algorithms

MDPI

Article

Cross-Language Plagiarism Detection System Using Latent Semantic Analysis and Learning Vector Quantization †

Anak Agung Putri Ratna *, Prima Dewi Purnamasari, Boma Anantasatya Adhi, F. Astha Ekadiyanto, Muhammad Salman, Mardiyah Mardiyah and Darien Jonathan Winata

Department of Electrical Engineering, Faculty of Enginering, Universitas Indonesia, Depok 16424, Indonesia; prima.dp@ui.ac.id (P.D.P.); boma@ee.ui.ac.id (B.A.A.); astha.ekadiyanto@ui.ac.id (F.A.E.); salman@eng.ui.ac.id (M.S.); mardiyah@ui.ac.id (M.M.); darien.jonathan@ui.ac.id (D.J.W.)

* Correspondence: ratna@eng.ui.ac.id; Tel.: +62-21-727-0078

† This paper is an extended version of a conference paper entitled "Analysis of the Effect of Term–document Matrix on the Accuracy of Latent Semantic Analysis-Based Cross-Language Plagiarism Detection" presented at the International Conference on Network, Communication and Computing 2016 (ICNCC 2016) at Kyoto, Japan, 17–20 December 2016.

Academic Editor: Andras Farago
Received: 31 March 2017; Accepted: 10 June 2017; Published: 13 June 2017

Abstract: Computerized cross-language plagiarism detection has recently become essential. With the scarcity of scientific publications in Bahasa Indonesia, many Indonesian authors frequently consult publications in English in order to boost the quantity of scientific publications in Bahasa Indonesia (which is currently rising). Due to the syntax disparity between Bahasa Indonesia and English, most of the existing methods for automated cross-language plagiarism detection do not provide satisfactory results. This paper analyses the probability of developing Latent Semantic Analysis (LSA) for a computerized cross-language plagiarism detector for two languages with different syntax. To improve performance, various alterations in LSA are suggested. By using a linear vector quantization (LVQ) classifier in the LSA and taking into account the Frobenius norm, output has reached up to 65.98% in accuracy. The results of the experiments showed that the best accuracy achieved is 87% with a document size of 6 words, and the document definition size must be kept below 10 words in order to maintain high accuracy. Additionally, based on experimental results, this paper suggests utilizing the frequency occurrence method as opposed to the binary method for the term–document matrix construction.

Keywords: Latent Semantic Analysis; learning vector quantization; plagiarism detection system

1. Introduction

High-level technology makes people desire comfort—even in negative ways. Papers, journals, and assignments are widely spread over the internet, which makes them accessible to most scholars. Easy access to information on the internet for academic purposes may lead to unethical actions, such as plagiarism. One way to plagiarize is by translating a paper into different language. With several editings, people can easily submit their plagiarized papers to local publishers. Plagiarism describes the appropriation of other persons' ideas and intellectual or creative work and passing them of as one's own [1]. Observations of plagiarism behavior in practice reveal a number of commonly found methods for illegitimate text usage, which are characterized below.

"Disguised plagiarism" subsumes practices intended to mask copied segments [2].

"Technical disguise" summarizes techniques for hiding plagiarized content from being automatically detected by exploiting weaknesses off current text-based analysis methods, e.g., by substituting

characters with graphically identical symbols from foreign alphabets or inserting letters in white font color [3].

"Undue paraphrasing" means the intentional rewriting of foreign thoughts in the vocabulary and style of the plagiarist without giving due credit for concealing the original source [2].

"Idea plagiarism" encompasses the usage of a broader foreign concept without due source acknowledgement [4].

"Self-plagiarism" characterizes the partial or complete reuse of one's own previous writings not being justified by scientific goals, e.g., for presenting updates or providing access to a larger community, but primarily serving the author, e.g., for artificially increasing citation counts [5,6].

Hence, it is necessary to create countermeasures; one of which is checking the submitted papers for any indication of plagiarism and apply penaties to its authors. Manual plagiarism detection; however; is time consuming and requires a lot of effort, diminished its effectiveness. An automated plagiarism detection system; so-called computer-assisted plagiarism detection; is needed to able to detect cross-language similarities between the documents which are currently under development.

Automated plagiarism detection is the process of determining plagiarism in a text-based paper by using software. Some methods can be used to perform textual plagiarism detection automatically, such as checking the grammatical structure of the document (grammar based), using vector-space models to detect similarities between two documents (semantic based), using a reference corpus of documents whose contents may have been plagiarized, using a clustering method, or combining two or more of these techniques [7].

Currently, the majority of academic sources (such as journals, proceedings, and books) are written in English, and the action of plagiarism often involves language translation process. In order to provide better tools for local publishers, the plagiarism detection system developed in this research focuses on bilingual plagiarism detection. This paper discusses a case study in which the document tested for plagiarism is in the Indonesian language while the suspected source of plagiarism is written in English. Since the sentences' structure between Indonesian and English are very different, structure- and grammar-based detection techniques are highly sophisticated. In this research, the detection method is conducted using a semantic-based technique.

The cross-language plagiarism detection system needs to be a semantic-based computer-assisted plagiarism detector and the choice of algorithm used in the system is therefore Latent Semantic Analysis (LSA). The algorithm does not only find the similarity between two documents, but it is also supported by a language translation process. Since the detection is based on semantics, current work only focuses on word translation without paying attention to grammatical rules. This is because semantics only views a document as interrelated words and the frequency of those words within, removing the necessity for applying accurate language rules.

The method proposed in this research uses LSA, which previously showed good results in an essay grading system. The reason for using LSA in our bilingual plagiarism detection system is based on the lack of natural language processing systems for the Indonesian language; consequently, the best method would be by paying attention only to the lexical features of plagiarism detection; this is supported by LSA. Because LSA only pays attention to the existence of words in a document and does not regard the grammar, the word-by-word translation should be sufficient, and this will make the translation process relatively simple. The classification part of the detection system is conducted using a learning vector quantization (LVQ) algorithm, which is a neural network algorithm to group data based on the distance from the input vector to the classification vector.

We have developed a bilingual plagiarism detection system using LSA and LVQ. The success rate of the proposed methodology is measured by the value of precision, recall, and F-measure. Precision and recall are two calculations that are widely used to measure the performance of a system or method. They are used in this paper to measure the performance of natural language processing, which is used for the relevant documents. Precision is the degree of accuracy of the information expected by the user compared to the results given by the system (how precise the prediction of natural language

processing is). Recall is the success rate of the system in rediscovering updates of information in the test language [8].

This paper is organized as follows: Section 2 reviews the techniques implemented in our proposed algorithm, including LSA and LVQ techniques. Section 3 explains the methodology for conducting automated cross-language plagiarism and how LSA and LVQ are implemented in the plagiarism checker. Section 4 explains the experiments and results of our proposed methodology in recognizing plagiarism. Finally, Section 5 concludes the paper.

2. Literature Review

2.1. Latent Semantic Analysis (LSA)

LSA is a method used to extract and perform contextual usage representation of the meaning of words using statistical computation, which is applied to a large collection of texts [9]. In LSA algorithms, there are mathematical matrices that describe the frequency with which terms occur in the document collection—or the so-called term–document matrix. In the term–document matrix, the rows of the matrix represent a collection of words in the document, and the columns represent the number of documents [10]. LSA is able to understand the lexical features of a document by using the concept of Singular Value Decomposition (SVD); SVD can reveal the document's context, not the synonym of words in the document. SVD transforms the original data by reordering the dimensions and sorting them [11]. Vector space models have been shown to be highly-effective at detecting meaningful semantic entities in text [12] as more generally have methods based on complexity analysis [13]. A term–document matrix built by LSA can be used for a probabilistic model to observe the similarity of the documents. Therefore, the LSA has often been used to build a knowledge-based system over the parallel corpus to find occurrences of plagiarism.

The matrix of word associations or term document matrix can be used to create a probabilistic model for rating the likelihood of an observed term, but the dimensionality of the matrix can become so high that rapid searches using the entire matrix can become excessively time-consuming and computationally-demanding. SVD mitigates this problem by reducing the dimensionality of the term document matrix, effectively compressing all the relevant information into a far more compact and wieldy form. Within this smaller framework, semantically-meaningful clusters of association become clearer (in machine learning terms, the signal to noise ratio is increased) [14].

The first step in running the LSA method is to represent the text as a matrix with words as rows and document (or other contexts) as columns. The matrix formed is called the term–document matrix [15]. Documents in the term–document matrix can be defined independently. They can be selected using a fixed number of words, such as 5, 10, or 20, or defined as a non-fixed-sized document bounded only by a sentence or ended with a full-stop [16]. Each cell contains the number of occurrences of a word on a row from the particular document on the corresponding column. An overview of the term–document matrix is shown in Figure 1.

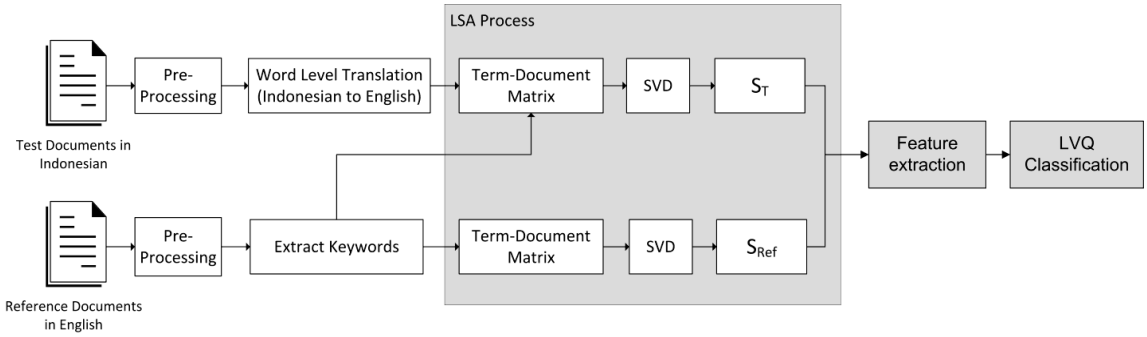


Figure 1. Block diagram of the proposed automated cross-language plagiarism detection system.

The next step of LSA is performing SVD of the matrix for factor analysis. SVD decomposes the matrix into three matrices. One of these matrices is a diagonal matrix containing the scale value that multiplies the row matrix (first matrix) and column matrix (third matrix) back into the initial matrix whose dimensions are equal to the dimensions of the matrix. However, the diagonal matrix has values which provide interesting significance compared to the original matrix [9].

The original method of LSA compares documents stored in the columns of the matrix using Spearman's rho correlation coefficient (ϱ), with a range of $-1 \leq \varrho \leq 1$. The correlation between two words in the documents can be checked to see whether they are relevant (positive correlation) or irrelevant (negative correlation). A detailed explanation of the original LSA is included in [10].

2.2. Learning Vector Quantization (LVQ)

Learning Vector Quantization (LVQ) is a supervised classification method in which each unit of output represents a predefined class (prototype) [17]. LVQ, one of the neural network algorithms, is an algorithm that adapts the results of biological studies; it indicates that different types of input sensors will be mapped to the appropriate groups on their respective areas in the cerebral cortex, which are referred to as topographic maps. LVQ also adapts the Kohonen network. LVQ has initialization, competition, cooperation, and adaptation components.

LVQ's training process modifies a portion of the Self Organizing Map (SOM). This algorithm is used to create a separate representation of each output neuron. This means that any input that is expected to go into the same group must have common features that can be grouped to the right output neuron. The goal of LVQ is to improve the performance (recognition rate) of SOM by using target data to supervise the process while keeping the main characteristic of SOM, which is its high speed [17].

LVQ is also based on competitive learning, where there is only one winner neuron that will be "activated" each time. The difference between LVQ and SOM is that LVQ checks whether the winning neuron is the same as the expected target. The suitability of a winning neuron to the targetted output determines what kind of update the weights are going to have—to make the input closer to the output if the winning neuron is the same as the expected output, or to make the input farther away from the output if the winning neuron is wrong. The determination of the winning neuron is done by finding the Euclidean distance from the input to each neuron. The neuron that has the shortest distance to the input will be treated as the winning neuron.

Determining plagiarism using LVQ requires a training phase. Training is done to get the correct weight values for the corresponding neural network so that the program can recognize each input and group it correctly. Training is done only once, and it uses all the results from the plagiarism detection system. This is different from the normal distribution method in which the normal distribution method is executed once for each paragraph.

In the training phase, learning vector quantization will initialize the weights by taking the input value, assuming that the weight corresponding to the input will have the distance of zero. That way, the initialized weights are already close to their ideal values. However, since weight initialization only uses a portion of the input, the weights must be trained to be able to adjust to other inputs. Weight adjustment is done by finding the nearest Euclidean distance from the input to each neuron using the formula described in Equation (1).

$$D_j = \sqrt{\sum_{k=1}^{m} \left(w_{kj} - x_{ik} \right)^2} \tag{1}$$

An index of the neuron that has the nearest Euclidean distance to the corresponding input is then checked to see if that neuron is the legitimate winner. If it is, the weights are modified to decrease the distance between the corresponding input, and the neuron is then represented by the index. If not, the weights are then modified to increase the distance between the corresponding input and the neuron

represented by the index [18]. The weight modification follows (2) if the minimum distance D_j matches the class target; when it does not match, (3) is applied for weight modification.

$$W_{ji}(new) = W_{ji}(old) - \alpha\left(x_i - W_{ji}(old)\right) \quad (2)$$

$$W_{ji}(new) = W_{ji}(old) + \alpha\left(x_i - W_{ji}(old)\right) \quad (3)$$

3. Methodology of the Automatic Cross-Language Plagiarism Detection System

Figure 1 shows a block diagram of the plagiarism detection system for two different languages. As shown in the diagram, the test document and the reference document will be first pre-processed. At the pre-processing stage, words are separated from their sentences. Then the test document (which is written in Indonesian) is translated into English word by word, ignoring the sentence's structure and grammar. After all the words are translated, a term–document matrix is formed and processed using SVD. The result of SVD for the test document is a singular matrix S_T; for the reference document, the result of SVD is a singular matrix S_{Ref}. After both singular matrices are generated, the feature extraction is conducted; as the last stage, LVQ is used as the classifier. A more detailed process is explained below.

3.1. Pre-Processing

At first, punctuations are removed; then, sentences are broken down into words (array of words), and stop words are removed. Stop words (i.e.,: prepositions, auxiliaries, etc.) must be cleaned out because they pose no important semantic meaning [8]. The frequent occurrence of stop words will disrupt the LSA algorithm and causes the algorithm became biased in determining the semantic context of the corresponding document. The result of this stage is one array for each document, where the elements are the words.

3.2. Word Level Translation for the Test Paragraph

The translation process from Indonesian to English is conducted for each of the array's elements, and it is performed word by word by using a dictionary database. The dictionary database is stored in our own server, and it does not use an online or commercial database. The translation process is only applied to the test document, which is written in Indonesian; this step is not performed for the reference document, which is already written in English.

3.3. Extracting Keywords from Reference Paragraph

Keywords are the important words that should be noticed; in this system, keywords are taken from the reference document only. The keywords are useful in determining the term–document matrix.

3.4. LSA Process

This step is the heart of the plagiarism detection system. In the first stage, each test and reference paragraph is separately tranformed into a term–document matrix. A term–document matrix is a mathematical matrix that describes the frequency of the occurrence of keywords (terms) in the documents. In the term–document matrix, the row is a collection of words, and the column is the frequency that they appear in documents. The definition of document in the term–document matrix can be different. They can be defined as a non-fixed size document bounded only in a sentence or ended by a full-stop [16]; however, in this paper, we used the fixed size document, and observe its behavior over variations in sizes (number of words) every 5 words from 5 up to 25.

Each term–document matrix from the test document and the reference document is processed separately using SVD, and the results of this process are S_T vector for the test document and S_{Ref} vector for the reference document, which will be processed further in the feature extraction process.

Furthermore, the S_T and S_{Ref} vectors are processed to form feature vectors so that they can be classified using a classifier.

3.5. Feature Extraction

This work compares three types of features, which are the ratio of the Frobenius norm on both vectors; the angle between the vectors using slice method; and the angle between the vectors using pad method as depicted in Figure 2. An illustration of the slice and pad procedures are as follows: Assume that there are two diagonal matrix SVD results, both of which will be compared (i.e., matrix A and matrix B), where matrix A is three-dimensional and matrix B is two-dimensional.

$$A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \; B = \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}$$

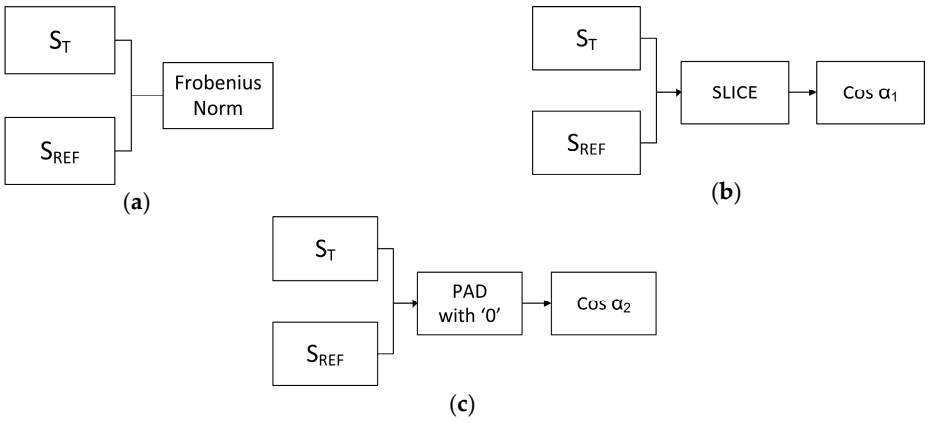


Figure 2. Features for classification: (**a**) the Frobenius norm; (**b**) angle α_1, between vectors after slicing the longer vector and (**c**) angle α_2, between vectors after padding '0' over the shorter vector.

Shown in Figure 3, the angle between the two vectors cannot be calculated due because they are in difference dimensional spaces. Therefore, the dimension of there two vectors must be aligned. The slice procedure projects the higher dimension vector to the lower one. As a result, matrix A becomes

$$A' = \begin{pmatrix} 3 & 0 \\ 0 & 4 \end{pmatrix}, \; B = \begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}$$

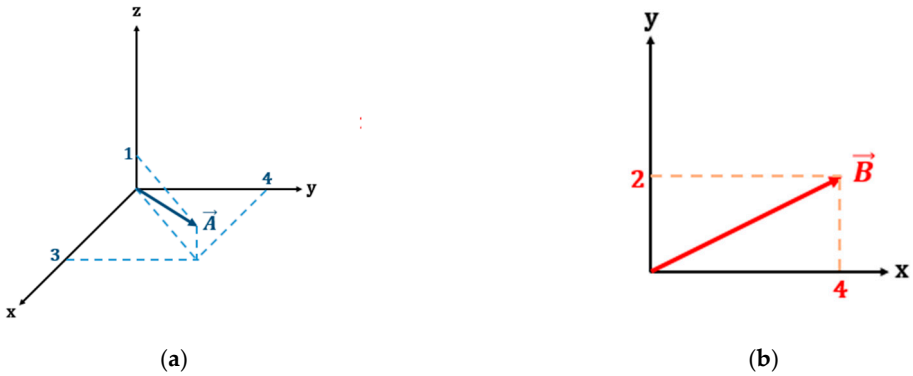
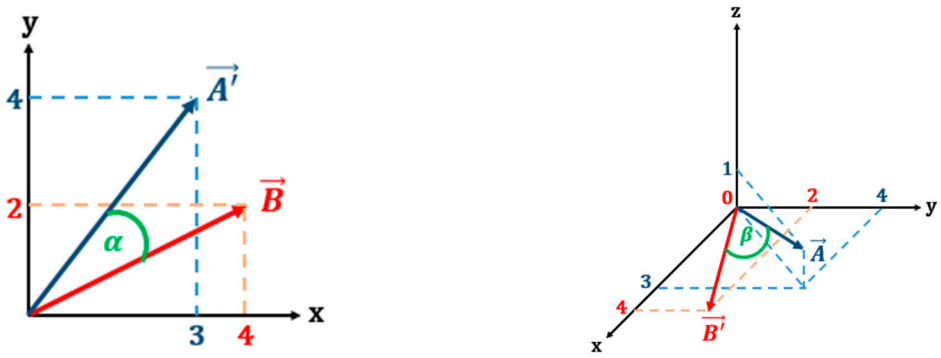


Figure 3. Vector of (**a**) diagonal A, and (**b**) diagonal B.

The pad procedure transforms the lower dimension vector into the higher one of course with zero value as the result of its projection into the new axis. Hence, matrix B becomes

$$A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{pmatrix}, B' = \begin{pmatrix} 4 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

After both matrices lie in the same space, the angle between the two diagonal vectors can be calculated. The angles resulting from slice and pad processes are illustrated by Figure 4.



(**a**) (**b**)

Figure 4. Illustration of (**a**) Slice, and (**b**) pad.

Angle calculation between two vectors can be easily calculated using dot product of two vectors, taken from (4)

$$\cos(A, B) = \frac{\sum_{i=0}^{n} A_i B_i}{|A||B|} \quad (4)$$

For example, according to (4), the cosine of the slice angle between vector A and B in Figure 4 is 0.8945, while the cosine of pad angle between them is 0.8771.

3.6. Classification Using LVQ

Weight modification is done using the variable alpha, which is lessened as the training continues, in order to increase the precision of the weight modifications. The weight modification will continue until the stopping criteria are fulfilled. The stopping criteria can be either the number of repetitions (epoch) in the training process or after the alpha drops below a certain number. The update for the variable alpha done for each training iteration follows (5); because the value of $n < 1$, the alpha lessens after each update.

$$\alpha(t+1) = n\alpha(t) \quad (5)$$

In the testing phase, LVQ uses the weights obtained in the training phase. The process uses the same algorithm as in the training phase; however, the testing phase stops at the determination of the winning neurons (neurons that have the shortest Euclidean distance with a particular input). The output of this testing phase is the pair of its input with the output neuron (namely, whether there is plagiarism or not). The pseudocode of the LVQ classifier is described in Algorithm 1.

Algorithm 1 Learning Vector Quantization Algorithm

Training Phase:
Initialize weights w_{ij}
Define: epoch_max, α_min
If ((epoch < epoch_max) AND (α > α_min))
  For each vector (x_i) dimension k = 1:K
    For each j dimensional output neuron,
      Calculate *Euclidean distance* between input x_i to the w_{ij} based on (1)
    end
    Determine index *j* that gives the minimum distance D_j
    If (x_i = y_i)
      Update weight according to (2)
     else
      Update weight according to (3)
     end
     Modify the pace of learning based on (5)
  end
end
Training Phase:
For each vector data (x_i) dimension k = 1:K
  For each j dimensional output neuron,
    Calculate *Euclidean distance* between input x_i to the w_{ij} based on (1)
  end
  Determine index *j* that gives the minimum distance D_j
  Output: class of data x_i is *j*
end

4. Experiments and Results

The experiments were carried out with 10 scientific papers written in English, which were used to evaluate the proposed methodology. From these 10 papers, there were a total 184 paragraphs extracted. The original English paragraphs were treated as the "reference documents". All documents were manually translated into Indonesian using a human translator. The translated documents were used as "test documents". For each paper, the comparisons were conducted between all pair combinations of paragraphs. For example, paper number 1 consists of 10 paragraphs, paragraph number 1 (translated into Indonesian) was compared to all 10 paragraphs in English. Thus, there should be one pair that indicates plagiarism and nine pairs that indicate no plagiarism; this is illustrated in Figure 5. Since the papers are taken from different scientific ares, it doest not make sense to perform comparison on different papers. Comparisons are performed only over paragraphs within the same paper. Thus, from the original 10 papers with 184 paragraphs, there were 4050 pairs of paragraph comparisons; 184 were marked as plagiarism, and 3856 of them were marked as not plagiarism.

The plagiarism detection process was conducted using the proposed cross-language detection system illustrated in Figure 1. In this system, the inputs in the form of test documents and reference documents were pre-processed, and the test documents were translated using a dictionary database. Then, the LSA was performed, and the features extracted were the Frobenius norm, the angle of two vectors with slice, and the angle of two vectors with pad. The classification was conducted using LVQ.

Table 1 provides an example of three feature values extracted after the LSA process, namely Fnorm, α_slice, and α_pad for test documents 3001 and 3002 compared to reference documents 3001–3002.

The evaluation of the proposed cross-language automated plagiarism detection system was conducted by a comparison with the manual/human evaluation, which was presented in three values of accuracy. These three values were precision, recall, and F-measure. They were then analyzed and used to draw conclusions.

Table 1. Example of the feature value extracted after the LSA process for test documents no #3001 and no #3002.

test_doc	ref_doc	Fnorm	α_Slice	α_Pad	test_doc	ref_doc	Fnorm	α_Slice	α_Pad
3001 *	3001	60.6977	5.95616	5.95616	3002	3001	53.8028	21.0574	21.0574
3001	3002	35.3553	47.6318	47.6318	3002 *	3002	55.9017	24.8395	24.8395
3001	3003	0	90	90	3002	3003	0	90	90
3001	3004	0	90	90	3002	3004	32.1634	35.0656	35.0656
3001	3005	28.4747	7.26953	7.26953	3002	3005	0	90	90
3001	3006	0	90	90	3002	3006	0	90	90
3001	3007	0	90	90	3002	3007	0	90	90
3001	3008	0	90	90	3002	3008	0	90	90
3001	3009	14.2857	58.706	58.706	3002	3009	0	90	90

* supposed to be classified as plagiarism because the test_doc is equal to ref_doc.

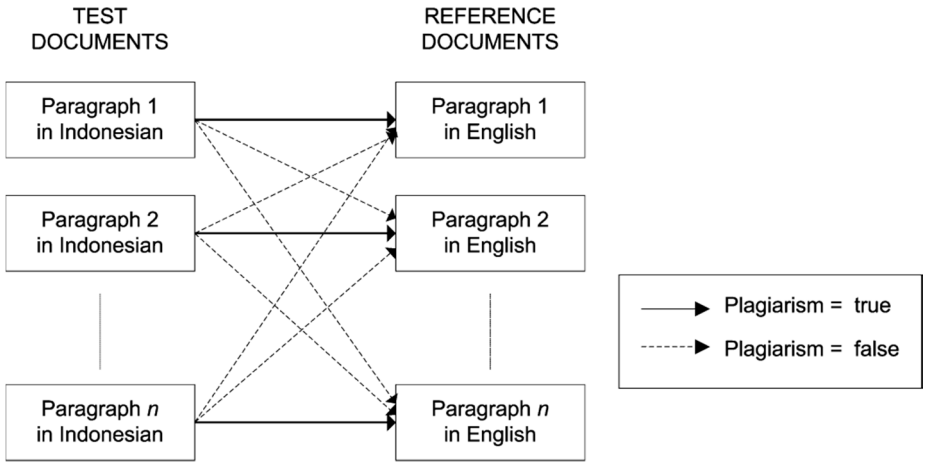


Figure 5. Testing scenario for each paper for the system developed.

4.1. Comparison of Various Learning Rate Parameters for LVQ

In accordance with the theory in LVQ for classification, the value of alpha and n both have the same function, which is to speed up the learning process; the experiment over the variation of these two variables is conducted together with the same sequence respectively. By using the default values, which are $\alpha = 1$ and $n = 5$, the variation is applied by multiplying both parameters with values of 10^{-1}, 10^{-2} and 10^{-3}. Thus, there are three variations in the experiment, which are: (1) $\alpha = 0.1$ and $n = 0.5$, (2) $\alpha = 0.01$ and $n = 0.05$ and (3) $\alpha = 0.001$ and $n = 0.005$. The result of this experiment is presented in Table 2 and the corresponding graph in Figure 6.

Table 2. Variation results of the alpha and n values in the evaluation of the plagiarism detection system using the learning vector quantization algorithm.

Multiplication Factor	Precision	Recall	F-Measure
10^{-1}	0.49833	0.76804	0.60446
10^{-2}	0.45482	0.77835	0.57414
10^{-3}	0.26725	0.93814	0.416

Based on Table 2 (the results of which are depicted in the graph in Figure 6), it can be concluded that the values of alpha and n are directly proportional to precision (increasing alpha and n means increasing precision), but the values of alpha and n are inversely proportional to recall (increasing alpha and n means decreasing recall). F-measure increases with the increase of alpha and n. Figure 6

also shows that a higher alpha value means fewer false positives (due to fewer positives) but also fewer true positives (due to more positives being judged as negative). The analysis related to the variation results of variable alpha and n can be explained as follows: increasing the values of alpha and n will increase the iterations needed for alpha to surpass the limit value to stop the learning process. More iterations mean more corrections to weights in the algorithm. Moreover, the data used for training are from all 10 papers, with many more "not/negative" values than "plagiarism/positive" (194 for positives and 3856 for false negatives; yield ratio of 1:19.876), causing the algorithm to learn to recognize negative results (not indicated as plagiarism) rather than positive results (indicated as plagiarism). For comparison, if the values of alpha and n are, respectively, 0.1 and 0.5, the training iteration will take place 24 times; whereas if alpha and n are, respectively, 0.01 and 0.05, the training iteration only occurs five times. For alpha and n with values of 0.001 and 0.005, the training iteration occurs only three times. In addition, according to Equations (2) and (3), small values of alpha and n means that there is not much difference in weight after each correction. Hence, the training results in weights that are similar to the initial weight, which came from the first and second entries of the input. There are not enough data to learn whether a text should be labelled as 'not/negative' (not indicated as plagiarism). This explains why, for example, smaller values for alpha and n mean fewer negative values and low score precision (too many false positives).

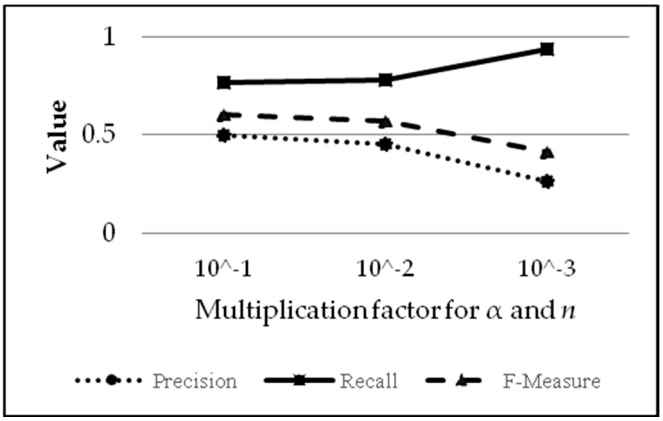


Figure 6. Effect of variation in the alpha and n values at information retrieval measurement values from the evaluation of a plagiarism detection system using the learning vector quantization algorithm.

4.2. Comparison of the Feature Types of the LSA

To find the best feature type after the LSA process, this study focused on comparing three types of feature values, namely the Frobenius norm (f), Cos α with slice (s), Cos α with pad (p), and a combination of them. The results of this experiment are shown in Table 3.

Table 3. Results of various features using the Frobenius norm (f), Cos α with slice (s), and Cos α with pad (p).

Feature *	Precision	Recall	F-Measure
Frobenius norm (f)	0.381253	0.654639	0.465479
Cos α with slice (s)	0.280162	0.331615	0.123266
Cos α with pad (p)	0.158213	0.32646	0.142965
fsp	0.4068	0.828179	0.531536
fs	0.406962	0.797251	0.526073
fp	0.408638	0.792096	0.528116
sp	0.172015	0.333333	0.141452

* f = Frobenius norm, s = slice, p = pad.

Table 3 shows the test results using various combinations of the Frobenius norm (f), Cos α with slice (s), and Cos α with pad (p) as the feature. The combination of the feature value was accomplished using the AND operation. From this experiment, it can be seen that the combination of two or more feature values is better than using a single feature value. Moreover, the combination of all three feature values provides the highest results of precision, recall and F-measure. The combination of the Frobenius norm with Cos α and with either slice or pad also provides a comparable result. In contrast, using the combination of Cos α with slice and Cos α with pad provides a very low accuracy. This result implies that both the Frobenius norm value and Cos α are important features in the LSA-based, cross-language automated plagiarism detection system; however, using a single feature will significantly reduce the accuracy.

4.3. Comparison of the Term–Document Matrix Definition

In this work, we investigate the optimum number of term–document matrix definitions, which is the optimum number of words to be defined in one document to allow deterministic resource allocation to be applied onto the algorithm for the sake of processing performance. A document can be defined as a sentence, a paragraph, or a fixed number of words in a sequence. In the developed system, we defined the term–document matrix based on a fixed number of words in a sequence; in this experiment, we varied the number of words to 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, and 25 words. The results of this experiment can be seen in Figures 7 and 8.

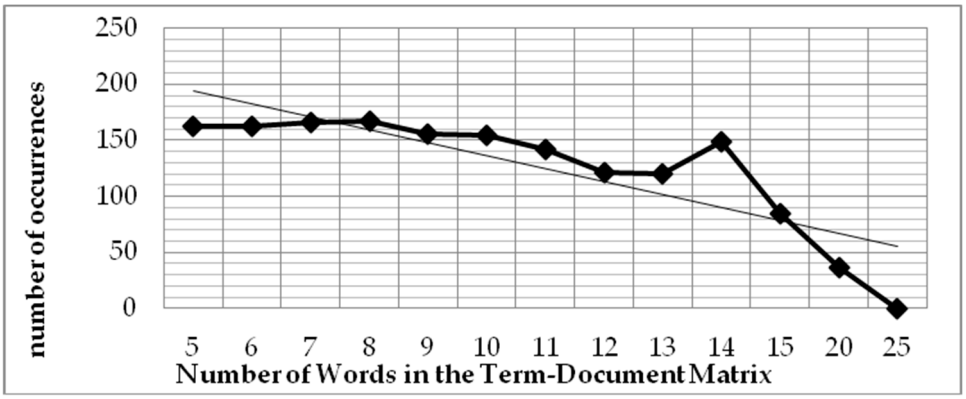


Figure 7. The number of true positive results using Learning Vector Quantization.

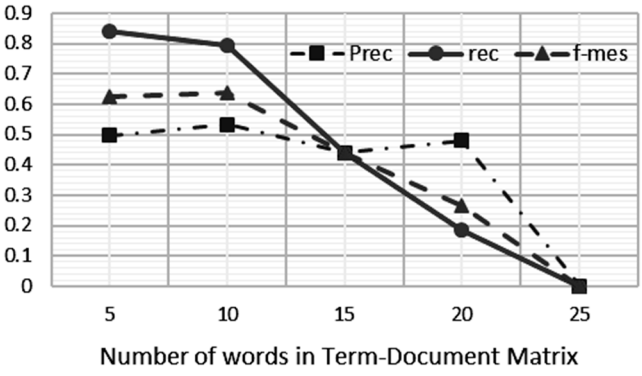


Figure 8. Precision, recall and F-measure results over various term–document matrix definitions.

Figure 7 shows the impact of decreasing the values of precision, recall and F-measure on increasing the number of words used in the term–document matrix. According to the human rater's assessment, the number of words in the document that should be declared plagiarism is 192 out of 4050 words.

This means that there are 3858 words that are not considered plagiarism. The system became less effective in detecting plagiarism when the term–document matrix increased. A bigger term–document matrix size will suppress the number of false positives and false negatives, but the weight of the harmonic precision and recall is reduced. Thus, the LSA precision and relevance of LSA as well as the accuracy of the detection tend to be worse when the size is larger.

4.4. Comparison of Frequency and Binary Occurrence in the Term–Document Matrix Definition

All of the previous experiments were performed using the binary method. The same set of samples were also used to perform experiments using the frequency method. The difference between them lies in the values used in the term–document matrix formation. The binary method uses the values of 1 and 0 to indicate the existence or absence of words in the document, while the frequency method uses the word count of the document. The results of term–document matrix using the binary method are depicted in Table 4; using the frequency method is depicted in Table 5.

Table 4. Results of LSA output processing with the binary method.

Size	**5**	**10**	**15**	**20**	**25**
True Neg	3691	3775	3753	3774	3801
True Pos	10	88	3	1	4
TRUE	3701	3863	3756	3775	3805
False Neg	184	106	191	193	190
False Pos	165	81	103	82	55
FALSE	349	187	294	275	245
Precision	0.057143	0.52071	0.028302	0.012048	0.067797
Recall	0.051546	0.453608	0.015464	0.005155	0.020619
F-Measure	0.054201	0.484848	0.02	0.00722	0.031621

Table 5. Results of LSA output processing with the frequency method.

Size	**5**	**10**	**15**	**20**	**25**
True Neg	3792	3797	3801	3808	3810
True Pos	103	96	58	36	25
TRUE	3895	3893	3859	3844	3835
False Neg	91	97	136	158	169
False Pos	64	60	55	48	46
FALSE	155	157	191	206	215
Precision	0.616766	0.615385	0.513274	0.428571	0.352113
Recall	0.530928	0.497409	0.298969	0.185567	0.128866
F-Measure	0.570637	0.550143	0.37785	0.258993	0.188679

Compared with the binary occurrence in the term–document matrix definition, the frequency-based occurrence improved the accuracy by about 10% for precision, recall, and F-measure. The true positive value (88) in binary was slightly lower than the frequency method (103). Nevertheless, the most beneficiary fact in favoring the frequency method is that all the false detection values in binary method are much greater than the frequency method. Since the binary method only detects the presence or absence of the word in a paragraph, it is more prone to mistakes compared to detecting plagiarism based on the frequency of words in a paragraph.

The overall comparison of the human rater, binary method, and frequency method can be seen in Figure 9. As expected, there are significant differences between the binary method and the frequency method in all term–document matrix sizes as well as in the best value obtained by both. Moreover, based on Tables 4 and 5, the binary method performance is definitely worse than the frequency

method in precision, recall and F-measure. These results led to the conclusion that representing the term–document matrix in frequency brought a significant improvement to the system's accuracy.

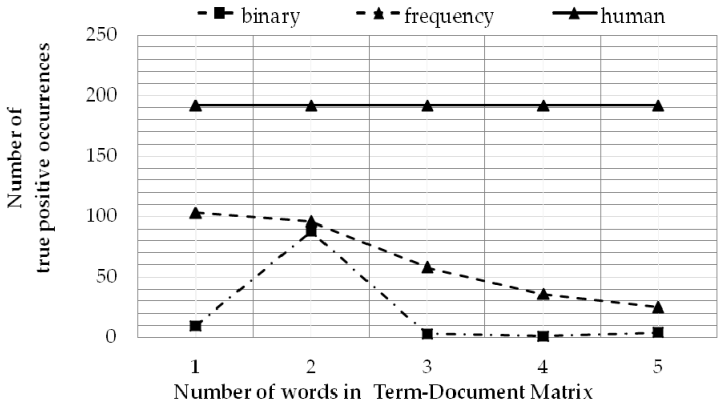


Figure 9. LSA processing results (number of true positive occurrences) of the binary method and the frequency method using the normal distribution.

5. Discussion

One of the greatest drawbacks to this system is the lack of a good translation dictionary; thus, a paragraph that should be judged as plagiarism is perceived differently after the translation process is executed. In order to improve translation quality, a better translation dictionary is needed—for example, using a licensed or paid Application Programming Interface (API), which has better standards and is able to perform a better translation process.

Development of the system can also be done by doing double evaluation. Double evaluation involves evaluating the tested paragraph compared to the reference paragraph and evaluating it again by rendering the reference paragraph as a tested paragraph and evaluating the tested paragraph as a reference paragraph. The indication of plagiarism can be determined if and only if both the results are indicated to be a plagiarism occurrence. It has been shown that this method increases the accuracy significantly (up to 85% for F-measure). However, to be able to do the double evaluation, an excellent translation dictionary is needed; this is because it is also necessary to translate the reference paragraph to Indonesian and the tested paragraph to English.

One possible further development is to more extensively examine the statistical properties of the data, providing the possibility of testing with other statistical methods. The development can be done by examining the data returned from the result of the plagiarism detection system—for example, by using the Jarque–Bera test, which tests the skewness and kurtosis of the data.

6. Conclusions

In conclusion, a decrease in the values of alpha and n positively impacts the relevance (recall) of the system, but it negatively impacts the precision of the system. Overall, the decline in the values of alpha and n negatively impacts the accuracy of the program plagiarism detection system, which is based on the LVQ algorithm. The LSA method can be used to assess similarities between two texts by comparing the length and calculating the angle between both matrices that result from the LSA operation. Both the Frobenius norm and angle (Cos α) are needed as the detection features after the LSA process. By using an appropriate combination of methods and parameters, the system achived the accuracy up to 65.98%. The increase of the term–document matrix size has been shown to result in a negative impact on the accuracy of LSA-based, cross-language plagiarism—even when it involves the Frobenius norm, slice, and pad when using LVQ processing. The best accuracy achieved is 87% with a document size of 6 words, and it reaches 0% (the worst) when it is increased to 25 words.

Results on the term–document matrix construction strategy suggested that utilizing the frequency method (as opposed to the binary method) will improve the detection performance. For further system development, a better and standardized translation dictionary is needed to open other possibilities (such as double evaluation) in developing the system.

Acknowledgments: This research was partly supported by the Ministry of Research and Higher Education of Indonesia under the grant PUPT 2016.

Author Contributions: A.A.P.R. conceived and designed the algorithm and experiments; B.A.A, M.M., and D.J.W. performed the derivation of the algorithms and conduct the experiments; P.D.P., M.S. and F.A.E. analyzed the data and wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gipp, B.; Meuschke, N. Citation Pattern Matching Algorithms for Citation-Based Plagiarism Detection: Greedy Citation Tiling, Citation Chunking and Longest Common Citation Sequence. In Proceedings of the 2011 ACM Symposium on Document Engineering, Mountain View, CA, USA, 19–22 September 2011.
2. Lancaster, T. Effective and Efficient Plagiarism Detection. Ph.D. Thesis, South Bank University, London, UK, 2003.
3. Kakkonen, T.; Mozgovoy, M. Hermetic and Web Plagiarism Detection Systems for Student Essays—An Evaluation of the State-of-the-Art. *J. Educ. Comput.* **2011**, *42*, 135–159. [CrossRef]
4. Maurer, H.; Kappe, F.; Zaka, B. Plagiarism-A Survey. *J. Univers. Comput. Sci.* **2006**, *12*, 1050.
5. Bretag, T.; Saadia, M. Self-Plagiarism or Appropriate Textual Re-Use? *J. Acad. Eth.* **2009**, *7*, 193. [CrossRef]
6. Errami, M.; Sun, Z.; Long, T.C.; George, A.C.; Garner, H.R. Deja vu: A Database of Highly Similar Citations in the Scientific Literature. *Nucleic Acids Res.* **2009**, *37*, D921–D924. [CrossRef] [PubMed]
7. Ali, A.; Abdulla, H.; Snasel, V. Overview and Comparison of Plagiarism Detection Tools. In Proceedings of the Dateso 2011: Annual International Workshop on DAtabases, TExts, Specifications and Objects, Pisek, Czech Republic, 20 April 2011.
8. Olson, D.; Delen, D. *Advanced Data Mining Techniques*; Springer: Berlin, Germany, 2008; p. 138.
9. Landauer, T.K.; Foltz, P.W.; Laham, D. An Introduction to Latent Semantic Analysis. *Discourse Processes* **1998**, *25*, 259–284. [CrossRef]
10. Dumais, S. Latent Semantic Analysis. *Annu. Rev. Inf. Sci. Technol.* **2004**, *38*, 188–230. [CrossRef]
11. Britt, M.A.; Wiemer-Hastings, P.; Larson, A.A.; Perfetti, C.A. Using Intelligent Feedback to Improve Sourcing and Integration in Students' Essays. *Int. J. Artif. Intell. Educ.* **2004**, *14*, 359–374.
12. Seaward, L.; Matwin, S. Intrinsic Plagiarism Detection Using Complexity Analysis. In Proceedings of the 25th Annual Conference of the Spanish Society for Natural Language Processing, SEPLN 2009, San Sebastian, Spain, 8–10 September 2009; pp. 56–61.
13. Zechner, M.; Muhr, R.; Kern, M.; Granitzer, M. External and Intrinsic Plagiarism Detection Using Vector Space Models. In Proceedings of the 3rd PAN Workshop. Uncoveringplagiarism, Authorship And Social Software Misuse, San Sebastian, Spain, 8–10 September 2009; pp. 47–55.
14. Alsallal, M.; Iqbal, R.; Amin, S.; James, A. Intrinsic Plagiarism Detection Using Latent Semantic Indexing and Stylometry. In Proceedings of the 2013 Sixth International Conference on Developments in eSystems Engineering (DeSE), Abu Dhabi, UA, 16–18 December 2013; pp. 145–150.
15. Asuncion, G.-P.; Jerome, E. *The Semantic Web: Research and Application*; Springer: Berlin, Germany, 2005.
16. Lappin, S.; Fox, C. *The Handbook of Contemporary Semantic Theory*; John Wiley & Sons: Hoboken, NJ, USA, 2015.
17. Kohonen, T. The Self-Organizing Map. *Proc. IEEE* **1990**, *78*, 1464–1480. [CrossRef]
18. Soleman, S.; Purwarianti, A. Experiments on the Indonesian plagiarism detection using latent semantic analysis. Proceedings of 2014 2nd International Conference on Information and Communication Technology (ICoICT), Bandung, Indonesia, 28–30 May 2014.

© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).