Journal of Information & Computational Science 6: 2 (2009) 1033-1041
Available at http:// www.joics.com

Improved Fuzzy Set Information Retrieval Approach on Duplicate Webpage Detection

Yuchen Zhou [a,*], Zuoda Liu [a], Beixing Deng [a], Xing Li [a,b]

[a] *Department of Electronic Engineering, Tsinghua University, Beijing 100084, China;*

[b] *CERNET Network Center, Tsinghua University, Beijing 100084, China*

Abstract

Similar Web pages are easily found on Internet. The redundancy of information severely slows down internet applications such as crawl module of search engine, and could lead to waste of storage in the indexing procedure. In this paper, we proposed a content-based approach for detecting webpage duplications. The algorithm contains three parts: i) pre-processing, excluding HTML tags and unrelated information; ii) use a query-combined fuzzy set information retrieval approach to find out the correlation between every two documents; iii) a threshold is set and duplicate webpages are eliminated. Original algorithm of duplication detection is revised and focused mainly on performance optimization. Testing results shows that the performance is greatly improved with an acceptable sacrifice on quality**.**

Keywords: Fuzzy Set; Duplicate Detection; Information Retrieval

1 Introduction

Search engine has been confronting a lot of document duplication problems these days, especially in BBS or news searching. BBS users are likely to quote other users' comment, which creates redundancy; Different media may report same news, which also leads to duplication. Besides, plagiarism in technical articles also degraded the overall performance of search engine as well as academic environment.

Till now, we have lots of measures for detecting totally same articles or pure quoted articles, but we still lack the approach to detect literally different but semantically same articles. Generally

[*] Corresponding author.
Email addresses: zyc0030_cn@sina.com (Yuchen ZHOU)

1548-7741/ Copyright © 2009 Binary Information Press
March, 2009

speaking, if we want to detect the similarity degree of two documents, we have to extract the core sentences or terms of these articles, find a way to measure the correlation between these sentences using an article-article correlation function, and eventually set up threshold for judging duplications. This algorithm is first introduced by Rajiv Yerra at BYU[1], however, it is focused on English document retrieval and the performance is unacceptable.

The following section of this paper is organized as: Part II we discuss related work of duplication detection. Part III fuzzy set information retrieval approach is introduced. Part IV performance problem of original algorithm is presented and performance improvement is made. Part V we present the result and performance analysis of our algorithm. Part VI is conclusion.

2 Related Work

A very easy approach for detecting similar webpages is to evaluate the size or MD5 of the documents/extracted documents. This is obviously the fastest way of comparing documents; however, a slight change of a proposition in the document may result in two totally different size/MD5s.

As a result, feature sentence based duplication detection method[2][3] is introduced. It incorporate clustering concept into this module, and tries to convert an article/sentence to a vector using VSM (vector space model). Then it computes the cosine angle of the vectors and correlation value is derived. However, the vocabulary base for deriving those vectors is only empirical and different content of webpages may require different set of vocabulary bases. Thus, in order to keep up with present update speed of webpages, this algorithm has to maintain a frequently updated library for vocabulary bases.

Another detection strategy is introduced by Giuseppe[4]. Unlike previous approaches which might require pre-processing to eliminate HTML tags, this approach utilizes these tags and compute the LevenShtein Distance of those terms. The idea is innovative; however, different HTML writing style may results in errors in detection.

The SVM(Support vector machine) clustering approach is widely used in graphic processing and pattern recognition, where we map the feature space into a higher dimension one and find a hyperplane to perfectly discriminate the two categories. We could also use this tool to detect webpage duplications[5], though this approach might be too strong, complicated and more importantly, too time-consuming for search engines.

Our algorithm could compensate those disadvantages mentioned above. The details are shown in the next section.

3 Fuzzy Set Information Retrieval Approach

In this approach we first employ pre-processing module to remove HTML tags, digits, English and unrelated information in webpages. After pre-processing, the document should not exist any non-Chinese characters

After pre-processing, for each sentence in article A and article B, we calculate the sentence correlation:

We divide the two sentences into separate words using a method without dictionary database. Taking a sample sentence as example:

服务中心餐饮部(Fu Wu Zhong Xin Can Yin Bu) should be divided into 7 separate words:

服务(Fu Wu), 务中(Wu Zhong), 中心(Zhong Xin), 心餐(Xin Can), 餐饮(Can Yin) and 饮部 (Yin Bu).

This dividing technique does not require frequent update of dictionary.

After dividing those sentences into words, we compute the word-word correlation of sentence C and D one by one using the following equation:

$$\mathbf{W(i,j) = Word\ Correlation(i,j) = \frac{n_{i,j}}{n_i+n_j-n_{i,j}}} \quad (1)$$

Where ni,j is the number of articles consisting both term i and j, ni is the number of articles that consist term i, and nj the number of articles that consist term j.

After calculating the word-word correlation we need to derive word-sentence correlation and further, sentence-sentence correlation. The correlation of sentence j on behalf of term i is given below:

$$\boldsymbol{\theta_{i,s} = 1 - \prod_{j\in s}(1 - W(i,j))} \quad (2)$$

Where W(i,j) is defined in (1).

To reach sentence-sentence correlation we need to sum and regularize word-sentence correlation.

$$\boldsymbol{\varphi_{s,t} = \frac{\sum_{i\in s}\theta_{i,t}}{n}} \quad (3)$$

Where s and t are two pending sentences and $\theta_{i,t}$ is defined in (2).

We set up a threshold for Φs,t to determine whether the two sentences are highly related:

$$\rho_{s,t} = \begin{cases} 1 \text{ if } \varphi_{s,t} > \mathit{Threshold}\ 1 \\ \quad 0 \text{ else} \end{cases}$$

After all the sentence-sentence correlation is derived, we use the following statistic model to compute the article-article correlation:

$$AC'(e,f) = \begin{cases} \dfrac{\sum_s \sum_t \rho_{s,t}}{n_e} & \mathit{if} \displaystyle\sum_s \sum_t \rho_{s,t} < n_e \\ 1 & \text{else} \end{cases}$$

Where AC'(e,f) is the Article Correlation of f on behalf of e, and ne is the total number of sentences in article e.

However, the AC matrix may not be symmetrical. Actually in most cases it is not symmetrical, so we should apply a new model—Dempster Shafer rule to make it symmetrical, which is:

$$\mathbf{AC(e,f)} = \frac{\mathbf{AC'(e,f) * AC'(f,e)}}{\mathbf{1 - AC'(e,f) * AC'(f,e)}}$$

In previous definition we have restricted the maximum value of AC' to 1, so AC(e,f) is a positive value, indicating the correlation between article e and f.

Eventually we have a judging threshold for article-article duplication detection:

Document e and Document f are the same if AC(e,f) > Threshold 2

If we sum this procedure up using a pipeline diagram, it could be expressed as below:

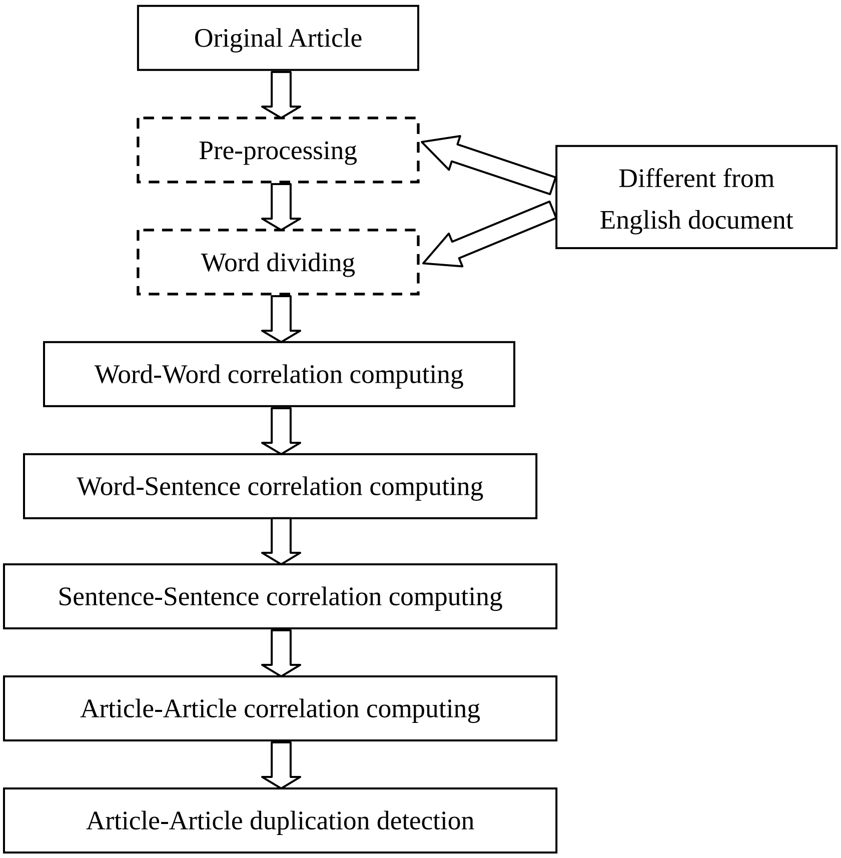

Original Article
Pre-processing
Word dividing
Different from English document
Word-Word correlation computing
Word-Sentence correlation computing
Sentence-Sentence correlation computing
Article-Article correlation computing
Article-Article duplication detection

Fig.1

4 Algorithm Revising and Performance Improvement

We could calculate time complexity of the naïve FSA(fuzzy set approach) as follows:

Consider 2 articles A and B, both with N sentences and each sentence contains M words.

For each sentence we need to check the word-word correlation, if we assume the time consumption for this procedure lasts S seconds.

If there are total P articles that match the query term, the entire process may take as long as:

$$T(A,B) = C_P^2 C_M^2 C_N^2 S$$

As we know, the word-word correlation could only be derived from (1), where $n_{i,j}$ could only be available in index files. This means we have to access 2 index files every time we try to get word-word correlation.

A test was run using the naïve algorithm: we have a small-scale indexing server and search engine designed for a university (http://www.xcu.edu.cn) (approximately 500 total webpages), for a given query ('纳米(nanometer)'), the search engine returned 26 results. Afterwards we tried to run the duplication detection program, but it did not end even after 30 minutes have passed.

We have to bear in mind that this was only a small-scale website. If the amount of webpages is to be increased by a multiple of n, the time complexity would be increased by n2.

After these analyses I have introduced several mechanism to increase the performance of naïve algorithm.

1) Query term combined FSA:

Since this duplication detection module is specifically designed for search engine systems, we should take advantage of this. We may not know what exactly an article's main idea is about, we may not know what discipline a paper is in, but we know the users' query terms and his interest, and this must be of special connection with the results returned from the search engine. Henceforth when we try to derive article-article correlation, we only deal with those sentences with query terms in them.

For example, if a user type '纳米(nanometer)' into the search box, and the search engine returned 2 results: Article A containing 250 sentences, 2 of which consist '纳米(nanometer)'; Article B containing 500 sentences, 6 of which consist '纳米(nanometer)'. This mechanism makes the duplication detection module 'blind' to the rest 248 sentences in article A, and the 494 sentences in article B.

Experiment shows that focusing on those sentences greatly decreases the time-consumption. The quality is not significantly reduced, and in some cases, even improved because unrelated information was ignored.

2) Combine full copy detection method with original approach:

Many search results have been totally the same with only a URL difference, because of the alias system, the search engine was not able to rule them out. In this case we should add an easy full copy detection module before we carry our FSA algorithm out.

3) Set limitations for minimum and maximum sentence length:

In most cases a website might contain many single-word tags indicating buttons/hyperlinks. Those single-words are taken for sentences after the pre-processing procedure. However these information does not usually make any sense. Eliminating those information further increases the processing speed.

4) Avoiding duplicate work:

We keep a LUT (look-up table) live after every article-article correlation function is called. If previous work had found out that A and B are similar, why do we have to do duplicate work on both A and B? This simple idea actually decreases processing time greatly if the duplication rate is rather high.

5) Ignore unimportant documents:

Statistics show that above 70% of user only look at the first page of results from search engine. If we could decrease the total number of articles to be processed, it would be absolutely very effective in performance. We may take the first 10 results into consideration only. However this optimization is a double-edged sword: if the duplication rate is high, we are facing the problem of presenting only 2-3 articles every page to the users. Also, the previous results may be similar with the next page results, which is ignored by this mechanism.

After All these optimizations, the architecture of this module is refined as follow:

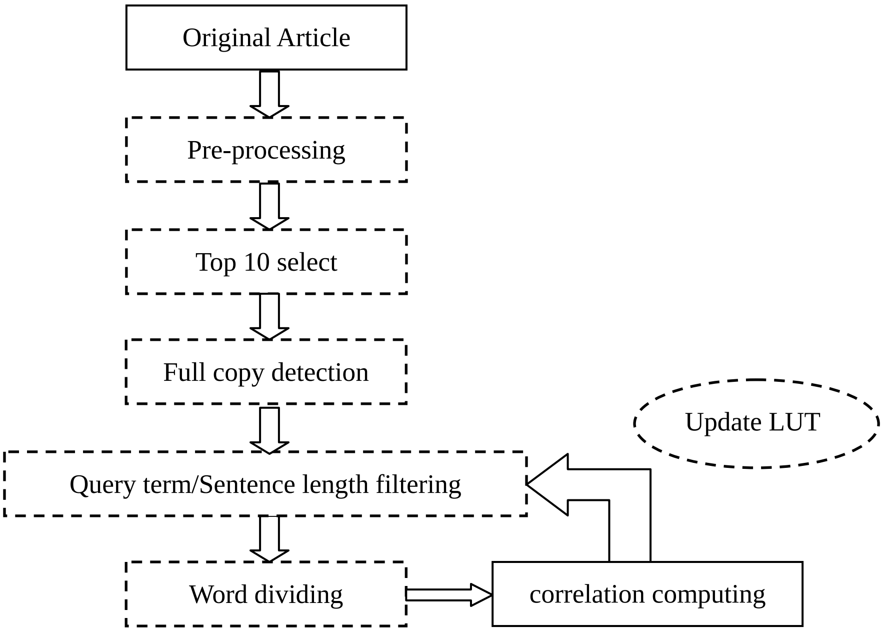


Fig.2

5 Results and Performance Analysis

5.1 Performance analysis:

Due to heavy calculation and redundant work, naïve FSA is not able to finish the duplication detection of 30 query results in 30 minutes. This is not applicable to online searching. As we mentioned before, detection strategy has been changed and now the performance could be shown in the following graph:

Query term is randomly selected.

Table.1

Method	Query Term	Query results	Time consumed	Degree of Similarity
Naïve	纳米(nanometer)	26	>30 min	14/26
Improved	纳米(nanometer)	26	28 sec	14/26
Improved	开发 搜索引擎 (develop search engine)	66	24 sec	27/66

Since we have been focusing on the 'Top Ten' webpages of the query results, the time-consumption is relatively stable.

Table.2

Method	Query Term	Query results	Time consumed
Naïve	纳米(nanometer)	26	>30 min
Improved (w/o Top Ten)	纳米(nanometer)	26	2min 13sec
Improved (w/o Top Ten)	开发 搜索引擎 (develop search engine)	66	Approx. 14min

If we do not incorporate 'Top Ten' algorithm the performance will be greatly degraded.

5.2 Result analysis:

To indicate directly what the results look like, we may refer to a figure from[1]:

Figure 3 shows how the top ten sites are related to each other. X and Y axis are documents, and Z axis is the correlation between X and Y.

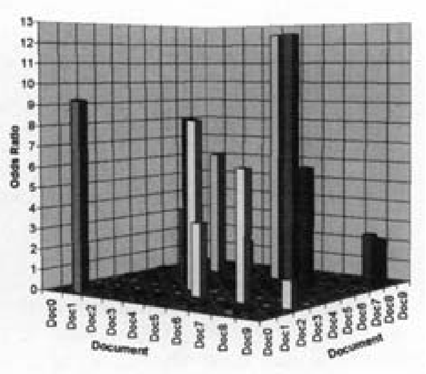


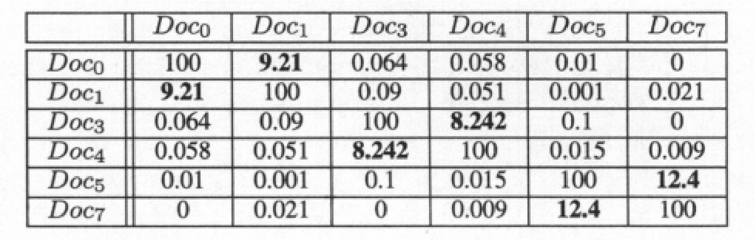
Figure 3

Here we make the correlation between a document and itself 0, which helps us to see clearly the relationships.

As is shown on the graph, some documents have relatively high value of correlation with others, which indicate their similarities with others are high.

If we compare the result of the naïve algorithm with the improved algorithm:

Table.3



	Doc_0	Doc_1	Doc_3	Doc_4	Doc_5	Doc_7
Doc_0	100	**9.21**	0.064	0.058	0.01	0
Doc_1	**9.21**	100	0.09	0.051	0.001	0.021
Doc_3	0.064	0.09	100	**8.242**	0.1	0
Doc_4	0.058	0.051	**8.242**	100	0.015	0.009
Doc_5	0.01	0.001	0.1	0.015	100	**12.4**
Doc_7	0	0.021	0	0.009	**12.4**	100

Table.4

	Doc1	Doc2	Doc3	Doc4	Doc5	Doc6	Doc7	Doc8	Doc9	Doc10
Doc1	0	100	100	5.0	5.0	0.5	0	0	0	0
Doc2	--	0	--	--	--	--	--	--	--	--
Doc3	--	--	0	--	--	--	--	--	--	--
Doc4	2.0	2.0	2.0	0	100	0	0.3	0.3	0.3	0.3
Doc5	--	--	--	--	0	--	--	--	--	--
Doc6	0	0	0	0.5	0.5	0	0	0	0	0
Doc7	0	0	0	0.2	0.2	0.2	0	100	100	100
Doc8	--	--	--	--	--	--	--	0	--	--
Doc9	--	--	--	--	--	--	--	--	0	--
Doc10	--	--	--	--	--	--	--	--	--	0

As is shown above, Figure 4 shows an example of output of the original algorithm. This figure is taken from [1]. The results are clear: Doc0 and Doc1 are similar; Doc 3 and Doc 4 are similar; Doc 5 and Doc 6 are similar.

From our result (Figure 5, result for the query term '纳米(nanometer)'), we know that if two document are judged to be absolutely same, the document that haven't been processed would be ignored afterwards. The '—' symbol stands for 'ignored'.

If we set the threshold to be at 1, then we could reach the following conclusion:

	Doc 1 2 3	Doc 4 5	Doc 6	Doc 7 8 9 10
Doc 1 2 3		similar	--	--
Doc 4 5	similar		--	--
Doc 6	--	--		--
Doc 7 8 9 10	--	--	--	

The actual content of these documents are provided below:

Doc1 2 3 : 许昌学院表面微纳米材料研究所 (Institute of surface Micro/Nanometer material, Xu Chang academy)

Doc4 5 : 研究所（室）-许昌学院 (Brief introduction to important institute of Xu Chang academy)

Doc 6 : 许昌学院科研外事处 (Diplomacy department of research institute of Xu Chang academy)

Doc 7 8 9 10 : 许昌学院 (Index page of Xu Chang academy)

As our results have pointed out, doc 1 2 3 are similar with doc 4 5, which proves to be correct after we have inspected the content of these webpages.

6 Conclusions

We introduced an existed approach to detect duplicated webpages, however the existed approach is too slow, so we did some optimizations on performance: i) combine query terms with duplicate detection module; ii) Combine full copy detection method with original approach; iii) Set limitations for minimum and maximum sentence length; iv) Avoiding duplicate work; v) Ignore unimportant documents. The performance is greatly improved with an acceptable sacrifice on quality.

References

[1] Rajiv Yerra and Yiu-Kai Ng, “Detecting similar HTML documents using a fuzzy set information retrieval approach”, 2005 IEEE International Conference on Granular Computing, 2005, p 693-699

[2] Chen Ji Li, “Duplicated Webpages deletion based on feature code” 2005.

[3] Peng Yuan, “Research on deletion of feature sentence extraction based duplicate web pages”, 2004

[4] Giuseppe Antonio Di Lucca, “An Approach to Identify Duplicated Web Pages”, Proceedings of the 26th Annual International Computer Software and Applications Conference, 2002.

[5] Cheng Jun, “Statistics-based Text Classification”.