Noise Elimination from the Web Documents by Using URL paths and Information Redundancy

Byeong Ho Kang and Yang Sok Kim
School of Computing, University of Tasmania
Sandy Bay, Tasmania 7001, Australia
{yangsokk, bhkang}@utas.edu.au

Abstract—**Noise data in the Web document significantly affect on the performance of the Web information management system. Many researchers have proposed document structure based noise data elimination methods. In this paper, we propose a different approach that uses a redundant information elimination approach in the Web documents from the same URL path. We propose a redundant word/phrase filtering method for single or multiple tokenizations. We conducted two experiments to examine efficiency and effectiveness of our filtering approaches. Experimental results show that our approach produces a high performance in these two criteria.**

I. Introduction

NOWADAYS lots of information is available on the Web and, furthermore, new information is continually created by distributed publishers around the world. Distributed Web publication introduces some additional problems that the traditional publication does not have, including information finding or gathering, systematic information management, and information sharing amongst the most significant. Web crawling and monitoring techniques are usually used for the purpose of information finding and/or gathering. Machine learning (ML) based, or knowledge engineering (KE) based text/document classification approaches, are usually used for the systematic management of collected information. Lastly, various notification systems are used to share collected and classified information. This paper focuses on the problem that is related to the classification task.

Classification systems use document features to perform their classification. Though other document features like hyperlinks or document structure can be used for the classification process, text data of Web documents are mainly used for input data of ML and/or KE based text/document classifiers. However, real datasets coming from Web sites usually contain lots of noisy data such as advertisement text, navigational text, or hyperlink text. These are inserted to attract customers (business purpose) or give high level content usability or accessibility (functional purpose).

Previous research showed that though the levels of noise sensitivity are different, the performances of various ML algorithms are significantly affected by noise data [1, 2]. For this reason, extracting information or eliminating noisy information from Web documents has been studied by many researchers [3-10].

Current techniques are mainly based on machine learning and natural language processing approaches to learn extraction rules from manually labeled examples. Nowadays many researchers are proposing structure based noise elimination or core content extraction. Basically their method splits the Web documents into small sections by using various tags, eventually deciding which section is core and non-core by using various information metrics. However, the method that we propose does not use any structure analysis. Instead we focus on the redundancy of Web documents from same URL path and on the elimination of redundant information for feature extraction. Though our approach can be applied as preprocessing for either Web crawling or Web monitoring, this paper focuses on the Web monitoring and document classification context, because in this context the URL paths that noisy data elimination system process is a finite set and the size of set is smaller set than a Web crawling system would process.

This paper consists of following the contents: Section 2 summarizes related research results. Section 3 illustrates our method for noise data elimination. Section 4 describes our experiment design and the results. Lastly, in Section 5 we provide conclusions and further work.

II. Related Work

A. Noise Data Types

Yi and Liu [11] grouped noise data of Web documents into two categories according to its granularity. Global noises are redundant Web pages over the Internet such as mirror sites and legal or illegal duplicated Web pages. Local noises (or intra-page redundancy) exist in the Web page with banners, advertisements, and navigational guides being examples of local noises. Global noise elimination research is related to Web page level filtering technologies. This paper only focuses on the local noise elimination method.

We classified the Web document data as following three types: *core information, redundant information and hidden information*. Core information is the content that a user wants to view from a Web page. For example, the main article in the news article Web page is core information. This information is mainly used for text classification tasks. Redundant information is added to enhance Web content accessibility or business attractiveness. Inserting this information is promoted officially by W3C [12] or profit seeking companies. Web documents also contain the 'hidden information' like HTML tags, script language and programming comments, which is called 'hidden' because it is not seen by end users. Users only can see it by performing the "view source" action. In this paper, we view hidden and redundant information as local noise data.

B. Feature Extraction and Noise Data

Feature extraction is critical to most text classification tasks whether they use ML or KE based techniques. It transforms all raw documents into a suitable representation, where a document is represented by a set of extracted features and their associated weights. Extracted features are used to find target concept descriptions of categories. Feature extraction begins by dividing *input data* into separate terms, called 'tokens'. Tokenization is simple for white-spaced languages, like English, because a word is a string of characters with white space before and after. When we process Web documents, noise data is also in the input data. Therefore, without removing such data, the efficiency of feature extraction and finally text classification is certainly degraded [4, 6, 11].

There are two types of tokens – single token and multiple token – which are used in the classification tasks. Single tokens are most frequently used, where information about dependency and relative positions of different tokens are not employed. Multiple tokens consist of more than one token, so it is possible to make use of the dependencies and relative positions of component tokens [13]. It is still debated by researchers whether multiple tokens improve the accuracy of text classification or otherwise [14]. Some experiment results indicate that multiple tokens are better [15, 16], while other research shows just the opposite [17, 18]. In our research, the redundant word elimination filtering approach is for the single token applications and redundant phrase elimination filtering approach is devoted to both the single token and the multiple token applications.

C. Noise Data Elimination

Lin and Ho [4] proposed a method that detects informative content blocks from Web documents. After analyzing all Web pages in Taiwan, they found that about 50% of them use <table> tag as a template to layout their Web pages. For this reason, they use <table> tag to segregate Web pages into small content blocks. Then they extract features (meaningful keywords) from these content blocks and calculate the entropy value of each feature. According to the entropy value of each feature in the content block, the entropy value of the block is defined. By analyzing the information measure, they proposed a method to dynamically select the entropy-threshold that partitions blocks into either informative or redundant. Similar approaches are proposed by Gupta et al.[7], Yi et al.[6], Bar-Yossef and Rajagoalan[19], Debanth et al.[3], Kao et al. [8] and Song et al.[5]. They are based on same idea that there exist informative blocks and non-informative blocks in the Web pages, and noise elimination can be accomplished by extracting informative blocks from Web pages, but use different measures to detect information blocks from Web document. For this reason, these approaches are ***structure based noise elimination approaches***.

Some researchers focus on the ***information extraction based noise elimination approach*** to remove noise data from Web documents. Wrapper [20, 21] and SoftMealy [22] extract the structural information from Web documents by using manually generated templates or examples. Other information extraction based approaches are proposed by [23], [24], and [25], in which they proposed methods that extract the "gist" of Web page by summarizing Web documents. These systems are not scalable and merely applied to specific Web applications, as they usually require domain-dependant natural language processing knowledge and the annotation process of corpora is usually performed manually [4].

We do not follow either the structure based approach or information extraction based approach in our research; rather, we focus on the contents in the Web pages, because in the Web monitoring context noisy data tend to appear in the successive incoming Web pages. To this end, we implemented three different noise data elimination filters – tag based filter, word based filter, and sentence based filter. In the following Section, we describe the implementation details of our method.

III. IMPLEMENTATION

A. Web Information Monitoring System

We used a Web monitoring system for our research, which was developed by a group of researchers at University of Tasmania for an advanced Web information management. If users register target Web sites, it revisits them periodically and reports any new information. In addition, the system supports a specific KE based text classification. The classifier is implemented with the Multiple Classification Ripple Down Rules (MCRDR) knowledge acquisition algorithm [26]. The classifier uses production rule similar to that of the traditional rule-based system. Each rule consists of two parts – condition and conclusion. The condition is a set of keywords and the conclusion consists of one or more categories or null. The classifier evaluates the existence of certain keyword sets in the documents and indicates a category, or several categories, under which the document would be classified. However, the MCRDR classifier is different from traditional rule-based systems because the rule structure has a rule-exception based structure which localizes validation and verification process. Each rule has coherent case or cases called ‘cornerstone cases’, which are used to create a specific rule. The cornerstone cases contain context information and make easier the validation and verification process.

Fig.1 illustrates the MCRDR rule structure, which is a n-ary tree structure. A classification recommendation (conclusion) is provided by the last rule satisfied in a pathway. All children of the satisfied parent rule are evaluated, allowing for multiple conclusions. The conclusion of the parent rule is only given if none of the children are satisfied [26-28]. In Fig.1., the first four different recommendations are proposed by the system (see bold box with blue line).

There are three types of rules in the MCRDR classifier – ground-breaking rule, refining rule, and stopping rules. A ground-breaking rule is created under the root node to make a new branch under the root node (e.g., rule 1 ~ 4, 11 in Fig. 1). A refining rule is created under the ground breaking rule, or other refining rule, to make an exception of the current rule (e.g., rule 5 ~ 8 in Fig. 1). A stopping rule is created under the ground breaking rule, refining rule, or other stopping rule. If a case (document) is fired by the stopping rule, it does not classify into the folder that its parent rule indicates (e.g., rule 9 in Fig. 1). More detailed information about our Web monitoring system is in [29], [30], and [31]. We used this system to collect Web pages source and to conduct our experiment.

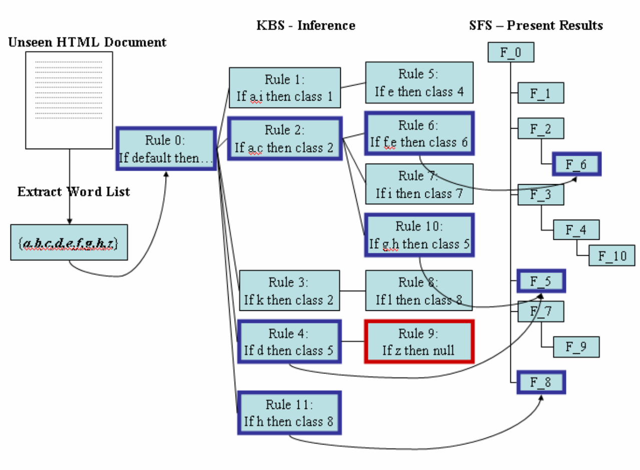
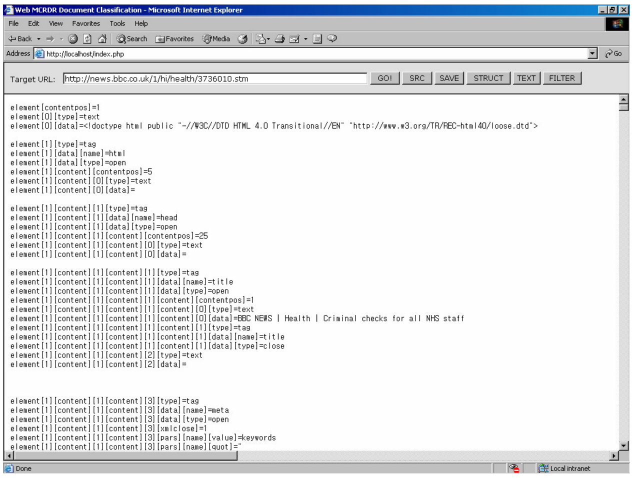


Fig. 1. The MCRDR classifier’s rule structure, inference, and rule types

B. HTML Parser

The HTML parser is critical when we want to extract specific data from the Web document. There are various HTML parsing approaches for special purposes such as HTML SAX or XML parser. Though bad HTML sources can be properly displayed with some grammatical errors, this causes errors when the parser system parses Web documents. Therefore, an appropriate HTML error correction mechanism should be employed for the HTML parser implementation. We used a parser that uses automata and HTML grammar. It was originally developed by A. Y. Kalmykov and is publicly available on the Web (http://anton.concord.ru/). We modified the parser to extract appropriate information from Web page sources. This parser corrects syntactic errors by using HTML grammar and generates an element tree (see Fig.2).


Web MCRDR Document Classification - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/index.php
Go
GO! SRC SAVE STRUCT TEXT FILTER
Done
Local intranet

Target URL: http://news.bbc.co.uk/1/hi/health/3736010.stm

```
element[contentpos]=1
element[0][type]=text
element[0][data]=<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN" "http://www.w3.org/TR/REC-html40/loose.dtd">

element[1][type]=tag
element[1][data][name]=html
element[1][data][type]=open
element[1][content][contentpos]=5
element[1][content][0][type]=text
element[1][content][0][data]=

element[1][content][1][type]=tag
element[1][content][1][data][name]=head
element[1][content][1][data][type]=open
element[1][content][1][content][contentpos]=25
element[1][content][1][content][0][type]=text
element[1][content][1][content][0][data]=

element[1][content][1][content][1][type]=tag
element[1][content][1][content][1][data][name]=title
element[1][content][1][content][1][data][type]=open
element[1][content][1][content][1][content][contentpos]=1
element[1][content][1][content][1][content][0][type]=text
element[1][content][1][content][1][content][0][data]=BBC NEWS | Health | Criminal checks for all NHS staff
element[1][content][1][content][1][content][1][type]=tag
element[1][content][1][content][1][content][1][data][name]=title
element[1][content][1][content][1][content][1][data][type]=close
element[1][content][1][content][2][type]=text
element[1][content][1][content][2][data]=



element[1][content][1][content][3][type]=tag
element[1][content][1][content][3][data][name]=meta
element[1][content][1][content][3][data][type]=open
element[1][content][1][content][3][xmlclose]=1
element[1][content][1][content][3][pars][name][value]=keywords
element[1][content][1][content][3][pars][name][quot]="
```

Fig. 2. HTML source parsing result

C. Noise Data Elimination System

Three noise data filtering methods are proposed in this research – a tag based filer, a redundant words elimination filter, and a redundant phrases elimination filter. Whereas the tag filter is commonly used in the noise data elimination process, the redundant word filtering method is mainly used in a single tokenization situation, while the redundant phrases elimination filter is used in single or multiple tokenization, process.

1) Tag based filtering: Though HTML tags do not strictly define content information, some text data between tags can be removed or preserved by using HTML tags, because some HTML tags intuitively represent certain data types. For example, <title> tag is used to represent a document’s title and <h1>, <h2>, …, <h6> tags are used to represent headlines of the document’s content. There are negative and positive tags. If the text is enclosed by negative tags, it is regarded as noise data, and is not used by the document classifiers. On the other hand, if the text is surrounded by positive tags, it is regarded as core data and should be used by the classifiers. In this research, we only use negative tags to remove hidden noise data. We defined the following tags as negative tags:

Hyperlink tag (`<a>hyperlink text</a>`). Hyperlink in Web documents can help with document classification. Kuo and Wong [32] proposed an algorithm to classify Web documents into subsets based on hyperlinks in documents and their contents. Border et al. [33] reported that using link information dramatically improved classification accuracy. Chakrabarti et al. [34] also showed that an approach based on iteratively re-labeling pages using hyperlink information was successful using data from both Web pages and the U.S. Patent database. However, as Chakrabarti et al illustrated, there are risks that naïve use of terms in the hyperlink of a document can even degrade accuracy and requires more careful use. For this reason, we used the hyperlink tag as a negative tag though the hyperlinked documents may contain similar contents.

Select tag (`<select> <option> text </option> </select>`). Document association in HTML documents can be performed by the selection tag. In this case, options can be used to link other related contents. A tag based filter removes option text from phrase lists.

Style tag (`<style>style text</style>`). Phrases that are enclosed by style tags are removed by a tag based filter.

Javascript (`<script language="javascript"> … </script>`). Javascript is a client-side scripting language that may be utilized in conjunction with HTML. Javascript codes are enclosed by <script> tag and javascript events can be used like attributes in HTML tags. A tag filter eliminates javascript from the phrase lists.

Programming Comments (`<!-- … -->`). Various programming comments are included in the HTML source and they are eliminated.

After applying the tag based filtering method, the following two filtering methods are employed for different application contexts.

2) Redundant words filtering method: This approach is based on the fact that Web pages from the same URL path contain common redundant words. Redundant words filtering method works as follows: if the Web monitoring server collects a new Web page (W_a) from the target monitoring site, the filtering system analyzes its URL path and finds whether there exists a Web page from the same URL path. If not, the URL and the Web page source are stored. If there exists a Web page (W_b) from same URL path, firstly the tag filter is applied and then remaining HTML tags are removed from them.

Let the text file that is obtained from W_a be T_a and from W_b be T_b. T_a and T_b are processed to get single token set S_a and S_b. The redundant words filter (W_{filter}) is minimum co-occurrence words and its number from two single token sets. For example, if S_a = {(a, 5), (b, 2), (c, 4), (d, 3)} and S_b = {(a, 3), (b, 1), (e, 2), (f, 5), (g, 3)}, the W_{filter} = {(a, 3), (b, 1)}. In this example, the first letter is word and the number is frequency of that word. If third article's single token set is S_c = {(a, 3), (b, 3), (g, 4), (h, 2), (I, 3)}, the single token is represented as S_c' = {(b, 2), (g, 4), (h, 2), (I, 3)} after the redundant words filtering method is applied. The redundant word filter generation process is repeated until the system obtains a stable redundant words filter set. The algorithm for the redundant words filtering is illustrated in Fig. 3.

3) Redundant phrase filtering method: This approach is based on the same assumption that is used in redundant words filtering method, except that this method uses phrase instead of individual words in the HTML source. In this procedure, the phrase is defined as a group of words between HTML tags. After eliminating the tag filtering text by using the tag filtering method, the text in the end node of the parsed tree is elicited from the Web source.

```
Input: HTML pages ($H_i$, $H_j$) from same URL path
Output: Redundant words filter set
begin
  Get text ($T_i$, $T_j$) after tag filter is applied
  Perform single tokenization to get word and it number
    $S_i$ = Set of single tokens and its number of $T_i$
        {$W_{i0}$, $W_{i2}$, …, $W_{iN}$}
    $S_j$ = Set of single tokens and its number of $T_j$
        {$W_{j0}$, $W_{j2}$, …, $W_{jM}$}
  for each word in $S_i$
    if $W_{ii}$ is in $S_j$
      if $W_{ii}$'s count is less then $W_{ij}$
        $W_{ii}$ and its count becomes an element of
        redundant word filter set
      else
        $W_{ij}$ and its count becomes an element of
        redundant word filter set
end
```

Fig. 3. Redundant words elimination filtering algorithm

Comparing each phrases from two sources, common phrases become elements of the redundant phrase set. This process is repeated until the system achieves a stable redundant phrases set. The algorithm for the redundant phrase filtering method is illustrated in Fig. 4.

```
Input: HTML pages ($H_i$, $H_j$) from same URL path
Output: Redundant phrase filter set
begin
  Apply tag filter ($H_i'$, $H_j'$)
  Get phrase set from filtered sources
    $P_i$ = Set of phrases of $H_i'$
        {$P_{i0}$, $P_{i2}$, …, $P_{iN}$}
    $P_j$ = Set of phrases of $H_j'$
        {$P_{j0}$, $P_{j2}$, …, $P_{jM}$}
  for each phrase in $P_i$
    if $P_{ii}$ is in $P_j$
      $P_{ii}$ becomes an element of redundant phrase set
end
```

Fig. 4. Redundant phrase elimination filtering algorithm

IV. Experiments

A. Data Set Used

Experiments in this research, over five month, were conducted using data sets collected by the Web Monitoring System, called WebMon, from recognized Web-based health information sites (BBC, ABC, and WebMD). The WebMon system collects newly uploaded information from the registered Web site. Originally 7,780 articles were collected from three Web sites. 757 documents were randomly selected for the first experiment and 500 for the second experiment. TABLE I summarizes the data sets used. These data sets were selected because they were real data in the Web monitoring context, meaning that they were not machine generated. All documents in these data sets were written by humans for some purpose other than the purpose of testing text mining systems. Though there are several corpora publicly available (e.g., Reuters-21578 and 20-Newsgroups), these corpora were not appropriate for our research, because they were not HTML formatted and were already cleaned.

TABLE I
Experiment Data Sets

Sites	Experiment 1	Experiment 2
BBC	270	255
WebMD	237	245
ABC	250	-
Total	757	500

B. Evaluation Method

1) Efficiency of Filtering: The aim of the first experiment was to measure the efficiency of filters, namely, focusing on the question - “does the system correctly eliminate noise data from Web documents?” This experiment was concentrated on the redundant phrase filtering method. All selected data were processed by the filtering system and the results were verified by the user. The system may propose that it is core content or non-core content when a phrase are core content or not and. Conventional performance metrics were defined and computed from these contingency tables. These measures are recall (r), precision (p), fallout (f), accuracy (Acc) and error (Err) (see TABLE II).

TABLE II
Contingency Table For Evaluation

	YES is correct	NO is correct
Assigned YES	270	255
Assigned NO	237	245

$r = a/(a + c)$ if $a + c>0$, otherwise undefined;
$p=a/(a + b)$ if $a + b>0$, otherwise undefined;
$f = b/(b + d)$ if $b + d >0$, otherwise undefined;
$Acc = (a + d)/n$ where $n=a + b + c + d >0$;
$Err = (b + c)/n$ where $n=a + b + c + d >0$.

2) Effectiveness of Filtering: The second experiment focused on the effectiveness of the filtering system, namely “how the noise filtering system helps a Web information management system to work effectively”. The effectiveness of a system is measured by its correctness. The MCRDR classifier was employed for this experiment. Three data sets were created by employing three different filtering methods: Data set 1 was processed by tag based filtering method and redundant phrases filtering method. Data set 2 was processed by tag based filter and redundant words filtering method. Data set 3 was not processed by any filter.

The experiment included the following procedures: Firstly, Data set 1 was classified by using the MCRDR document classifier. Secondly, Data sets 2 and 3 were automatically classified by using the knowledge base that was created by the classification of Data set 1. Lastly, the inference results of Data sets 1, 2, and 3 were compared to measure the comparative effectiveness. The effectiveness can not be evaluated by an absolute measure because the correctness of classification is not an absolute measure, as it is influenced by various factors, such as subject (user), situation, cognitive factors, and temporal factors [35]. For this reason, correctness is only measured comparatively by assuming that Data set 1’s inference results are correct.

V. Results

A. Filtering Efficiency

The filtering system generates all phrase lists when a document is requested to process. The tag based filtering module eliminates phrase/s that are enclosed by negative tags such as hyperlink text, option text, javascript, and programming comments. The redundant phrase filtering module generates a set of filtering phrases for each Web site. TABLE III summarizes the number of redundant phrases of each monitoring Web site and indicates the number of words eliminated by the tag filter and redundant phrase filter. The number of filtering phrases on the BBC is significantly greater than those on of the WebMD or the ABC Web sites. This means the BBC contains more redundant information compared to other two Web sites. However, noise data that are eliminated by the redundant filter are smaller than that of other Web sites, caused by the fact that the BBC has more unique URL paths than the other Web sites.

TABLE III
Filtering Results

Sites	Total Phrases	Filtering Phrase	Noise Data		Core Data
			Tag	Redundant	
BBC	48,915	103	9,083	2,060	37,772
WebMD	27,730	20	3,839	2,821	21,070
ABC	32,195	27	2,364	2,114	27,717
Average	36,280	50	5,095	2,332	28,853

We used four metrics (recall, precision, fallout, and accuracy) to measure the efficiency of the filters. The accuracy of BBC and WebMD are very similar and the accuracy of ABC is better than that of both the WebMD and the BBC. On average, the recall is 95.2, the precision is 88.8, and the accuracy is 97.3. This result shows acceptable effectiveness in eliminating noisy data from Web documents compared to other structure based approaches [3-5].

TABLE III
FILTERING EFFICIENCY

Sites	Recall	Precision	Fallout	Accuracy
BBC	93.8	87.1	2.6	96.9
WebMD	93.4	89.5	2.4	96.9
ABC	98.3	89.7	2.1	98.0
Average	**95.2**	**88.8**	**2.4**	**97.3**

B. Filtering Effectiveness

Firstly Data Set 1 is classified by the Web based MCRDR document classifier. 239 rules were created with 708 condition keywords and the average keywords per rule were 2.96. In total 82 folders were created under eight top categories; alternative medicine, drug information, disease, demographic groups, pregnancy, sexual health, social and family issues, and well being. Average articles per rule were 4.12 and average articles per folder were 12.01. TABLE IV and V illustrate Data Set 1's classification results.

TABLE IV
DATA 1'S CLASSIFICATION RESULTS (RULES & ARTICLES)

Conditions	1	2	3	4	4>	Total
Rules	21	54	100	51	13	239
Articles	269	306	296	84	30	985
Articles per Rule	**12.8**	**5.7**	**2.9**	**1.7**	**2.3**	**4.1**

TABLE V
DATA 1'S CLASSIFICATION RESULTS (FOLDERS & ARTICLES)

Conditions	1	2	3	Total
Folders	3	45	34	82
Articles	7	681	297	985
Articles per Folder	**2.3**	**15.1**	**8.7**	**12.0**

The articles in Data Sets 2 and 3 were automatically classified using Data Set 1's classification knowledge base. Classification results are summarized in TABLE VI. The inference results of Data Set 2 are very similar, which means the redundant phrases filter and the redundant words filter were identical in both feature extraction and inference task, but the inference results of Data Set 3 were very different. The MCRDR classifier suggested almost more than three times the inference results. This means the inference results of Data Set 3 may have many erroneous inference results.

TABLE VI
INFERENCE RESULTS COMPARISON

	Articles	Data 1	Data 2	Data3
BBC	255	502 (1.97)	470 (1.84)	1,226 (4.81)
WebMD	245	483 (1.97)	496 (2.02)	1,828 (7.46)
Total	**500**	**985** **(1.97)**	**966** **(1.93)**	**3,054** **(6.11)**

Note: The number in () is average articles per rule

VI. CONCLUSIONS

The goal of this research was to investigate and develop core content extraction methods from Web documents for Web information management without using document structure. We considered three different noise data filtering methods - tag based filtering method, redundant word filtering method, and redundant phrase filtering method. We used the tag based filtering method before we processed the redundant word/phrase filtering method. We conducted two experiments to evaluate our approach from the effectiveness and efficiency view point. Our experiment results show that our approach shows results in an encouraging outcome.

In spite of the favorable results that we obtain, we need to conduct more expanded experiments to a further evaluation of our approach. In our experiment, we measured filtering effectiveness by comparing the classification results. However, we need to measure the cost of misclassification for a better understanding of effectiveness. The cost can be measured by the amount of remedying rules that are needed in revising current rule bases.

REFERENCES

1. Lopresti, D. *Performance evaluation for text processing of noisy inputs*. in *2005 ACM symposium on Applied computing*. 2005. Santa Fe, New Mexico: ACM Press New York, NY, USA.
2. Kalapanidas, E., et al. *Machine Learning Algorithms: A study on noise sensitivity*. in *1st Balcan Conference in Informatics 2003*. 2003. Thessaloniki.
3. Debnath, S., P. Mitra, and C.L. Giles. *Automatic extraction of informative blocks from webpages*. in *2005 ACM symposium on Applied computing*. 2005. Santa Fe, New Mexico: ACM Press New York, NY, USA.
4. Lin, S.-H. and J.-M. Ho. *Discovering informative content blocks from web documents*. in *SIGKDD '02*. 2002. Edmonton, Albert, Canada.
5. Song, R., et al. *Learning block importance models for web pages*. in *13th international conference on World Wide Web*. 2004. New York, NY, USA: ACM Press New York, NY, USA.
6. Yi, L., B. Liu, and X. Li. *Eliminating Noisy Information in Web Pages for Data Mining*. in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-2003)*. 2003. Washington, DC, USA.
7. Gupta, S., et al. *DOM-based content extraction of HTML documents*. in *International World Wide Web Conference*. 2003. Budapest, Hungary: ACM Press New York, NY, USA.
8. Kao, H.-Y., J.-M. Ho, and M.-S. Chen, *WISDOM: Web intrapage informative structure mining based on document object model.* IEEE Transactions on Knowledge and Data Engineering, 2005. **17**(5): p. 614 - 627.
9. Freitag, D., *Information extraction from HTML: application of a general machine learning approach.* Proceedings Fifteenth National Conference on Artificial Intelligence (AAAI-98). Tenth Conference on Innovative Applications of Artificial Intelligence, 1998: p. 517-523.
10. Grieser, G., et al., *A unifying approach to HTML wrapper representation and learning.* Discovery Science. Third International Conference, DS 2000. Proceedings (Lecture Notes in Artificial Intelligence Vol.1967), 2000: p. 50-64.
11. Yi, L. and B. Liu. *Web page cleaning for Web mining through feature weighting*. in *Proceedings of Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*. 2003. Acapulco, Mexico.
12. Chisholm, W., G. Vanderheiden, and I. Jacobs, *Techniques for Web Content Accessibility Guidelines 1.0.* 2000.

13. Liao, C., S. Alpha, and P. Dixon. *Feature preparation in text categorization*. in *Australasian Data Mining Workshop*. 2003. Lakeside Hotel, Canberra.
14. Sebastiani, F., *Machine learning in automated text categorization.* ACM Computing Surveys, 2002. **34**(1): p. 1-47.
15. Sahami, M., *Learning limited dependence Bayesian classifiers.* KDD-96 Proceedings. Second International Conference on Knowledge Discovery and Data Mining, 1996: p. 335-338.
16. Dumais, S., et al., *Inductive learning algorithms and representations for text categorization.* Proceedings of the 1998 ACM CIKM International Conference on Information and Knowledge Management, 1998: p. 148-155.
17. Lewis, D.D. *Feature selection and feature extraction for text categorization*. in *Speech and Natural Language Workshop*. 1992. San Mateo, California: Morgan Kaufmann.
18. Scott, S. and S. Matwin, *Feature engineering for text classification.* Machine Learning. Proceedings of the Sixteenth International Conference (ICML'99), 1999: p. 379-388.
19. Bar-Yossef, z. and S. Rajagopalan. *Template Detection via Data Mining and its Applications*. in *WWW 2002*. 2002. Honolulu, Hawaii, USA.
20. Kushmerick, N., D.S. Weld, and R. Doorenbos, *Wrapper induction for information extraction.* IJCAI-97. Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, 1997: p. 729-735.
21. Kushmerick, N., *Wrapper induction: efficiency and expressiveness.* Artificial Intelligence, 2000. **vol.118, no.1-2**: p. 15-68.
22. Chun-Nan, H. and D. Ming-Tzung, *Generating finite-state transducers for semi-structured data extraction from the Web.* Information Systems, 1998. **vol.23, no.8**: p. 521-538.
23. Shen, D., et al. *Web-page classification through summarization*. in *27th annual international ACM SIGIR conference on Research and development in information retrieval*. 2004. Sheffield, United Kingdom: ACM Press New York, NY, USA.
24. Kolcz, A., V. Prabakarmurthi, and J. Kalita. *Summarization as feature selection for text categorization*. in *tenth international conference on Information and knowledge management*. 2001. Atlanta, Georgia, USA: ACM Press New York, NY, USA.
25. Berger, A.L. and V.O. Mittal. *OCELOT: a system for summarizing Web pages*. in *23rd annual international ACM SIGIR conference on Research and development in information retrieval*. 2000. Athens, Greece.
26. Kang, B.H., P. Compton, and P. Preston, *Validating incremental knowledge acquisition for multiple classifications.* Critical Technology: Proceedings of the Third World Congress on Expert Systems, 1996: p. 856-868.
27. Compton, P. and R. D. *Extending Ripple-Down Rules*. in *12th International Conference on Knowledge Engineering and Knowledge Managements (EKAW'2000)*. 2000. Juan-les-Pins, France.
28. Martinez-Bejar, R., et al., *An easy-maintenance, reusable approach for building knowledge-based systems: application to landscape assessment.* Expert Systems with Applications, 2001. **vol.20, no.2**: p. 153-162.
29. Park, S.S., S.K. Kim, and B.H. Kang. *Web Information Management System: Personalization and Generalization*. in *the IADIS International Conference WWW/Internet 2003*. 2003.
30. Kim, Y.S., et al. *Adaptive Web Document Classification with MCRDR*. in *International Conference on Information Technology: Coding and Computing ITCC 2004*. 2004. Orleans, Las Vegas, Nevada, USA.
31. Park, G.C., et al. *An Automated WSDL Generation and Enhanced SOAP Message Processing System for Mobile Web Services*. in *Third International Conference on Information Technology : New Generations (ITNG 2006)*. 2006. Las Vegas, Nevada, USA.
32. Kuo, Y.-H. and M.-H. Wong. *Web Document Classification based on Hyperlinks and Document Semantics*. in *PRICAI Workshop on Text and Web Mining*. 2000.
33. Broder, A.Z., R. Krauthgamer, and M. Mitzenmacher, *Improved classification via connectivity information.* Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, 2000: p. 576-585.
34. Chakrabarti, S., B. Dom, and P. Indyk, *Enhanced hypertext categorization using hyperlinks.* SIGMOD Record, 1998. **vol.27, no.2**: p. 307-318.
35. Kowalski, G., *Information Retrieval Systems: Theory and Implementation*. 1997: Kluwer Academic Publishers.