CopyCaptor : Plagiarized Source Retrieval System using Global word frequency and Local feedback

Notebook for PAN at CLEF 2013

Taemin Lee[1], Jeongmin Chae[2], Kinam Park[3], and Soonyoung Jung[4*]

[1] Department of Computer Science Education, Korea University, Republic of Korea
[3] Computer Software Engineering, Soonchunhyang University, Republic of Korea
{taeminlee[1], onacloud[2]}@korea.ac.kr, spknn@sch.ac.kr[3], jsy@korea.ac.kr[4*]

Abstract. In this paper, we present a plagiarized source retrieval system called CopyCaptor using global word frequency and local feedback to generate an effective query for finding plagiarized source documents from the given suspicious document on PAN'13 source retrieval task. The system achieved 3rd place in competition with 0.33 F1 score, 0.50 precision and 0.33 recall on the test which find appropriate source documents of 58 suspicious documents from approx. 1 billion web pages.

1 Introduction

Plagiarism is referred as "use or close imitation of the language and thoughts of another author and the representation of them as one's own original work"[1] and it becomes a significant problem. Since mid-1990s, automatic plagiarism detection methods have been developed. Many of them used a relatively small corpus (approx. 100~10000 documents). However, in the real situation, plagiarized documents are made from web resources (at least, 4.63 billion web pages[2]). Therefore, a plagiarism detection method should consider characteristic of web resources.

Since 2012, PAN (Plagiarism analysis, Authorship identification and Near-duplicate detection) research community made two core tasks 'source retrieval' and 'text alignment' for plagiarism detection. First, 'source retrieval' focuses finding source documents from the given suspicious document using a web search engine. Second, 'text alignment' focuses on finding similarity between two documents in a plagiarism detection manner. Detailed information about these tasks is described in [3]. We believe the former one is the key task to solve detecting plagiarized documents in consideration of web resources.

In this paper, we propose a plagiarized source retrieval system called 'CopyCaptor' which uses global word matrix and local feedback for 'Source retrieval' task on PAN'13. In section 2, we describe the framework and method based on heuristics. Section 3 shows its results of evaluation and section 4 summarizes the result of our research and discusses about the future work.

2 Method

To find the source documents from a suspicious document, we developed a plagiarized source retrieval system called 'CopyCaptor' based on a query generation method using global word frequency and local feedback. In this section, we present components of CopyCaptor and describe it briefly.

2.1. Framework of CopyCaptor

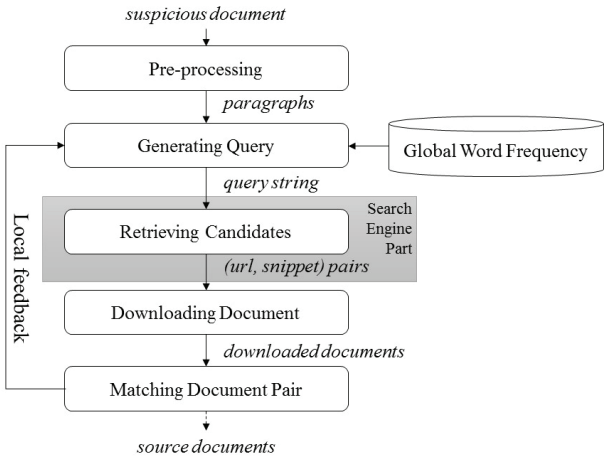

Figure 1 Architecture of CopyCaptor

Figure 1 illustrates the CopyCaptor system for the plagiarized source retrieve task on PAN'13 competition. The CopyCaptor consists five components, 'Pre-processing', 'Query generating', 'Retrieving candidates', 'Downloading document' and 'Matching document pair'.

In the 'Pre-processing', suspicious document divided into paragraphs, and each word in paragraphs are tokenized and stemmed. Also, stop-words are removed. In the 'Query generating' process, our system made query strings with contiguous *k* words of each paragraph, and select the most unique query string. A detailed method and the definition of uniqueness of a query are described in the section 2.2.

For the 'Retrieving candidate', our system uses Indri search engine[4] because it supports structured query operators from INQUERY. We also retrieve snippet of candidate documents using ChatNoir[5] API. If our system seen the same snippet beforehand or snippet has no words, its URL is discarded. In the 'Downloading document', we download *top-k* URLs from candidates and also download URLs which frequently appeared in candidates URLs from different query strings.

Of course, downloaded documents are weather related to suspicious document or not. Therefore, in 'Matching document pair', we align (suspicious document, download document) pairs using simple n-gram match method. If the matching ratio (how many share the same n-gram between suspicious and a download document pair) is over some threshold (e.g. 5% or 100 words), we accept it as a source document.

When the most part of the suspicious document appeared in source documents or a number of querying on a suspicious document is over the number of paragraphs, the system stop retry querying and returns gathered source documents. Otherwise, our system retry generating query with local feedbacks.

2.2. Generating query using global word frequency and local feedback

Our generating query method acquired on some heuristics from an analysis of PAN 13' training data set. We set up three heuristics as follows:

1. The most unique query is the best query.
2. A query should be differ from the previously executed queries.
3. A query formed with contiguous words in a phrase.

To find out 'most unique query', we define the uniqueness of a query as follow:

Definition 1. The uniqueness of a Query: *Given a query formed with words* $\boldsymbol{Q_W}$ *(*w_1*,* w_2 *...* w_n*), and global frequency of words* $\boldsymbol{Q_{GWF}}$ *(GWF[*w_1*], GWF[*w_2*], ..., GWF[*w_n*]), we will say* ***uniqueness of a query*** *is inverse proportional to the product of* $\boldsymbol{Q_{GWF}}$**.**

GWF(Global Word Frequency) is a dictionary that contains a word as a key and an occurrence of a word in different documents on a corpus as a value. Lower occurrence means more uniqueness on a corpus. There are a lot of corpora to get a global word frequency. In this paper, we use google n-gram corpus[6] because of its generality and a wide span of topics. From the above definition, we can calculate a uniqueness of a query string and rank query strings.

We get a local feedback from previous executed queries to generate a more effective query. Because a suspicious documents comply with multiple source documents and the source documents have different words, we prefer choosing a word not exists in previous query string and not exists in match words from 'Matching Document Pair' process. Query strings are made by contiguous *k* words in our system.

From these rules, we design a generating query algorithm using global word frequency and local feedback. Figure 2 depicts its pseudocode.

3 Experiment

We implemented CopyCaptor system using PHP language. Therefore, our system is well attached on internet service and can be simply meshed up with other web services. Using the system, we evaluated of the performance of the proposed system with 'Source Retrieval' task on PAN'13 corpora. PAN'13 corpora contain 40 suspicious documents for training and 58 suspicious documents for the test. We used training corpus to find out appropriate parameters only. We used parameters for the system as follows: *search engine* = Indri, *number of query words* = 8, *number of n of n-gram* = 4, *matched penalty* = 2. Our system uses Indri search engines built on ClueWeb09 corpus which contains approx. 1 billion web pages. Overall, CopyCaptor system achieved 3rd rank in source retrieval task with 0.34 F1 score. The detailed experiment result of the system is shown in Table 2.

Generating Query Algorithm
Input *paragraph, GWF, previous_queries, matched_words, matched_panelty, #_query_words*
Output Q_{str}

```
1   Qstr = NaN;     Qw = Queue();     Qgwf = Queue();       max_uniqueness = 0;
2   Foreach word in paragraph
3       if (word exists previous_queries) continue;
4       if (word exists matched_words) Qgwf.EnQueue(GWF[word] * matched_panelty);
5       else Qgwf.EnQueue(GWF[word]);
6       Qw.EnQueue(word);
7       if (Qw.Count() == #_query_words)
8           uniqueness = 1 / Product(Qgwf);
9           if (uniqueness > max_uniqueness)
10              Qstr = implode(" ", Qw);
11              max_uniqueness = uniqueness;
12          Qw.DeQueue();     Qgwf.DeQueue();
13  return Qstr;
```

Figure 2 Pseudocode of generating query algorithm

Table 1 Experiment results of proposed method on the PAN'13 corpuses

Retrieval Performance			***Workload***		***Time to 1st Detection***	
F1	*Precision*	*Recall*	*Queries*	*Downloads*	*Queries*	*Downloads*
0.35	0.50	0.33	44.04	11.16	7.74	1.72

4 Conclusion

In this paper, we design and implemented a system called CopyCaptor for source retrieval task on PAN'13. Retrieval performance of CopyCaptor shows that the system is based on simple heuristics but it well suited for solving the problem. However, also results shows that the research in this field is not yet conquered. Furthermore, The performance of our proposed system will be improved by applying of the better text alignment algorithm because matching results from text alignment affects query generation by local feedback.

Reference

1. *Random House Webster's Unabridged Dictionary* 1995: Random House.
2. *The size of the World Wide Web (The Internet)*. 2013 [cited 2013 06.14]; Available from: http://www.worldwidewebsize.com/.
3. Martin Potthast, T.G., Matthias Hagen, Martin Tippmann, Johannes Kiesel, Efstathios Stamatatos, Paolo Rosso, and Benno Stein, *Overview of the 5th International Competition on Plagiarism Detection.* CLEF 2013 Evaluation Labs and Workshop, 2013.
4. Strohman, T.a.M., Donald and Turtle, Howard and Croft, W Bruce, *Indri: A language model-based search engine for complex queries.* Proceedings of the International Conference on Intelligent Analysis, 2005. **2**(6): p. 2-6.
5. Potthast, M.a.H., Matthias and Stein, Benno and Graßegger, Jan and Michel, Maximilian and Tippmann, Martin and Welsch, Clement, *ChatNoir: a search engine for the ClueWeb09 corpus.* Proceedings of the 35th ACM SIGIR, 2012: p. 1004.
6. Google. *Google books Ngram Viewer raw data Version 20120701*. 2013 [cited 2013 06.13]; Available from: http://goo.gl/3IIA9.