Early-Detection System for Cross-Language (Translated) Plagiarism

Khabib Mustofa and Yosua Albert Sir

Department of Computer Science and Electronics,
Universitas Gadjah Mada
khabib@ugm.ac.id, yosuasir@gmail.com

Abstract. The implementation of internet applications has already crossed the language border. It has, for sure, brought lots of advantages, but to some extent has also introduced some side-effect. One of the negative effects of using these applications is cross-languages plagiarism, which is also known as *translated plagiarism*.

In academic institutions, translated plagiarism can be found in various cases, such as: final project, theses, papers, and so forth. In this paper, a model for web-based early detection system for translated plagiarism is proposed and a prototype is developed. The system works by translating the input document (written in Bahasa Indonesian) into English using Google Translate API components, and then search for documents on the World Wide Web repository which have similar contents to the translated document. If found, the system downloads these documents and then do some preprocessing steps such as: removing punctuations, numbers, stop words, repeated words, lemmatization of words, and the final process is to compare the content of both documents using the modified sentence-based detection algorithm (SBDA). The results show that the proposed method has smaller error rate leading to conclusion that it has better accuracy.

Keywords: translated plagiarism, sentence-based detection algorithm (SBDA), modified-SDBA,Google API.

1 Introduction

Since it was first introduced, internet has brought several changes in human life, not to mention in academic environment. The existence of search engines makes students and teachers easy in finding materials for enhancing their knowledge, but in other point of view, it also facilitates any attemps of academic misconduct such as plagiarism.

Generally, academic misconduct and plagiarism may happen within several types:

1. *copy-paste* : copying part or the whole content of document
2. *paraphrasing* : changing the grammar, changing the order of constructing sentences, or rewriting the documents' content using synonim

K. Mustofa et al. (Eds.): ICT-EurAsia 2013, LNCS 7804, pp. 21–30, 2013.

3. *translated plagiarism*: translating part or the whole content of document from some language to some other language

The above approaches can fall into plagiarism if the writer does not provide correct citation or without mentioning the source document references ([1]). Plagiarism type (1) or (2) can easily be done, and type (3) is little bit more complicated as the writer still needs to translate the source into different language. According to [1], several tools already exist to suspect or detect plagiarism type (1) or (2), such as : *Turnitin, MyDropBox, Essay Verification (EVE2), WcopyFind, CopyCatch, Urkund*, and *Docoloc*, while plagiarism of type (3) was discussed in ([2]), eventhough without revealing the quantitative accuracy.

This paper will discuss a model and establishment of a system for early detecting translated plagiarism. The system works under the following constraints:

- source document is written in Bahasa Indonesia.
- the system is not a "silver bullet" to translated plagiarism. The output of the system is not an absolute judgement whether a palgiarism does exist.
- the algorithm used in comparing documents is based on *sentence based detection algorithm*
- to enhance the accuracy of detection process, the algorithm is slightly modified by incorporating synonyms of the words constructing sentences. This approach is carried out during stemming and lemmatization process.

2 Problem Formulation

Suppose there exists a suspect document D_q, written in Bahasa Indonesia. On the other hand, there is also a set of vast amount of documents (reference documents), written in other language, Ω, available on the web repository. In this case, for simplicity, we will assume that all documents, $\forall d_i \in \Omega$, are written in English. When there exist statements in or part of a document D_q, whose translation is similar in meaning to statements from some documents in Ω, *how can we find and identify such statements, either in suspected document and also in reference documents*?

Using sentence-based detection algorithm, the target can be obtained by computing similarities which result from comparing each statements in suspected document, $\forall s_q \in D_q$, with all statements in all reference documents, $\forall s_r \in D_r : \forall D_r \in \Omega$. Theoretically, the above process is feasible, but in practice, special treatment should be incorporated as the size or dimension of Ω is big enough and also both the D_q and D_r are in different languages.

The following questions give us guidance in understanding the approach to be discussed further in this paper:

1. How is the system architecture to achieve the goal of detecting translated plagiarism?
2. How to translate D_q (in Bahasa Indonesia) into D_q^* (in English) , where later the statements in D_q^* will be compared with statements in D_r ?

3. How to reduce the size of Ω, for example by constructing Θ, where $\Theta \subset \Omega$, such that $|\Theta| \lll |\Omega|$ and $\forall D_x \in \Theta$, D_x is a document which has similar (not necessarily all) content with D_q^*
4. How to calculate the similarity between $\forall s_q \in D_q^*$ and $\forall s_r \in D_x : \forall D_x \in \Theta$?

Figure 1 shows the architecture of the system as the realization to answer the above questions.

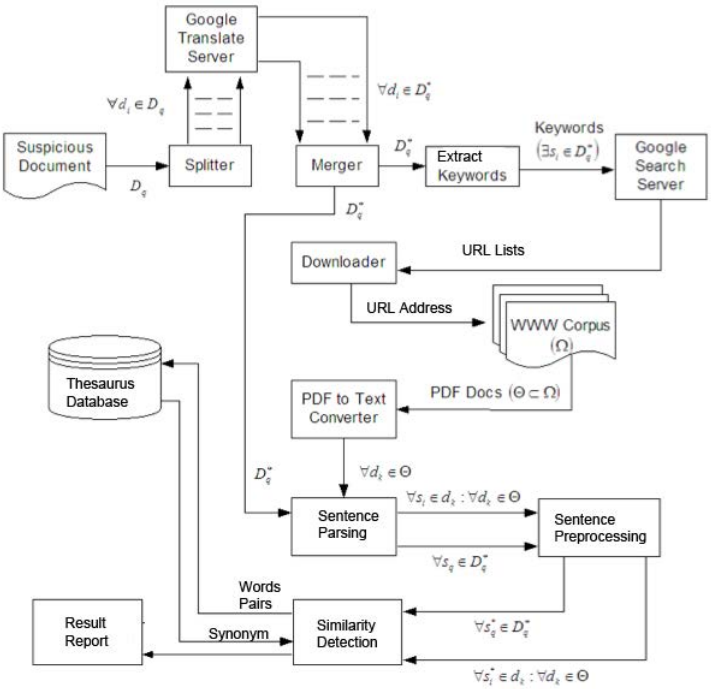


Fig. 1. Architecture of the Early-Detection System for Translated Plagiarism

3 Related Works

In general, the methods for detecting plagiarism on text documents can be categorized into two ways: *document fingerprint comparison* and *full-text comparison* [3]. In the first approach, contents of the input documents (both suspected and reference documents) are converted into a more compact form, based on some predefined method, and then the detection process is carried out by comparing the fingerprints without comparing the whole contents.

Kent and Salim ([2]) explored the first approach by forming *4-grams document fingerprinting* and then comparing the fingerprints using *Jaccard Distance*. The result showed that the approach incurred disadvantages:

1. vulnerable against changes in words order or sentence order. Changes in words or sentence order, even just a small changes, may result in significant changes in fingerprints.
2. unable to detect small change that may be aded by plagiarist, such as word insertion or deletion

White and Joy ([3]) implemented the second approach by using *sentence-based detection algorithm*. In this method, the detection process on plagiarism between two documents is approached by calculating similarities of all pairs of sentences constructing the documents.

This paper will discuss an approach which extends the second approach by adding feature of incorporating synonym to enhance the capability of finding sentences eventhough the sentences have been modified by changing some words with their synonyms. This means, compared to the method implemented by Kent and Salim ([2]), the proposed approach will overcome the disadvantages of using fingerprinting. While compared to White and Joy ([3]) which implements *sentence-based detection algorithm*, the proposed approach will enhance the method by adding capability of invetigating synonims of words appearing in the sentences.

4 Methodology

4.1 Documents Similarity Measures

A document can be viewed as a series of tokens which may come in form of letters, words or sentences. If it is assumed that there is a parser capable of parsing the document contents, ignoring puctuation marks, formatting commands and capitalization, then the output of such parser is a canonical sequence of tokens [4]. In his paper [4], Broder introduced two metrics for measuring the similairty between documents A and B : *resemblance* ($r(A, B)$) and *containment* ($c(A, B)$), expressed as follows:

$$c(A, B) = \frac{|d(A) \cap d(B)|}{|d(A)|} \tag{1}$$

$$c(B, A) = \frac{|d(A) \cap d(B)|}{|d(B)|} \tag{2}$$

$$r(A, B) = \frac{|d(A) \cap d(B)|}{|d(A) \cup d(B)|} \tag{3}$$

where

- $d(X)$ symbolizes set of token in document X
- $r(A, B)$ has value $x \in \Re, x \in [0, 1]$. If $r(A, B) = 1$ then $d(A) = d(B)$
- $c(A, B)$ has value $x \in \Re, x \in [0, 1]$. If $c(A, B) = 1$ then $d(A) \subseteq d(B)$. Two documents A and B are said to be identical if and only if the set of all tokens in A is subset of the set of all tokens in B and vice versa.

– Assumed that the canonical sequence of token is in form of a sentence, $|d(x)|$ indicates the number of sentences in document X (length/size of the set), and $|d(A) \cap d(B)|$ can be considered as common sentences found in document A and B.

As an illustration, given that

Document A : *a rose is a rose is a rose*
Document B : *a rose is a flower which is a rose*

Both sentences can be pre-processed as shown in table 1, and the value of $c(A, B) = 87.5\%$), $c(B, A) = 77.78\%$ and $r(A, B) = 70\%$

Table 1. Example of documenst preprocessing to obtain *resemblance* and *containment*

A	B	$d(A) \cap d(B)$	$d(A) \cup d(B)$
a:3	a:3	a:3	a:3
rose:3	rose:2	rose:2	rose:3
is:2	is:2	is:2	is:2
	flower:1		flower:1
	which:1		which:1

4.2 Sentence-Based Detection Algorithm

In order to apply this method, in which the comparison of all pairs of sentences from both document should be done, three steps must be performed:

1. Documents Preprocessing. This step includes: decapitalization, removing stop words, removing duplicate words
2. Computing Sentences Similarity. Assume that A and B are documents to be compared, comprised of sentences. we use the following symbols to shorten the upcoming discussion.
   – $d(X)$: a set of sentences in document X
   – s_i^x : i^{th} sentence of document X
   – $d(s_i^x)$: set of words comprising s_i^x, can also be denoted as $s_i^x = \{w_1^x, w_2^x, \ldots, w_n^x\}$

   The measure of similarity can be computed from the following equations:

$$common_{words} = \begin{cases} 1 & \text{if } w_i^a \text{ and } w_j^b \text{ is identical} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$SimScore(A, B) = \left(|common_{words}| \times \frac{|d(A)| + |d(B)|}{2 \times |d(A)| \times |d(B)|} \right) \times 100\% \tag{5}$$

$$SimSent(A, B) = \begin{cases} SimScore(A, B) & \text{, if } SimScore(A, B) \geq SimTh \\ & \text{OR } |common_{words}| \geq ComTh \\ 0 & \text{, otherwise} \end{cases} \tag{6}$$

   where $ComTh$ represents *Common Threshold* and $SimTh$ indicates *Similarity Threshold*

3. Computing Documents Similarity. Whenever the whole pairs of sentences from document A and B have been examined and sentences similarity scores have been obtained, the document similarity score can be calculated by summing all sentences similarity scores.

4.3 Modified Sentence-Based Detection Algorithm

In the previous section, from equation (4), (5) and (6), it is clear that the existence of common words clearly contributes to the similarity scores. Suppose that within the two sentences to be compared there exist words form the first sentence having similar meaning with some words in the second sentence. *How if we treat those pairs of words also as common words (with different weight of commonness)*?

Assuming the above question works, we can derive the proof that changing some words in the first sentence into their synonyms will affect the similarity measures as long as the synonyms are in the second sentence.

Suppose the first sentence is the i^{th} sentence of document A, denoted by $s_i^a = \{w_1^a, w_2^a, w_3^a, \ldots, w_n^a\}$, and the second sentence is the j^{th} sentence of document B, denoted by $s_j^b = \{w_1^b, w_2^b, w_3^b, \ldots, w_m^b\}$. Then:

1. $common_{OLD}(s_i^a, s_j^b) = s_i^a \cap s_j^b$
2. $Syn(w)$ =synonym of w
3. $Diff(s_i^a, s_j^b) = s_i^a - s_j^b$
4. $Diff(s_j^b, s_i^a) = s_j^b - s_i^a$
5. $SynWORD(s_i^a, s_j^b) = \{w_k | w_k \in Diff(s_i^a, s_j^b) \wedge Syn(w_k) \in Diff(s_j^b, s_i^a)\}$
6. $|common_{NEW}| = |common_{OLD}| + |SynWORD(s_i^a, s_j^b)| \times 0.5$, as synonyms are considered common words with different weight of commonness (in this case 0.5)

Hence

$$(SynWORD(s_i^a, s_j^b) \neq \{\}) \Longrightarrow (|common_{NEW}| > |common_{OLD}|)$$

Reformulation of equation (4) and (5) by incorporating synonyms yields slightly different forms:

$$common_{words} = \begin{cases} 1 & \text{if } w_i^a \text{ and } w_j^b \text{ is identical} \\ 0.5 & \text{if } w_i^a \text{ is synonym of } w_j^b \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$SimScore(A, B) = \left(\frac{2 \times |common_{words}|}{|d(A)| + |d(B)|} \right) \times 100\% \tag{8}$$

In standard Sentence-Based Detection Algorithm, $SimScore(A, B)$ is calculated using asymmetrical average method (equation 5), while in modified algorithm it is expressed using *Dice* method (equation 8).

The measure or score of documents similarity is then defined as combination of two asymmetric similarity scores [5]:

$$Sim(A,B) = \left\langle \frac{|d(A) \cap d(B)|}{|d(A)|}, \frac{|d(A) \cap d(B)|}{|d(B)|} \right\rangle \tag{9}$$

The same approach is also used in [6] and [7]. By using this form, two-way similarity score can be obtained and gives better picture of the relationship between the two document. Suppose we have two documents A and B, and $|d(A)| = 120$ sentences, $|d(B)| = 160$ sentences and $|d(A) \cap d(B)| = 80$ sentences, then the similarity score of document A and B, $Sim(A,B) = \langle 0.667, 0500 \rangle$ which can be interpreted that two third of sentences in A can also be found in B and half of sentences in B can be found in A.

4.4 Illustration of Algorithm Usage

In this section, we will look on how both methods (standard Sentence-Based Detection and Modified Sentence-Based Detection) are applied to the same dataset and see the differencces of their usages.

Let us assume that document A and B have the following contents:

> A : Face detection is one of the crucial early stages of face recognition systems are used in biometric identification.
> B : Face detection is one of the most important preprocessing step in face recognition systems used in biometric identification.

and assign *similarity threshold* 80, *common threshold* 6 and *stop words* {the, is, on, in, are}. After pre-processing step (decapitalization, removing stop words, removing duplicate words), we have:

1. $SentenceObject_A = \{face,\ detection,\ one,\ crucial,\ early,\ stages,\ recognition,\ systems,\ used,\ biometric,\ identification\}$
2. $SentenceObject_B = \{face,\ detection,\ one,\ most,\ important,\ preprocessing,\ step,\ recognition,\ systems,\ used,\ biometric,\ identification\}$
3. $common_{words}(A,A) = \{face,\ detection,\ one,\ crucial,\ early,\ stages,\ recognition,\ systems,\ used,\ biometric,\ identification\ \}$
4. $common_{words}(A,B) = \{face,\ detection,\ one,\ recognition,\ systems,\ used,\ biometric,\ identification\}$
5. $Diff(SentenceObject_A, SentenceObject_B) = \{crucial,\ early,\ stage\}$
6. $Diff(SentenceObject_B, SentenceObject_A) = \{most,\ important,\ preprocessing,\ step\}$.

   By pairing each words in $Diff(SentenceObject_A, SentenceObject_B)$ with each words in $Diff(SentenceObject_B, SentenceObject_A)$ and consulting with translation service (such as Google Translate), additional information about synonym is obtained, $SynWord(A,B) = \{important, step\}$ (as "important" is synonym of "crucial" and "step" is synonym of "stage").

Calculating Similarity Using Sentence-Based Detection Algorithm. Based on the summary of preprocessing result above, the following scores can easily be obtained:

1. $SimScore(A, A) = \left(11 \times \frac{11+11}{2\times11\times11}\right) \times 100 = 100$ and $SimSent(A, A) = 100$
2. $SimScore(A, B) = \left(8 \times \frac{11+12}{2\times11\times12}\right) \times 100 = 69.70$ and $SimSent(A, B) = 69.70$
3. Similarity between A (as reference) and B is $Sim(A, B) = \frac{69.70}{100} \times 100\% = 69.70\%$

Calculating Similarity Using *Modified* Sentence-Based Detection Algorithm (SBDA). Based on the summary of preprocessing result above, the following scores can easily be obtained:

1. $|common_{NEW}| = |common_{OLD}| + |SynWord(A, B)| \times (0.5) = |8| + |2| \times (0.5) = 9$
2. Based on Eq . 8, $SimScore(A, B) = \left(\frac{2\times9}{11+12}\right) \times 100 = 78.26$
3. Similarity between A and B is $Sim(A, B) = \langle 81.82\% : 75\% \rangle$. This score can be interpreted as: 81.82% of sentences in document A can be found in document B, and 75% of sentences in document B can be found in A.

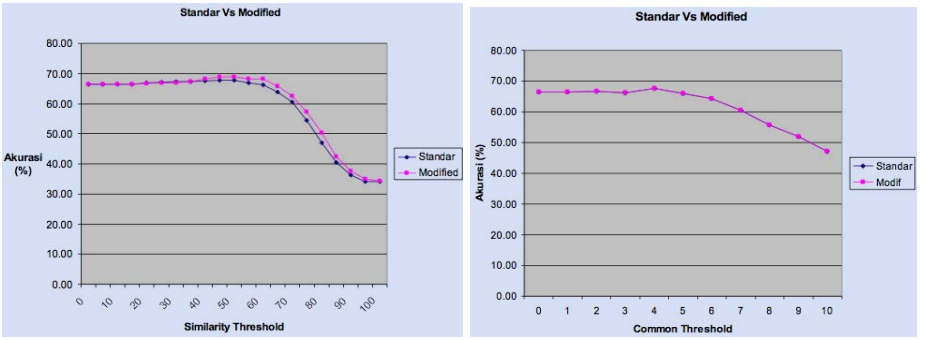
5 Implementation, Testing and Results

Modules Implementation. The system is built by extending and reusing some existing tools. Based on the architecture given at Figure 1, the following are modules developed and tested against some libraries:

1. **Translation Module.** As Google Service restricts the length of translated text, this module first splits long text into possibly several chunks of size maximum 4KB. Each chunk is sent to Google as a request and then the result is combined again to construct the whole document translation.
2. **Document-Searching Module.** Searching of documents on the web repository using search engine requires keywords. This module uses *Named-Entity Recognition (NER)* from Standford, available at `http://nlp.stanford.edu/software/stanford-ner-2009-01-16.tgz`. The tools will search, identify and extract entities of type *person, location* and *organization*. Those types are unique, difficult to plagiarize and suitable to be taken as keywords. This module will perform keyword extraction, look for documents in web repository (based on keywords extracted) and, then download the found PDF documents.
3. **Text Extraction from PDF Documents.** The contents of PDF documents just downloaded are then extracted using existing tools *xpdf version 3.2*.
4. **Content Preprocessing.** This module is responsible for: *decapitalization*, eliminating stop words and punctuation symbols, eliminating repitition of words and lemmatization.

Testing and Results. For the sake of testing, all reference datasets (*corpus*) used in this research are deterministic, in the sense that the similarity degrees of the datasets have been apriori known.

1. **Unit Testing.** Reference Dataset used in this unit testing is taken from *Microsoft Research Paraphrase Corpus*, available at `http://research.microsoft.com/en-us/downloads/607d14d9-20cd-47e3-85bc-a2f65cd28042`. This dataset consists of 1725 pairs of sentences extracted from thousands documents, having been justified by human annotator to classify whether the pair of sentences is a paraphrase or not, resulting 1147 pairs are identified as paraphrases and 578 are not paraphrases. Based on the dataset, by adjusting several values of *similarity threshold* or *common threshold*, as depicrted in Fig. 2, the optimal value for similarity threshold is 50%, and common threshold is 4.



(a) Accuracy vs Similarity Threshold (b) Accuracy vs Common Threshold

Fig. 2. Accuration versus *Similarity Threshold* or *Common Threshold*

Table 2. Average Asymmetric Similarity and Error Rate between Modified and Standard Method

Doc ID	Average Asymmetric Similarity			Error Rate	
	Modified	Standard	Actual	Modified	Standard
1	56.82	42.66	68.18	129.16	651.53
2	61.88	44.89	66.31	19.58	458.60
3	51.24	31.50	65.22	195.30	1136.70
4	51.23	17.45	70.93	388.29	2860.65
5	48.04	29.15	70.22	491.73	1686.33
6	56.97	27.34	68.37	129.85	1683.05
7	60.58	49.09	80.77	407.84	1003.94
8	53.35	31.82	69.77	269.45	1439.82
9	63.20	60.24	72.22	81.45	143.64
10	46.46	47.51	71.80	641.86	590.00
Root Mean Square Error (RMSE)				16.60	34.14

2. **Integration Testing.** For the integration and functional testing, ten documents from `http://pps.unnes.ac.id` are taken as samples. These samples document are processed, and then their similarity are calculated. Table 2 shows the result of similarity test, revealing the outperformance of the proposed method (modified SBDA) against the standard method.

6 Conclusion

Plagiarism is a serious misconduct in academic environment, thus it must be anticipated. The advancement in internet technology and services have been enabling users to more easily conduct plagiarism, but on the other hand, such condition also provides environment to easier check whether any attemp to plagiarism has happened. This paper has shown an approach to early detection of translated plagiarism. The prototype was developed by integrating online services, online repository and implementing modified sentence based detection algorithm. From the result, the following can be concluded:

1. The *modified SBDA* show higher accuracy compared to standard SBDA. This is indicated by smaller value of *error-rate*
2. The modification of standard SBDA is carried out by incorporating synonym conversion. Converting words into their synonyms will *increase the count of common words* between the documents compared, contributing to the better accuracy in document similarity measurement.

References

1. Maurer, H., Kappe, F., Zaka, B.: Plagiarism - a survey. Journal of Universal Computer Science 12(8), 1050–1084 (2006)
2. Kent, C.K., Salim, N.: Web based cross language plagiarism detection. Journal of Computing 1(1) (2009)
3. White, D.R., Joy, M.S.: Sentence-based natural language plagiarism detection. Journal on Educational Resources in Computing 4(4) (2004)
4. Broder, A.Z.: On the resemblance and containment of documents. In: Compression and Complexity of Sequences, SEQUENCES 1997, pp. 21–29. IEEE Computer Society (1997)
5. Monostori, K., Finkel, R., Zaslavsky, A., Hodász, G., Pataki, M.: Comparison of Overlap Detection Techniques. In: Sloot, P.M.A., Tan, C.J.K., Dongarra, J., Hoekstra, A.G. (eds.) ICCS-ComputSci 2002, Part I. LNCS, vol. 2329, pp. 51–60. Springer, Heidelberg (2002)
6. Yerra, R.: Detecting similar html documents using a sentence-based copy detection approach. Master's thesis, Brigham Young University (2005)
7. Smith, R.D.: Copy detection systems for digital documents. Master's thesis, Brigham Young University (1999)