Informative Content Extraction By Using Eifce [Effective Informative Content Extractor]

Chaw Su Win, Mie Mie Su Thwin

Abstract: Internet web pages contain several items that cannot be classified as the "informative content," e.g., search and filtering panel, navigation links, advertisements, and so on. Most clients and end-users search for the informative content, and largely do not seek the non-informative content. As a result, the need of Informative Content Extraction from web pages becomes evident. Two steps, Web Page Segmentation and Informative Content Extraction, are needed to be carried out for Web Informative Content Extraction. DOM-based Segmentation Approaches cannot often provide satisfactory results. Vision-based Segmentation Approaches also have some drawbacks. So this paper proposes Effective Visual Block Extractor (EVBE) Algorithm to overcome the problems of DOM-based Approaches and reduce the drawbacks of previous works in Web Page Segmentation. And it also proposes Effective Informative Content Extractor (EIFCE) Algorithm to reduce the drawbacks of previous works in Web Informative Content Extraction. Web Page Indexing System, Web Page Classification and Clustering System, Web Information Extraction System can achieve significant savings and satisfactory results by applying the Proposed Algorithms.

Index Terms: Informative Content Extraction, Main Content Extraction, Web Page Segmentation

1 INTRODUCTION

THE World Wide Web serves as a huge, widely distributed, global information service center for news, advertisements, consumer information, financial management, education, government, e-commerce, and many other information services. However, Internet web pages typically contain a large amount of non-informative content such as advertisements, search and filtering panel, headers, footers, navigation links, and copyright notices, etc. Most clients and end-users search for only the informative content and the need of Informative Content Extraction from web pages becomes evident. To extract the informative content of the web page correctly, the informative content and the non-informative content of the web page must be known clearly. The informative content is the main content of the web page that gives some information to the user e.g., articles about technology, health or education, etc. The non-informative content of the web page contains fixed description noise such as site logos, copyright notices, privacy statements, etc., service noise, irrelevant services, such as the weather, stock or market index, etc., navigational links, advertisements, header, footer and so on. To distinguish between the informative and non-informative content in a web page, it needs to segment the web page into semantic blocks. There are several kinds of methods for Web Page Segmentation. The most popular ones are DOM-based Segmentation [2], Location-based Segmentation [3] and Vision-based Page Segmentation [5], [6]. These methods are distinguished from one another by considering various factors as the partition basis. Actually in the DOM-based Segmentation Approach, DOM tends to reveal presentation structure other than content structure, and is often not accurate enough to discriminate different semantic blocks in a web page. In the Location-based Segmentation Approach, a kind of layout template cannot be fit into the layouts of all web pages. So, Vision-based Approaches have been used for Effective Web Page Segmentation. However, these Approaches also have some drawbacks. And a good Vision-based Web Page Segmentation is urgently required. So this paper proposes Effective Visual Block Extractor (EVBE) Algorithm to reduce the drawbacks of previous works in Web Page Segmentation. A web page structure and layout varies depending on different content types and the presentation style designed by the web developer. Thereby informative content positions of the web pages differ in variety of websites. Even there might be some content in page view that are beside each other but actually in DOM tree they are not in the same level and same parent. So finding the informative content in this area needs complicated and costly algorithms. Almost all algorithms have been proposed are tag dependent. They could only look for informative content among specific tags such as <TABLE> or <DIV>. Algorithms that could simulate a user visiting a website have high probability for extracting informative content as result. So this paper proposes Effective Informative Content Extractor (EIFCE) Algorithm to automatically extract Informative Content Block from web pages effectively. The Proposed Algorithm simulates a web page user visit and how the user finds the informative content block in the page. The Proposed Algorithm is intended for extracting the informative content of the web page effectively. The rest of this paper is organized as follows. Section 2 introduces an overview on the literature review. In Section 3, the background theory of the Proposed Approach is described. The Proposed Approach for Web Informative Content Extraction is presented in Section 4. Section 5 proposes Effective Web Page Segmentation Approach. Effective Informative Content Extraction is presented in Section 6. Section 7 provides Evaluation of the Proposed Approach. Finally, concluding remarks are expressed in Section 8.

- *Chaw Su Win is currently pursuing Ph.D. degree program in information technology in University of Technology (Yatanarpon Cyber City), Myanmar. E-mail: chawsuwin1982@gmail.com*
- *Mie Mie Su Thwin is currently working at Myanmar Computer Emergency Response Team in MPT, Myanmar. E-mail: drmiemiesuthwin@mmcert.org.mm*

2 LITERATURE REVIEW

VIPS algorithm [6] uses visual cues to produce content structure from DOM structure and with this content structure it fills the gap between DOM structure and the conceptual structure of the web page. The algorithm uses obtained content structure and tries to simulate how actual user finds the main content by blocking the page based on structure and visual delimiters. This algorithm mainly depends on visual separators although in some cases visual separators are misleading or ambiguous. It does many loops to reach its desire granularity. Vision-based Page Segmentation (VIPS) algorithm needs some improvements in its most important part, Visual Block Extraction. So, Extended VIPS Algorithm [18] defines additional terms and detects visual cues for extending Visual Block Extraction part. In this technical report, deficiencies of VIPS algorithm are explained, and new rules are defined. NEWIE [14] is proposed as a new approach to eliminate the noise in web pages for improving Informative Content Extraction. The approach is DOM-based to eliminate the noise and defines the rules which are noise in web page based on the observation of the noise which are normally located in web page. Info Discoverer Algorithm [4] is proposed to efficiently and automatically discover intra-page redundancy and extract informative contents of a page. The researchers concentrated on HTML documents with <TABLE> tags. Based on HTML tag <TABLE>, a page is partitioned into several content blocks. Based on DOM, a coarse tree structure is obtained by parsing an HTML page based on <TABLE>. Each internal node shows a content block containing one or more content strings as its leaf nodes. After parsing a page into content blocks, features of each block are extracted. Here features mean the meaningful keywords. After extracting features, entropy value of a feature is calculated according to the weight distribution of features appearing in a page cluster. Next step is calculation of entropy value of a content block. It is given by summation of its features entropies. The entropy of a content block is the average of all entropy values in that block. By using this, a content block is identified as informative or redundant. If the entropy value is higher than a threshold or close to 1, the content block is redundant as most of the block features appear in every page. If it is less than a threshold, the content block is informative as features of the page block are distinguishable from others. Site Style Tree (SST) [7] considers non-content blocks as local noise in the web page. A tree structure is used to capture common presentation styles and actual contents of the pages in the given web site. A Style Tree can be built for the site by sampling the pages of the site. Each HTML page corresponds to a DOM tree where tags are internal nodes and the detailed texts, images or hyperlinks are the leaf nodes. A DOM tree can represent the presentation style of a single HTML page, but it is difficult to study the overall presentation style and content of a set of HTML pages. It is difficult to clean pages based on individual DOM trees. So a tree structure known as Style Tree is used for this purpose. Two importance values, presentation importance and content importance, are used to find importance of an element node. The presentation importance is used to detect noises with regular presentation styles while content importance is used to detect those main contents of the pages that may be represented in similar presentation styles. Hence, the importance of an element node is given by combining its presentation importance and content importance. CE [9], [10] considers all detailed pages of a website as pages with the same class. It runs a learning phase with two or more pages as its input and finds the blocks that their pattern repeats between input pages and marks them as non-informative blocks then stores them in storage. These non-informative blocks are mostly copyright information, header, footer, sidebars and navigation links. Then it eliminates non-informative patterns from the structure of its input pages based on the stored patterns in its storage for specific class of input pages. Finally from the remaining blocks in the page it will return the text of block containing the most text length. CE needs a learning phase so it couldn't extract the main content from random one input web page. FE [9], [10] extracts the text content of block that has the most probability of having text so it will work fine in web pages that text content of main content dominates other types of content. In addition FE could return just one block of the main content, so K-FE [9], [10] is proposed in order to return k blocks with high probability of having the main content. Algorithm steps of K-FE and FE are the same except the last part. In K-FE, the algorithm final section sorts the blocks depending on their probability then it uses k-means clustering and takes high probability clusters. CoreEx [11] is proposed as a heuristic technique for automatically extracting the main article from online news site web pages. It uses a DOM tree representation of each page, where every node in the tree represents an HTML node in the page. The amount of text and the number of links in every node are analyzed and a heuristic measure is used to determine the node (or a set of nodes) most likely to contain the main content. For every node in the DOM tree, two counts are maintained. textCnt holds the number of words contained in the node and linkCnt holds the number of links in or below the node. A node's score is a function of its textCnt and linkCnt. If two nodes reach identical scores, the node higher in the DOM tree is selected. NIT [12] is a new method for information extraction from web pages. The method is based on statistical analysis of web page content and intended mainly for text corpus making. The NIT method is an automatic statistical-based algorithm using the web page structure for information extraction. It transforms the source document into the hierarchical structure DOM tree. Each DOM node represents one web page element. The NIT method is based on detection of most useful nodes in the DOM tree. The useful nodes are included into the extraction result and they represent an ideal plain text extract in the best case. Generally most of the existing content extraction approaches that used heuristic rules introduced only one main feature to distinguish main content from noisy information. Combination of different features usually can lead to a better content extraction as different kinds of web pages have different characteristics. Samuel Louvan [13] proposed a new hybrid approach that consist of Machine Learning and developed heuristic approaches namely Largest Block String (LBS), String Length Smoothing (SLS), and Table Pattern (TP). With the Machine Learning approach, many kinds of features can be used and the learning algorithm may learn the parameter automatically. Moreover, by using this approach different learning results can be applied for different types of websites.

3 Background Theory

3.1 Web Page Segmentation

Several methods have been explored to segment a web page into regions or blocks [2], [4]. In the DOM-based Segmentation Approach, an HTML document is represented as a DOM tree. Useful tags that may represent a block in a page include P (for paragraph), TABLE (for table), UL (for list), H1~H6 (for heading), etc. DOM in general provides a useful structure for a web page. But tags such as TABLE and P are used not only for content organization, but also for layout presentation. In many cases, DOM tends to reveal presentation structure other than content structure, and is often not accurate enough to discriminate different semantic blocks in a web page. Another intuitive way of page segmentation is based on the layout of web page. In this way, a web page is generally separated into 5 regions: top, down, left, right and center [3]. The drawback of this method is that such a kind of layout template cannot be fit into all web pages. Furthermore, the segmentation is too rough to exhibit semantic coherence. Compared with the above segmentation, Vision-based Page Segmentation (VIPS) excels in both an appropriate partition granularity and coherent semantic aggregation. By detecting useful visual cues based on DOM structure, a tree-like vision-based content structure of a web page is obtained. The granularity is controlled by the Degree of Coherence (DoC) which indicates how coherence each block is. VIPS can efficiently keep related content together while separating semantically different blocks from each other. Visual cues such as font, color and size, are used to detect blocks. Each block in VIPS is represented as a node in a tree. The root is the whole page; inner nodes are the top level coarser blocks; children nodes are obtained by partitioning the parent node into finer blocks; and all leaf nodes consist of a flat segmentation of a web page with an appropriate coherent degree. The stopping of the VIPS algorithm is controlled by the Permitted DoC (PDoC), which plays a role as a threshold to indicate the finest granularity that we are satisfied. The segmentation only stops when the DoCs of all blocks are not smaller than the PDoC [8].

3.2 Informative Content Extraction

Informative Content Extraction is the process of determining the parts of a web page which contain the main textual content of this document. A human user nearly naturally performs some kind of Informative Content Extraction when reading a web page by ignoring the parts with additional non-informative contents, such as navigation, functional and design elements or commercial banners − at least as long as they are not of interest. Though it is a relatively intuitive task for a human user, it turns out to be difficult to determine the main content of a document in an automatic way. Several approaches deal with the problem under very different circumstances. For example, Informative Content Extraction is used extensively in applications, rewriting web pages for presentation on small screen devices or access via screen readers for visually impaired users. Some applications in the fields of Information Retrieval and Information Extraction, Web Mining and Text Summarisation use Informative Content Extraction to pre-process the raw data in order to improve accuracy. It becomes obvious that under the mentioned circumstances the extraction has to be performed by a general approach rather than a tailored solution for one particular set of HTML documents with a well-known structure [19].

4 Proposed Approach for Web Informative Content Extraction

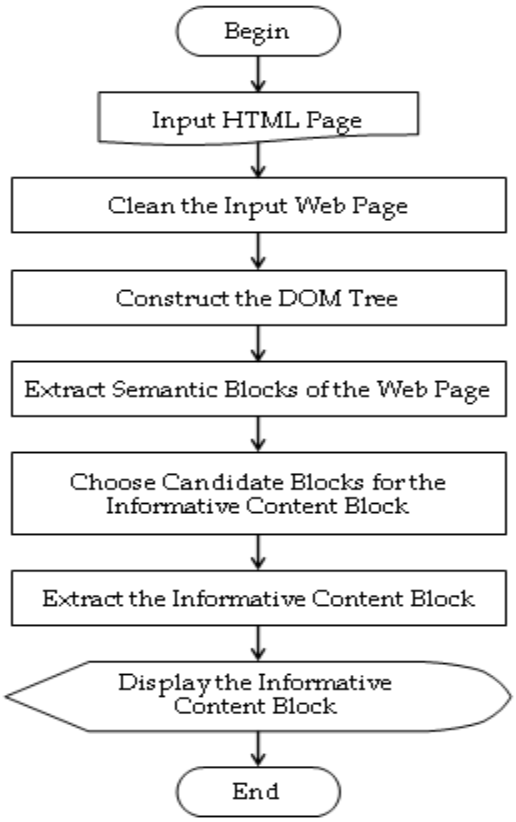

Fig. 1. Flow diagram for the proposed approach

Fig. 1 shows flow diagram for the Proposed Approach. It uses DOM tree of the input web page as its input and returns the Informative Content Block of the web page as its output. Five steps are needed to be carried out for effective Web Informative Content Extraction: Cleaning the Web Page, DOM Tree Construction, Visual Block Extraction, Choosing Candidate Blocks and Finding Informative Content Block. In this paper these five steps are presented in two Phases, Effective Web Page Segmentation and Effective Informative Content Extraction. The following sections demonstrate how to clean an HTML page, how to create the DOM tree of the input web page, how to extract semantic blocks of the web page, how to choose candidate blocks of the web page and how to extract the Informative Content Block effectively. The terms that are used in the Proposed Algorithms of these steps are defined as follows.

Terms Definition of the Proposed Algorithms
DOM = DOM tree of the HTML page
BLT = Block Tree
BL = Block
RE = Root Element
CLS = Children
CL = Child
IFCBL = Informative Content Block
CBL = Candidate Blocks for the Informative Content Block
GP = General Parameters
ImportValue = Importance Value of the Block

5 EFFECTIVE WEB PAGE SEGMENTATION

For Effective Web Page Segmentation, the following three steps:

1. Cleaning the Web Page
2. DOM Tree Construction and
3. Visual Block Extraction are needed to be carried out.

In the Visual Block Extraction Phase, the Proposed Approach tries to overcome the problems of DOM-based Approaches and reduce the drawbacks of previous works by applying the proposed Effective Visual Block Extractor (EVBE) Algorithm and its efficient rules. EVBE Algorithm and its efficient rules are proposed in Web Page Segmentation Phase to help for getting effective result in Web Informative Content Extraction.

5.1 Cleaning the Web Page

Most web pages are not well-formed documents. They contain invalid tag structure such as there is an opening tag with no corresponding closing tag and vice versa. Some HTML tags are nested in wrong order and also some tags are mixed up. In order to construct the DOM tree of the input web page correctly, HTML file needs to be well-formed. Therefore these invalid tag structures are needed to be cleaned before processing them. The CleanHTML Method for cleaning web page to construct the proper DOM tree effectively is as follows.

```
CleanHTML (HTMLpage)
    for each HTML tag in page
      begin
        if the tag is missing or mismatched
          then detect and correct this tag
        else if the tags are nested in the wrong order
          then correct the tag order
        else if there are tags lacking close '>'
          then fix this case
        end if
      end
    end for
```

5.2 DOM Tree Construction

The Document Object Model (DOM) is a standard for how to access, change, add, or delete HTML elements. The DOM presents an HTML document as a tree-structure. By using DOM tree and its visual properties as a source of Web Page Segmentation instead of web page, more control can be achieved while segmenting the page. Web pages are composed of HTML tags and their contents, such as text, image or hyperlinks etc. Each HTML page corresponds to a DOM tree where tags are internal nodes and the actual text, images or hyperlinks are the leaf nodes. The ConstructDOMTree Method that carries out to construct the DOM tree from the cleaned HTML document is as follows.

```
ConstructDOMTree (cleanedHTML)
    begin
        parse the cleaned HTML and
        construct the DOM tree
    end
```

Fig. 2 shows HTML document and its corresponding DOM tree. In the DOM tree, each solid rectangle is a tag node and the shaded box is the actual content of the node.

```
<HTML>
  <BODY bgcolor=WHITE>
    <TABLE width=800 height=200 >
      ...
    </TABLE>
    <IMG src="image.gif" width=800>
    <TABLE bgcolor=RED>
      ...
    </TABLE>
  </BODY>
</HTML>
```

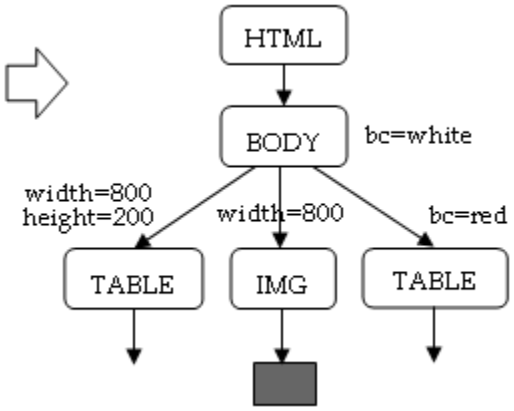

Fig. 2. HTML document and corresponding DOM

5.3 Visual Block Extraction

The approach for the extraction of semantic blocks from web page by applying EVBE Algorithm is presented in this section. The Algorithm uses the DOM structure of the input web page and the visual properties of each DOM node for effective extraction of semantic blocks from web page as shown in Fig. 3. The Algorithm recursively traverses the input DOM tree in pre-order manner and returns the Block Tree and the General Parameters of each block as its output. General Parameters contain general total value of specifications for text, links etc. that will be used to find the informative block later.

```
Algorithm: Effective Visual Block Extractor (EVBE)

Input: DOM
Output: GP, BLT
Type: Recursive

begin
        ExtractBlock (RE, out BLT)
        If RE has CL then
            For each CL in RE's CLS do
                If CheckValidity (CL) = True then
                    RE ← CL
                    EVBE (RE, out GP, out BLT)
                End If
            End For
        End If
end
```

Fig. 3. EVBE algorithm

The first line of the Algorithm applies ExtractBlock Method by using the Root Element of the DOM tree as its input. It returns the Block Tree for the Root Element. If Root Element has Child, for each Child of Root Element the Algorithm checks whether the Child is valid for creating sub-Block Tree or not. If it is valid, EVBE Algorithm is used again for constructing sub-Block Tree of the Child node. Each node in Block Tree clusters one or more nodes from the DOM tree.

Extract Block Method

From the root node of the DOM tree, the process is started to extract semantic blocks from the DOM tree based on visual cues. It checks the input DOM node to determine whether it should form a new block or not. It makes this decision by checking the nature of the current node or the visual distance of the current node in contrast with its parent and in some cases its sibling node. If the input DOM node is a separator node, the new block is created and the Algorithm flags its direct sibling node as the parent of new block. If the visual distance with its parent is valid, the current DOM node is added to the block of its DOM parent node. If the distance with its parent or in some cases with its sibling node is not valid, a new block is created and the Algorithm specifies the current DOM node as the parent of new block. The visual distance is the number of differences on the visual presentation styles of the DOM nodes such as width, height, font-size, background color, float, clear etc.

Sub-Block Tree Creation for Child Node

For each DOM child of the input element if the child is valid for creating sub-Block Tree, the recursive EVBE Algorithm is applied. The recursive algorithm makes sub-Block Tree for each child node and their subordinating child nodes.

EVBE Algorithm

Several algorithms have proposed DOM-based Approaches to segment a web page into semantic blocks. However, these algorithms cannot often provide satisfactory results. VIPS is a Vision-based Algorithm to segment a web page into semantic blocks automatically. This algorithm mainly depends on visual separators although in some cases visual separators are misleading or ambiguous. It does many loops to reach its desire granularity. The nature of EVBE Algorithm is different from VIPS Algorithm. The Proposed Algorithm mainly depends on the nature of the current node or the visual distance of the current node in contrast with its parent and in some cases its sibling node for effective Web Page Segmentation. The definition of nodes and some important visual cues which are used for developing the efficient rules of the Proposed Algorithm are as follows.

Definition of Nodes

***Inline Node*:** Nodes, which do not cause to a new line in a page, are inline nodes e.g., A, STRONG, BIG, EM, etc.

***Line-break Node*:** Line-break nodes cause to a new line in a page e.g., DIV, P, TABLE, TR, UL, etc.

***Invalid Node*:** Invalid nodes are those do not appear visually in the page, such as PARAM, SCRIPT, STYLE, TITLE, <!–...–>, !DOCTYPE, etc.

***Partially Valid Node*:** These nodes affect the page layout only if they have a text node or any other visible children. Partially valid nodes consist of inline nodes and some of the line-break nodes such as FORM, HEAD, TABLE, DD, DT, etc.

***Valid Node*:** Valid nodes are those appear in the page layout. They consist of partially valid nodes with visible children and the remaining of the line-break nodes that do not include in partially valid nodes. These line-break nodes appear in the page layout, even if they do not have any valid child.

***Text Node*:** The DOM node corresponding to free text, which does not have an html tag, is a text node.

***Virtual Text Node*:**

1. Inline node with only text node children is a virtual text node.
2. Inline node with only text node and virtual text node children is a virtual text node.

***Float*:** Some nodes can be grouped into the same block or divided into different blocks according to the nature of float values and in some cases together with the use of the width and clear properties of the nodes.

***Clear*:** The clear property of the node is considered together with the float property when deciding which nodes should be grouped into the same block or divided into different blocks.

***Width*:** The width property of the node is very useful to support other property of the node, such as float when making decision for creating blocks.

By using the above definitions and visual cues, the efficient rules for effective Web Page Segmentation are produced. Some basic rules of the Proposed Algorithm are as follows.

Rules of the Proposed Algorithm

***Rule 1*:** Remove all the invalid nodes and partially valid nodes which do not have any child.

***Rule 2*:** If a node has only text node or invalid children, no block will be extracted. Otherwise same rules are applied to each child.

***Rule 3*:** If node has only one child,

1. If child is a text node or virtual text node, then no block will be extracted.
2. If child is line-break node, then same rules are applied to the child node.

***Rule 4*: I**f all the children are virtual text nodes of a node, node is put into block pool.

***Rule 5*:** If a node contains a child whose tag is HR or any of valid nodes which has no child, then the node is divided into two as the nodes before the separator and after the separator. For each side of the separator, two new blocks are created and children nodes are put under these blocks.

***Rule 6*:** If node is a table and some of its columns have different background color than the others, divide the table into the number separate columns and construct a block for each piece.

***Rule 7*:** If the first child of the node or the last child of the node is a DIV with no child and with nonzero height property and background-color property which is not white, then create two blocks: one of which for its parent node and the other for its parent previous or direct sibling node.

***Rule 8*:** If a node has two adjacent children that the first child has float: left with or without clear: both property and the second child has float: left or right with clear: none property or without clear property and the total width of these two children is less than the width of the parent node, create a block for each adjacent child. Also create a block for the children without float value. Then same rules are applied to the children of each block.

***Note*:** The parent node and two adjacent children may be DIV. Two adjacent children can have equal float value.

***Rule 9*:** If a node has a child, whose at least one of border-top-width and border-bottom-width values is nonzero, divide this node into two blocks. Put the sibling nodes before the node with nonzero border into the first block and put the siblings after the node with nonzero border into the second block.

1. If child has only nonzero border-top-width, put the child into second block.
2. If child has only nonzero border-bottom-width, put the child into first block.
3. If child has both nonzero border-top-width and nonzero border-bottom-width, create a third block and put it between two blocks.

***Rule 10*:** If a node has at least one of nonzero border-top-width and nonzero border-bottom-width values and both of the nonzero border-left-width and nonzero border-right-width values, create a block for the node.

Some rules of the Proposed Algorithm are similar with the basic rules from other Vision-based Algorithms. New efficient rules are presented in the Proposed Algorithm to get satisfactory results in Web Page Segmentation.

Effective Visual Block Extraction by applying the Proposed Rules

Fig. 4(a) shows the expected result on a sample page after applying the Proposed EVBE Algorithm. There are four expected semantic blocks in the page namely BL1, BL2, BL3 and BL4 marked by red rectangles. Here BL stands for block. Each red rectangle represents the semantic visual block of the page.

Fig. 4(a). Expected segmentation result on a sample page

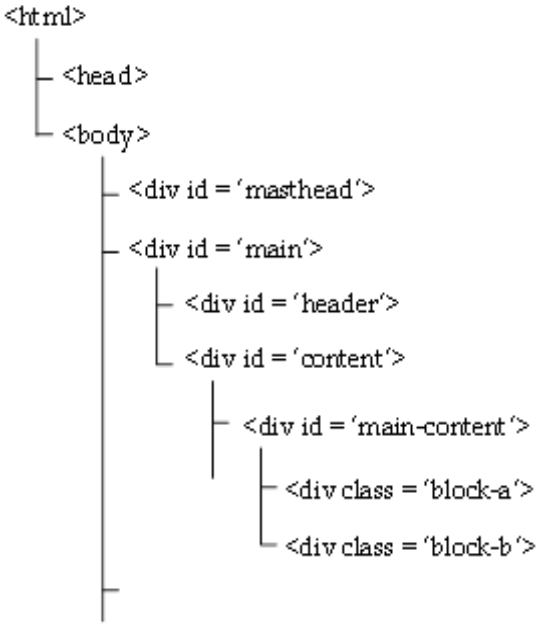

Fig. 4(b). Corresponding DOM of the sample page

Fig. 4. Visual block extraction of the sample page

Fig. 4(b) shows a part of the DOM tree to give an overview of the block extraction process. The visual block extraction process starts from the <body> node of the DOM tree. In the block extraction process, when the node is met, it is checked whether it should be divided or not according to Rule 2. The visual distance between the current node and its children is valid and they are not divided into different blocks. And the Algorithm checks the visual distance

of the current node in contrast with its direct sibling node. The background color and some important visual properties of the two nodes are very different and the current node is extracted as a separate block BL1. In node, there are two children, and . The children of have valid visual distance with its parent. The node has child . In node, there are two adjacent children and . The first child has float: left property and the second child has float: right with clear: none property and the total width of these two children is less than the width of the parent node. So node is extracted as a separate block BL2. And according to Rule 8, a block for each adjacent child is created as block BL3 and BL4. Therefore after applying EVBE Algorithm, four semantic blocks, BL1, BL2, BL3 and BL4 are extracted as the semantic visual blocks of the sample page. Some Vision-based Algorithms mainly use the concept of parent-child relationship of the DOM nodes in their Web Page Segmentation. The Proposed EVBE Algorithm applies not only the concept of parent-child relationship but also the concept of siblings' relationship of the DOM nodes for effective Web Page Segmentation. It does not depend on any tag type. It just has an iteration to segment the input web page into semantic blocks. The Proposed Algorithm is intended to overcome the problems of DOM-based Approaches and reduce the drawbacks of previous works in Web Page Segmentation.

6 Effective Informative Content Extraction

```
Algorithm: Effective Informative Content Extractor (EIFCE)

Input: DOM
Output: IFCBL

begin
        EVBE (DOM, out GP, out BLT)
        {Final Computation for each total of GP}
        ChooseCandidateBlocks (BLT, GP, out CBL)
        {Output: CBL}
        FindInformativeBlock ( GP, CBL, out IFCBL)
        {Output: IFCBL}
end
```

Fig. 5. EIFCE algorithm

After segmenting the web page into semantic blocks correctly, the Informative Content Block of the web page can be extracted effectively. Fig. 5 shows the Effective Informative Content Extractor (EIFCE) Algorithm, the Proposed Algorithm for Effective Informative Content Extraction. The first line of the Algorithm applies EVBE Algorithm for Effective Web Page Segmentation. It returns the Block Tree and General Parameters of each block as its output. Then the Algorithm applies ChooseCandidateBlocks function for choosing Candidate Blocks for the Informative Content Block. And then it applies the FindInformativeBlock method to detect the Informative Content Block from the Candidate Blocks. Finally it returns the Informative Content Block of the web page as result. Two steps, Choosing Candidate Blocks and Finding Informative Content Block for Effective Informative Content Extraction are presented in the following sections.

6.1 Choosing Candidate Blocks

```
Algorithm: ChooseCandidateBlocks Function

Input: GP, BLT
Output: CBL
Type: Recursive

begin
        For each BL in the BLT do
            If GPIsValid(BL) then
                CBL := BL;
            End If
        End For
end
```

Fig. 6. ChooseCandidateBlocks function

As shown in Fig. 6, the input for the ChooseCandidateBlocks function is the output of EVBE Algorithm, General Parameters of each block and the Block Tree. The output is the Candidate Blocks for the Informative Content Block. The method checks each block whether it is valid for choosing Candidate Blocks for the Informative Content Block. It is determined by checking the General Parameters of each important feature of each block. If the block met the least conditions to be chosen as candidate for the Informative Content Block, it is added to the Candidate Blocks list. Otherwise it is assumed as non-informative content block. Finally the blocks in the Candidate Blocks list are returned as the Candidate Blocks for the Informative Content Block**.**

6.2 Finding Informative Content Block

The inputs for the FindInformativeBlock function are the Candidate Blocks, the output of ChooseCandidateBlocks function and the General Parameters of each Candidate Block as shown in Fig. 7. The output is the Informative Content Block of the web page. Firstly the method calculates the Importance Value of each block in the Candidate Blocks list. The Importance Value is calculated by using the concept of the following function f proposed in [1].

$$f_p(b) = \frac{\text{the size of block } b}{\text{the distance between the center of } b \text{ and the center of the screen}} \quad (1)$$

f is a function that assigns an Importance Value to every block b in page p. The Importance Value is the ratio of the possibilities of informative content and the possibilities of non-informative content. Both upper and lower part of the function is replaced with the reasonable General Parameters of each block. The bigger the value of the possibilities of informative content, the more the Informative Content Block it can be. The bigger the value of the function f is, the more important the block b is. Then it compares the Importance Values of the Candidate Blocks. Finally the block with the greatest Importance Value is returned as the Informative Content Block of the web page.

```
Algorithm: FindInformativeBlock Function

Input: CBL, GP
Output: IFCBL
Type: Recursive

begin
      For each BL in CBL do
          {
            Calculate the ImportValue of BL
          }
      End For
      For i=1 to i < no: of CBL do
          {
            Compare the Import Value of CBL
            IFCBL := BL with the greatest ImportValue
          }
      End For
end
```

Fig. 7. FindInformativeBlock function

7 Evaluation of the Proposed Approach

For further Effective Informative Content Extraction, it needs to segment the web page into semantic blocks correctly. By applying the Proposed EVBE Algorithm, the blocks such as BL3 and BL4 can be extracted easily. However, VIPS algorithm cannot segment them as separate blocks when the Permitted Degree of Coherence (PDoC) value is low. It can segment them as separate blocks only if PDoC value is high. However, when the PDoC value is high, it segments the page into many small blocks although some separate blocks should be a single block. It is unreasonable and inconvenient for any further processing. Although BL3 contains the informative content of the web page, BL4 doesn't contain any informative content of the page. Actually the content nature of BL3 and BL4 is different and they should be segmented as separate blocks. However, when the PDoC value is low, VIPS algorithm assumes BL3 and BL4 as a single block. The great rules of EVBE Algorithm can reduce the drawbacks of previous works and can help for getting finer results in Web Page Segmentation. Some solutions proposed DOM-based Approaches to extract the informative content of the web page. Unfortunately DOM tends to reveal presentation structure other than content structure, and is often not accurate enough to extract the informative content of the web page. CE needs a learning phase for Informative Content Extraction from web pages. So it couldn't extract the informative content from random one input web page. FE can identify Informative Content Block of the web page only if there is a dominant feature. So the Proposed Approach intends to introduce EIFCE Algorithm which could extract the informative content that is not necessarily the dominant content and without any learning phase and with one random page. It simulates the concept of how a user understands the layout structure of a web page based on its visual representation. Compared with DOM-based Informative Content Extraction Approaches, it utilizes useful visual cues to obtain a better extraction of the informative content of the web page at the semantic level. The efficient rules of the Proposed EVBE Algorithm in Web Page Segmentation Phase can help for getting finer results in Web Informative Content Extraction.

8 Conclusion

Web pages typically contain non-informative content, noises that could negatively affect the performance of Web Mining tasks. Automatically extracting the informative content of the page is an interesting problem. By applying the Proposed EVBE and EIFCE Algorithms, the informative content of the web page can be extracted effectively. Automatically extracting Informative Content Block from web pages can help for increasing the performance of Web Mining tasks. The empirical experiment of the Proposed Approach is planned as the future work.

Acknowledgment

I would like to thank my supervisor, Dr. Mie Mie Su Thwin, for her excellent guidance, where the initial scope of the research was defined and throughout the process of writing this paper.

References

[1]. J. Han and M. Kamber, "Data Mining: Concepts and Techniques, Second Edition," pp. 630-637, 2006.

[2]. J. Chen, B. Zhou, J. Shi, H. Zhang, and Q. Fengwu, "Function-Based Object Model Towards Website Adaptation", In the Proceedings of the Tenth World Wide Web conference (WWW10), Budapest, Hungary, May 2001.

[3]. M. Kovacevic, M. Diligenti, M. Gori, M. Maggini, and V. Milutinovic, "Recognition of Common Areas in a Web Page Using Visual Information: a possible application in a page classification," In the Proceedings of 2002 IEEE International Conference on Data Mining (ICDM'02), Maebashi City, Japan, December 2002.

[4]. S.-H. Lin and J.-M. Ho, "Discovering Informative Content Blocks from Web Documents," In the Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD'02), 2002.

[5]. D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, "Extracting Content Structure for Web Pages based on Visual Representation," In the Fifth Asia Pacific Web Conference (APWeb2003), Springer Lecture Notes in Computer Science, 2003.

[6]. D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, "VIPS: a Vision-based Page Segmentation Algorithm," Technical Report, MSR-TR-2003-79, 2003.

[7]. L. Yi, B. Liu, and X. Li, "Eliminating Noisy Information in Web Pages for Data Mining," In Proceeding of the 9th ACM SIGKDD International Conference, 2003.

[8]. R. Song, H. Liu, J.-R. Wen, and W.-Y. Ma, "Learning Important Models for Web Page Blocks based on Layout and Content Analysis," SIGKDD Explorations, Volume 6, Issue 2, Microsoft Research Asia, 49 Zhichun Road, Beijing, 100080, P.R. China and Department of Computer Science, University of Toronto, Toronto, ON, Canada, 2004.

[9]. S. Debnath, P. Mitra, N. Pal, and C.L. Giles, "Automatic Identification of Informative Sections of Web Pages," In IEEE Transactions on Knowledge and Data Engineering, 17(9): 1233-1246, 2005.

[10]. S. Debnath, P. Mitra, and C.L. Giles, "Identifying Content Blocks from Web Documents," Penn State University, USA, 2005.

[11]. J. Gibson., B. Wellner, and S. Lubar, "CoreEx: Content Extraction from Online News Articles," In Proceeding of the 17th ACM IKM Conference, 2008.

[12]. M. Toman, "Comparison of Approaches for Information Extraction from the Web," In Proceeding of the 9th International PhD Workshop on Systems and Control: Young Generation Viewpoint, Slovenia, 2008.

[13]. S. Louvan, "Extracting the Main Content from HTML Documents," [Online], http://wwwis.win.tue.nl/bnaic2009/papers/bnaic2009_paper_113.pdf, 2009.

[14]. T. Win and K.N.N. Tun, "Noise Elimination for Improving Web Information Extraction," In the Proceedings of the Seventh International Conference on Computer Applications, 2009.

[15]. M. Asfia, M.M. Pedram, and A.M. Rahmani, "Main Content Extraction from Detailed Web Pages," In International Journal of Computer Applications (0975 – 8887), 2010.

[16]. Y. Yesilada, "Web Page Segmentation: A Review," eMINE Technical Report Deliverable 0 (D0), 2011.

[17]. Y. Yesilada, "Heuristics for Visual Elements of Web Pages," eMINE Technical Report Deliverable 1 (D1), 2011.

[18]. E. Akpınar and Y. Yesilada, "Vision Based Page Segmentation: Extended and Improved Algorithm," eMINE Technical Report Deliverable 2 (D2), unpublished, Middle East Technical University, Ankara, Turkey, 2012.

[19]. T. Gottron, "Evaluating Content Extraction on HTML Documents," Institut für Informatik, Johannes Gutenberg-Universität, Mainz, Germany.