Significant Word-based Text Alignment for Text Reuse Detection

Lucia D. Krisnawati, Klaus U. Schulz

***Abstract*—**One of challenges in text reuse detection is how to detect the source-reused passage pairs which have a wide range of similarity degree by one method or a single alignment algorithm only. However, the academic texts are rich with terminologies which are hardly altered when one reuses the existing texts in his/her writings. In this paper, we introduce and analyze the use of significant words filtered through word local weighting from the field of text summarization to align source-reused passages. We base our alignment process on the paragraph segmentation which is filtered by the use of their weighted and binary vectors. We demonstrate that the proposed text alignment method is capable of detecting the source-reused passage pairs which are obfuscated by means of literal copy, copy and shake, and paraphrases from light, medium to heavy levels.

***Keywords*—**Text alignment, text reuse, plagiarism detection, seeds.

I. Introduction

Text reuse is defined as the reuse of existing written sources in the creation of a new text [1]. Basically, there are 2 types of text reuse: the acceptable and unacceptable ones. The acceptance of a text reuse is determined by 3 factors which comprise the attribution to its sources or former authors, the length and portion of a reused text, and the text genre. These factors are intertwined closely. For example, a reuse of a news in different newspapers is acceptable in spite of the lack of attribution to its sources and its highly reused portion. In academic writings, such text reuse arises the accusation of *plagiarism*, which is a form of unaccepted text reuse.

Both types of text reuse are reproduced by obfuscating the original ones. The obfuscation techniques determine the degree of similarity between the source and reused texts. The literal copy and slight modification result in the almost identical texts, while smartly-done paraphrase and summary produce either topically or semantically related texts.

Finding nearly identical documents becomes the task of near-duplicate detection [2], while matching topically related documents with queries is the subject area of the information retrieval (IR). For this reason, the task of text reuse detection is assumed to lie between the tasks of IR and near-duplicate detection [3]. However, we argued that the task of text reuse detection is to find out texts whose similarity degree spans from semantically similar texts to nearly identical ones. This depends on the obfuscation types and the reuse portion. Figure 1 describes the task of text reuse detection on the similarity spectrum..

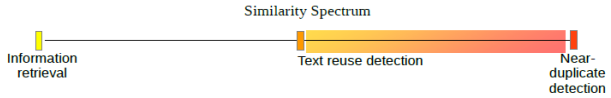

Fig. 1 Similarity degree of text reuse compared with Information Retrieval and Near-duplicate detection

The challenges of text reuse detection are manifested into 2 subtasks: retrieving a small set of documents which are likely the sources of reuse, and extracting source-reused passages [4], [5] which have a wide range of similarity degree. In this work, we are interested in the source-reused passage extraction by means of text alignment and focused our study on solving the problems of aligning Indonesian texts. Due to the lack of standard and publicly-available corpus for assessing text reuse detection systems in Indonesian, this study aims at providing such evaluation corpus.

II. Related Works

Given a suspicious document, text Alignment task is to analyze further a set of source candidate documents, which are retrieved by the source retrieval subtask. The majority approaches of text alignment adopt the building blocks proposed in [6] which comprise of seeding, seed extension, and filtering. Seeding refers to 'matches' between a suspicious document containing the reused passages, d_{plg}, and a source document, $d_{src} \in D_{src}$ using seed heuristics [6], [7].

In general, **seeds are generated** to match either the content, structure, or style of text pairs. In *content-based matching*, the seed heuristics could be n-grams [8], [9], word k-skip n-grams [10], sentences [9], or fingerprints [11]. Seeds commonly used for matching *structural similarity* are stopword n-grams [12], word-pair orders [8], [9], and citation pattern [13], while those for matching *stylistic similarity* could take form of sentence length, token length, and function word frequency [8].

The basic idea of **seed extension** is to present the whole passage rather than some multiple chunks of separate seeds. so far, there are 4 approaches applied i.e. rule-based approaches [12], [9], [11], clustering [14], [10], classification [20], and dynamic programming [14].

Recently, researches on *text reuse detection in Indonesian* are steadily developing. In this study, we did a survey to 16 research reports on Indonesian text reuse detection and found out that the majority of them (68.75%) deal with near-

Lucia D. Krisnawati, IT Dept., Duta Wacana Christian University, Jogjakarta, Indonesia. krisna@staff.ukdw.ac.id
Klaus U. Schulz, CIS, LMU, Munich, Germany, schulz@cis.uni-muenchen.de

duplicate detection. Among 31.75% systems concerning on detecting the text reuse, only a handful of them distinguished their tasks into the source retrieval and analysis subtasks. Besides, fingerprints generated through Rabin-Karp algorithm and selected through winnowing algorithm become the favourite seeds [15], [16]. Other seeds take form of tokens [17], or phrases [18]. The comparison or seed extension approaches could be grouped into rule-based comparison [16]-[18], and classification [15].

III. The Proposed Methods

The building blocks of our text alignment comprise of text segmentation, seed generation, paragraph similarity, seed processing, and filtering which adapts the ones in [7].

A. Text Preprocessing and Segmentation

Text normalization was done by eliminating non-readable characters, numbers, case folding, and normalizing white spaces. A single paragraph break symbol was used to segment a document into paragraphs. The short segments will be merged into their successive paragraphs. In preprocessing, we applied 2 types of stopwords: the frequency-based and semantic-based ones. A semantic stop list takes account of words which semantically have little value for retrieval process [19]. For stemming, we made use of IDNStemmer which is a variant of Porter Stemmer for Indonesian[1].

B. Seed Generation

A paragraph is a collection of sentences having a single theme which is expressed through several keywords. We assumed that in academic text reuse, these keywords are rarely altered but their surrounding words are highly to be the objects of alteration. Based on this assumption, this study borrowed the local word scoring proposed in [21] to select paragraph keywords.

We modified the locality of word local scored into a paragraph and used 2 statistical criteria: the relative term frequency (TF) and a paragraph count (ParCount) which refers to the number of paragraphs containing the significant word normalized by the total number of paragraphs in a text. The computation of TF is also adapted to a paragraph TF which is normalized by the total number of words in that particular paragraph. The adapted word local score (WlScore) is then defined as in (1).

$$WlScore = \alpha * TF + (1 - \alpha) * ParCount \quad (1)$$

Where α is a parameter weight in the range of (0, 1).

The significant words were obtained by removing the 'insignificant' ones through a word local score threshold [21] which is defined in (2).

$$WlScoreThreshold = \frac{\sum_i WlScore(i)}{\#\ text\ words} * PF \quad (2)$$

where *i* represents the word index, and PF stands for Pruning Factor. PF could be defined to decide how many percentage of words will be used as paragraph seeds. By increasing PF, less words will be selected which is good at matching heavily obfuscated paragraphs. The seeds generated from (1) and (2) are then indexed. Figure 2 illustrates the seed index.

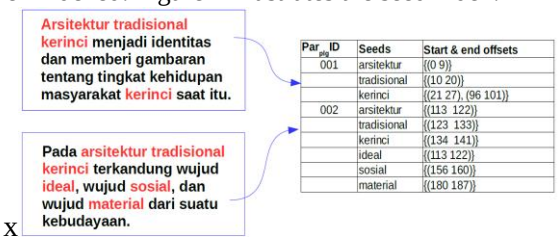

Fig. 2 An illustration on seed index for two short segments representing two paragraph segments.

C. Paragraph Similarity

In this study, the seeds are aimed to serve dual functions, i.e. as paragraph queries and as a heuristic match for a source-reused passage pair having a wide range of similarity degree.

Using seeds as queries, the similarity between each paragraph of a d_{plg} and d_{src} could be measured. For similarity measures, we came up with applying Dice and Jaccard coefficients. In our setting, Dice coefficient was implemented to capture the source-reused passage pairs modified through paraphrase and summary which needs only a handful of queries. The Dice coefficient borrowed from [22] could be seen in (3).

$$S_{Dice} = \frac{2 \sum_{i=1}^{d} P_i Q_i}{\sum_{i=1}^{d} P_i^2 \div \sum_{i=1}^{d} Q_i^2} \quad (3)$$

where P_i refers to a candidate paragraph vector, and Q_i represents the paragraph query vector.

The second similarity metric is aimed to capture as many similar terms as possible to anticipate paragraph reuses with obfuscation types of *copy and paste*, or a slight modification. The simple but famous *Jaccard coefficient* was used to serve this purpose. Only pairs of paragraphs whose scores are above 0.35 for Jaccard **and** 0.4 for Dice would be processed further.

IV. Seed Processing

The seed processing comprises of seed matching, seed merging, and seed extension. The seed matching was carried out by looking up the seed index. Given pairs of paragraph IDs, the seed matching function derives the seeds of suspicious paragraph to match terms of source paragraphs. Whenever a match occurs, the start and end offsets of matched seeds will be saved into a matched seed table.

A. Seed Merging

The computation of seed merging and extension is performed by looking up the seed tables and verifying the defined rules and parameters. The rules and parameters setup were based on the following considerations:

1) **Giving space for any text modification** which might be done by any obfuscation techniques
2) **Defining a gap between seeds** whose length should not be longer than a length of a short paragraph.

[1] The IDNStemmer was written by A.F. Wicaksono and B. Muhammad.

3) **Avoiding seed repetition** within a paragraph which indicates the absence of text reuse.

Based on this considerations, seed merging was performed in a two-step merging process. In the first step, the neighboring seeds whose distance is less than the gap parameter, α, are merged. α is defined to be 35 characters in d_{plg} and 50 in d_{src}. In this step, the defined α value produced short sequences of seeds. This is intentionally done as a longer gap will result in greedy seed merging. On the second step, we used 2 parameters to remerge the seeds. The sequence length (len) and the gap, β. This time, the β was set to be 75 characters and len is equal to 35.

B. Seed Extension

If seed merging joins the seeds within a paragraph scope, the seed extension merges the merged sequence pairs beyond the paragraph boundaries. The seed extension algorithm is based on the relations defined in [11] which identify 4 relation categories as follows:

1) **Containment** identifies a match within another match. Assuming that we have 2 pairs of merged sequences with $\{(s1, e1, l1) \rightarrow (a1, b1, ln1), (s2, e2, l2) \rightarrow (a2, b2, ln2)\}$ where *s, e, l* stands for start, end offsets, and length of a source sequence, while *a, b, ln* refer to the same things in a suspicious paragraph. The second match pair is said to be within the first if $s2 \geq s1$, $e2 \leq e1$, and $l1 \geq l2$ [11].
2) **Overlap** describes the condition where only a part of a match is within another match. 2 pairs of merged sequences are said to be overlapped if $e2 \geq e1 \geq s2 \geq s1$ [11].
3) **Near-disjoint** identifies pairs of matches which share no common offset but the distance between them is within a defined gap threshold (θ), i.e. if $s2 - e2 \leq \theta$.
4) **Far-disjoint** describes two pairs of merged sequences whose distance is beyond the gap threshold.

Theoritically, each relation in a source-merged sequence has a possibillity being aligned to a reused-merged sequence under 4 different relations. Thus, there would be 16 relation possibilities. Due to our paragraph-based merging technique, the extension algorithm extends only consecutive merged sequence pairs with near-disjoint relation for the source sequences to overlapping, near-disjoint, or containment relations in consecutive reused sequences.

C. Filtering

The last task is to filter any detected sequence pairs which are too short, as they often lead to a high false positive detection. For this purpose, a rule-based filtering technique was developed. Based on the observation on our test document corpus, we removed all passages which are less than 125 characters for the source passages aligned with passages which are less than 150 characters in suspicious passages.

V. EVALUATION FRAMEWORK

Evaluating the performance of a text reuse detection system requires 2 things: an evaluation corpus and evaluation concepts. As there has been no publicly available corpora for evaluating text reuse detection system in Indonesian, this study constructed a medium-scale evaluation corpus.

A. Building the Evaluation Corpus

As an effort of standardizing our evaluation corpus, this study did a survey on corpus building strategies to 152 research reports. We found out that there have been 2 institutions which continually evaluate text reuse. These institutions are PAN shared task and HTW research center[2]. Their methods and strategies play an important role in our strategies of evaluation corpus building.

Our evaluation corpus is a collection of source documents and suspicious or test documents. Those texts were acquired either manually or by authomatic web grabbing. The source document corpus take a form of bachelor theses, articles, papers in proceedings and journals, and comprise of 2014 documents.

The test documents were created through two methods as in [23]:

1) **Algorithmic generation** which creates documents by random text operation and semantic word variation. This results in artificial text reuses.
2) In **Simulation**, the test documents were produced by human writers and addressed as simulated text reuses.

The random text operation was performed by deleting, inserting, deleting and inserting words, and shuffling the word orders. In the *insertion process*, the inserted words were taken from an Indonesian root word lexicon[3]. *The semantic word variation* was performed by making use of Wordnet Bahasa[4]. These 2 processes resulted in 128 artificial test documents.

The main goal of creating *simulated text reuses* is to have test documents which emulates the real situation of text reuse. However, many research groups created test documents containing only one obfuscation type per document. This contradicts the real case in which a single suspicious document may contain several reused passages with different types of obfuscation. Therefore, the simulation process in this study was aimed to produce test documents containing various types of obfuscation per document. The simulation involved 37 persons and produced 105 test documents with 4 types of obfuscation: verbatim copy, copy and shake, paraphrase, and summary. Their length varies from 300-1200 words. Besides these cases, our test document corpus is also completed with 10 documents containing no-reused passages.

B. Evaluation Measures

For assessing the performance of the proposed methods, this study made use of evaluation measures proposed in [23], [7], which assess system performance on the character, case, and document levels. In [7], a plagiarism or text reuse case is defined as a quadruple $s = \{s_{plg}, d_{plg}, s_{src}, d_{src}\}$ where s_{plg} is a passage in a d_{plg} which is a reused version of a source passage s_{src} in a d_{src}. $s \in S$ refers to a quadruple set defined in a gold

[2] HTW stands for Hochschule f\"ur Technik und Wirtschaft, Berlin

[3] This lexicon was downloaded from http://stop-words-list-bahasa-indonesia.blogspot.de/2012/09/daftar-kata-dasar-bahasa-indonesia.html

[4] Wordnet Bahasa, Nanyang Technological University (NTU), Singapore and it is available as a free resource in http://wn-msa.sourceforge.net.

label of a given d_{plg}. Correspondingly, $r = \{r_{plg}, d_{plg}, r_{src}, d_{src}\}$ is used to represent a reported detection outputted by the system [23].

1) Character-Level Measures

In this measure, $s \in S$ is used as references to characters of d_{plg} and d_{src} which specify passages s_{plg} and s_{src}, so does $r \in R$. r is said to detect s iff $s \cap r \neq 0$, $r_{plg} \cap s_{plg} \geq 150$ characters, and $r_{src} \cap s_{src} \geq 125$ characters. Based on these sets and restriction, the macro-averaged precision and recall are defined exactly as in [23], and can be seen in (4) and (5).

$$prec_{char}(S,R) = \frac{1}{|R|} \sum_{r \in R} \frac{|\cup_{s \in S} s \cap r|}{|r|} \quad (4)$$

$$rec_{char}(S,R) = \frac{1}{|S|} \sum_{s \in S} \frac{|\cup_{r \in R} s \cap r|}{|s|} \quad (5)$$

where $s \cap r$ equals to an intersection between s and r which refers to the number of similar characters in both sets, if r detects s. Otherwise the intersection value will be zero.

2) Passage-level Measures

One drawback of character-level measures is that it cannot inform from which passage pairs are the detected characters. The passage-level measures are introduced to addressed this drawback We used the same threshold for passage length to be evaluated: 125 characters for r_{src} and 150 characters for r_{plg}.

Let S, and R denote to the same references as mentioned earlier, but s and r refer to a pair of passages instead of a set of passage's characters. Thus, the precision and recall on the passage level are defined as follows:

$$prec_{pass}(S,R) = \frac{|R_S|}{|R|} \qquad rec_{pass}(S,R) = \frac{|S_R|}{|S|} \quad (6)$$

where S_R refers to passages in S which are detected by R, and R_S refers to passage pairs in R which detect S.

3) Document-level Measures

The measures on document level try to assess the detection performance on a wider scale and to see whether each $d_{src} \in D_{src}$ for a given d_{plg} are detected. The minimum requirement for a detection of a source document d_{src} in R to be considered as a true positive detection is that this document contains at least one accurate detection of a source-reused passage pair. Let D_S denotes pairs of source-suspicious documents defined in S, and D_R denotes the detected pairs of source-suspicious documents in R. Based on these sets, the document-level precision and recall are defined as follows:

$$prec_{doc}(S,R) = \frac{|D_S \cap D_R|}{|D_R|} \qquad rec_{doc}(S,R) = \frac{|D_S \cap D_R|}{|D_S|} \quad (7)$$

4) Measures for the Obfuscation Types

The paragraph-level measures gives a general evaluation, i.e. the obfuscation types of the detected passage pairs remain unknown. To address this drawback, we introduced the recognition measure for the obfuscation type, abbreviated into *obtype recognition*.

In order to compute the obtype recognition, a new attribute *obtype* was added to $s \in S$ as an annotation for the obfuscation for that passage pair. As its consequence, s is extended into sextuple: $s = \{srcOfst, srcLen, d_{src}, plgOfst, plgLen, obtype\}$ while r remains to be a quintuple, $r = \{srcOfst, srcLen, d_{src}, plgOfst, plgLen|$. Let $\mathbf{S_C}$ denotes a set of passage pairs having a specific obfuscation type in *S*, and $\mathbf{R_C}$ denotes passage pairs from a specific obfuscation type in *R*, where S and R refer to the same sets used in the former measures. The obtype recognition of a single obfuscation type for the whole test documents is defined as follows:

$$reco_{optape}(S,R) = \frac{\sum_{i=1}^{|D_C|} |S_C \cap R_C|}{\sum_{i=1}^{|D_C|} |S_C|} \quad (8)$$

where D_C refers to the total number of test documents containing one specific obfuscation type, eg. paraphrase or copy.

5) Measures for no-reuse Cases

In measuring no-reuse cases, we perceived that precision and recall measures become inappropriate measures. Therefore, we took the advantage of Boolean function. For the convenience of notation, we abbreviated this measure into *noReU*.

In order to computer a noReU rate, most attributes in S were defined to be empty, except for the attributes of source document and its length. Unlike S, the set of detected cases reported in R has only two possibilities,, whether it is empty or assigned values as a result of a false detection. Based on this probability, each tuple attribute in *r* will be assigned a boolean value 1 if its tuple attributes are assigned, otherwise it has a boolean value 0. Thus, each $r \in R$ has only the following possible boolean values {0, 0, 0, 0, 0} or {1, 1, 1, 1, 1}. Unlike in *r*, The tuple attributes in *s* have only the following boolean values {0, 0, 0, 1, 1}.

The boolean value of a set pair s and r, bol(s, r) is computed by adding each attribute value of s to r. The bol(s, r_i) is assigned 1 if the addition operation results in 1 for all of its tuple elements, i.e. {1, 1, 1, 1, 1}, otherwise a zero value. i in r_i refers to the index in R cardinality. Based on the value of bol(s, r_i), the Boolean value of bol(S, R) from a given d_{plg} is defined as follows:

$$bol(S,R) = \begin{cases} 1, & if \exists bol(\vec{s},\vec{r}) \in bol(S,R) whose\ value\ is\ 1 \\ 0, & otherwise \end{cases} \quad (9)$$

Based on (9), the noReU score which is actually a macro-average of bol(S, R) is then computed as follows:

$$noReU(S,R) = 1 - \frac{\sum_{j=1}^{N} bol(S_j,R_j)}{N} \quad (10)$$

where N refers to the total number of tested d_{plg} with no-reuse cases.

VI. RESULT AND DISCUSSION

To evaluate the performance of our text alignment, we run an *oracle experiment*. For retrieving source candidate documents, we relied on the source retrieval module whose early concepts were described in [24]. Thus, given a d_{plg}, the source retrieval module will retrieve a set of candidate documents which become the inputs of our Text Alignment module.

We did the experiments on two *seed units*, i.e. word unigram (token) and character 5- to 7-grams. We took an observation on the use of frequency-based and semantic-based stopwords and their variation with stemming. As its results, there were 4 variants of token seeds which were codified with TK followed by numbers 1 to 4 (TK1-TK4), where TK1 stands for token normalized by frequency-based stopwords, TK3 represents token normalized with semantic stopwords. TK2 & TK4 are stemmed tokens of TK1 & TK2. In N-grams, we applied character n-stopgrams due to the characteristics of Indonesian.

To see how good the performance of the proposed methods, we conducted a comparison experiment between our prototype, *PlagiarIna*, to Alvi's algorithm [11]. The reason why Alvi's algorithm was chosen among othes are:

1) Alvi´s algorithm makes use of Rabin-Karp algorithm the most text reuse detection systems in Indonesian do.
2) It has been tested in PAN'14 test corpora.
3) Both Alvi's algorithm and PlagiarIna applied rule-based approaches for seed extension.

Due to space limitation, only some experiment results on simulated test documents could be presented in Table I while Table II presents the experiment results on both systems tested on artificial test documents.

TABLE I
EXPERIMENT RESULTS ON PLAGIARINA AND ALVI'S PERFORMANCE TESTED ON SIMULATED TEXT REUSE CORPUS

Systems	Methods	Char-based			Pass-based			Doc-based		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
PlagiarIna	TK1	**.76**	**.60**	**.67**	**.76**	**.66**	**.67**	.89	.72	.80
	TK2	.75	.59	.66	.74	.58	.61	**.92**	**.74**	**.82**
Alvi's		**.76**	.45	.57	.75	.52	.60	.87	.68	.76

Table I shows that averagely the F1 scores of *PlagiarIna* are relatively higher than F1 score of Alvi's algorithm. However, the character n-grams performance was quite dissatisfying and the highest score which was achieved by 7-grams are still lower than Alvi's algorithm. The rational is that the gap parameter is set to a fixed value for all seeds.

The experiment results of both systems on artificial test documents were presented in Table II. It shows that PlagiarIna outperforms Alvi's algorithm in all obfuscation types (deletion, insertion, deletion and insertion, shuffle, synonymy replacement) for all levels of measures. For artificial test documents, the passage level scores have the same values with the document level measures. The rationale is that the random obfuscation was performed in the document level. Thus, a document is treated as a long single passage.

TABLE II
EXPERIMENT RESULTS ON PLAGIARINA AND ALVI'S PERFORMANCE TESTED ON ARTIFICIAL TEXT REUSE CORPUS

Obfuscate	Systems	Char-based			Pass-based			Doc-based		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Delete & Insert	PlagiarIna	**.95**	**.84**	**.89**	**.91**	**1**	**.95**	**.91**	**1**	**.95**
	Alvi's	.83	.40	.53	.45	.83	.52	.83	.83	.83
shuffle	PlagiarIna	.57	.08	.14	.58	.66	.66	.58	.66	.52
	Alvi's	0	0	0	0	0	0	0	0	0
synonym	PlagiarIna	.61	.83	.70	.83	.83	.83	.83	.83	.83
	Alvi's	.33	.06	.10	.25	.33	.28	.33	.33	.33

From Table I & II, it can be seen that *PlagiarIna's* rates tested on artificial test documents are much higher than its rates on simulated ones. This indicates that algorithmically obfuscated texts present few problems to PlagiarIna. In contrast, texts obfuscated by human writers still become challenges for our prototype system. Some possible explanations for this are that firstly human writers tend to obfuscate texts on the **different levels of linguistic structure** such as on morphological, lexical, and syntactic structures, while algorithmic obfuscation occurs on the lexical level only. Secondly, test documents belonging to artificial plagiarism cases contain only one type of obfuscation per document, while those in simulated plagiarism cases tend to comprise **different obfuscation types per document**.

The recognition rates on obfuscation types of simulated test documents are presented in Table III. In this experiments, we classified the degree of paraphrase into light (L), medium (M) and heavy paraphrase (H). Table III shows only PlagiarIna's highest score which was achieved by TK3 and the lowest scores produced by TK2. However, TK2 outperforms Alvi's algorithm which is proved by its higher scores on the obfuscation types of copy, 3 levels of paraphrase, and copy and shake. However, Alvi's algorithm score on summary is insignificantly higher than PlagiarIna's scores achieved by all seed units.

TABLE III
THE ALVI''S AND PLAGIARINA'S RECOGNITION RATES ON THE OBFUSCATION TYPES TESTED ON SIMULATED TEST DOCUMENTS.

Systems	Meth	Copy	para-L	para-M	para-H	shake	smry
PlagiarIna	TK3	.90	.91	.81	.44	.74	.37
	TK2	.81	.88	.48	.48	.70	.37
Alvi's		.68	.55	.42	.42	.64	.45

In detecting no-reuse cases, 3 methods of PlagiarIna reach its maximum rates, 1 for seed units TK2, TK4, and character 7-grams. In general, some PlagiarIna's methods produce rates that higher than Alvi's score except for TK1 (see table IV).

TABLE IV
PLAGIARINA'S AND ALVI\S DETECTION ON NO-REUSE CASES

	PlagiarIna					Alvi's
	TK1	TK2	TK3	TK4	7GR	
Rates	.80	1	.90	1	1	.90

Being tested in our simulated test document corpus, Alvi's recall rates range from 0.45 to 0.68. Its maximal recall rate, 0.68, is as high as its maximal recall rate, when it was tested on PAN corpus, 0.67 [7], [11]. Tested in our corpus, its precision rates range from 0.75-0.87, whose upper range, 0.87 is insignificantly lower that its precision rate tested in PAN corpus which reaches 0.90. Alvi's detection rate on no-reuse case reaches 0.90, which is a very high score. However, it is less high than its score tested in PAN corpus (under no-plagiarism case) which is able to reach the optimal rate, 1.0 [7]. Based on these rates, it could be boldly concluded that the complexity of our evaluation corpus has reached an international standard level.

The recognition rates of the obfuscation types on three level of paraphrases, shake, and copy which are higher than Alvi's scores prove that our paragraph-based alignment me-thod works well. Furthermore, it is capable of detecting heavily-paraphrased and summarized texts without applying any semantic analysis. Another strength of our alignment method is that it produces no overlap detections. Yet, its drawback lies on its passage boundary detection. Based on significant words as seeds, the detected source-suspicious passage pairs may start and end on these significant words, which syntactically may produce nonsense start or end of sentences. It would be better if the start and end of all detected source-suspicious passages are also the start and end of complete sentences in which these significant words occur.

VII. CONCLUSION

This study has shown that the proposed paragraph-based alignment method is capable of detecting both short and long segments of text reuses. The use of significant has proven to be a competitive technique in detecting heavily paraphrased text. Another strength of our proposed alignment method compared to string-based or fingerprinting techniques is that it produces almost no-overlapping detection. One drawback of this method lies on its passage boundary which may produce an unproper start and end of a sentence.

This study has proved also that the complexity of a test document corpus correlates highly with the text reuse detection system's performance. This is validated by the higher rates on all obfuscation types in all levels of measures for artificial test document corpus than the simulated one. Last but not least, this study has successfuly provided a standard evaluation corpus for assessing text reuse detecion systems for Indonesian.

REFERENCES

[1] P. Clough, R. Gaizauskas, S. S. Piao, and Y. Wilks. "METER: measuring textrReuse," in *Proc. 4th Annu. Meeting of the Association of Computational Linguistics (ACL)*, 2002, pp. 152-159

[2] M. Potthast, and B. Stein. "New issues in dear-duplicate detection," in *31st Conf. German Classification Society*, 2008. Pp. 601-609.

[3] M. Bandersky, and W. B. Croft. "Finding text reuse on the web," in *Proc. 2nd ACM International Conf*., pp 262-271, Feb 2009. https://doi.org/10.1145/1498759.1498835

[4] B. Stein, S. M. zu Eissen, and M. Potthast. Strategies for "Retrieving plagiarized documents," in *SIGIR'07, ACM*, Amsterdam, July 2007.

[5] M. Potthast et al. "An overview of the 5th international competition on plagiarism detection," in *CLEF 2013 Evaluation Labs and Workshop*, P. Forner, R. Navigili, and D. Tullis, Eds. Pp. 85-98, September 2013.

[6] M. Potthast et al. "An overview of the 4th international competition on plagiarism detection," in *Notebook Papers of CLEF 2012 Labs and Workshop*, P. Forner et al., Eds., September 2012.

[7] M. Potthast et al. "An overview of the 6th international competition on plagiarism detection," in *Notebook Papers of CLEF 2014 Labs and Workshop*, September 2014.

[8] D. Bär, T. Zescht, and I. Gurrevych. "Text reuse detection using composition of text similarity (Technical paper)," in *Proc. COLING'12*, 2012, pp. 167-184.

[9] L. Kong, Z. Lu, H. Qi, and Z. Han. "Detecting high plagiarism obfuscation plagiarism exploring multi-features via machine learning," in *U- and e-service, Sciences and Technology J*,, vol. 7, no. 4, pp 385-396, 2014.

[10] P. Gross, and P. Modaresi. "Plagiarism detection alignment by merging context seeds," in *PAN CLEF´14 Labs and Workshop*, Sept. 2014.

[11] F. Alvi, M. Stevenson, and P. Clough. "Hashing and merging heuristics for text reuse detection," in *PAN CLEF´14 Labs and Workshop*, Sept. 2014

[12] E. Stamatatos. "Plagiarism detection using stopword n-grams," in *American Society for Information Science and Tech. J.*, vol. 62, no. 15, pp. 2512-2527, 2011.

[13] B. Gipp. *Citation-based Plagiarism Detection: Detecting Disguise and Cross-language Plagiarism Using Citation Pattern Analysis.* Wiesbaden: Springer Verlag, 2014.

[14] D. Glinos. "A Hybrid architecture of plagiarism detection," in *PAN CLEF´14 Labs and Workshop*, Sept. 2014

[15] Z. F. Alfikri and A. Purwarianti. "The Construction of Indonesian-English Cross Language Plagiarism Detection," in *Computer Science and Information J.*, vol. 5, no. 1, pp. 16-23, 2012

[16] A. F. Suryata, A. T. Wibowo, and A. Romadhany. "Performance efficicency in plagiarism indication detection system using indexing method with data tree 2-3," in 2nd Int. Conf. Information and Communication Tech. (IcolICT), IEEE, pp. 403-408, 2014.

[17] C. Vania, and M. Adriani. *Proc. PAN CLEF 2010*, http://ceurws.org/vol-1776/CLEF2010wn-PAN-VaniaEt.pdf

[18] S. Soleman, and A. Purwarianti. "Experiment on the Indonesian plagiarism detection using latent semantic analysis," in *2nd Int. Conf. Information and Communication Tech. (IcolICT)*, IEEE, pp. 413-418, 2014. https://doi.org/10.1109/icoict.2014.6914098

[19] F.Z. Tala. *A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia* (Master's Thesis). University of Amsterdam, Netherland, 2003.

[20] L. Kong et al. "Source Retrieval and Text Alignment Corpus Construction for Plagiarism Detection," in *PAN at CLEF'15 Proc*., 2015, http://pan.webis.de/clef15/pan15-web/proceedings.html.

[21] M. Kiabod, M. N. Drehkodi, and S. M. Sharafi. "A novel method of significant words identification in text summarization," in *Emerging Tech. Web Intelligence J*., vol 4, no. 3, pp.252ß258, 2012.

[22] S.H. Cha. "Comprehensive survey on distance/similarity measures between probability functions," in *Math. Models and Methods in Applied Sciences J*., vol. 1, no. 4, pp. 300-307, 2012.

[23] M. Potthast, B. Stein, A. Baron-Cedono, and P. Rosso. "An evaluation framework for plagiarism detection," in *Proc. 2nd International Conf. on Computational Linguistics (COLING'10)*, pp. 997-1005, 2010.

[24] L.D. Krisnawati, and K. U. Schulz. "Plagiarism detection for indonesian texts," in *15th Int. Conf. Information Integration and Web Services (iiWAS'2013)*, pp. 595ß599, 2013. https://doi.org/10.1145/2539150.2539213