An Approach to Main Web Content and Keyword Extraction to Develop a Vietnamese Contextual Advertising Engine

Nguyen Quy Minh, Le Hoai Bac
Faculty of Information Technology, University of Natural Science, VNU Ho Chi Minh City, Viet Nam.
lhbac@fit.hcmus.edu.vn

***Abstract*: Contextual advertising is a next-generation of online marketing. A contextual advertising system scans the text of a web page for keywords and returns advertisements to that web page based on what the user is viewing. For example, if the user is viewing a web page pertaining to sports, the user may see advertisements for sports-related things, such as sport equipments or sporting events. To do so, we face two main problems: detecting the main content of the web page, and extracting keywords from Vietnamese text automatically. For the first problem, we proposed a method of web page segmentation by employing histogram scheme. For the second problem, we have measured and combined the local statistic together with global statistic of each term to estimate the importance of a term, and then determine the main keywords. Our methods were evaluated and compared with others based on the following measures: precision, recall, and F-measure. Received results are very positive.**

Keywords: page segmentation, keyword extraction, contextual advertising.

I. INTRODUCTION

In traditional advertising model, advertisers need to manually contact the web master to put their advertisements (ads) into the appropriate location on the web page. This is actually not efficient because the ads may be placed on articles unrelated to it. This also means that their ads are shown to the irrelevant user. Contextual advertising is a new model of targeted advertising for advertisements. The ads themselves are selected and served by automated systems based on the current content of the web page.

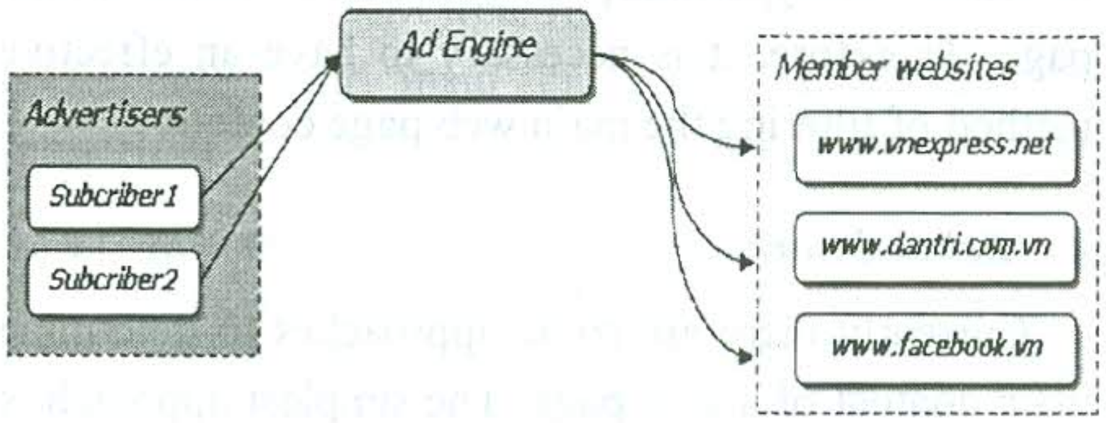

Figure 1. Contextual advertising model

In this paper, we try to develop a middle engine to automatically distribute the ads from advertisers to relevant Vietnamese web pages. Firstly, we must process to extract the main content of the web page. While the presentation of HTML format is convenient for human users, this is not particularly convenient for the automatic processing because it contains a large amount of irrelevant information. Secondly, we extract Vietnamese keywords from the main content that has just extracted out in the first step above. Then we make the comparison with the available keywords of ads to pick out the ads matching the keywords of the web page. These methods will be discussed in detail in the next sections.

II. CONTENT EXTRACTION FROM THE WEB PAGE

The fact that a huge amount of information is uploaded into Internet today has triggered a demand to research and organize such information effectively in a fast manner. Several applications such as Search Engine, RSS, Feedback system, document summarization, bilingual search, etc. require the content of the Internet to be gathered, processed and

stored quickly and efficiently. This effort is often hampered by the use of structure tags in HTML and XML. These tags are meaningful only to the browser that renders the web page, but bear little semantic meaning to the end user. Besides, a real big challenge to the application developers is that not all contents of the web page are necessary. Web pages are usually "noised" with different kinds of information. It will not be effective if we just simply filter out HTML markup tags. It still contains a lot of unnecessary information such as menu bars, advertisements, messages, or hyperlinks, etc. It makes a "noise" in the page. Therefore, it is necessary to have an effective method of filtering the main web page content.

A. Related work

Currently there are some approaches to determine main content of a web page. The simplest approach is filtering all text contents of the page by analyzing the HTML tag in the page to build a document tree (DOM)[1] reflecting the page content. Nodes on the DOM tree represent different elements of the web page. Therefore, the main content of the web page can be extracted by combining nodes with tag "TEXT". This approach can extract all text content of the web page only, but cannot filter the junk contents. The second approach is comparing two formats [10] of two web pages based on format recognition method. This approach compares a sample standard web page with a page to be filtered for content, to determine the common format of two web pages, and then filter out the regions which are decided to be the main content on the sample page. However, the different sample web pages must be available for each domain, which is also considered a drawback of this approach. Another approach is to evaluate the node importance [4,5]. This approach is based on previous method of analyzing HTML tag to create a DOM tree. We will determine which node contains the main content of the web page, and gives a certain score to each node after processing the natural language of that node. The rule of giving score is: firstly, give score for nodes labeling with tag TEXT only because only these nodes contain real contents, other nodes are summarized by these nodes; secondly, based on the number of sentences of that node's contents (more sentences, higher score), and the node must contain at least one paragraph (although the determination of one paragraph is still heuristic). Score of the parent node will be the sum of its child nodes. Therefore, the main content will be determined by finding the deepest node on the tree with the highest score. This approach is effective, but some main contents may be missed if they are located on different independent nodes of the DOM tree. Another approach is to analyze the ratio of tag TEXT and tag HTML on each row of the web page [8]. Then the main content is determined by selecting rows with highest ratio of tag TEXT. However, it is still not sure how to determine one "row" of the page. Another approach is based on visual segment of the web page. This approach will divide the page into different segments by vision-based approach, using VIPS algorithm (Vision-based Page Segmentation) [2] developed by Microsoft Lab. The automatic division of the web page is based on the continuity of nodes on DOM tree and some heuristic judgments. Then we will decide main contents of the web page by determining the importance of each segment of the web page with machine learning or based on defined characteristics of the segment [6]. This approach is considered as one strong and most effective approach, but it is still complicated and difficult to implement if the input is HTML tag only and without the support of the browser.

B. Histogram-based segmentation of web page

This paper will approach the page content segmentation by exploiting histogram schema. This approach will be partly overcome the disadvantages of the above approaches. By analyzing the DOM tree of the web page, we can extract all text nodes together with its weight and represent them as a histogram.

[1] Document Object Model (DOM): is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML and XML documents.

Then, we can enhance that extracted histogram by using mean filter method to remove text nodes with low weight and avoid missing text nodes with high weight. Finally, based on the heuristic statement that "main content of the web page is the area containing the highest concentration of text", we conducted the clustering on this histogram by using K-means method and select the highest cluster, which is a collection of text nodes, to be considered as the main content of the web page. Figure 2 describes this process.

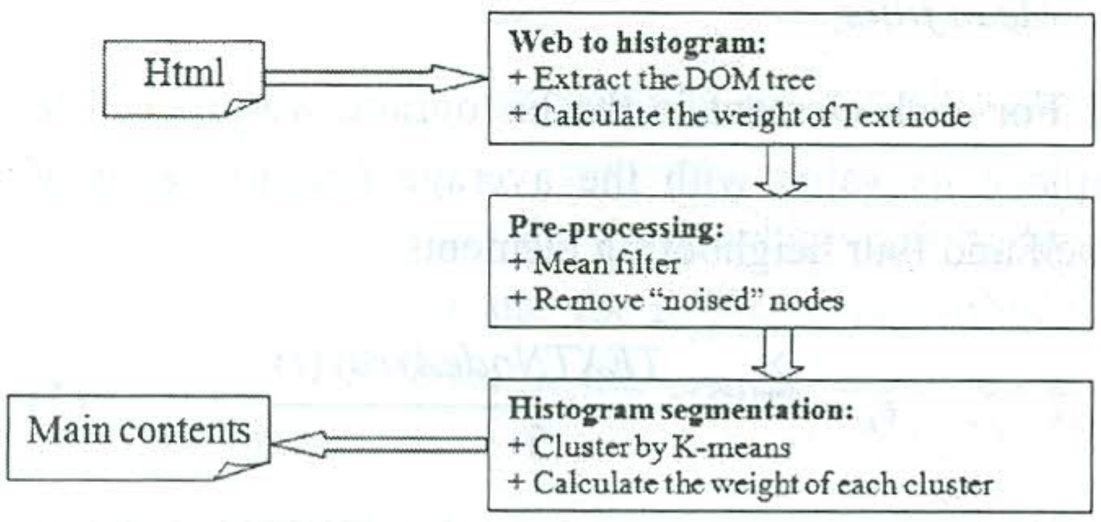

Figure 2. Model of extracting main contents

1) Represent web page as a histogram

We digitize a web page as a histogram scheme. Each node in the DOM tree will be processed by using depth-first search tree approach. First, delete all invisible nodes that cannot be seen on the browser by users, such as node labeling with tags <SCRIPT>, <STYLE>, etc. After that, we extract all Text nodes because only these nodes will contain the valuable content. Finally, we re-arrange these nodes as an array of Text nodes together with their weights. Figure 3 describes pseudo-code of the algorithm.

The weight of node

The weight of a node is considered as the importance of that node in DOM tree. In this paper, it is measured as a heuristic method that is the size of node (length of the text contained in the node). Note that depending on context, we may improve the accuracy of algorithm with a more accurate description by combining more other attributes, such as the position of the node, the format of node, node seamless with the surrounding, etc. The higher measurement, the more accuracy of the algorithm.

```
Input
    DOM ← HTML source code
Begin
Remove all InvisibleNodes.
      For each node in DOM tree:
            If (node is VirtualTextNode) then
                        TEXTNodeArray[i] ←
Weight (node)
            Else
                        TEXTNodeArray[i] ← 0
      Return TEXTNodeArray
End
```

Figure 3. Web to histogram algorithm

In the above algorithm, we also define some type of nodes in the DOM tree. In this paper, we divide the nodes into four main categories: InvisibleNode, InlineNode, TextNode, and VirtualTextNode.

- InvisibleNode: this node cannot be seen by user, it is just meaningful for the browser (e.g. nodes with tag <SCRIPT>, <STYLE>, empty node, line break node, etc).
- InlineNode: the node with inline text Html tags, which only affect the appearance of text and can be applied to a string of characters without introducing any line breaks (e.g. nodes with tag <B>, <I>, <FONT>, etc).
- TextNode: the node corresponding to the free text, which does not have an html tag (e.g. the text "I go to school" is a text node).
- VirtualTextNode: this node like TextNode, but it can contain InlineNode. VirtualTextNode detection is very important. If it is not identified correctly, we may lose text nodes which have short content and thus skew the result.

Based on the algorithm described in Figure 3, we will be able to build an array of text nodes containing text extracted from the page. Each element of array is a text node. Then we will perform the histogram scheme from this array. As an example, consider a news article posted on 18/8/2009 at http://vnexpress.net/GL/Doi-song/2009/07/3BA119E9/. This web page is similar to many other pages on the Internet. The title banners, images, navigations, and advertisements take up most of the space on the web while its main content is only confined to relatively space in the left. Ads, links, and administrative

information are placed at the bottom of the page. When we analyze this page with the algorithm, we obtain the histogram chart.

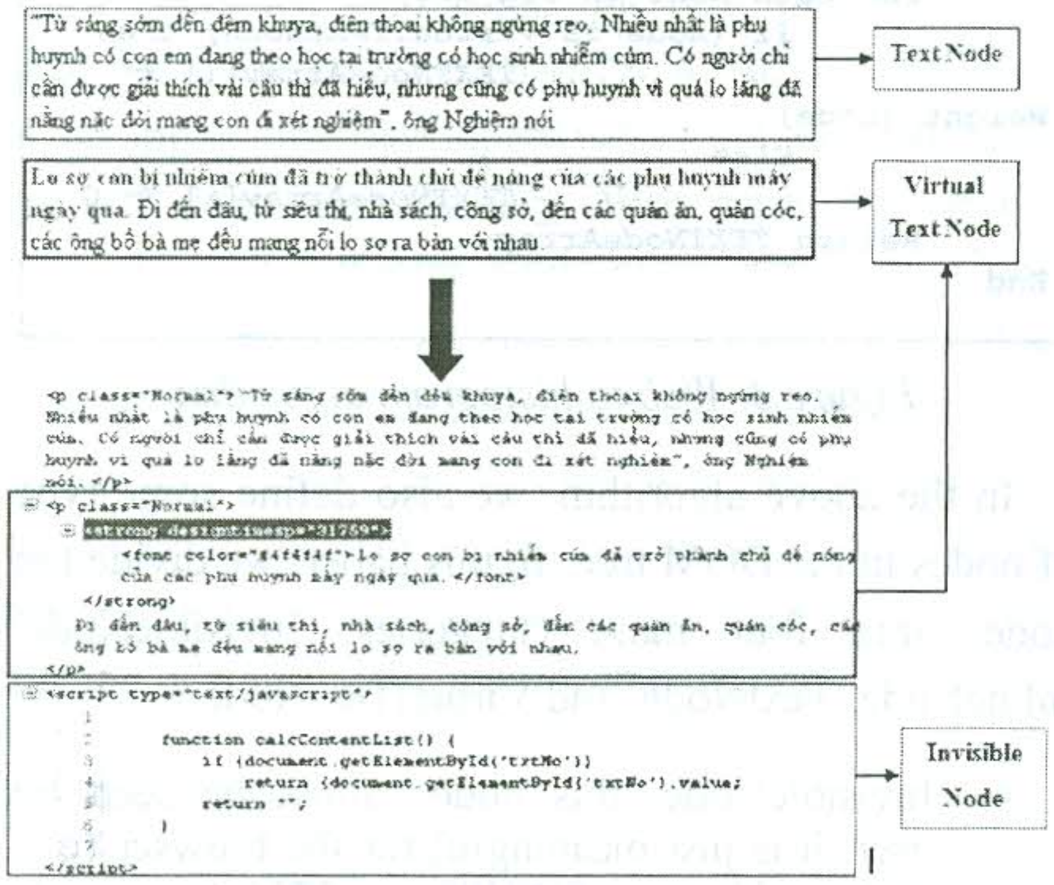

Figure 4. Example of node types

In the Figure 4, the X axis is the order of nodes in the array (it is also the order of the nodes in the DOM tree), and the Y axis is the weight of nodes (length of the node's content). Based on the diagram, we found that the area containing nodes located from position 23 to 67 in Figure 5 denotes the main content of the page. Evaluated on some other sites, we also have a similar comment. Therefore, based on this idea, we will conduct to extract the main content of the page by collecting the content of all nodes from 23 to 67, which are the most concentrated high-density nodes. To do this, we apply the clustering technique to filter out these nodes properly.

2) Histogram pre-processing

Before doing clustering, we apply the Mean filter algorith[2] to enhance the histogram schema first. This will help us avoid the loss of important short nodes like node containing article title, brief content, etc... that may be lost in the process of clustering. Furthermore, this step also helps us remove redundant nodes as well.

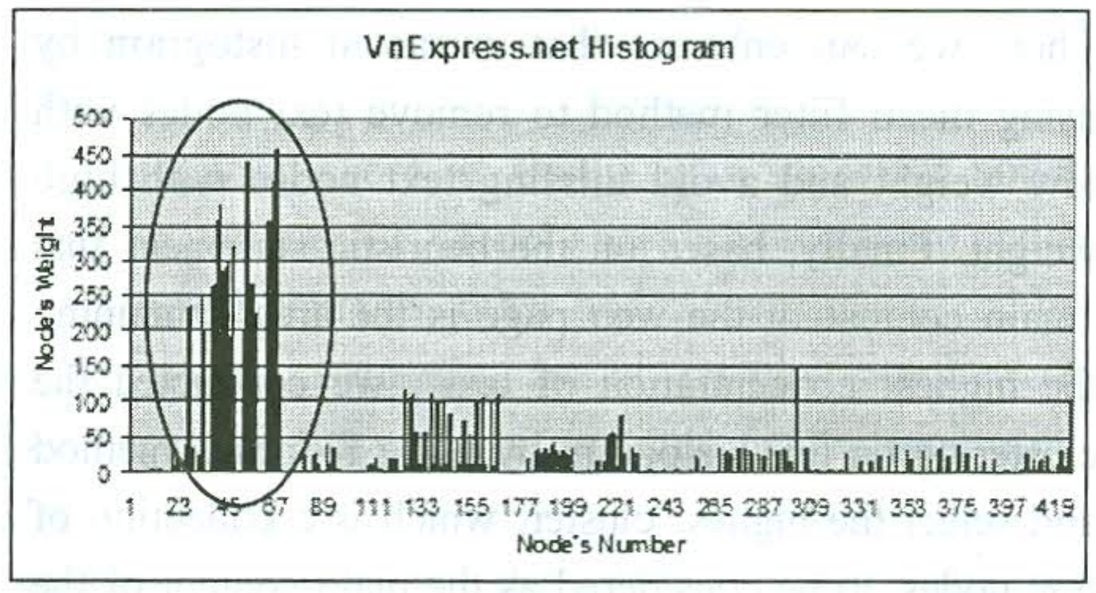

Figure 5. Histogram of VnExpress page

Mean filter

For each element in the histogram, we proceed to replace its value with the average (mean) value of itself and four neighboring elements.

$$e_k = \frac{\sum_{i=k-2}^{i=k+2} TEXTNodeArray(i)}{5} \quad (1)$$

With e_k is the element k in the TEXTNodeArray array.

After applying the mean filter algorithm, we have the following histogram (Figure 6):

The result shows that the histogram in Figure 6 is more smoothly than in Figure 5. All elements which have low weight scattered from position 89 onwards were smooth down while elements from position 23 to 67 were enhanced more prominent than in Figure 5. This makes us easily remove the redundant contents (the elements that are located below the average threshold) and focus on the remaining elements, which could contain the main content of the page.

Reduce noise

The average threshold is defined as the average value of all elements in the histogram. Average threshold is represented by the red horizontal line in above histogram (Figure 6). We remove all nodes which have the value below the average threshold by setting its value to zero.

[2] Reduce noise by using Mean filter: http://homepages.inf.ed.ac.uk/rbf/HIPR2/mean.htm

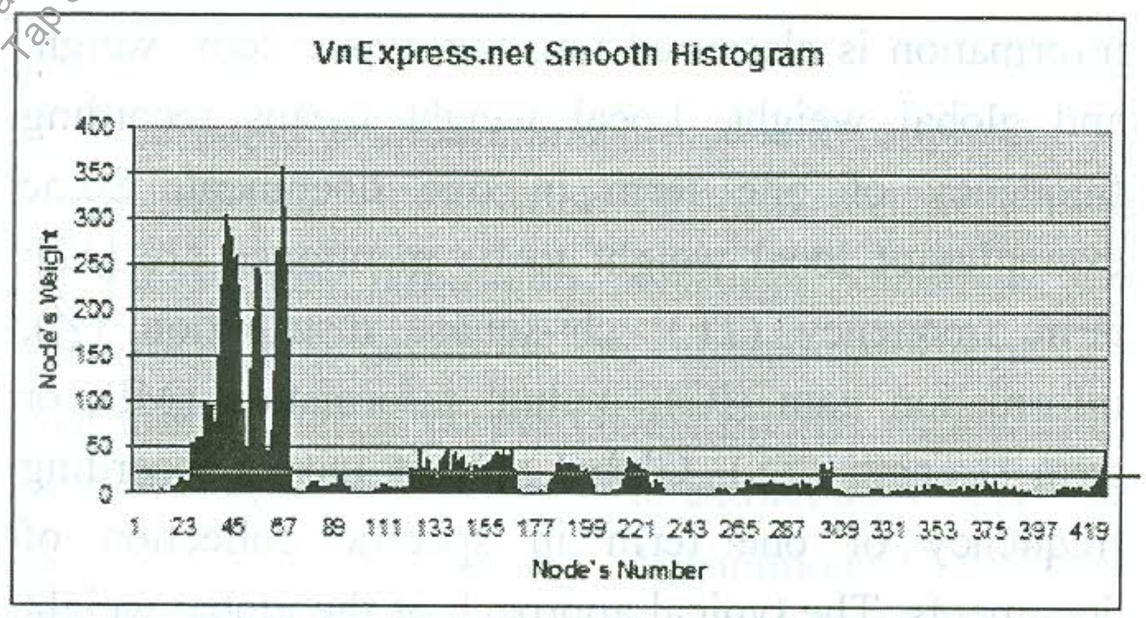

Figure 6. Histogram after using mean filter

3) Histogram clustering

After doing the pre-processing, we use clustering technique to gather all elements in schema into groups, and then extract the group which gets highest weight. We decided to use the K-Means algorithm to group all relevant nodes so that we can easily extract these nodes later.

Table 1. Experimental results of extracting main content with various evaluations

Methods / Measures	Average Precision (%)	Average Recall (%)	Average F_1-measure (%)
No histogram preprocessing, group into 2-clusters	45.57	86.52	58.41
No histogram preprocessing, group into 3-clusters	45.58	86.51	58.42
No histogram preprocessing, group into 4-clusters	45.56	86.52	58.41
Histogram preprocessing, group into 2-clusters	69.47	79.94	71.32
Histogram preprocessing, group into 3-clusters	74.10	80.32	76.04
Histogram preprocessing, group into 4-clusters	75.28	78.60	76.03

With n elements *(xi, yi)* in this scheme, it is easy to use K-Means to collect them into 3-clusters (the distance that used in the K-Means is the Euclidean distance). We selected 3 as the number of clusters for K-Means. We could also select 2, 4… as the number of clusters, but the experimental result below show that 3 is the most suitable (Table 1). This is appropriate since the layout of the web page is usually separated into 3 main areas: the introduction area, the main area containing the main content, and the rest area containing other minor information like advertisements, links, etc. Then we determine the "highest cluster" by calculating the average value of each cluster, then select the cluster that has taken the highest average value. This is also the cluster that contains the main content of web page. From this extracted cluster, we will easily retrieve the text content from its nodes. Finally, we remove all redundant HTML tags (if any, caused by InlineNode) to get the complete result (Figure 7).

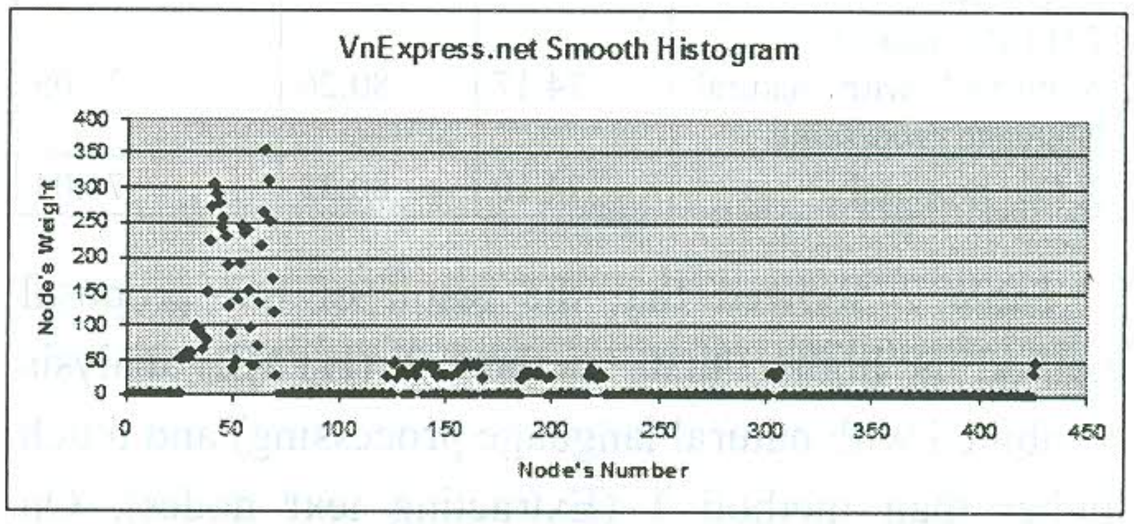

Figure 7. Represent in points of histogram. Page is segmented by clustering all relevant nodes into groups

4) Experimental result

Experimental data are the first 57 Vietnamese web pages taken from Google when searching with the keyword "thông tin" (information). Next, we detected and selected the main content of each page manually and stored it in a separate file so that we can automatically compare it with the result of our proposal model. To verify the correctness of our model, we evaluated the result based on the following measures: precision, recall and F1-measure.

$$\text{Precision } P = \frac{\text{No. of accurate extracted characters}}{\text{Total number of extracted characters}} \quad (2)$$

$$\text{Recal } R = \frac{\text{No. of accurate extracted characters}}{\text{Total number of characters of right content}} \quad (3)$$

$$F_1 - measure = 2.\frac{P.R}{P+R} \quad (4)$$

"No. of accurate extracted characters" mentioned above are calculated by measuring the "longest

common substrings"[3] between the result of our model and the result selected by hand. We must also remove all line break and white space characters before making the comparison to ensure that the comparison is accurate. After running the test, we have the following results:

Table 2. Compare the result with other methods

Methods / Measures	Average Precision (%)	Average Recall (%)	Average F1-measure (%)
1.Extracting text nodes	45.58	86.52	58.41
2.HTML analysis combined with natural language processing	74.17	80.26	75.05
3. Our proposal	74.10	80.32	76.04

Table 2 showed that the result of our proposal method is higher than method 2 (HTML analysis combined with natural language processing) and much higher than method 1 (Extracting text nodes). On average it achieves the highest measure (F1 measure) although in some cases its precision is lower than with method 2, but not significantly.

III. KEYWORD EXTRACTION FROM VIETNAMESE DOCUMENT

After extracting the main content of the web page, we continue extracting keywords. The extracted content of web page is considered as a combination of many keywords. Then we will match these keywords with the ones of advertisements to find out the most suitable advertisements for the page's content.

A. *Related work*

Some approaches have been proposed for the automatic extraction of keywords; and two most popular approaches are Statistics and Machine Learning.

Statistics-based approach

This approach records the most frequent terms to decide keywords of the document. Such recorded information is classified into two types: local weight and global weight. Local weight means recording frequency of one term in one document. Some examples of local weight to be mentioned are [12]: term frequency (TF), chi-square distribution (χ^2), information gain (IG), mutual information (MI), or term strength (TS). Global weight means recording frequency of one term in specific collection of documents. The typical approach of the global weight is IDF (Inverse Document Frequency) method [15], which is used to measure inverse frequency of one term in a specific document collection. Therefore, we need many available documents to apply the global weight approach.

The current effective approach of extracting keywords based on combination of local weight and global weight is TF.IDF (Term Frequency × Inverse Document Frequency) [15]. The TF.IDF weight evaluates how importance a term is to a document in a collection or corpus. The basic principle of TF.IDF is: "The importance increases proportionally to the number of times a term appears in the document (TF) but is offset by the frequency of the term in the corpus (IDF)". In other words, one term becomes less important if it is included in many different documents. For example, the term "chúng ta" (we) is not likely to be keyword because it is too popular. Based on this, we can re-evaluate the importance of a term in document.

Machine learning-based approach

This approach trains a machine to recognize keywords based on grammatical and syntactical characteristics. This approach has been used in many applications: Taeho Jo [7] train the Neural network based on TF.IDF to determine keyword. Witten [3] applies the Naïve Bayes algorithm in the KEA system. Hulth [1] applies the supervised learning RDS system combining with labeling words.

[3] http://en.wikipedia.org/wiki/Longest_common_substring_problem

B. Combination of $\chi^2 \times IDF$

The paper proposes another statistic approach by evaluating the combination of local weight χ^2 and global weight IDF. This approach combines the statistical information in the internal document (χ^2) and in the external corpus (IDF). Firstly, we make the pre-process on the document to extract all Vietnamese terms; then we calculate χ^2- distribution of each term in document, and calculating its IDF weight in specific training documents[4]. So we can calculate the weight W of each term "t" by combining these two factors: $W(t) = \chi^2(t) \times IDF(t)$. Finally, the terms with highest W will be the keywords of the document.

1) Pre-processing

In this paper, a term is defined as a word or a word sequence. A word sequence is also written as a phrase. A keyword is also a term. Before extracting the keywords, we must perform the term segmentation for Vietnamese text first. Vietnamese language is an alphabetic script that usually separates the word by space character. However, unlike other languages, in Vietnamese spaces are not only used to separate the word, but they are also used to separate the term containing some words. This is a problem for extracting Vietnamese terms from text. For example, the Vietnamese sentence "Tổ quốc ta đẹp như tranh vẽ" contains the following terms: Tổ quốc | ta | đẹp | như | tranh vẽ.

We used the Vietnamese word segmentation tool developed by Le Hong Phuong, N.T.M.Huyen, Azim Roussanly and Ho Tuong Vinh [17]. This tool provides the result of Vietnamese word segmentation with high accuracy (96% - 98%). After segmenting the document into different terms, we conduct to remove all stop words. The remaining words will be the candidate terms for the system.

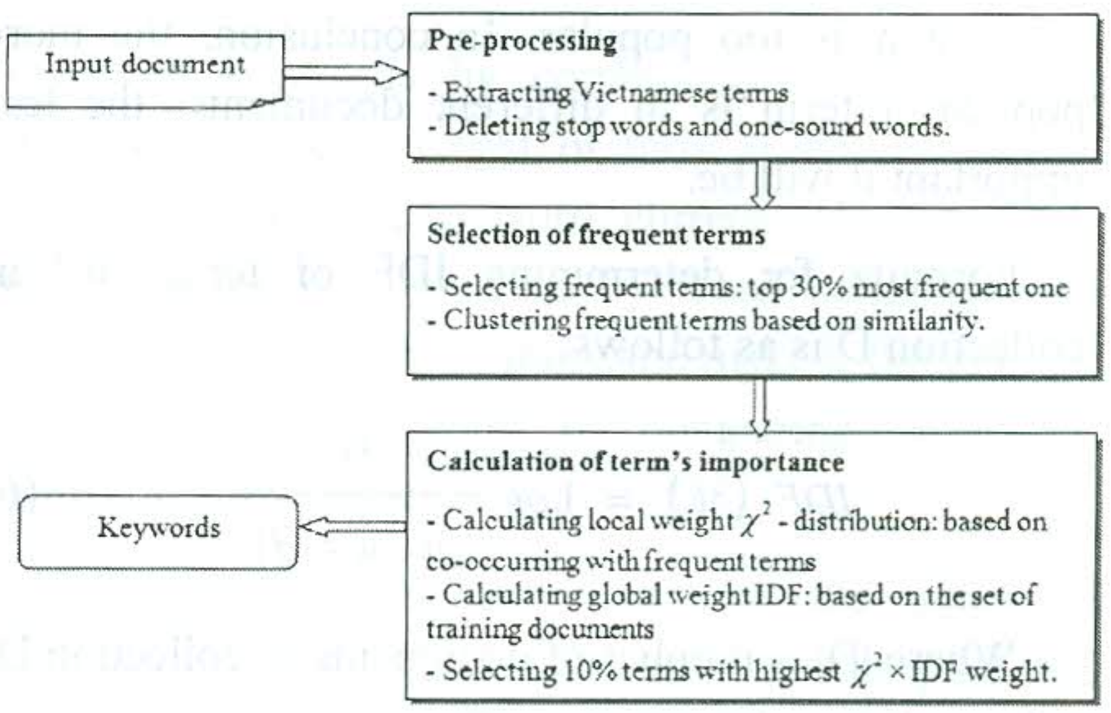

Figure 8. Model of extracting keyword on Vietnamese document

2) Local weight χ^2

χ^2- distribution is a statistic measurement of one term in document, determined by measuring its co-occurring with frequent terms in document [14,9]. Firstly, we extract frequent terms in document, denoted as G. Then we count the co-occurrence of each term in document with G. It is assumed that the term co-occurring with the G may also be the keyword of the document. Then, we can calculate the χ^2- distribution of each term w in document with the following formula:

$$\chi^2(w) = \sum_{c \in G} \left\{ \frac{(freq(w,c) - n_w p_c)^2}{n_w p_c} \right\} - \max_{c \in G} \left\{ \frac{(freq(w,c) - n_w p_c)^2}{n_w p_c} \right\} \quad (5)$$

In which: The G set is clustered based on similarity of terms [14]: G = [C1, C2, …,Cn]; freq*(w, c)* = co-occurrence frequency of term w with group c; nw = total number of terms in sentences included *w; pc* = frequency of group *c, pc* = *nc* / Ntotal; Ntotal = total number of terms in the document.

After evaluating, we can see that not all popular terms are important terms.

3) Global weight IDF

As mentioned above, this weight is based on assumption that if a term is referenced in many different documents, that term becomes less important

[4] Training documents is 1000 documents downloaded from Vietnamese repository ViCorpora: http://sourceforge.net/ projects/ ngonngu/files/ ViCorpora/

because it is too popular. In conclusion, the more popular a term is in different documents, the less important it will be.

Formula for determining IDF of term "w" in collection D is as follows:

$$IDF\ (w) = Log\frac{|D|}{|\{d : w \in d\}|} \quad (6)$$

Where |D| is number of documents in collection D; |{d:w ∈d}| are documents in D containing term w.

For example, suppose we have 100-term document containing 5 terms "doctor" and 5 terms "we", so we have frequency: TF ("doctor") = TF ("we") = 5 / 100 = 0.05. Then assuming that that document is included in a collection of 1000 documents, 200 of which contain term "doctor" and 500 contain term "we". Then we can determine IDF ("doctor") = log (1000 / 200) = 0.7 and IDF ("we") = log (1000/500) = 0.3. Therefore, we see that TF.IDF ("doctor") = 0.05 × 0.7 = 0.035 and TF.IDF ("we") = 0.05 × 0.3 = 0.015. Conclusion can be made that: term "doctor" is more important than "we" although they both appear five times in the document.

The paper used a collection of 1000 documents (about 4MB) which is downloaded from Vietnamese repository ViCorpora [16]. These documents are processed, analyzed, filtered in advance and stored into one single file to speed up the calculation process. Then paper combines this global weight IDF to re-evaluate χ^2 which has been calculated before to determine the word's importance.

4) Experimental result

We test with data as a collection of documents including 20 summaries of Vietnamese articles from local conferences with different methods. Each article is copied and re-formatted properly, then put into one file. Keywords of these documents are kept in other files, considered as the result files so we can compare with the result of keyword extraction automatically by our system. In order to verify the correctness of this model, we evaluated the results based on following measures: precision, recall and F2-measure.

$$\text{Precision P}=\frac{\text{No. of accurate extracted keywords}}{\text{Total number of extracted keywords}} \quad (7)$$

$$\text{Recal R}=\frac{\text{No. of accurate extracted keywords}}{\text{No. of accurate accurate keywords of document}} \quad (8)$$

$$F_2 = 5.\frac{P.R}{4.P+R} \quad (9)$$

Testing the algorithm, we have the following results:

Table 3. Result of automatic extraction of keyword compared with other methods

Method / Measures	Average Precision (%)	Average Recall (%)	Average F_2 (%)
χ2 with term co-occurrence	21.87	60.91	44.31
TF.IDF	22.33	58.59	43.71
Our proposal	23.12	63.71	46.55

This result can be seen as promising, because we get a higher result than other methods (even when the document collection is not too large).

IV. RUNNING THE ONLINE ADVERTISING ENGINE – ADENGINE

Combining the above mentioned methods, we have developed successfully an automatic advertising engine, named AdEngine®. AdEngine will automatically receive and distribute advertisements to member web pages appropriately based on detected keywords. Therefore, the advertisement will be automatically adjusted in the web page's content.

The AdEngine allows the subscribers (advertisers) to register their online ads to the system. The ad contains its title, content, link to site, and keywords. Then these ads will be automatically distributed and displayed on the member website network that its publisher (web master) agrees to show the ads of AdEngine. The ad will be displayed through a client script that the publisher has been provided from

AdEngine. The publisher will embed or insert that script into the desired position on their web pages. The script will automatically analyze the content of the current page that contains it to extract the main keywords and get the appropriate ads from the AdEngine server that match with these keywords of the page. Then the ads will be shown up at that position of the web page.

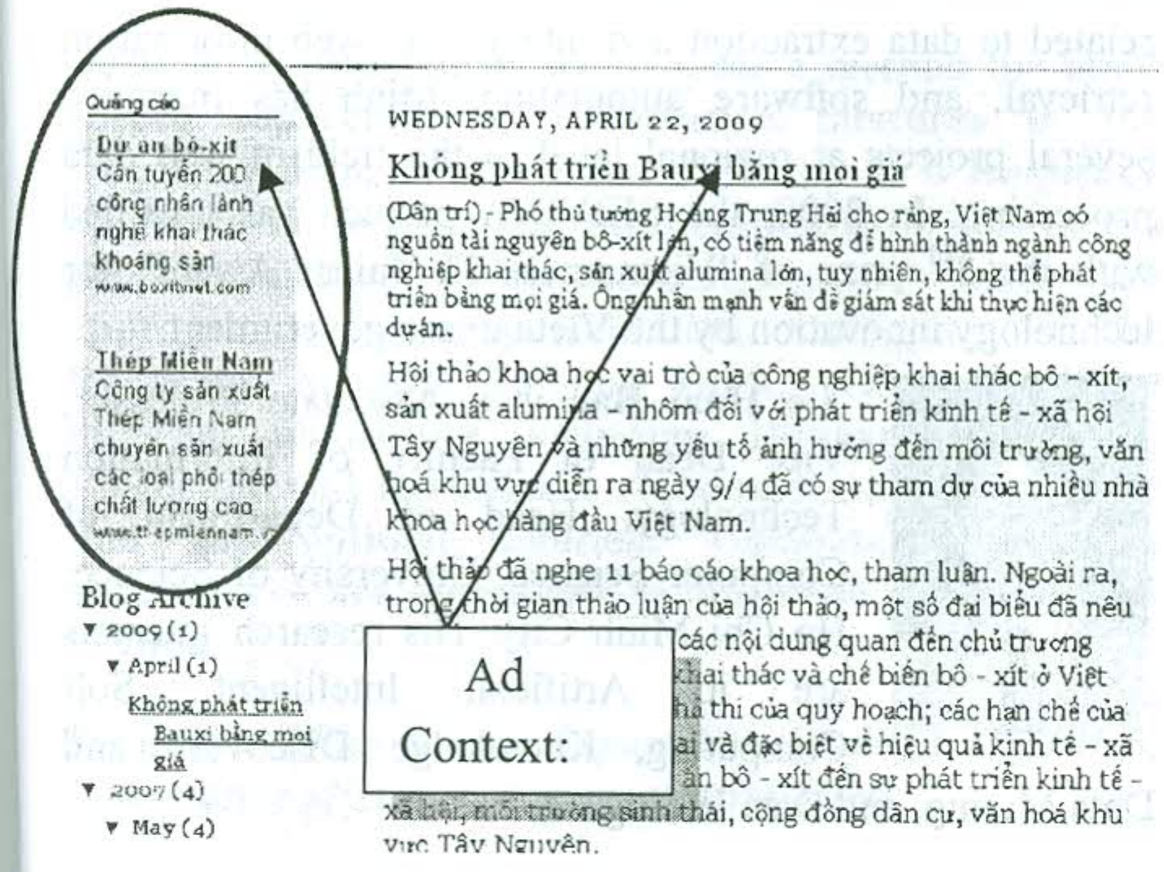

Figure 9. The web page with their ads loaded from AdEngine

V. Conclusion and Future Works

The paper has described the actual situation of online advertising in Vietnam and proposed possible solutions. The article also proposes a method of extracting the main content of a web page by histogram segmentation, as well as another method of automatically extracting Vietnamese keywords with new measurement $\chi^2 \times$ IDF.

In the future, it is necessary to develop and improve the algorithm to further increase the precision. In terms of extracting the main content of a web page, we could evaluate with different filters such as Gauss, beside the Mean filter. Besides, we should try the method of clustering with other algorithms such as EM to check if we can detect the number of clusters more exactly. In terms of extracting keywords, we could improve more the correctness of extracted keywords by combining additional term measurement: the weight of term in the document. About the matching keyword, currently we are simply using the method of word by word comparing. In order to improve the accuracy and make it flexible, we should try the method of comparing based on semantic relevance eg. Okapi BM25.

In addition, we can also use the synonym dictionary or WordNet dictionary to compare the keywords more exactly in the meaning.

References

[1] A.Hulth, "Improved automatic keyword extraction given more linguistic knowledge", In: Proc of EMNLP03, 2003.

[2] Deng Cai, Shipeng Yu, Ji-Rong Wen, Wei-Ying Ma, "VIPS: a vision-based page segmentation algorithm", Microsoft Research, Redmond, WA, 2004.

[3] Ian H.Witten, Gordon W.Paynter, Eibe Frank, Carl Gutwin, Craig G.Nevill-Manning, "KEA: practical automatic keyphrase extraction", in: Proc of Digital Libraries, 1999, pp. 254–256.

[4] Manuel Álvarez, Alberto Pan, Juan Raposo, Fernando Bellas, Fidel Cacheda, "Extracting lists of data records from semi-structured web pages", University of A Corunã, Spain, 2008.

[5] Ngo Quoc Hung, "Searching bilingual documents English-Vietnamese automatically from Internet", Master thesis, University of Sciences, Vietnam, 2008, pp.5–10.

[6] Ruihua Song, Haifeng Liu, Ji-Rong Wen, Wei-Ying Ma, "Learning block importance models for web pages", Microsoft Research Asia, 2004.

[7] Taeho Jo, Malrey Lee, Thomas M.Gatton, "Keyword extraction from documents using a neural network model", in: Proceedings of the 2006 International Conference on Hybrid Information Technology, 2006, pp. 194–197.

[8] Tim Weninger and William H. Hsu, "Text extraction from the web via text-to-tag ratio", in: 19th International Conference on Database and Expert Systems Application, 2008, pp. 23–28.

[9] Tran Viet Cuong, Nguyen Van Tuan, Nguyen Hoang Tu Anh, "Vietnamese keyword extraction based on term co-occurrence", Vietnam National University, 2006.

[10] Thomas M.Breuel, "Information extraction from HTML documents by structural matching", Second

International Workshop on Web Document Analysis, Edinburgh, 2004.

[11] Vibhanshu Abhishek, "Keyword generation for search engine advertising using semantic similarity between terms", Fair Isaac Corporation, Bangalore, India, 2007.

[12] Yang and Pedersen, "A comparative study on feature selection in text categorization", ICML97, 1997.

[13] Ying Li, Arun C.Surendran, and Dou Shen, "Data mining and audience intelligence for advertising", Microsoft AdCenter Lab, Redmond, WA 98074 USA, 2007.

[14] Y.Matsuo, "Keyword extraction from a single document using word co-occurrence statistical information", National Institute of Advanced Industrial Science and Technology, 2003.

[15] http://en.wikipedia.org/wiki/Tf-idf

[16] http://sourceforge.net/projects/ngonngu/files/ViCorpora

[17] Le Hong Phuong, Nguyen Thi Minh Huyen, Azim Roussanaly, Ho Tuong Vinh, A hybrid approach to word segmentation of Vietnamese texts, in: Second International Conference, LATA 2008, Tarragona Spain, 2008, pp. 240–249.

AUTHORS' BIOGRAPHIES

Nguyen Quy Minh is a project manager at software development center of LARION Computing, Ltd. He used to be a senior researcher at Software Engineering Laboratory of University of Science, Vietnam National University. He earned his Master's Degree in Computer Science from University of Science and also received his MCSD certification from Microsoft. His research interests are related to data extraction and integration, web information retrieval, and software automation. Minh has managed several projects at regional level in the field of web data processing. In 2008, the SIRVINA project was awarded with the 3rd prize of "Vietnamese IT Talent Award" for technology innovation by the Vietnamese government.

Le Hoai Bac is a Associate Professor, Vice Dean of Faculty of Information Technology, Head of Department of Computer Science, University of Science, Ho Chi Minh City. His research interests are in Artificial Intelligent, Soft Computing, Knowledge Discovery and Data Mining, and Data Hiding.