JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 11, No. 2, June 2017, pp. 39-48

Main Content Extraction from Web Pages Based on Node Characteristics

Qingtang Liu, Mingbo Shao, Linjing Wu*, Gang Zhao, and Guilin Fan*
School of Educational Information Technology, Central China Normal University, Wuhan, China
liuqtang@mail.ccnu.edu.cn, shao@mingbo.de, wlj_sz@126.com, zhaogang@mail.ccnu.edu.cn, fanguilin@qq.com

Jun Li
School of Information Engineering, Hubei University for Nationalities, Enshi, China
lijun72213@163.com

Abstract
Main content extraction of web pages is widely used in search engines, web content aggregation and mobile Internet browsing. However, a mass of irrelevant information such as advertisement, irrelevant navigation and trash information is included in web pages. Such irrelevant information reduces the efficiency of web content processing in content-based applications. The purpose of this paper is to propose an automatic main content extraction method of web pages. In this method, we use two indicators to describe characteristics of web pages: text density and hyperlink density. According to continuous distribution of similar content on a page, we use an estimation algorithm to judge if a node is a content node or a noisy node based on characteristics of the node and neighboring nodes. This algorithm enables us to filter advertisement nodes and irrelevant navigation. Experimental results on 10 news websites revealed that our algorithm could achieve a 96.34% average acceptable rate.

Category: Information Retrieval / Web

Keywords: Content extraction; Web page; Text density; Hyperlink density

I. INTRODUCTION

With the development of the World Wide Web, the Internet has become the most important information resource for us. The explosion of web information makes the "rich data, poor knowledge" issue more critical. When we browse a webpage, a mass of irrelevant information such as advertisement, irrelevant navigations and trash information are included on the screen. This collectively irrelevant information not only poses a heavy burden to users, but also generates issues for web applications such as search engines and mobile applications [1]. Besides, with the development of mobile Internet, the requirements of resources for small-screen devices increase with each passing day [2]. If we can acquire content of a web page rapidly, we can develop many semantic applications based on it such as search engines, web page classifications [3], clustering, mobile web browsers, etc. How to quickly and accurately extract main content from web pages has become a hot research area in information extraction.

There are two types of commonly-used main content

Open Access http://dx.doi.org/10.5626/JCSE.2017.11.2.39 http://jcse.kiise.org

Received 16 October 2016; **Revised** 02 June 2017; **Accepted** 07 June 2017
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

extraction methods: methods based on templates and methods based on blocks. Methods based on templates need to extract a template from a group of similar web pages. Then this template can be used to extract useful information from web pages. Experimental results reveal that these kinds of methods are effective and precise [4]. However, there are limitations: first, a template can only be used on a specific kind of web page. Second, pages with different structure need to use different templates. Third, compared to its effect, the cost to learn how to use a template from web pages is relatively high. Besides these factors, web access has a strong randomness for users. So, methods based on templates cannot extract main content in real time if the web page is new. Methods based on blocks mainly contain these kinds of algorithms: document object model (DOM) based page segmentation [5-8], vision-based page segmentation [9, 10], specific tag based page segmentation [11, 12], hybrid methods [13], and semantic based page segmentation. DOM based page segmentation uses hierarchical relations in tags to extract the main content [5, 14]. Xpath can be used to locate content nodes in html where DOM is a kind of XML [15]. Text density is the most popular characteristic in DOM-based methods. However, only using text density may not be adequate. Text advertisement and comments may also have high text density, but they do not belong to the main content. Microsoft Research Asia proposed a function-based object model (FOM) to represent content on a web page [16]. Based on FOM, they proposed a Vision-based Page Segmentation Algorithm (VIPS) [9] to analyze the visual structure of web pages. Vision-based page segmentation uses visual tags and attributes such as background color, font color, font size, bold combined with hierarchical relation to cut the page into blocks. This kind of method can yield positive results in main content extraction. However, vision-based page segmentation needs to save vast visual information, so performance will decrease if page complexity grows rapidly. In the early days, the tag '<table>' was used to organize content on a web page [11, 12]. So, researchers could divide the web page into several blocks according to the tag <table>. This algorithm is simple and effective for the early pages. However, with the development of html5, is used more often than <table>. So, the <table>-based algorithm cannot handle those web pages that use instead. Hybrid methods combine two or more characteristics such as text density and visual information [13]. The combination method can yield a relatively positive effect. But time complexity is also relatively higher than single methods. Semantic-based page segmentation segments the page into different blocks according to semantic information of the page. Semantic information includes keywords [17], text features [18], vector space model, etc. Semantic based page segmentation can yield positive results on content extraction. But we need to process the text in the page to yield semantic information with the use of natural language processing technology. This may increase the complexity of content extraction.

Together, studies on web page subject extraction have already lasted for years. According to experimental results in reference [13], the average F-measure value of body text extraction method [19] is 80.83%. The k feature extractor based on DOM block segmentation method [5] is 68.73%. And the hybrid method in reference [13] can reach 95.21% in certain corpus. However, there are still shortcomings in existing algorithms. If we construct templates manually, then we can only extract certain kinds of web pages that depend on templates. If the templates are constructed by machine learning, the training process may have relatively high time and space complexity. Vision-based page segmentation methods also need to handle complex visual characteristics computing. So, in this paper, we propose an automatic main content extraction algorithm based on node characteristics. Node characteristics include text density and hyperlink density. And we also consider the characteristics of the neighbor nodes to identify continuous blocks. This algorithm does not need to train or calculate visual characteristics. And it can address most of the common web pages. Experimental results reveal that it can yield positive performance in news web pages. The remainder of this paper is organized into three sections. In Section II, we introduce the core algorithm of this paper. Section III depicts experimental results of this algorithm on some famous news websites. And finally, Section IV summarizes the conclusion and suggests future research.

II. MAIN CONTENT EXTRACTION ALGORITHM BASED ON NODE CHARACTERISTICS

In general, the news page always contains the following parts: navigation bars, main content, advertisement, copyright information, etc. Most users are mostly concerned with main content of a page. Other parts can be defined as noisy information. Generally, to acquire a better reading experience, similar information on a web page always has a certain degree of continuity [20]. That means, main content nodes are always near other main content nodes. Meanwhile, noisy information is always followed by noisy information. So, according to these characteristics, we propose a main content extraction algorithm based on the characteristics of the node and neighboring nodes. In this algorithm, we present a web page with the DOM. A node represents a pair of matched html tags and the content between them, such as ' ' and ' ', ' ' and ' ', '<table>' and '</table>', etc. We will calculate some characteristics of each node in DOM. Then according to the characteristics of each node and neighboring nodes, we will estimate if the node is a main content node or not. After the estimation of all

nodes, we can return main content nodes to users.

In this section, we first introduce the feature analysis of web pages. Then we present two key characteristics of the node in the document object model. And last, we describe the core algorithm based on two key characteristics.

A. Feature Analysis of Web Page

DOM is a common tool to represent web pages. In DOM, web pages are represented as a set of tags and the hierarchical relationship between these tags. According to the function of each tag, we classify the html tags into four categories:

1) Tags for interaction. This kind of tag allows users to create a message and interact with the server or the presupposed functions in the page. The commonly used interaction tags contain <script>, <noscript>, <applet>, <object>, <embed>, <input>, etc.

2) Tags for style. This kind of tag is used to personalize the style of the content on the page. Typical tags include: <b> for bold and <i> for italic.

3) Tags for metadata description. This kind of tag is used to describe the metadata and basic attributes for the whole web page. Typical tags are <head>, <title>, <meta>, <link>, <style>, etc.

4) Tags as a container. This kind of tag is always to arrange the content of a web page. In general, the core content is always put in these tags. Frequently-used tags contain , <table>, , , <td>, <tr>, etc.

In this study, our target is extracting the main content from a web page. So, we will construct the DOM according to the container-kind tags. Other kinds of tags such as tags for interaction, tags for style and tags for metadata will be blocked by filters at the first step.

After the construction of DOM for a web page, we need to choose characteristics to describe the web page. From microcosmic to macrocosmic, we can classify characteristics into four levels: (1) Basic characteristics of the node [13]. For example: the text length in a pair of and . The number of hyperlinks in a pair of and . (2) Basic characteristics of the whole page [8]. For example: the text density of a whole web page. (3) Rendering characteristics of the web page. Such as: the visual characteristics of a web page that can be observed in a browser. And (4) a common template for all the web pages in a website. We can find that the first two kinds of characteristics are easy to get. They can be obtained by calculation of the web page. Conversely, the last two kinds of characteristics are more difficult to get. They require more external information. Rendering of a web page may need external CSS files. And to find a template for a website needs to analyze many web pages in this site. So, to improve the performance of the algorithm, we use the first two kinds of characteristics to depict a web page. The exact calculation method is introduced in the following section.

B. The Calculation Method of the Characteristics

According to the analysis in Section II-A, we choose text density $\delta(b)$ and hyperlink density $\theta(b)$ to depict a web page. Their definitions are as follows:

DEFINITION 1. Text density $\delta(b)$ measures the number of text in a node b in DOM. Node b is a pair of container-kind tags. The calculation method is shown in formula (1):

$$\delta(b) = \begin{cases} \dfrac{T'(b)}{(L(b)-1)*maxLen} & L(b) > 1 \\ \dfrac{T(b)}{maxLen} & L(b) \le 1 \end{cases} \quad (1)$$

In this formula, $L(b)$ represents the number of lines in node b. $T(b)$ represents the number of characters when the number of lines in this node is 1. $T'(b)$ represents the number of characters without the last line when the number of lines in this node is more than 1. According to our typesetting habits, if the number of the lines in a paragraph is more than 1, then the last line will have long or short white spaces left. And in this line, the number of characters will be less than the other lines. So, we subtract 1 from the total number of lines in the formula to reduce the effect of the last line on the text density. *maxLen* represents the number of characters in one line on the screen at most.

DEFINITION 2. Hyperlink density measures the number of hyperlinks in a node b in DOM. The calculation method is shown in formula (2):

$$\theta(b) = \frac{Ta(b)}{T(b)} \quad (2)$$

In this formula, $Ta(b)$ represents the number of characters which are in the tags <a> and . $T(b)$ represents the total number of characters in this node.

C. Main Content Extraction Algorithm for the Web Page

1) Framework of Main Content Extraction

The framework of main content extraction is shown in Fig. 1. The main parts include: DOM Generator, DOM Processor, Node Fusion, Node Characteristics Analyzer, Node Filters, Filtering Result Analyzer, and Content Generator.

The basic process of the framework is as follows:

First step: DOM Generator generates a document object from the web page and saves a copy into the deposit of DOM.

Second step: DOM processor will do some pre-treatment on the document object that is generated in the first step. Pre-treatment includes: it will change the document object into a regular xhtml format, and remove the con-

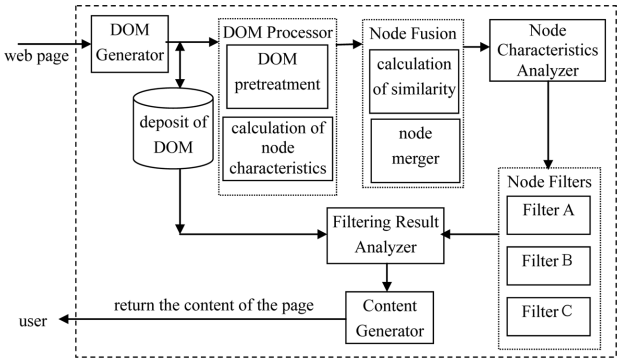


Fig. 1. Framework of main content extraction.

tent-irrelevant tags from the DOM such as tags for interaction, style and metadata. The content in the <title> tag and <h1> tags will be saved as the title of this page. After pretreatment, Dom Processor will calculate the characteristics of each node in the Dom and save them for the next step. The characteristics include text density $\delta(b)$ and hyperlink density $\theta(b)$ of each node.

Third step: Based on results of the second step, Node Fusion is used to merge those similar nodes according to a certain similarity estimation algorithm. The similarity estimation algorithm will be introduced in Section II-C-2.

Fourth step: After node fusion, Node Characteristics Analyzer is used to classify all the nodes according to the text density and hyperlink density. The algorithm will be introduced in Section II-C-3.

Fifth step: Node Filters are used to filter noisy nodes and part of multiple-layer container-kind-nodes. The specific method is introduced in Section II-C-4.

Sixth step: At last, Content Generator will return the information extracted in the fifth step according to the network and processing capacity of the terminal devices. If the terminal capacity and network are adequate, Content Generator will return both the text content and multimedia information such as image and video. Otherwise, Content Generator will only return text information.

2) The Mechanism of Node Fusion

Node Fusion includes two parts: the similarity estimation and node merger. The specific algorithm is as follows:

First step: We will get the text density $\delta(b)$ and hyperlink density $\theta(b)$ of each node from the deposit.

Second step: We will calculate the difference between each two neighboring nodes. The difference between node a and node b can be calculated by the following formula.

$$difference(a, b) = |(\delta(a) - \delta(b)) * \alpha| + |(\theta(a) - \theta(b)) * \beta| \quad (3)$$

In this formula, α and β represent weights of text density and hyperlink density in similarity. The sum of α and β is 1. Based on the experience, hyperlink density has more impact than text density in estimation of node type. So, in our experiment, we set α=0.2 and β=0.8.

Third step: Then we will estimate if the nodes are similar or not. If the difference between two nodes is less than empirical value ε, then we determine that these two nodes are similar and merge them into one node. After merging, we renew text density and hyperlink density of the new node. Otherwise, we determine that the two nodes are not similar enough to merge.

3) Node Characteristics Analyzer

Node Characteristics Analyzer is used to estimate the node type of each node in DOM. According to the characteristics of each node and its neighboring nodes. Node Characteristics Analyzer will estimate if the node is a

```
Input: A node T and its neighboring nodes
Output: flag of the node type
1: if t.linkdensity<θ
2:     if t.textdensity<δ
3:         if t.next.textdensity<δ
4:             if t.pre.textdensity<δ
5:                 flag=noisy node
6:             else
7:                 flag=content node
8:             endif
9:         else
10:            flag=content node
11:        endif
12:    else
13:        flag=content node
14:    endif
15: else
16:    flag=noisy node
17: endif
18: return flag
```

Fig. 2. Pseudocode of node characteristics analyzer.

content node or a noisy node. The principle of the Node Characteristics Analyzer is based on the following two empirical rules:

(1) Compared with noisy node, the content node always has higher text density and lower hyperlink density. The reason is that content node always has more text and less hyperlinks than catalog and advertisement.

(2) Nodes that belong to the same type always stay together. That means, if one node is a content node, then its neighboring node will be more likely to be a content node. Noisy nodes have the same phenomenon.

Based on the above rules, we formulate a set of rules to estimate the type of the node according to its characteristics and neighboring nodes. The pseudocode is shown in Fig. 2.

We set the value of θ as 0.333, and the value of δ as 0.5. The value of θ means that if more than one-third of the text in a node are hyperlinks, then the node is more likely to be a noisy node. The value of δ means that if the length of text in this node exceeds half of the screen, then the node is more likely to be a content node. Combined with the characteristics of its neighboring nodes, the results will be more accurate.

4) Node Filters and Filtering Result Analyzer

The specific working procedures of Node Filters and Filtering Result Analyzer are as follows:

(1) Space node filter is used to filter all the blank and invalid nodes which have no text content.

(2) Tag filters are used to filter nodes which contain invalid tags. These tags include invalid P, SPAN, TD, DIV, etc. Invalid tags refer to those tags which contain other container-kind tags. We will delete the outer tags and reserve the container-kind tags inside them.

(3) After filtering, Filtering Result Analyzer is used to analyze the results to generate main content of the web page. The mechanism is as follows: after filtering, get all the text from the content nodes, and calculate the text density and number of punctuation marks. If the text density is bigger than 0.3 and number of punctuation marks is bigger than 2, then return the text as main content of the web page, else go to step (4).

(4) Get an original DOM of the web page, then get all the <a> and pairs and return them as a catalogue.

III. EXPERIMENT AND ANALYSIS

A. Experimental Data Source

To test effectiveness and accuracy of the algorithm, we chose 976 news webpages from 10 websites for our experimental data. To get an impression of these pages, we calculate the average number of hyperlinks and characters in the main content of each website. The basic information of the 10 websites is shown in Table 1.

B. Evaluation Criterion

We use the following criterion to evaluate the experimental results.

To judge the extracted results, we first extract the main content by manual labor. Two graduate students are invited to extract the main content of the web pages separately manually. If their results on a web page are the same, then this result will be saved as a standard manual result. If their results are not the same, they will be asked to negotiate with each other until they can reach a consistent result. Then this result will be saved as the standard manual result. Then we will compare the manual results and the machine-extracted results. According to the differences between them, we divide the results into four categories:

(1) If the machine-extracted result is the same as the

Table 1. Data source

Website	URL	Number of webpages	Average number of hyperlinks per page	Average number of characters in the main content per page
SINA	http://www.sina.com/	116	136	1,239.5
JRJ	http://www.jrj.com.cn/	98	142	1,341
IFENG	http://www.ifeng.com/	107	235.5	749.3
163	http://www.163.com/	107	406.7	2,496.5
SOHU	http://www.sohu.com	103	291.5	648.5
QQ	http://www.qq.com	103	149.2	599.3
HUANQIU	http://www.huanqiu.com	63	70.5	659
XINHUA	http://www.xinhuanet.com	93	73.5	2,701.3
CHINANEWS	http://www.chinanews.com	88	156.5	1,501
HEXUN	http://www.hexun.com	98	76.5	1,034

manual result, then we consider that this is an accurate extraction.

(2) If the program can precisely predict that the web page contains no main content, then we consider that this is an accurate extraction. These kinds of situations may include: the webpage is used to show pictures and has few descriptions. The webpage is an interactive multimedia page such as flash or video.

(3) If the difference between the manual result and the machine-extracted result is small, and there is no influence for users to understand the main content, then we consider that this is an acceptable extraction. The specific criteria are as follows:

$$\frac{|N_{manual}-N_{machine}|}{N_{manual}}<5\% \quad (4)$$

In this formula, N_{manual} represents the number of words of the manual result. $N_{machine}$ represents the number of words of the machine extracted result.

(4) Other situations except the above three situations are considered as wrong extractions.

Based on those four situations, we define two indicators to measure the accuracy of extraction.

$$Precision\ rate=\frac{N_{accurate}}{N_{total}} \quad (5)$$

$$Acceptable\ rate=\frac{N_{accurate}+N_{acceptable}}{N_{total}} \quad (6)$$

In these two formulae, $N_{accurate}$ represents the number of web pages which are accurate extractions. $N_{acceptable}$ represents the number of web pages which are acceptable extractions. N_{total} represents the total number of web pages.

Besides, to evaluate the efficiency of algorithms, we also record the time that the program needs to process each page. Because the time to connect the http server may be varied and affected by different factors such as the speed of the Internet, the response time of the server, we first download all the pages from the websites and store them onto hard disk. Then these pages are extracted by the awaited algorithms and the cost of time is recorded. The average cost of time of all the pages in each website is calculated as part of the results.

C. Experimental Result

To check the effectiveness of the algorithm, we use two kinds of algorithm on the same dataset. The first one is our algorithm: the automatic extraction based on node characteristics algorithm. The other algorithm is the extraction algorithm based on vision and text leafs ratio. The visual segmentation references the VIPS [9]. This algorithm first splits the web page into several blocks according to the vision information. Then the percentage of text nodes in all the leaf nodes in each block is used to judge whether this block is main content or not. The basic process of the algorithm is shown in Fig. 3.

Input: DOM of the webpage
Output: main content of the webpage
1: get the outmost node D of the DOM.
2: add D into the list Q
3:***while*** (Q is not empty)
4: get next node q in the list Q
5: ***if*** (q has no child anymore || all the children nodes in q are text nodes || size of q is less than relative size threshold)
6: add q into list C
7: remove q from list Q
8: ***continue;***
9: ***endif***
10: ***if*** q contains tag "<hr/>"
11: spilt q into several sub nodes according to the "<hr/>" tags
12: remove q from list Q
13: add all the new sub nodes into Q
14: ***endif***
15: ***if*** q is an inline node and q has a child which is a line-break node
16: spilt q into several sub nodes according to the "line-break" node
17: remove q from list Q
18: add all the new sub nodes into Q
19: ***endif***
20: ***if*** q has a child which has the different background color with it
21: spilt this child from the parent node
remove q from list Q
22: add all the new sub nodes into Q
23: ***endif***
24: add q into list C
25: remove q from list Q
26:***endwhile***
27: sort all the nodes in C according to the original sequence of the nodes in the original DOM
28:***for*** each node c in C
29: ***if*** c contains "<h>" tag
30: keep the text in c as title
31: ***endif***
32: ***if*** the text leafs ratio in c exceeds the threshold
33: add the text in c into main content
34: ***end if***
35:***endfor***
36: ***return*** main content

Fig. 3. Pseudocode of the vision and text leafs ratio algorithm.

Table 2 shows the results of the extraction algorithm based on vision and text leafs ratio. And Table 3 shows the results of the automatic extraction based on node characteristics which we proposed. The results include the precision rate, acceptable rate and average cost of time for each website.

From the experimental results, we can see that the average acceptable rate of extraction algorithm based on vision and text leafs ratio is 94.42% and the acceptable rate of our algorithm is 96.34% which is a little higher.

Table 2. Experimental results of the extraction algorithm based on vision and text leafs ratio

Website	Total webpages	Accurate extractions	Acceptable extractions	Wrong extractions	Precision rate (%)	Acceptable rate (%)	Average cost of time (ms)
SINA	116	99	10	7	85.34	93.97	402
JRJ	98	80	13	5	81.63	94.90	381
IFENG	107	91	11	5	85.05	95.33	298
163	107	90	12	5	84.11	95.33	407
SOHU	103	90	7	6	87.38	94.17	403
QQ	103	91	9	3	88.35	97.09	371
HUANQIU	63	50	11	2	79.37	96.83	346
XINHUA	93	78	10	5	83.87	94.62	315
CHINANEWS	88	70	12	6	79.55	93.18	345
HEXUN	98	74	13	11	75.51	88.78	351
Total	976	813	108	55	83.02	94.42	361.9

Table 3. Experimental results of automatic extraction based on node characteristics algorithm

Website	Total webpages	Accurate extractions	Acceptable extractions	Wrong extractions	Precision rate (%)	Acceptable rate (%)	Average cost of time (ms)
SINA	116	104	8	4	89.66	96.55	360
JRJ	98	85	11	2	86.73	97.96	345
IFENG	107	87	16	4	81.31	96.26	280
163	107	89	15	3	83.18	97.20	387
SOHU	103	88	13	2	85.44	98.06	376
QQ	103	90	8	5	87.38	95.15	347
HUANQIU	63	53	7	3	84.13	95.24	329
XINHUA	93	79	11	3	84.95	96.77	291
CHINANEWS	88	70	13	5	79.55	94.32	322
HEXUN	98	78	16	4	79.59	95.92	329
Total	976	823	118	35	84.19	96.34	336.6

As to the precision rate, our method is also higher than the algorithm based on vision and text leafs ratio. The value of the former is 84.19%, while the latter is 83.02%. The reason is that if the noisy block is mostly composed of text such as text advertisement or user comments, the algorithm based on vision and text leafs ratio may misjudge it as main content. But our method can identify it according to its neighboring nodes if its neighboring nodes are advertisement.

Besides, the average cost of time of the algorithm based on vision and text leafs ratio for the 10 websites is 361.9 ms, and our algorithm is 336.6 ms. From our analysis, the main reason is that our algorithm discards tags for style at the beginning. This step will block computation complexity when the web page is complex.

After extraction, we remove most of the advertisement, copyright information and irrelevant navigation. The main content of each web page and the most important navigation such as HOME and GO BACK are reserved. The average size of web pages after extraction is 4 kB. This can meet the requirements even for mobile devices using GPRS.

However, there are still some situations that our algorithm cannot handle well. For example, if most of the content in the web page is generated by JavaScript dynamically, then our algorithm cannot extract this kind of content.

D. Practicability on Mobile Devices

To display practicability of our algorithm, we designed a web agent for mobile devices. When the mobile device

Table 4. Survey of the practicability of the web browse agent (unit: %)

Question	Strongly agree	Agree	No opinion	Disagree	Strongly disagree
I think this agent is useful for internet surfing.	26.09	65.22	0.00	4.35	4.35
The main content extracted for me is accurate in most of the time.	21.74	65.22	4.35	8.70	0.00
The processing speed of the agent is acceptable for me.	17.39	65.22	13.04	4.35	0.00
When the Internet connection is slow, I would prefer to use this agent instead of getting the original page.	34.78	52.17	4.35	4.35	4.35

submits a web URL request to the agent, the agent will extract the main content of the web page, and then return the main content to the device. We invited 23 undergraduate students to use this agent on their mobile phones and browse 15 web pages from at least 5 websites. Then we asked them to fill a small questionnaire which is a 5-point Likert-type scale to gather their opinions about the agent. Table 4 is the result of this survey.

From the results, we can see that 91.30% of users strongly agree/agree that the agent is useful for them. The 86.96% of users strongly agree/agree that the extracted main content is accurate in most of the time and they would prefer to use it when the Internet connection is not very good. The 82.61% strongly agree/agree that the processing speed is acceptable. We conducted an interview to determine why users chose to disagree or strongly disagree. The reasons mainly focused on the following two aspects: the first situation was that if the page was pictures news, then just providing a little text was not enough to get the main idea. The second situation was that some users suggested that if the main content was lengthy, they would prefer to get an illustrated picture and abstract. This will be our focus in the next phase.

IV. CONCLUSION

Content extraction of web pages has critical theoretical and practical significance for information processing and retrieval. Especially, it can be used in web surfing for mobile devices. The purpose of this paper is to propose an automatic main content extraction method of web pages. This method uses two indicators to depict the characteristics of web pages: text density and hyperlink density. Based on these two indicators, we designed an extraction algorithm to traverse all nodes in the DOM and classify all nodes into two categories: content nodes and noisy nodes. All the text in the content nodes will be extracted as the main content of the web page.

To verify the validity of the algorithm, we choose 976 web pages from 10 news websites as experimental data. Experimental results revealed that our algorithm can get 96.34% as acceptable rate. After extraction, the file size of the web page can be reduced to 4 kB on average. And it can be used in practical applications. With the help of this algorithm, mobile devices using GPRS can also capture good web surfing experiences. In the next phase, we will focus on how to extract the abstract and compress audio and video for mobile devices.

ACKNOWLEDGMENTS

This paper was supported by National High Technology Research and Development Program "Key Technologies and System in Elementary Mathematics Problem Solving" (No. 2015AA015408), National Science & Technology Supporting Program "Research on interactive virtual exhibition technology for Tujia Nationality's Brocade Culture" (No. 2015BAK27B02), Chinese Education Ministry's New Century Excellent Talents Supporting Plan (No. NCET-13-0818), and Fundamental Research Funds for the Central Universities of CCNU (No. CCNU16A05023 & CCNU15A02020). This work was also supported by Hubei Collaborative Innovation Center of Basic Education Information Technology Services Project "Research on Key Technologies in Smart Interaction Classroom" and National Natural Science Foundation of China (No. 61562025).

REFERENCES

1. A. Bhardwaj and V. Mangat, "An improvised algorithm for relevant content extraction from web pages," *Journal of Emerging Technologies in Web Intelligence*, vol. 6, no. 2, pp. 226-230, 2014.
2. J. O. Wobbrock, J. Forlizzi, S. E. Hudson, and B. A. Myers, "WebThumb: interaction techniques for small-screen browsers," in *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*, Paris, France, 2002, pp. 205-208.
3. W. Petprasit and S. Jaiyen, "E-commerce web page classification based on automatic content extraction," in *the Proceedings of 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Songkhla, Thailand, 2015, pp. 74-77.
4. A. Schieber and A. Hilbert, "Process model for content extraction from Weblogs," *International Journal of Intelligent Information Technologies*, vol. 10, no. 2, pp. 20-36, 2014.
5. S. Debnath, P. Mitra, and C. L. Giles, "Identifying content

blocks from web documents," in *International Symposium on Methodologies for Intelligent Systems*. Heidelberg: Springer, 2005, pp. 285-293.
6. S. Gupta, G. Kaiser, D. Neistadt, and P. Grimm, "DOM-based content extraction of HTML documents," in *Proceedings of the 12th International Conference on World Wide Web*, Budapest, Hungary, 2003, pp. 207-214.
7. F. Sun, D. Song, and L. Liao, "DOM based content extraction via text density," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Beijing, China, 2011, pp. 245-254.
8. D. Insa, J. Silva, and S. Tamarit, "Using the words/leafs ratio in the DOM tree for content extraction," *Journal of Logic and Algebraic Programming*, vol. 82, no. 8, pp. 311-325, 2013.
9. D. Cai, S. Yu, J. R. Wen, and W. Y. Ma, "VIPS: a vision based page segmentation algorithm," Microsoft Corporation, Redmond, WA, *Technical Report MSR-TR-2003-79*, 2003.
10. L. Q. Chen, X. Xie, W. Y. Ma, H. J. Zhang, H. Q. Zhou, and H. Q. Feng, "DRESS: a slicing tree based web representation for various display sizes," Microsoft Corporation, Redmond, WA, *Technical Report MSR-TR-2002-126*, 2002.
11. Z. Ahmad and J. L. Hong, "Mobile web browsing techniques," in *Neural Information Processing*. Heidelberg: Springer, 2012, pp. 283-291.
12. S. H. Lin and J. M. Ho, "Discovering informative content blocks from Web documents," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Canada, 2002, pp. 588-593.
13. D. Song, F. Sun, and L. Liao, "A hybrid approach for content extraction with text density and visual importance of DOM nodes," *Knowledge and Information Systems*, vol. 42, no. 1, pp. 75-96, 2015.
14. A. Pouramini and S. Nasiri, "Web content extraction using contextual rules," in *Proceedings of 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, Tehran, Iran, 2015, pp. 1014-1018.
15. B. Sun, "A study of main contents extraction from web news pages based on xpath analysis," *Journal of the Korea Society of Computer and Information*, vol. 20, no. 7, pp. 1-7, 2015.
16. J. Chen, B. Zhou, J. Shi, H. Zhang, and F. Qiu, "Function-based object model towards website adaptation," in *Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, 2001, pp. 587-596.
17. J. Park, J. Kim, and J. H. Lee, "Keyword extraction for blogs based on content richness," *Journal of Information Science*, vol. 40, no. 1, pp. 38-49, 2014.
18. Z. Y. Xiong, X. Q. Lin, Y. F. Zhang, and Y. A. Man, "Content extraction method combining web page structure and text feature," *Computer Engineering*, vol. 39, no. 12, pp. 200-203, 2013.
19. A. Finn, N. Kushmerick, and B. Smyth, "Fact or fiction: content classification for digital libraries," in *Proceedings of Joint DELOS-NSF Workshop: Personalization and Recommender Systems in Digital Libraries*, Dublin, Ireland, 2001.
20. W. Song and M. Kim, "A text block context information based multiple Web contents extraction," in *Proceedings of IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Paris, France, 2015, pp. 1-8.

Qingtang Liu

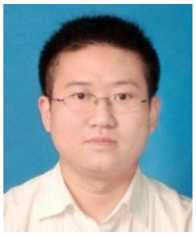
Qingtang Liu received his Ph.D. degree from Huazhong University of Science and Technology in 2005. He is working as a Professor at Central China Normal University in the School of Educational Information Technology. His research interests include data mining, text processing and e-learning.

Mingbo Shao

Mingbo Shao received his Master's degree from School of Educational Information Technology, Central China Normal University in 2012. He is working as a software engineer in Ant Financial Services Group. His research interests include mobile learning and cloud computing.

Linjing Wu

Linjing Wu received her Ph.D. degree from Central China Normal University in 2013. She is working as a Lecturer at Central China Normal University. Her research interests include educational data mining, text processing and e-learning.

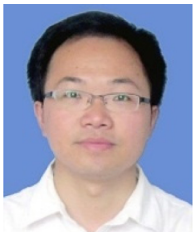


Gang Zhao

Gang Zhao received his Ph.D degree from Huazhong University of Science and Technology in 2005. He is working as a Professor at Central China Normal University in the School of Educational Information Technology. His research interests include machine learning, data mining and computer vision.

Jun Li

Jun Li is a professor at School of Information and Engineering, Hubei University for Nationalities. He received a Master's degree in Computer Science from Huazhong University of Science & Technology in 1999. His research interests include image processing, computer graphics and digital protection of intangible cultural heritages.



Guilin Fan

GuiLin Fan is a PhD candidate in School of Educational Information Technology, Central China Normal University. His research interests include educational data mining, learning analysis and natural language processing.