Main Content Extraction from Web Documents Using Text Block Context

Myungwon Kim, Youngjin Kim, Wonmoon Song, and Ara Khil

Dept. of Computing, Soongsil University,
Seoul, Korea Republic
{mkim,liebulia,gtangel,ara}@ssu.ac.kr

Abstract. Due to various Web authoring tools, the new web standards, and improved web accessibility, a wide variety of Web contents are being produced very quickly. In such an environment, in order to provide appropriate Web services to users' needs it is important to quickly and accurately extract relevant information from Web documents and remove irrelevant contents such as advertisements. In this paper, we propose a method that extracts main content accurately from HTML Web documents. In the method, a decision tree is built and used to classify each block of text whether it is a part of the main content. For classification we use contextual features around text blocks including word density, link density, HTML tag distribution, and distances between text blocks. We experimented with our method using a published data set and a data set that we collected. The experiment results show that our method performs 19% better in F-measure compared to the existing best performing method.

Keywords: Web Document Analysis, Content Extraction, Tag Distribution, Block Distance, Context.

1 Introduction

As portable smart phones became widely distributed, users are able to access the Internet faster and more conveniently. Also, the development of new web standards and publishing tools enables the producers of the web documents to express their messages in easier and more diverse ways. The web documents with extremely free and diverse styles are generated in a fast speed not only by business organizations but also by individual users via web blogs or SNS. To provide users what they want promptly and precisely in the web service environment, where many types of web documents are increasing explosively, it is needed to precisely categorize, analyze, and understand web documents.

Recently, for this purpose, researchers have investigated automatic classification of the pre-designated areas of web documents such as the main content, advertisements, comments, menus and other. Moreover, studies have been actively conducted on auto-extraction of the main content of a web document which contains the most important information.

H. Decker et al. (Eds.): DEXA 2013, Part II, LNCS 8056, pp. 81–93, 2013.

In the early studies, the researchers analyzed structural features of the HTML documents by reorganizing the HTML-based web documents in the form of DOM (Document Object Model) [1] tree structure and analyzed the results [2-4]. Recently, some alternative methods were introduced in order to extract the main content out of an HTML document based on its context, overcoming the limitation of the structural analysis of the HTML documents, such as: identifying the main content based on a linguistic model, which is obtained through learning emerging words and messages in the target document [5-8], and extracting the main content using distinct features such as tags and hyper-links in the target HTML document [9-12]. However, since all of these previous works tested their performances using the restricted types of data such as news and blogs, the applicability to the current web environment containing various types of documents, remains questionable.

In this paper, we propose a method that can complement the weaknesses of the existing methods. The proposed method extracts the main content of the target document more accurately, by decomposing the target HTML document into text blocks and determining each text block whether it belongs to the main content or not based on the contextual features of the text block such as HTML tag distribution around the text block and the information of its neighboring text blocks. The method is applicable to the actual web environment which is full of diverse types of documents.

2 Related Works

2.1 Classification/Extraction of the Main Content from Web Documents

The most representative method to extract the main content from a web document is the DOM tree structure analysis, using the structural features of the HTML documents [2-4]. Especially, in [2], they convert an HTML document into a DOM tree based on visually identifiable blocks for page separation and extraction of the main content block. In the DOM tree, the attributes of each node consist of its width, height, frequency of its appearance in the document. Then, the nodes are grouped based on their attributes and finally each group is determined whether it belongs to the main content or not based on the group features. The main purpose of this method is to investigate the possibility of extracting the main content automatically through machine learning without using any template information of each web page. However, since it only utilizes visibly identifiable information on the screen and structural features of the documents that are interpretable to a tree structure, when irrelevant information such as advertisement, spam message, and a list of articles constitutes the bigger part of the target document, its performance of extracting the main content degrades rapidly.

To overcome this weakness, other methods were proposed that utilize HTML tags and text messages as distinctive attributes on the pattern of the HTML document composition [5-8, 10-12]. [5] and [7] convert an HTML document into a list of tags and text tokens, analyze the order and context of those items using probability models and extract the document regions which match to one of the known patterns as the main content of the document. [11] and [12] propose a process that groups the lines of the target document using the appearance frequencies of the HTML tags and non-tag texts, then decides the eligibility of each group as the main content. The bottom line of these methods is that they attempt to analyze the document on the basis of the text tokens and patterns, beyond the structural analysis. However, the use of the text tokens of documents causes a risk to constantly revise and expand the database for learning to handle newly published web documents.

Other methods that were recently introduced utilize the characteristics of the HTML documents as web documents, which differentiate them from others in extracting the main content [9, 13]. [9] focuses on the fact that the main content of web documents usually consists of texts. It first segments text blocks out of the target HTML document. Then it decides whether a text block belongs to the main content based on the attributes such as the number of words and hyper-linked words in the text block, and the information of its neighboring text blocks. By doing so, the method shows a relatively high performance. However, it was verified by only using limited source of data such as news or blogs. It is still questionable whether it is applicable in the current web environment where many different kinds of web documents exist.

2.2 Main Content Extraction Based on Word/Link Density in Text Blocks

The best known open system for main content extraction from web documents is Boilerpipe proposed by the L3C research center [9, 13]. Boilerpipe consists of four steps. The first step is to divide the target HTML document into text blocks using some empirical rules. In the second step, it derives attributes of each text block by calculating each text block's word density and link density corresponding to the number of words and links contained in the text block, respectively. The next step is to build a classification model by machine learning based on the attributes of each text block and the word densities and link densities of its neighboring text blocks. Finally, using the model it decides which text block is the main content of the target document.

To divide an HTML document into text blocks, Boilerpipe utilizes HTML tags. All HTML tags in the document become group separators and all text segments separated by them are configured into text blocks. The only exception is tag <a> because it is only used to insert the hyper-link information and does not bring any structural changes. Therefore, <a> tags are only used for calculating link density in an HTML document.

After separating all text blocks, for each text block TB_i, the word density ($D_{WORD}(TB_i)$) and link density ($D_{LINK}(TB_i)$) are calculated according to equations (1). In the equations, $Word(TB_i)$, $Sentence(TB_i)$ and $LinkedWord(TB_i)$ denote the set of all words, the set of all sentences and the set of hyper-linked words by <a> tag in the text block, respectively. [9] identifies sentences in a text block by the number of words, and each sentence is assumed to have 80 words.

$$D_{WORD}(TB_i) = \frac{|Word(TB_i)|}{|Sentence(TB_i)|}, \quad D_{LINK}(TB_i) = \frac{|LinkedWord(TB_i)|}{|Word(TB_i)|} \tag{1}$$

Each text block has six attributes for machine learning and classification including the word density and link density of itself and the word densities and link densities of its neighboring text blocks.

In [9], the authors identified all text blocks through the procedure described above in the L3S-GN1 data set, a collection of 621 news articles on the web gathered by Google search. They then checked each text block and marked whether it belonged to the main content of the article or not to build the data for machine learning. They measured the classification accuracy of the method with the decision tree based classification algorithm and the 10-fold cross validation. To verify the superiority of the method to others, they compared the result with other existing methods, and successfully achieved their goals. It implicates that the six attributes, which are attainable in a relatively short period of time can classify the main content more accurately. However, their verification procedure remains questionable because they took the accuracy of the non-main content classification into consideration when they calculated the accuracy of the main content classification. Since the non-main content take the larger part of the target data set, it is difficult to recognize that the reported accuracy represents the exact accuracy of the main content classification.

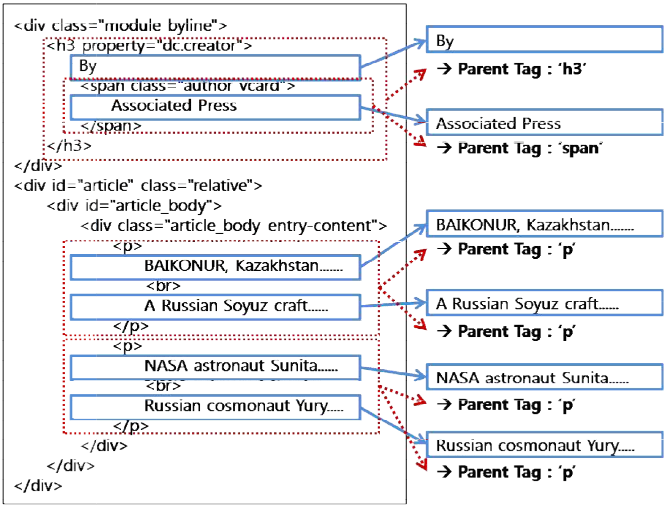
3 Main Content Extraction Using Text Block Context

In this section, we introduce a new method to extract the main content from HTML documents more accurately using the contextual features of text block. The contextual features consist of the HTML tag distribution around the text blocks and the neighboring text block information in addition to previously introduced and proven attributes, i.e. the word density and link density of the text blocks. Initially, following the procedure of [9], the target HTML document is divided into text blocks and the word density and link density are used as the attributes of each text block. The new method adds two additional aspects to the existing text block attributes for main content extraction: the HTML tag distribution and the neighboring text block information.

3.1 HTML Tag Distributions Around the Text Blocks

An HTML document is organized by predefined and tree-structured HTML tags. In the main content of an HTML document, which mostly consists of texts, HTML tags for the text format and paragraph separation are mainly used [14]. Based on these characteristics, we utilize parent tags, which are immediate upper-level HTML tags that contain text blocks in a tree structure, as one of the attributes to determine whether a given text block belongs to the main content or not.

Figure 1 is an example of parent tag extraction for some text blocks. Since a parent tag is an immediate upper-level tag that encompasses the given text block as mentioned before, the parent tag of "By", the first text line in the upper part of Figure 1, is <h3> and <span> is the parent tag of "Associated Press". In the case of the text blocks in the lower part of Figure 1, the parent tag of "NASA astronaut Sunita..." and "Russian cosmonaut Yuri..." are the <p> tag which is located right above them. The <br>

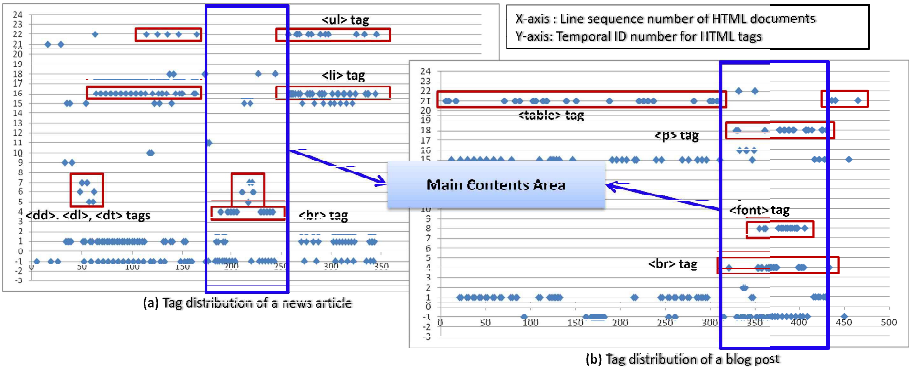


```
<div class="module byline">
    <h3 property="dc.creator">
        By
        <span class="author vcard">
            Associated Press
        </span>
    </h3>
</div>
<div id="article" class="relative">
    <div id="article_body">
        <div class="article_body entry-content">
            <p>
                BAIKONUR, Kazakhstan.......
                <br>
                A Russian Soyuz craft......
            </p>
            <p>
                NASA astronaut Sunita......
                <br>
                Russian cosmonaut Yury.....
            </p>
        </div>
    </div>
</div>
```

Fig. 1. Tag distribution around the text block: parent tag information

tag, located between these two text blocks cannot be the parent tag because it is a child tag of the <p> tag and it does not include these two text blocks.

Any HTML tag – total of 94 tags under the HTML standard 4.0.1 [14] - can be a parent tag of a text block. Among them, tags for text format processing and paragraphing are mostly used near the main content area of an HTML document. Therefore, in this paper, these tags which are frequently used around the main content are considered as the only candidate group of the parent tags. To enhance the effectiveness of the main content extraction process we propose, the syntactic and semantic information of all HTML tags [14] is checked, and their distribution and density information are analyzed with various examples in the Korean web such as news articles and blogs. Figure 2 shows the partial result of these examples.



(a) Tag distribution of a news article

(b) Tag distribution of a blog post

Fig. 2. Tag distribution of an HTML document

In the figure, the horizontal axis in graph (a) and (b) represents the sequence number of lines in the target document that is refined beforehand and the vertical axis represents the original identification number that was assigned to each HTML tag. For example, a point (x,y) on the two-dimensional graph – blue dots in the figure - represents that the HTML tag corresponding to the ID number 'y' appears in the 'x'-th line of the target document. With these conditions in mind, when you look at the graph (a), you can find that <br>, <dd>, <dl>, and <dt> mainly appear on the inside of the main content, while <ul>, <li>, and others are concentrated on the outside of the main content. (Similarly, in the graph (b), <br>, <font>, and <p> are located mainly on the inside of the main content.)

We conducted the same analysis as Figure 2 with 15 other arbitrarily collected HTML documents and defined 22 tags that mainly appeared on the inside of the main content area and less likely to appear in other areas. Table 1 is the list of 22 tags, and these tags were used for extracting parent tag information. When extracting the parent tags out of a document, if any extracted tag is one of those in Table 1, then it is separately marked. If not, it is classified into the same group as the rest. This procedure is necessary for learning a classification model during the machine learning process, focusing more on the main content text blocks than the non-main content text blocks and reducing the possibility of false learning by noises and irregularities.

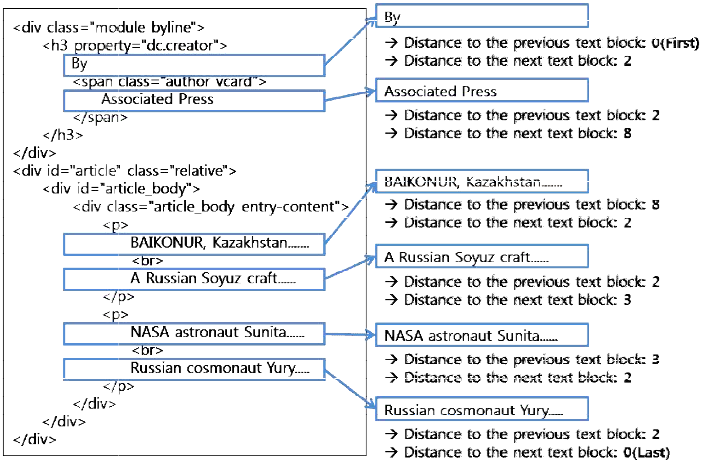
Table 1. Meaningful HTML tags of the text block's parent tag information

Tag Name	Html Function/Description	Tag Name	Html Function/Description
a	Insert a hyper link	img	Insert an image
b	Make the next text bold	li	Describe lists in <ul>, <ol>
blockquote	Indicate a quoted paragraph	ol	Produce an ordered list
br	Begin a new line	p	Paragraphing
dd	Describe definitions declared in <dl>	pre	Maintain document style
dl	List of definitions	q	Indicate a quoted word and phrase
dt	Describe terms declared in <dl>	table	Insert a table
font	Set font type	ul	Produce an unordered list
h1~h6	Heading (biggest size ~ smallest size)		

3.2 Neighboring Text Block Information

The second set of information for the main content extraction method we propose consists of two pieces of information: the neighboring text block information and the distance to the neighboring text blocks.

Since the main content of a web document shows up in one specific area of the whole document, texts in the main content are located closely in that area. It means if a text block is a part of the main content, its neighboring text block is likely to be as well. Also, if a text block is not a part of the main content, its neighboring text block is less likely to be so. Therefore, the distances from a given text block to its neighboring



```
<div class="module byline">
    <h3 property="dc.creator">
        By
        <span class="author vcard">
            Associated Press
        </span>
    </h3>
</div>
<div id="article" class="relative">
    <div id="article_body">
        <div class="article_body entry-content">
            <p>
                BAIKONUR, Kazakhstan.......
                <br>
                A Russian Soyuz craft......
            </p>
            <p>
                NASA astronaut Sunita......
                <br>
                Russian cosmonaut Yury.....
            </p>
        </div>
    </div>
</div>
```

By
→ Distance to the previous text block: 0(First)
→ Distance to the next text block: 2

Associated Press
→ Distance to the previous text block: 2
→ Distance to the next text block: 8

BAIKONUR, Kazakhstan.......
→ Distance to the previous text block: 8
→ Distance to the next text block: 2

A Russian Soyuz craft......
→ Distance to the previous text block: 2
→ Distance to the next text block: 3

NASA astronaut Sunita......
→ Distance to the previous text block: 3
→ Distance to the next text block: 2

Russian cosmonaut Yury.....
→ Distance to the previous text block: 2
→ Distance to the next text block: 0(Last)

Fig. 3. Distance to the neighboring blocks

text blocks and the neighboring text block's attributes are essential in deciding whether a certain text block belongs to the main content or not. In this paper, we propose a method that can extract the main content more accurately based on such information.

The distance from the current text block to its neighboring text block is measured by the number of lines between the text blocks in the preprocessed HTML file with each tag separated on a line. An example is shown in Figure 3. The distance from the first text block to its preceding text block is set to 0. Similarly, the distance from the last text block to its following text block is also set to 0. In a sense the distance between text blocks represents the density of text blocks.

Additionally, the information of two preceding and two following text blocks is also utilized. As mentioned before, the main content text blocks tend to be located nearby from one another and so do the non-main content text blocks. To take advantage of such an observation in classifying each text block we use the information of its surrounding text blocks as the context information.

3.3 Attributes for Classification and Classification Algorithm

In this paper, as mentioned above, we propose a method based on machine learning for automatic extraction of the main content from HTML documents using the combination of previously introduced attributes and newly introduced ones in this paper.

Total of 25 attributes are used, and five attributes from each text block were extracted. The attributes introduced in [9] such as word density and link density, and the ones proposed in this paper such as parent tags, the distance to the preceding text block, and the distance to the following text block. Additionally, as mentioned in Section 3.2, for each text block the information of the two preceding text blocks and two following text blocks are used as the context information as well. Therefore, the set of attributes for a text block is composed of 25 attributes from five text blocks including its two preceding and two following text blocks.

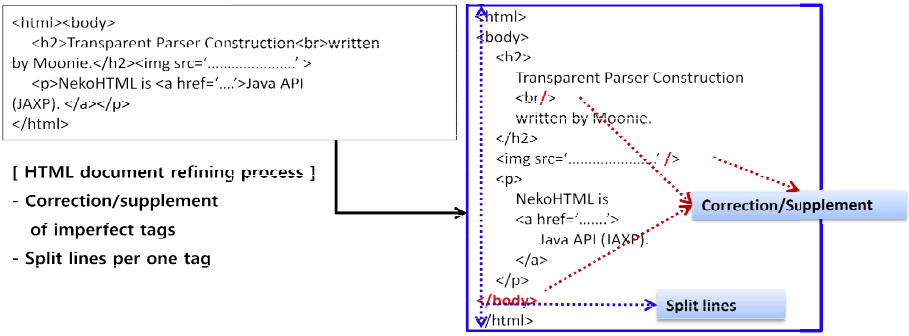
Machine learning for classification is based on the decision tree model which is widely used. A decision tree is composed of nodes and arcs which represent attributes and attribute values, respectively. A decision tree is constructed by recursively dividing the data space along the selected attribute into subspaces in each of which classes are well separated. When selecting an attribute to divide the target data into subgroups of good class separation, different algorithms use one of these: Chi-square statistics, Gini index, or entropy index [15, 16]. In this paper, we used J48 method provided by WEKA [17]. It is most widely used methods and one of the decision tree based on C4.5 algorithm which uses the entropy index for attribute selection [16, 17].

4 Experiments and Evaluation

In Section 4, we describe attribute extraction through preprocessing of web documents, experimental data for learning and evaluation, learning algorithm, and performance evaluation. We also verify the proposed method by comparing it with the existing methods using both public and privately collected data.

4.1 HTML Document Preprocessing for Attribute Extraction

Before identifying each text block and extracting its attributes, it is necessary to refine the improperly written HTML documents. Although most HTML documents abide to the grammatical rules of HTML, some of them ignore the HTML codes that do not affect their final display on the screen. Since most HTML documents do not separate lines clearly, they have poor readability, thus, it is difficult to discern the text blocks and to calculate the distance between text blocks. Consequently, these documents should undergo a refining process that corrects or supplements their grammatically improper tags and separates lines to make each line to have only one HTML tag or one text block. The Neko parser [18], an HTML parser, was used for the refining process, and Figure 4 is a partial excerpt of a refined HTML document.



```
<html><body>
   <h2>Transparent Parser Construction<br>written
by Moonie.</h2><img src='......................' >
   <p>NekoHTML is <a href='....'>Java API
(JAXP). </a></p>
</html>
```

```
<html>
<body>
  <h2>
    Transparent Parser Construction
    <br/>
    written by Moonie.
  </h2>
  <img src='......................' />
  <p>
    NekoHTML is
    <a href='.......'>
      Java API (JAXP).
    </a>
  </p>
</body>
</html>
```

Fig. 4. HTML document refining process

After the refining process, following the method of [9], all text blocks are identified using all tags as separators except <a>, then the word density and link density for each text block are calculated. Finally, the attributes proposed by this paper, i.e. the surrounding tag distribution and the information of the neighboring text blocks, are extracted and added to the attribute vector of each text block along with the word and link densities.

4.2 Experiment Data

To verify the proposed method, we used two data sets in this paper.

The first data set is L3S-GN1 [19], a public data set made by a German research center (L3S) to compare and evaluate the performance of the main content extraction from web documents. Based on the assumption that the news articles published by various agencies are a generic document with diverse web contents, the authors gathered 621 English news articles through Google web search and published the data. In the data each text block is marked manually whether it belongs to the main content or not.

The second data set is called 965-GWeb [20], a collection of 965 web documents we gathered. It is a collection of web documents gathered through Google search like L3S-GN1, but for a wide spectrum of document types, we collected various kinds of documents such as news articles, blog posts, SNS pages, wiki documents, image/video pages, etc. Then, we checked and marked each text block manually whether it is a part of the main content or not.

The number and proportion of all text blocks and main content text blocks are shown in Table 2

Table 2. Distribution of the training data's records (text block) and class (main content and non-main content)

Data Set	Number of HTML Documents	Number of Learned Records (Text Blocks)		
		Total Records	Main Content (%)	Non-Main Content (%)
L3S-GN1 [19]	**621**	84,198 (Average 136/doc)	13,209 (15.69%) (Average 21/doc)	70,989 (84.31%) (Average 114/doc)
965-GWeb [20]	**965**	141,808 (Average 147/doc)	22,308 (15.73%) (Average 23/doc)	119,500 (84.27%) (Average 124/doc)

To use the both data sets shown in Table 2 for learning and evaluation without any separate validation data, all data in the data set are split for 10-fold cross-validation and then used for learning and evaluation.

4.3 Learning Algorithm and Performance Measure

For automatic extraction of the main content area, a classification method was used to determine whether a certain text block belongs to the main content or not. As for the learning algorithm, J48 was used, which is a decision tree algorithm based on C4.5 [15, 16], included in WEKA [16, 17].

In the algorithm, two parameters are supported. The one is the reliability of the training data and the other is the number of minimum objects at the leaf node of tree. In this paper, while maintaining the reliability of the training data we collected and tested directly, as an effort to make the method remain applicable to the current web environment where new contents of various types are constantly increasing, we configured the confidence factor - 'confidenceFactor' option in J48 - 70% instead of 100%. As shown in Table 2, considering that the main content of one document consists of 20 text blocks on average, we set to 10 the number of minimum objects at the leaf node of decision tree - 'minNumObj' option in J48, which is about 50% of the average number of main content text blocks.

To measure the accuracy of the main content extraction, we used F-measure, which is the most widely used in the evaluation of classification systems. Normally, in multi-class classification evaluation, for the final F-measure to be calculated, the F-measure for each class is calculated first and then the F-measure for each class weighted by the size or significance of the class is summed over the whole classes. Since in this paper we have the two-class classification problem, i.e. the main content and the non-main content, we can acquire the final F-measure by summing up the F-measures for the main content and the non-main content, weighted by the number of data per class. But, even in a multi-class classification when the data are disproportionately distributed as shown in Table 2, the F-measure of the total data can be swayed by the F-measure of the larger class. For example, in Table 2, the F-measure of the main content is 0.5 and the F-measure of the non-main content is 0.9. Thus, the F-measure of both classes based on the number of data is 0.84, which is much higher than 0.5. This means that even with the accuracy of main content classification, the figure that matters is quite low. The final value is relatively high due to the influence of the larger and irrelevant class. In this paper, since the main purpose is to identify the main content area with accuracy, we only adopt the F-measure of the main content class to compare our method with the existing methods and eventually prove the validity of our method.

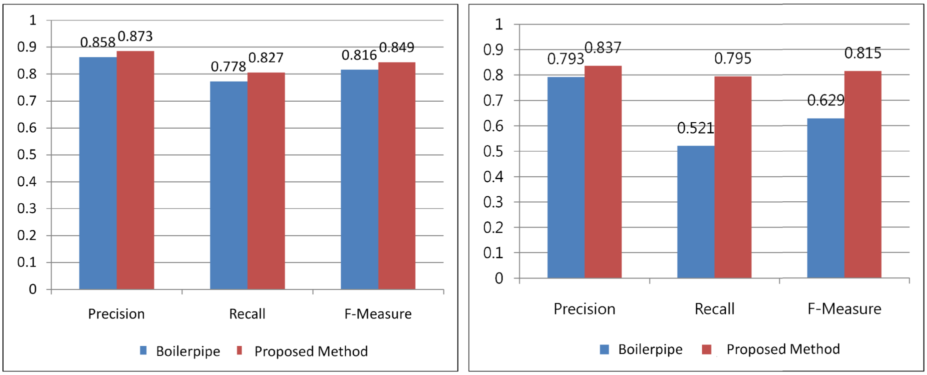
4.4 Performance Evaluation and Comparison

As mentioned in Section 2, according to [13], Boilerpipe [9] is the most efficient method for main content extraction. In this paper, we compared the proposed method with Boilerpipe using two data sets described in Section 4.2, However, since the reported performance of Boilerpipe is the result of multi-class classification with both classes of the main content and non-main content, to avoid the distortion resulting from any data disproportion, as mentioned in Section 4.3, the classification accuracy of Boilerpipe was re-calculated so that it only covers the main content class.

Table 3 and Figure 5 compare Boilerpipe and the proposed method in precision, recall, and F-measure, based on the two data sets.

Table 3. Performance comparison between Boilerpipe and the proposed method

	L3S-GN1 Data Set [19]			965-GWeb Data Set [20]		
	Precision	Recall	F-Measure	Precision	Recall	F-Measure
Boilerpipe [9]	0.858	0.778	0.816	0.793	0.521	0.629
Proposed Method	0.873	0.827	0.849	0.837	0.795	0.815



(a) Performance comparison based on L3S-GN1 dataset

(b) Performance comparison based on 965-GWeb dataset

Fig. 5. Performance comparison between Boilerpipe and the proposed method

As shown in the table and figure, the proposed method marked higher figures for all performance measures than Boilerpipe, known as the best performing method by far. For L3S-GN1, the F-measure of the proposed method is approximately 3% higher than Boilerpipe and 19% higher for 965-GWeb. The result proves the validity of the proposed method. Also, it shows that the proposed method can be more adaptable to various types of documents, because unlike Boilerpipe, the proposed method showed similar performance for both data sets.

Especially, the performance difference between Boilerpipe and the proposed method is significantly larger for the 965-GWeb data set than for the L3S-GN1 data set. The reason for this gap seems to be the fundamental difference between the two data sets. That is, L3S-GN1 is a collection of English news articles on the web, while 965-GWeb is a collection of web documents with various types and languages. It also means that the attributes such as word density and link density are effective for main content extraction of English news articles, while the proposed attributes of the neighboring text blocks have little contribution. However, the word density and link density may not be effective any more for the documents containing more diverse types of contents and languages, and the proposed attributes contribute significantly for main content extraction in this case.

5 Conclusion and Future Works

Extracting the main content, the important area of any document, out of the large group of web documents with free and diverse styles is a substantial and fundamental task to understand and analyze web documents better to achieve improved services. For this purpose, we proposed a new method to automatically extract the main content of a web document using the contextual features of text blocks. Through an experiment, we proved that the proposed method showed a better performance in F-measure than the currently known best performing method by 3% when applied to a publicly available data set and 19% when applied to the privately collected data set. The proposed method showed relatively similar performance for both data sets. The experiment result shows that the proposed method is efficient and more suitable for main content extraction in the current web environment.

We need future work to supplement the proposed method.

The method needs to be extended to automatically extract from web documents various document areas which users are interested in, including comments, advertisements, article lists or menu. For this, we need to find out more general attributes that are applicable to different areas of web documents. In addition, a study for performance enhancement of the main content extraction should be conducted. In this paper, we employed decision tree for classification. However, for enhanced performance an ensemble classifier should be investigated to integrate different classifiers.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (grant no. 2011-0010719)

References

1. (Februry 2013), `http://en.wikipedia.org/wiki/Document_Object_Model`
2. Deng, C., Shipeng, Y., Ji-Rong, W., Wei-Ying, M.: VIPS: a Vision-based Page Segmentation Algorithm. Microsoft Technical Report(MSR-TR-2003-79) (2003)
3. Suhit, G., Gail, E.K., David, N., Peter, G.: DOM-based Content Extraction of HTML Documents. In: 12th International Conference on World Wide Web, pp. 207–214 (2003)
4. Suhit, G., Gail, E.K., Peter, G., Michael, F.C., Justin, S.: Automating Content Extraction of HTML Documents. World Wide Web 8(2), 179–224 (2005)
5. Jeff, P., Dan, R.: Extracting Article Text from the Web with Maximum Subsequence Segmentation. In: The 18th International Conference on World Wide Web, pp. 971–980 (2009)
6. Stefan, E.: A lightweight and efficient tool for cleaning Web pages. In: The 6th International Conference on Language Resources and Evaluation (2008)
7. Stefan, E.: StupidOS: A high-precision approach to boilerplate removal. In: Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop, pp. 123–133 (2007)
8. Young, S., Hasan, J., Farshad, F.: Autonomic Wrapper Induction using Minimal Type System from Web Data. In: Artificial intelligence, pp. 130–135 (2005)
9. Christian, K., Peter, F., Wolfgang, N.: Boilerplate Detection using Shallow Text Features. In: The Third ACM International Conference on Web Search and Data Mining, pp. 441–450 (2010)

10. Jian, F., Ping, L., Suk Hwan, L., Sam, L., Parag, J., Jerry, L.: Article Clipper- A System for Web Article Extraction. In: 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 743–746 (2011)
11. Tim, W., William, H.H., Jiawei, H.: CETR - Content Extraction via Tag Ratios. In: 19th International Conference on World Wide Web, pp. 971–980 (2010)
12. Tim, W., William, H.H.: Text Extraction from the Web via Text-to-Tag Ratio. In: The 19th International Conference on Database and Expert Systems Application, pp. 23–28 (2008)
13. (July 2012), http://tomazkovacic.com/
14. W3C (February 2013), http://www.w3.org/TR/html401/
15. Jiawei, H., Micheline, K.: Data Mining: Concepts and Techniques. Morgan Kaufmann (2006)
16. Ian, H.W., Eibe, F.: Data Mining: Practical Machine Learning Tools and Techniques. Elsevier (2005)
17. Waikato Univ. (February 2013), http://www.cs.waikato.ac.nz/ml/weka/
18. Andy, C., Marc G.: (February 2012), http://nekohtml.sourceforge.net/
19. L3S Research Center (February 2013), http://www.l3s.de/~kohlschuetter/boilerplate/
20. (February 2013), http://121.78.244.168:8090/ice/index.jsp