WEB PLAGIARISM DETECTION BASED ON SEARCH RESULT SNIPPETS

Santipong Thaiprayoon and Choochart Haruechaiyasak

Human Language Technology Laboratory (HLT)
National Electronics and Computer Technology Center (NECTEC)
Thailand Science Park, Klong Luang, Pathumthani 12120, Thailand
{santipong.thaiprayoon, choochart.haruechaiyasak}@nectec.or.th

ABSTRACT

The problem of Web plagiarism is continuously growing due to the large amount of publicly available contents on the Web. Currently, there exist some available plagiarism checking services on the Web. These services are different in design and implementation details. In this paper, we propose another plagiarism detection framework by using the Google Search API. While most existing services support only English, our proposed framework is designed to support both Thai and English languages. Another major difference is, instead of downloading the whole Web pages, our approach performs the text pattern analysis on search result snippets to improve the system response time. In addition, the sliding window technique is applied to improve the system robustness against the modification in plagiarized texts. Based on the experimental results, the sliding window technique help improve the performance based on the F-measure over the no-sliding approach.

***Index Terms*—** Web plagiarism detection, copy and paste plagiarism, Google search API, sliding window technique.

1. INTRODUCTION

Today the Internet has become a common source for students to acquire additional knowledge outside the classroom. On the down side, many students plagiarize on their homework assignments by copying and pasting contents from the Web [1, 3, 9]. Many reported plagiarism cases showed that students use search engines to enter some keywords related to the assigned topic. From the search results, students click on the links to access the Web pages then copy-and-paste the contents without doing much modification or sentence restructuring.

With the high accessibility of the Web, the plagiarism problem has turned into a serious problem for educational institutions. At present, a manual approach used by teachers to check plagiarism is to select suspicious segments in a document, and then enter them into search engines such as Google, Yahoo, MSN and Ask. The downside of this manual approach is that it is labor intensive. Therefore, plagiarism detection tools have become an integral part for assisting teachers to find original sources in which the homework assignments were copied from.

There are a number of tools available as Web services for checking plagiarism for Web contents [6, 11]. Some of the well-known plagiarism detection tools include Plagium[1], Copyscape[2], SNITCH [7] and TurnItIn[3]. These software tools have been developed for plagiarism detection to reduce the effort of teachers who have to check the homework assignments regularly. Most of the plagiarism detection systems apply string matching algorithms to find the most likely matching text sources [10]. The searching is done exhaustively, i.e., the plagiarized document is compared against the entire Web pages returned as the results from search engines. Therefore, it is time consuming. To reduce the system response time, our proposed approach does not download the full Web contents from each search result. Instead, by using the Google Search API, we perform string matching on search result snippets (i.e., short text with highlighted search query).

However, another challenge is when the plagiarized texts contain rewording (i.e., changing words by synonyms or changing the word ordering in the text). The string matching task becomes more difficult to effectively detect plagiarism. In this paper, we propose a plagiarism detection framework which supports both Thai and English texts. To improve the system robustness for the case of text modification, the sliding window technique is applied under the proposed framework. With the sliding window, more text chunks can be generated from the suspicious text by shifting the text window with a fixed number of tokens. As a result, the query to search engines could yield more search coverage.

To evaluate the performance of the proposed framework, we constructed a plagiarism corpus from Thai Wikipedia web pages. The plagiarized texts are created based on some keywords in different class subjects such as social study, Thai history and science. Using this corpus, we performed some experiments to see the effect of varying window size (i.e.,

[1]Plagium (beta): plagiarism tracker & checker, *http://www.plagium.com*
[2]Copyscape plagiarism checker, *http://www.copyscape.com*
[3]TurnItIn: Internet Plagiarism Detection Service, *http://www.turnitin.com*

the number of tokens in a text chunk as query) and the size of sliding window (i.e., the number of shifting tokens on the original text). The remainder of this paper is organized as follows. In Section 2, we introduce our proposed framework and describe the string matching approach in details. In Section 3, we present the evaluation methodology and results. Finally, we present a conclusion with some future works in Section 4.

2. THE PROPOSED FRAMEWORK

In this section, we propose a framework for automated plagiarism detection and provides implementation details which support the framework, including Google Search API. Our framework incorporates the searching capabilities of search engines as the main feature for identifying plagiarized text chunks.

2.1. Sliding Window Technique

To improve the system robustness, we apply a sliding window technique [2, 4, 5] for detecting plagiarism within a suspicious document. This technique creates a series of search queries by shifting the text window with a specified size of sliding size (i.e., number of shifting tokens). The main reason for using the sliding window technique is to increase the effectiveness for checking the plagiarized texts which are partially edited and modified by the students to avoid the teachers suspicion. The details of sliding window technique can be best understood by using the following example.

Example 1: Suppose that a suspicious text, T, is input into the system, the first step is to tokenize T into a series of tokens, TT. Given that the window size is equal to 10 words and the sliding size is equal to 4 words, TT can be split into 6 text chunks, $c_1, \ldots, c_6$, as shown in Fig. 1.

```
T = "การใช้สารอินทรีย์ในวงจรอิเล็กทรอนิกส์ ซึ่งจะมี
     ต้นทุนในการผลิตถูกยิ่งกว่าการผลิตวงจรจากซิลิกอน
     ที่ใช้ในปัจจุบันและสามารถนำไปใช้ได้กับพื้นผิวที่โค้งงอ"

TT = "การ|ใช้|สาร|อินทรีย์|ใน|วงจร|อิเล็กทรอนิกส์| |ซึ่ง|จะ|มี|
      ต้นทุน|ใน|การ|ผลิต|ถูก|ยิ่ง|กว่า|การ|ผลิต|วงจรจาก|ซิลิกอน|
      ที่|ใช้|ใน|ปัจจุบัน|และ|สามารถ|นำ|ไป|ใช้|ได้|กับ|พื้น|ผิว|ที่|โค้งงอ|"

c1 = "การ|ใช้|สาร|อินทรีย์|ใน|วงจร|อิเล็กทรอนิกส์| |ซึ่ง|จะ|"
c2 = "อิเล็กทรอนิกส์| |ซึ่ง|จะ|มี|ต้น|ทุน|ใน|การ|ผลิต|"
c3 = "ทุน|ใน|การ|ผลิต|ถูก|ยิ่ง|กว่า|การ|ผลิต| |"
c4 = "กว่า|การ|ผลิต| |วงจรจาก|ซิลิกอน|ที่|ใช้|ใน|ปัจจุบัน|"
c5 = "ที่|ใช้|ใน|ปัจจุบัน|และ|สามารถ|นำ|ไป|ใช้|ได้|"
c6 = "นำ|ไป|ใช้|ได้|กับ|พื้น|ผิว|ที่|โค้งงอ|"
```

Fig. 1. An example of applying sliding window technique to generate text chunks from a suspicious text.

2.2. Processes under the proposed framework

We propose a framework of plagiarism detection based on sliding window technique. The process of plagiarism detection is illustrated in Fig. 2.

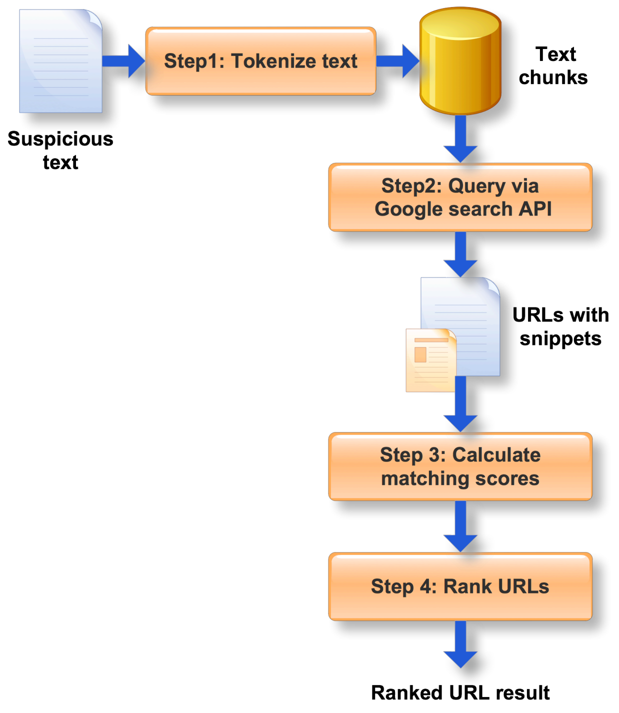

Fig. 2. The proposed plagiarism detection framework

The framework consists of the following four major steps. Given a suspicious document to be checked, the first step is to segment the text string in the document into a set of m text chunks, $C = \{c_1, c_2, \ldots, c_m\}$. The number of text chunks, i.e., m, is varied depending on two parameters: ws, the window size and ss, size of sliding window. The number of text chunks increases as the window size is increasing and the sliding size is decreasing.

The second step is to iteratively send each chunk as a query within double quotation marks (i.e., phrase query) to the Google Search API. By putting the chunk in quotation marks, we indicate to Google API that we only want to search for exact matches for which we are searching. We limit the number of returned search results to 10. A set of n hyperlinks is represented as $URL = \{url_1, url_2, \ldots, url_n\}$ and a set of n snippets as $S = \{s_1, s_2, \ldots, s_n\}$. Each url_i corresponds to an s_i.

On the third step, we calculate the matching score between each snippet and the original document. The score is calculated on the frequency of matching terms. Web pages containing a high number of matching terms would be given with high score. In the final step, the matching scores of every urls are ranked in decreasing order. Then the final ranked results are returned to the user. The matching score between a snippet and a tokenized text, $Matching\_Score(s_i, TT)$, can be calculated by dividing the number of matching words with the total number of words in the snippet. For the case of more than one snippet coming from the same URL, we calculate the average matching scores from all snippets belonging to the same URL. We give an example of how to calculate the matching score for each snippet as follows.

Example 2: Suppose that a suspicious paragraph, a tokenized text, TT, and two snippets, s_1 and s_2 are shown in Fig.3.

```
TT = "โดย|การ|ใช้|สาร|อินทรีย์|กับ|วงจร|อิเล็กทรอนิกส์| |โดย|มี|
         ต้น|ทุน|ใน|การ|ผลิต|ที่|ถูก|ยิ่ง|กว่า|การ|ผลิต|"

s1 = "การ|ใช้|สาร|อินทรีย์|ใน|วงจร|อิเล็กทรอนิกส์|"
s2 = "ซึ่ง|จะ|มี|ต้น|ทุน|ใน|การ|ผลิต|"
```

Fig. 3. The proposed plagiarism detection framework

The matching scores between each snippet and the tokenized text can be calculated as follows.

$$Matching\_Score(s1, TT) = (5/7) \tag{1}$$

$$Matching\_Score(s2, TT) = (6/8) \tag{2}$$

Suppose that s_1 and s_2 comes from the same url_i, then the ranking score of url_i can be calculated as follows.

$$Ranking\_Score(url_i) = ((5/7) + (6/8))/2) = 0.73 \tag{3}$$

We implemented the proposed framework with the PHP scripting language. Text_Diff package[4], an open-source program for comparing the file contents in the PHP environment and rendering the outputs in various formats. The software environment is based on the Apache HTTP server and the MySQL database server running on the Linux operating system. Our system applies the Google AJAX Search API[5] for finding the source of plagiarism Web pages. Our framework supports both Thai and English languages. For Thai language plagiarism detection, word segmentation is required to preprocess given texts into a series of terms before performing the query process.

3. EVALUATION METHODOLOGY AND RESULTS

To evaluate our proposed framework, we constructed a corpus from Thai Wikipedia. The evaluation is performed to compare the effect of selecting different window sizes and the sliding sizes.

[4]Text_Diff, *http://pear.php.net/package/Text_Diff*

[5]Google AJAX Search API, *http://code.google.com/apis/ajaxsearch/web.html*

3.1. Corpus

To evaluate the effectiveness of our framework, we design the corpus by using contents from Thai Wikipedia[6] which is a widely used resource for students. Also, Wikipedia is a huge collection of texts on almost any topic. Our corpus is created from several articles on three class subjects: Social, Thai history and Science. Each subject contains 40 relevant keywords. Each keyword is sent to the Google API for search relevant list of web links. From each Web links, we collect the contents by using a crawler and import into the database. The total number of documents in the corpus is approximately 1,500. We create the source of plagiarized documents which consists of two types that average 400 words or 15 sentences for each documents.

- Original document set: This document is generated with random sentences from the same subject without any modification.
- Modified document set: In this document set, each sentence from original document is modified by shuffling, removing, inserting, or replacing word or short phrases at random [8].

3.2. Results

We performed experiments using a corpus from Thai Wikipedia. Fig. 4 shows the average F-measure of different window sizes. We compare with the five different value size of window. The size of window at 13 words yields the best performance with the F-measure of 0.478.

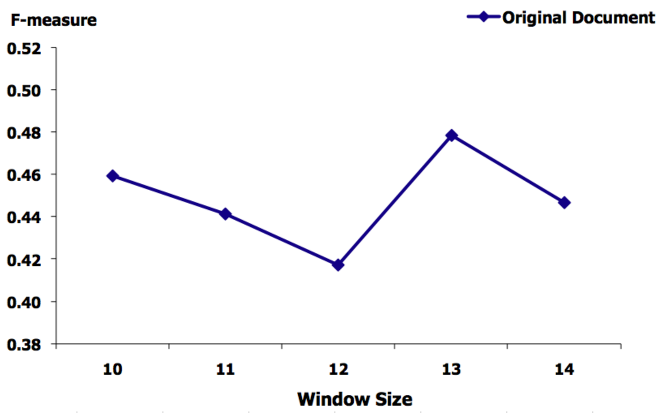

Fig. 4. Evaluation results with varying window size

The experimental results are summarized in Table 1. We test the original document set and modified document set with our framework by varying the sliding size, ss, from 4 to 6 words. From the table, the original document set is suitable

[6]Thai Wikipedia, *http://th.wikipedia.org/wiki/*

for sliding window technique at the size of 6 words. For the modified document set, the sliding size of 4 words yielded the best result.

Sliding Size	Original Document Set			Modified Document Set		
	P	R	F1	P	R	F1
9	0.3266	0.3852	**0.3535**	0.4100	0.1500	**0.2196**
8	0.3138	0.3789	**0.3432**	0.4200	0.1900	**0.2616**
7	0.3015	0.3667	**0.3309**	0.3900	0.2000	**0.2644**
13 No sliding	0.3327	0.3719	**0.3512**	0.3500	0.1700	**0.2288**

Fig. 5. The proposed plagiarism detection framework

Fig. 6 shows the results for the average F-measure, we can conclude that sliding window technique yields better performance than no-sliding window technique for both test sets.

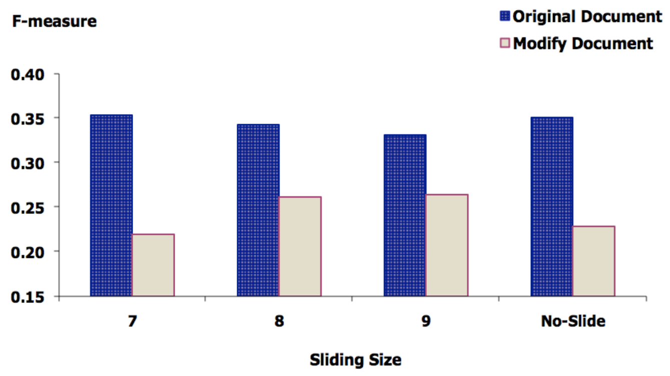

Fig. 6. Evaluation results with varying sliding size

4. CONCLUSION

In this paper, we propose a framework for automatic plagiarism detection of the Web contents. The proposed framework can be potentially useful for teachers to examine students homework assignments which may be copied from the Internet. The framework is developed by using the PHP scripting language including the Text_Diff package and Google AJAX Search API for finding the suspicious documents on the Internet. The framework applies the sliding window technique for solving the case of modification of the Web contents. The experimental results showed that a window size of 13 words yields the best performance on the corpus. For the original document corpus, the sliding size of 6 words yields the best result. For the modified document corpus, the sliding size of 4 words yields the best result. In future works, we will improve the framework by reducing the number of search queries to further reduce the system response time.

5. REFERENCES

[1] Anderson N., "Can Web-Based Plagiarism Detection Beat a Google Search?," Available at *http://arstechnica.com/tech-policy/news/2009/05/can-web-based-plagiarism-detection-beat-a-google-search.ars*, May 31, 2009.

[2] Butakov S. and Shcherbinin V., "On the number of search queries required for Internet Plagiarism Detection," Proceedings of 2009 Ninth IEEE International Conference on Advanced Learning Technologies, pp. 482–483, 2009.

[3] Colin J. Neil and Ganesh Shanmuganthan, "A Web-Enabled Plagiarism Detect Tool," *IT Professional*, pp. 19–23, September, 2004.

[4] Knight A., Almeroth K. and Bimber B., "An Automated System for Plagiarism Detection Using the Internet," *Proc. of World Conference on Educational Multimedia, Hypermedia and Telecommunications*, pp. 3619–3625, 2004.

[5] Liu, Y.-T., Zhang, H.-R., Chen, T.-W., Teng, W.-G, "Extending Web Search for Online Plagiarism Detection," *Proceedings of the IEEE Int. Conf. on Information Reuse and Integration (IRI 2007)*, pp. 164–169, 2007.

[6] Lukashenko R., Graudina V., and Grundspenkis J., "Computer-Based Plagiarism Detection Methods and Tools: An Overview," *Proc. of the 2007 international conference on Computer systems and technologies (CompSysTech07)*, 2007.

[7] Niezgoda, S. and Way, T. P., "SNITCH: A Software Tool for Detecting Cut and Paste Plagiarism," *Proc. of the 37th SIGCSE Technical Symposium on computer science education*, pp. 51–55, March, 2006.

[8] Potthast M., Stein B., Eiselt A., Barrn-Cedeno A., and Rosso P., "Overview of the 1st International Competition on Plagiarism Detection," *PAN09 Workshop*, pp.1–9, 2009.

[9] Ranatunga R., Atukorale A., and Hewagamage K., "An Integrated Framework for Detecting Plagiarism in e-Learning Systems," 2007.

[10] Vaughn M. Segers and Connan J., "An Online System for Plagiarism Detection," *Proc. of South African Telecommunication Networks and Applications Conference (SATNAC 2008)*, pp. 409–412, 2008.

[11] Plagiarism.org, available at *http://www.plagiarism.org*