. On the other hand, left-offset and right-offset are defined as the percentage of the total width of webpage's image. Figure 15 demonstrates the consideration of top-offset, bottom-offset, left-offset and right-offset corresponding to a given region R_i. After training the classifier with the regions contained within the webpages of training-set, we introduced the features of segmented regions from the webpages of testing-set to the SVM classifier to classify each region of all the webpages of testing-set.

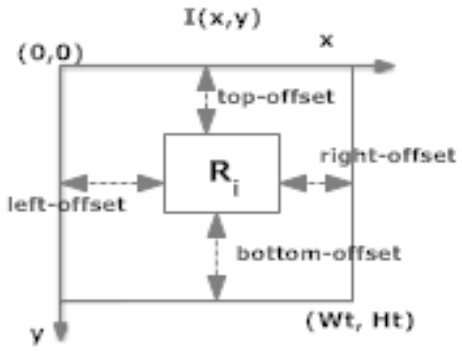


Figure 15. Spatial features of a region to be used for classification

4. EXPERIMENT RESULTS

The proposed methodology is implemented in Python-2.7 using the Eclipse SDK, version 4.2.1. The OpenCV library is used for Canny edge detection and Hough transform.

4.1 Dataset

Web pages of existing paths from *Walden's Paths* [6] were used for testing the proposed work. We collected 150 web pages from different paths. Out of these 150 web pages, 50 pages were used for training (the training set) and the remaining 100 pages were used for evaluation (the testing set). The corpus included web pages with a wide range of visual and structural layouts, which were used for the unbiased assessment of the proposed work.

4.2 Evaluation

For each of the 150 webpages of the dataset, ground truth was generated. Two of our lab members, each having more than 4 years of experience in web technology, went through all the webpages in the dataset and annotated different sections present in each of the webpages as Header, Footer, Body, Top-nav, Bottom-nav, Left-nav, Right-nav, Dynamic, and Contains-Image. Moreover, we stored the properties of each of these marked regions. We used ground-truth-established pages of the training set for experimental purposes, which helped us to derive the values of different parameters to be used for segmenting the webpage's image into various regions. Table 1 shows the values of different parameters, which are obtained based on the adjustments made in accordance to our training set.

Table 1. Parameters and their values

Parameter	Value
α	0.30
β	0.20
δ_1	0.05 * Ht
δ_2	0.10 * Wt
T_1	350
T_2	80
σ	150
C	5.0
γ	0.11

We have divided the evaluation into three parts. Firstly, we have evaluated the accuracy of the segmentation algorithm used for dividing a webpage's image into various regions. Then, evaluation of rule-based method to identify the regions of a webpage is performed. At last, we have evaluated the classifier-based method for identifying the regions.

After running the segmentation algorithm over the testing set, we matched the segmented regions of each webpage with the available ground truth in order to find the accuracy of the proposed method. Based on the ground truth, if a webpage contained a top-navigation bar and the proposed method could not segment that, we inferred that the method failed to identify an identifiable top-navigation region, which was considered as a *miss* for the Top-nav region. If the method could segment that region, it showed that an identifiable top-navigation section was identified by the method proposed, which was considered as a *hit* for the Top-nav region. Figure 16 demonstrates the way different parts

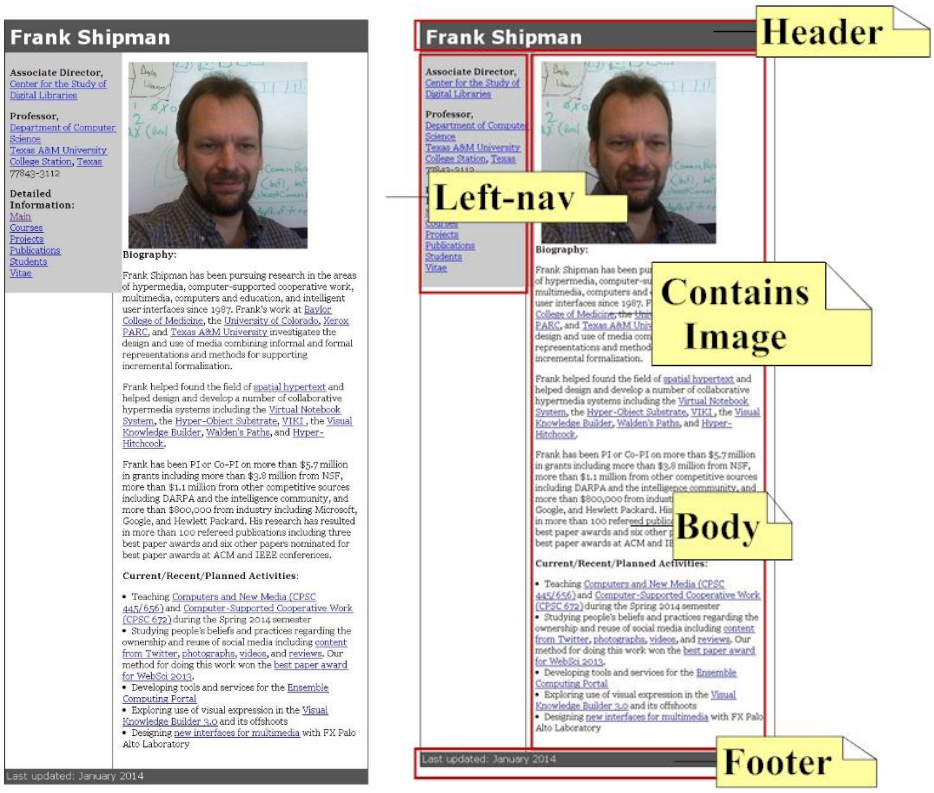


Figure 16. Ground truth generation [25]

of a webpage [25] were annotated and thus ground truth was generated corresponding to webpages of dataset. Further, Figure 17 substantiates the way proposed segmentation method is evaluated on top of established ground truth for a given webpage [25]. Following this approach of evaluation, we ran the segmentation algorithm over all the webpages of the testing-set and the result obtained is shown in Table 2, which shows that there are 93 headers present in the testing set and out of these 93 headers 83 are segmented properly. Out of 90 footers, 73 are segmented properly. Other rows can be interpreted in similar fashion.

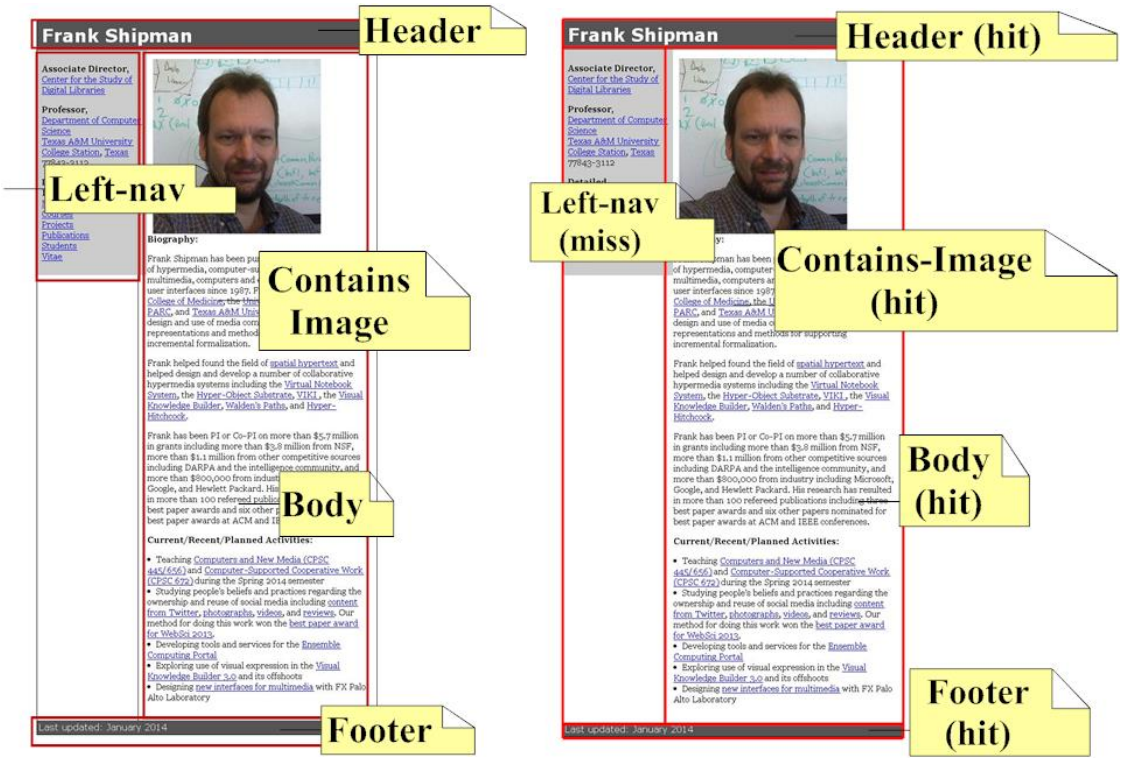


Figure 17. Evaluation method to compute hits and misses [25]

Table 2. Results of segmentation method

Region	No. of correctly segmented	Total in the testing-set
Header	83 (89.2%)	93
Footer	73 (81.1%)	90
Body	311 (83.2%)	374
Top-nav	59 (81.9%)	72
Left-nav	27 (61.4%)	44
Right-nav	20 (62.5%)	32
Bottom-nav	38 (66.7%)	57
Dynamic	69 (59.0%)	117
Contains image	271 (79.0%)	343
Overall	**951 (77.8%)**	**1222**

Figure 18. Accuracy of proposed segmentation method for different regions

We extended our evaluation by running the proposed rule-based method for identifying regions over the testing set. We considered only the correctly segmented regions of each of the webpages of the testing set. We compared the results obtained with the ground truth for each of these webpages. The results obtained are shown in Table 3.

Furthermore, we evaluated the classifier-based method for identifying regions using SVM classifier. Python's Scikit-learn library is used for classification using SVM. For SVM classifier, the value of C is set to be 5.0; the value of γ (*gamma*) as 0.11 i.e. one upon the number of features and RBF is used as the kernel function. Regions of ground-truth-established webpages of testing-set were used for training the classifier. The

proposed method was tested on the same testing-set, which was used for evaluating the rule-based method. The results obtained are shown in Table 4.

Table 3. Results of rule-based method to identify regions

Region	No. of correctly identified	No. of correctly segmented
Header	69 (83.1%)	83
Footer	58 (79.5%)	73
Body	199 (64.0%)	311
Top-nav	45 (76.3%)	59
Left-nav	18 (66.7%)	27
Right-nav	13 (65.0%)	20
Bottom-nav	27 (71.1%)	38
Dynamic	42 (60.9%)	69
Contains image	233 (86.0%)	271
Overall	**704 (74.0%)**	**951**

Table 4. Results of SVM-based method to identify regions

Region	No. of correctly identified	No. of correctly segmented
Header	73 (88.0%)	83
Footer	61 (83.6%)	73
Body	189 (60.8%)	311
Top-nav	46 (78.0%)	59
Left-nav	23 (85.2%)	27
Right-nav	14 (70.0%)	20
Bottom-nav	34 (89.5%)	38
Dynamic	33 (47.8%)	69
Contains image	237 (87.5%)	271
Overall	**710 (74.7%)**	**951**

Figure 19 compares the accuracies of rule-based and classifier-based methodologies for identifying regions in webpages. Using the classifier-based approach for identifying webpage regions showed significant improvement in the results. The results obtained using the rule-based method are satisfactory too. Moreover, the major advantage of rule-based approach is that it does not require any prior training in order to identify regions.

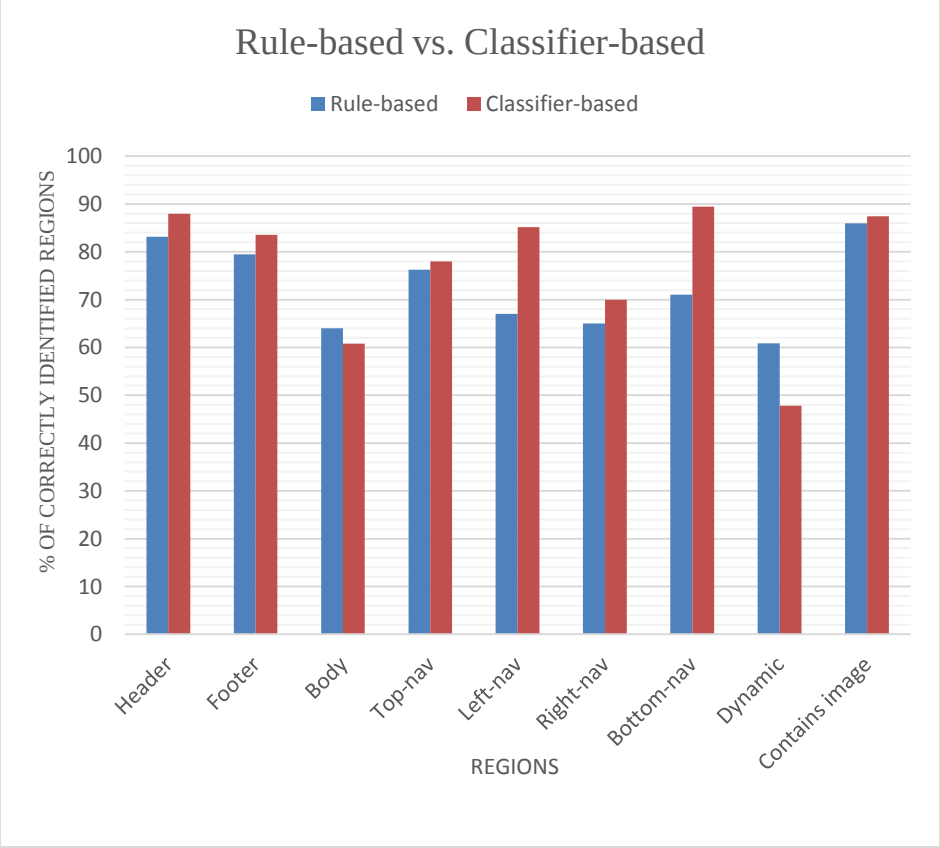


Figure 19. Comparing the accuracy of rule-based method vs. SVM-based method for identifying regions in a webpage

Based on our observation, we found that there is difference in the behavior of both the above mentioned approaches while identifying the regions. Some of the regions that were identified correctly by the rule-based method could not be identified correctly by SVM-based approach and vice-versa. Table 5 shows the results obtained when a region is identified correctly by rule-based and SVM-based approaches. These are the regions for which the behavior of rule-based approach and SVM-based approach are in sync.

Table 5. Results of correctly identified regions using both rule-based and SVM-based methods

Region	No. of correctly identified	No. of correctly segmented
Header	67 (80.7%)	83
Footer	57 (78.1%)	73
Body	168 (54.0%)	311
Top-nav	42 (71.2%)	59
Left-nav	17 (63.0%)	27
Right-nav	11 (55.0%)	20
Bottom-nav	26 (68.4%)	38
Dynamic	24 (34.8%)	69
Contains image	229 (86.0%)	271
Overall	**641 (67.4%)**	**951**

Moreover, it's an interesting idea to use both rule-based and SVM-based methods to identify the roles of different regions. In this case, we consider a region as correctly identified if the region is identified correctly by rule-based method or SVM-based method

or both. Results obtained with this approach is shown in Table 6. We see a drastic improvement in the accuracy of correctly identified regions as reported in Table 6.

Table 6. Results of correctly identified regions either using rule-based or SVM-based method

Region	No. of correctly identified	No. of correctly segmented
Header	75 (90.4%)	83
Footer	62 (84.9%)	73
Body	220 (70.7%)	311
Top-nav	49 (83.1%)	59
Left-nav	24 (88.9%)	27
Right-nav	16 (80.0%)	20
Bottom-nav	35 (92.1%)	38
Dynamic	51 (73.9%)	69
Contains image	241 (88.9%)	271
Overall	**773 (81.3%)**	**951**

5. CONCLUSION

Understanding the structure of web pages is important to a wide range of applications. This thesis presents a hybrid approach to segment web pages that combines image processing and markup analysis that achieved accuracy near 78% on a varied corpus. Once segments are recognized, understanding the role they play can further enhance applications. We presented and compared rule-based and SVM-based approaches to classifying web page segments. The two approaches performed similarly with 74-75% accuracy overall although each approach proved to be substantially better than the other for one or two classes of segments. Based on the behavior of rule-based and SVM-based approaches, we also presented the results for which both these methods are synchronized and results for the regions when either of these methods identified a region correctly.

While these results are promising, improving the OCR component of the system would improve segmentation results. Future work in role classification should both explore the effects of additional web page features on role classification performance and revisit the set of roles that are targets for classification.

Longer term, we plan to develop a similarity metric for web pages based on their structure in order to support the identification of important changes to web pages and the location of replacement pages when web pages go missing. We will need to experimentally determine how to weight the structural information that is provided by this form of analysis with the content-based metrics that have traditionally been explored as solutions to these problems.

REFERENCES

[1] Bar-Yossef, Z. and Rajagopalan S. Template detection via Data Mining and its Applications. In *Proceedings of the 11th International Conference on World Wide Web (WWW)*, 2002.

[2] Burges, C.J. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, pages 121-167, 1998.

[3] Cai, D., He, X., Ma, W.Y., Wen, J.R. and Zhung, H. Organizing WWW Images based on the Analysis of Page Layout and Web Link Structure. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, 2004.

[4] Cai, D., Yu, S., Wen, J. and Ma, W. Vips: a Vision-based Page Segmentation Algorithm, *Microsoft Technical Report*, 2003.

[5] Canny, J. A Computation Approach to Edge Detection. In *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, November, 1986.

[6] Center for the Study of Digital Libraries, Texas A&M University. Walden's Paths. http://www.csdl.tamu.edu/walden/index.html. [Online; Accessed: November, 2012].

[7] Chen, T. and Li, H. A Spectral Method for Image Co-segmentation. In *Proceedings of the International Conference on Computational Problem-Solving (ICCP),* October 2011.

[8] Comaniciu, D. and Meer, P. Mean Shift Analysis and Applications. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1197-1203, 1999.

[9] Debnath, S., Mitra, P. and Giles, C.L. Automatic Extraction of Informative Blocks from Webpages. In *Proceedings of the 2005 ACM Symposium on Applied Computing*, pages 1722–1726, 2005.

[10] Deng, G.; Cahill, L.W. An Adaptive Gaussian Filter for Noise Reduction and Edge Detection. In *Proceedings of the Nuclear Science Symposium and Medical Imaging Conference, IEEE Conference Record*, vol. 3, pages 1615 – 1619, 1993.

[11] Duda, R.O., Hart, P.E. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM*, vol. 15, no. 1, pages 11 – 15, January 1972.

[12] Felzenszwalb, P.F. and Huttenlocher, D.P. Efficient Graph-based Image Segmentation. *International Journal of Computer Vision*, vol. 59, no. 2, pages 167 – 181, September 2004.

[13] Feng, H.M., Liu, B., Liu, Y.M., Fang, Y. and Song, G. Framework of Webpage Analysis and Content Extraction with Coordinate Trees. *J Tsinghua Univ (Sci & Tech)*, vol. 45, pages 1767–1771, 2005.

[14] Finn, A., Kushmerick, N. and Smyth, B. Fact or Fiction: Content Classification for Digital Libraries. In *Proceedings of the Joint DELO-NSF Workshop on Personalization and Recommender Systems in Digital Libraries*, Dublin, 2001.

[15] Google. Tesseract-OCR. https://code.google.com/p/tesseract-ocr/. [Online; Accessed: December, 2013].

[16] Gupta, S., Kaiser, G., Neistadt, D. and Grimm, P. DOM based Content Extraction of HTML Documents. In *Proceedings of the 12th World Wide Web conference (WWW)*, Hungary, 2003.

[17] He, X., Cai, D., Wen, J.R., Ma, W.Y., and Zhang, H.J. ImageSeer: Clustering and Searching WWW Images Using Link and Page Layout Analysis, *Microsoft Research Technical Report*, 2004.

[18] Itseez. Open Source Computer Vision. http://opencv.org/. [Online; Accessed: September, 2013].

[19] Jiuxin, C., Bo, M. and Junzhou, L. A Web Page Segmentation Algorithm Based on Iterated Dividing and Shrinking. In *Proceedings of the IFIP International Conference on Network and Parallel Computing Workshops*, September 2007.

[20] Li, Y., Meng, X., Li, Q. and Wang, L. Hybrid Method for Automated News Content Extraction from the Web. In *Proceedings of the 7th International Conference on Web Information Systems Engineering*, pages 327–338, 2006.

[21] Lin, S.H. and Ho, J.M. Discovering Information Content Blocks from Web Documents. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2002.

[22] Maini, R. and Aggrawal, H. Study and Comparison of Various Image Edge Detection Techniques. http://wwwmath.tau.ac.il/~turkel/notes/Maini.pdf. [Online; Accessed: December, 2014].

[23] Peng, B., Zhang, L. and Zhang, D. Automatic Image Segmentation by Dynamic Region Merging. In *Proceedings of the IEEE Transactions on Image Processing*, vol. 20, no. 12, December 2011.

[24] Scikit-learn. Support Vector Machines. http://scikitlearn.org/stable/modules/svm.html. [Online; Accessed: January, 2014].

[25] Shipman, Frank. Frank Shipman Profile Page. http://www.csdl.tamu.edu/~shipman/. [Online; Accessed: January, 2014].

[26] Spengler, A. and Gallinari, P. Document Structure meets Page Layout: Loopy Random Fields for Web News Content Extraction. In *Proceedings of the 10th ACM symposium on Document engineering (DocEng)*, New York, 2010.

[27] Takagi, H., Saito, S., Fukuda, K., and Asakawa, C. Analysis of Navigability of Web Applications for Improving Blind Usability. In *Proceedings of the ACM Transactions Computer-Human Interaction*, September 2007.

[28] Texas A&M University. Texas A&M University Home Page. http://www.tamu.edu. [Online; Accessed: January, 2014].

[29] Vieira, K., Silva, A.S., Pinto, N., Moura, E.S., Cavalcanti, M.B., Jo and Freire, J. A Fast and Robust Method for Web Page Template Detection and Removal. In *Proceedings of the 15th ACM international conference on Information and knowledge management (CIKM)*, pages 258–267, New York, 2006.

[30] Wikipedia. Sobel Operator. http://en.wikipedia.org/wiki/Sobel_operator. [Online; Accessed: October, 2013].

[31] Yi, F. and Moon, I. Image Segmentation: A Survey of Graph-cut Methods. In *Proceedings of the International Conference on Systems and Informatics (ICSAI),* May 2012.