Plagiarism Detection for Indonesian Texts

Lucia D. Krisnawati

München 2016

Plagiarism Detection for Indonesian Texts

Lucia D. Krisnawati

Dissertation
an dem Centrum für Informations- und
Sprachverarbeitung (CIS)
Ludwig–Maximilians–Universität
München

vorgelegt von
Lucia D. Krisnawati
aus Madiun, Indonesien

München, 14 März 2016

Erstgutachter: Prof. Dr. Klaus U. Schulz
Zweitgutachter: PD Dr. Stefan Langer
Tag der mündlichen Prüfung: 18 Mai, 2016

Declaration of Authorship

I, Lucia D. Krisnawati, declare that this thesis entitled, *Plagiarism Detection for Indonesian Texts*, and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.

Signed: ________________________________

Date: August 29, 2016

The fear of the Lord is the beginning of knowledge,
But fools despise wisdom and instructions.
Proverbs 1:7
"Copy from one, it's plagiarism, Copy from two, it's research"
John Milton
"Good writers borrow, Great writers steal"
Oscar Wilde
"Self-Plagiarism is a style"
Alfred Hitchcock
"Plagiarists at least have the quality of preservation"
Benjamin Disraeli

Table of Contents

List of Figures

List of Tables

Abstract

As plagiarism becomes an increasing concern for Indonesian universities and research centers, the need of using automatic plagiarism checker is becoming more real. However, researches on Plagiarism Detection Systems (PDS) in Indonesian documents have not been well developed, since most of them deal with detecting duplicate or near-duplicate documents, have not addressed the problem of retrieving source documents, or show tendency to measure document similarity globally. Therefore, systems resulted from these researches are incapable of referring to exact locations of *"similar passage"* pairs. Besides, there has been no public and standard corpora available to evaluate PDS in Indonesian texts.

To address the weaknesses of former researches, this thesis develops a plagiarism detection system which executes various methods of plagiarism detection stages in a workflow system. In retrieval stage, a novel document feature coined as *phraseword* is introduced and executed along with word unigram and character n-grams to address the problem of retrieving source documents, whose contents are copied partially or obfuscated in a suspicious document. The detection stage, which exploits a two-step paragraph-based comparison, is aimed to address the problems of detecting and locating source-obfuscated passage pairs. The seeds for matching source-obfuscated passage pairs are based on locally-weighted significant terms to capture paraphrased and summarized passages. In addition to this system, an evaluation corpus was created through simulation by human writers, and by algorithmic random generation.

Using this corpus, the performance evaluation of the proposed methods was performed in three scenarios. On the first scenario which evaluated source retrieval performance, some methods using phraseword and token features were able to achieve the optimum recall rate 1. On the second scenario which evaluated detection performance, our system was compared to Alvi's algorithm and evaluated in 4 levels of measures: character, case (or passage), document, and obfuscation type. The experiment results showed that methods resulted from using token as seeds have higher scores than Alvi's algorithm in all 4 levels of measures both in artificial and simulated plagiarism cases. In recognizing the obfuscation type, our systems outperform Alvi's algorithm for copied, shaked, and paraphrased passages. However, Alvi's recognition rate on summarized passage is insignificantly higher than our system. The same tendency of experiment results were demonstrated on the third experiment scenario, only the precision rates of Alvi's algorithm in character and case levels are higher than our system. The higher *Plagdet scores* produced by some methods in our system than Alvi's scores show that this study has fulfilled its objective in implementing a competitive state-of-the-art algorithm for detecting plagiarism in Indonesian texts.

Being run at our test document corpus, Alvi's highest scores of recall, precision, Plagdet, and detection rate on no-plagiarism cases correspond to its scores when it was tested on PAN'14 corpus. Thus, this study has contributed in creating a standard evaluation corpus for assessing PDS for Indonesian documents. Besides, this study contributes in a source retrieval algorithm which introduces phrasewords as document features, and a paragraph-based text alignment algorithm which relies on two different strategies. One of them is to apply a local-word weighting used in text summarization field to select seeds for both discriminating passage pair candidates and for matching process. The proposed detection algorithm results in almost no overlapped detection. This contributes to the strength of this algorithm.

Zusammenfassung

Während Plagiarismus indonesische Universitäten und Forschungszentren zunehmend besorgt, wird die Verwendung von automatischen Plagiatserkennungssoftware immer notwendiger. Allerdings ist Plagiatserkennungssoftware für indonesische Dokumente noch unterentwickelt. Die meisten von ihnen befassen sich mit der Erkennung von Duplikaten oder annähernde Duplikattexten. Die bisherige Forschung addressiert jedoch nicht Probleme mit Abrufen von Quelldokumenten oder tendiert dazu, Dokumentähnlichkeit umfassend zu messen. Daher sind Plagiaterkenungssysteme in der Regel umfähig, zusammengehörige Quelltextabschnitte und plagiierte Textabschnitte zu ermitteln. Außerdem existiert keine öffentlichen und Standardkorpora, um indonesische Plagiatserkennungssoftware zu testen und zu bewerten.

Die vorliegende Studie entwickelt eine Plagiatserkennungssoftware, die verschiedene Methoden der Plagiatserkennung in mehreren Stufen (als Workflow-System) durchführt. Für die Abrufphase wird das neue Dokumentsmerkmal *phraseword* eingeführt. Phraseword wird zusammen mit Wort-Monogramm und Buchstaben-N-Grammen ausgeführt, um das Abrufen von Quelldokumenten zu ermöglichen, deren Inhalte teilweise kopiert oder verschleiert in verdächtigen Dokumenten enthalten sind. Das Ziel der Textabgleichsphase, die einen zweistufigen abschnittbasierten Vergleich nutzt, ist, Paare von Quellabschnitten und plagiierten Abschnitten aufzufinden. Die Saatgüter (*seeds*), die benutzt werden, um die Paare aus Quelltext und plagiiertem Text abzugleichen, werden durch eine lokale Termgewichtungstechnik selektiert. Damit sollen paraphrasierte und zusammengefasste Abschnitte erfasst werden. Zusätzlich zu diesen Ansätzen wurde ein Evaluierungskorpus erstellt. Dieser besteht aus einer Simulation menschlicher Texte (geschrieben durch Menschenhand) und algorithmischer Zufallsgeneration

Unter Verwendung dieses Korpus wurde in drei Szenarien die Leistung der vorgeschlagenen Methoden bewertet. Im ersten Szenario, das die Leistung des Abfragesystems bewertet, konnten einige Methoden, die phraseword und Tokenmerkmale verwenden, die optimale Recall-Rate 1 erreichen. Im zweiten Szenario, das die Leistung des Abgleichsverfahrens auswertet, wurde unser System mit dem Alvi-Algorithmus verglichen und bezüglich vier Messstufen bewertet: Buchstabe, Fälle (Abschnitt), Dokument, und verirrungstyp . Die Versuchsergebnisse zeigten, dass Methoden, die Token als Dokumentsmerkmale verwenden, für alle vier Messstufen höhere Recall-Rate als der Alvi-Algorithmus erzielten, sowohl für künstliche als auch simulierte Plagiatsfälle. Bei der Erkennung der Plagiatsfälle übertrifft unser System den Alvi-Algorithmus bei der Erkennung von kopierten, *Shake and paste*, und paraphrasierten Abschnitten. Allerdings ist die Erkennungsrate des Alvi-Algorithmus von zusammengefassten Abschnitten unwesentlich höher als die Erkennungsrate unseres Systems. Das dritte Experiment zeigte tendenziell gleiche Ergebnisse wie das zweite. Nur bei den Messstufen Buchstabe und Abschnitt waren die Präzisionsraten des Alvi-Algoritmus höher als die unseres Systems. Die höheren Plagdetraten von einigen Methoden unseres

System verglichen mit dem Alvi-Algorithmus zeigen, dass das Ziel dieser Studie einen neuen Algorithmus zur Plagiatserkennung für indonesische Texte zu entwickeln erfüllt ist.

Diese Studie hat einen internationalen Standard-Evaluierungskorpus zur Beurteilung von Plagiatserkennungssoftware für indonesische Texte bereitgestellt. Der Alvi-Algorithmus wurde erfolgreich auf unseren Testdokumentenkorpus angewendet: Die erzielten höchsten Recall-, Präzisions- und Plagdetraten und die Erkennungsrate für Nicht-Plagiatsflle stimmen mit den Raten überein, als Alvi's Algorithmus am Korpus PAN'14 getestet wurde. Außerdem leistet diese Studie einen Beitrag in der Form eines Source-Retrieval-Algorithmus, der Phrasewords als Dokumenteneigenschaften einführt, und eines absatzbasierten Text-Alignment-Algorithmus, der auf zwei unterschiedlichen Strategien beruht. Eine dieser Strategien ist die Anwendung der lokalen Wortgewichtungstechnik aus dem Bereich der Textzusammenfassung, um die Saatgüter für die Abschnitte auszuwählen. Die Saatgüter wurden benutzt, um gepaarte Quelltextabschnitte und plagiierte Abschnitte abzugleichen. Der vorgeschlagene Text-Alingment-Algorithmus führt zu fast keiner Mehrfacherkennung eines Abschnittpaares. Dies ist ein entscheidender Vorteil dieses Algorithmus.

Chapter 1

Introduction

The abundant availability of information and data in the Web affects the academic life tremendously. On one hand, one needs only a second to update oneself to current research findings and inventions. On the other hand, the ease of accessing research reports and replicating digital documents provide possibilities of commiting plagiarism as found in many student papers and final year project reports. Conventionally, an act of plagiarism could be recognized manually by relying on human cognition on the seemingly-similar texts or on the writing style that changes drastically. However, this kind of recognition demands a sharp memory on all articles, book and any other types of writings which have been read. Another requirement is that the process of reading should have occurred recently. Otherwise, it would be forgotten. With the amazing improvement on the computer network and the vast amount of source documents available in the Internet, the task of recognizing plagiarism is getting beyond the reach of any human cognition. To make it worse, proving a work as an act of plagiarism demands evidence of source documents. This situation gives rise to a need of an Automatic Plagiarism Detection (APD).

1.1 Research Motivation

In 2010, Indonesian public and academicians were shocked by the revelation of three separate cases of plagiarism which involved a full time professor and two lecturers from different outstanding universities [1]. Through these cases, the use of plagiarism checker has become an increasing need for the universities and research centers in Indonesia. However, we could not simply use the available widely-used plagiarism detection products such as *TurnItIn* or *PlagiarismChecker.com*. Inspite of its massive database that covers 45+ billion web pages, 400+ million student papers, and 130+ million articles[2] and its usage in more than 80 countries around the world, *TurnItIn* proves to be incapable of detecting plagiarism for Indonesian texts. The reason is that firstly Indonesian is excluded from the list of 19 languages which it supports. Secondly, its database contains no Indonesian texts. In fact, *TurnItIn* is a tool for checking text similarity on a document level as it can be seen in its report to teachers which provides a percentage of unoriginality of student's assignment. In contrast, *PlagiarismChecker* is a tool that matches copy-and-pasted student papers against

[1]*Saving Indonesia from Traps of Plagiarism*. Kompas Online, April 28, 2010. Retrieved from `http://english.kompas.com/read/2010/04/28/02563687` in April 2014

[2]TurnItIn content: Content Database `http://turnitin.com/en_us/features/originalitycheck/content`

those found in Google or Yahoo. It searches phrases rather than the whole paper, and thus it functions merely as a search engine for Indonesian texts.

Modelled to *TurnItIn*, the former accessible researches on Automatic Plagiarism Detection for Indonesian texts concentrate on measuring text similarity on a document level. This can be seen in [135] which calculates the document similarity by means of clustering, or in [94] that uses Naive Bayes to classify the plagiarized documents and calculates their similarity to source documents within the assigned classes. Measuring similarity on document level proves to be good in cases where a large number of similar portion is found on a pair of plagiarized and source documents. The drawback of this approach is that it will give poor similarity values in cases where the text is copied partially, or where the length of the copied passages cover only a small portion of a plagiarized document. In real cases, plagiarism is very often done smartly, for example by paraphrasing or obfuscating the texts so that only a small part of document are found similar. Thus, such methods prove to be unfit for the later cases. This motivated us to develop a plagiarism detector for Indonesian texts that is capable of detecting not only the copy-and-paste cases but also the obfuscated plagiarism cases on the level of passages.

1.2 Problem Setting and Scope of the Study

In the recent development of plagiarism detection, detecting duplicate and near-duplicate files has no longer been research challenges. The reason is that in duplicate and near-duplicate files we find the following phenomena:

a) the plagiarism often takes the form of a literal copy.

b) the portion of plagiarized text is large and may cover more than 70% of the document length which makes both documents almost identical.

c) the copy is taken from limited number of sources, even it is very often taken from one source only.

d) the duplicates and near-duplicates are mostly found in cases of website duplicates or on novice student's term papers.

In contrast, we found out the following phenomena in the real setting of academic plagiarism:

a) the plagiarism takes in various forms,

b) the plagiarized passages are very often modified in order to conceal the offenses [130]. They could be reduced to a smaller extent which covers a small portion of a suspicious document, which is a plagiarized version of a source document.

c) the number of source documents for a suspicious document is quite large or minimally more than one.

This real setting of academic plagiarism led us to identify our research problems which cover two main areas of plagiarism detection as follows:

1. **Source Document Retrieval**
   Different from Information Retrieval, retrieving source documents for a given plagiarized text requires elaborate strategies and techniques, so that the Plagiarism Detection system is able to retrieve not only sources having highly similar successive words and phrases but also sources whose passages are modified and partially copied. Referring back to the results of our former studies which applied word n-gram model, bag of word approach, and global similarity measurement (cf. subsection 2.2.2.1.4), we found out that two documents having high global similarity score may share no consecutive similar word n-grams where n is set to be greater than or equals to 4 [149, 187]. This result was in line with another research conducted by Stein and Eissen in [166] which applies fingerprints as document representation. One possible explanation is that those methods ignore the consecutive occurrences of similar words in some extent which become the main requirement in APD. Thus, the results of these studies led us to pose the following questions:

   1.1 What kind of strategies and methods are able to give high score on source documents whose contents are obfuscated and copied partially or fully in a suspicious document?

   1.2 What kind of information available in a text could be used to represent a document, so that such representation model is able to capture passage similarity even though those passages are obfuscated structurally and their word orders are either shuffled or preserved?

   1.3 How to formulate queries which represent “hidden plagiarized passages” and enable retrieving source documents characterized in problems 1.1 and 1.2 ?

2. **Passage Similarity Detection**
   In some APD systems as in [32, 94, 135, 146], the task of detection is terminated as source documents of a suspicious one are retrieved or identified. The high similarity or low distance scores between source-suspicious document pair is commonly used to filter the source documents. Such task scenario supports duplicate or near-duplicate detection only and therefore has not addressed the problem setting of academic plagiarism which demands similarity detection to a passage level. A common method which is used to locate plagiarized passages in both source and suspicious document is an exhaustive comparison using string or substring matching. Two weaknesses from this method are that firstly it is computationally expensive, and secondly it has difficulties in handling obfuscated passages [58, 129]. In locating and detecing passage similarity, this research focuses on answering the following questions:

   2.1 What kind of methods and strategies are able to locate a pair of source passage and its modified passage efficiently and effectively?

2.2 How to determine similar passage boundaries and what kind of parameters could be used to define this task?

The strategies and methods stated in research problems 1.1 and 2.1 should address the problems of source retrieval and detecting plagiarism in Indonesian texts, as the scope of this study is set to a monolingual, external plagiarism detection which takes Indonesian texts as its object. The plagiarism scope in this study refers to the academic plagiarism which excludes detecting duplicate websites or blogs. The reason is that there has been few researches conducted on plagiarism detection for Indonesian texts despite the great need for it. Besides, there has been no standardized corpus available to test the performance of plagiarism detection algorithms. Some former researches used either PAN'10 Corpus that contains documents in several European languages [181], or Clough and Stevenson corpus [94]. So, it does not address the problems of plagiarism detection for Indonesian texts, except that research was conducted by Indonesians.

In terms of plagiarism types, the methods implied in the research questions 1.1-1.3 and 2.1-2.2 should be able to detect plagiarism from the type of literal copy as well as the obfuscated texts by means of paraphrasing and summarizing. The ghostwritten texts, which very often cause polemic on whether they are included as a specific plagiarism type or not, share common characteristics with plagiarized texts, because many ghostwriters tend to reuse texts from their database or available texts by doing a slight, medium, or heavy modifications. Thus, the problem of detecting ghostwritten texts is automatically covered in the former problems, types and degree of plagiarism.

1.3 Research Objectives and Contributions

Based on the research problems and motivation mentioned in the previous sections, the objective of this research is *'to design, implement and evaluate an external plagiarism detection algorithm in Indonesian texts which is capable of detecting plagiarism on passage level with different kinds of obfuscation'*. This objective is carried out through the following tasks:

1. Conducting a thorough literary research on state-of-the-art APD in general and on the available APD in Indonesian texts in order to be able to propose a new concept of APD for Indonesian texts.

2. Designing a framework for alternate execution of various detection methods based on distinct document representations in a system workflow. The framework is schematized as a three-stage approach that consists of retrieval, detection, and post-processing stages [164]. In retrieval stage, a novel type of document representation which is coined as a 'phraseword', is introduced and executed along with other document representations to address the research problems numbered 1.2, 1.2 and 1.3. The detection stage exploits a two-step comparison applying different comparison measurements to address the research problems listed in number 2.1 and 2.2.

3. Finding and implementing a competitive state-of-the-art algorithm on plagiarism detection for Indonesian texts.

4. Creating a standard evaluation corpus for testing Indonesian plagiarism detection systems. So far, there has been no public and standard corpus available to evaluate Indonesian plagiarism detection systems. Researches on external plagiarism detection conducted by Indonesians either use the available evaluation corpora containing documents in western European languages, or use their own customized corpora.

5. Evaluating the performance of the the proposed methods and comparing the proposed system performance to one of state-of-the-art algorithms on external plagiarism detection systems.

The products, outputs, and realization of four objectives described above are meant to be the contributions of this research.

1.4 The Thesis Structure

The thesis is organized into two parts. The first part deals with literary research on plagiarism which covers chapter 2 and 3. The second part of the thesis presents the proposed framework, corpus building and the system evaluation which cover chapter 4, 5, 6 and 7. Chapter 1 presents the introduction that comprises the research motivation, problems, objectives and the organization of the thesis.

Chapters 2 is organized into two parts coomprising two closely related subtopics. The first part deals with the important concepts of plagiarism viewed from socio-historical perspectives, plagiarism taxonomy, and some possible plagiarism scenarios in the real setting. The second part of chapter 2 deals with Automatic Plagiarism detection and presents the definitions, types of plagiarism detection, the existing and direction methodologies on APD including state-of-the-art approaches.

Chapter 3 describes a concise overview of Indonesian language, its morphological and syntactical features. A review on the previous researches on Automatic Plagiarism Detection for Indonesian texts can also be found in this chapter.

Chapter 4 presents the proposed methodology conducted in this research. It covers the architecture of the system in a workflow, the algorithm applied in retrieval and detection stages. The implementation of various methods and the use of various document representations for source retrieval are presented here. These various methods and document representations are realized in a plug and play system which enables users to switch to different methods within a application program. This means that a user does not need to run and switch to a different program application whenever he/she switches to the different methods. The rest of chapter 4 presents the two-step text comparison which is attributed as text-alignment in detection phase.

Chapter 5 describes the process of corpus building for the sake of evaluating the system performance, the evaluation measurements and the experiment scenarios. The rest of the chapter discusses the similarity metrics used in both retrieval and text alignment subtasks.

Chapter 6 reports the evaluation and experiment results of the proposed methods. The performance comparison between the proposed methods and an algorithm included in the state-of-the-art could be also found in this chapter. This algorithm is implemented in a setting which enables it to be comparable and used to detect indonesian texts.

Chapter 7 sums up and concludes the conducted research. The outline of research contributions will be described in this chapter. Also, it presents an outline direction for the possible future research work.

Chapter 2

Plagiarism and Plagiarism Detection

Plagiarism detection (PD) has become a field of study that attracts the attention of many researchers in the last two decades. However, many references in automatic PD simply blame the advancement of the Internet, computer network, storage devices, and the ease of information sharing as the primary factors that encourage someone to slip into an act of plagiarism. Is it true that the act of plagiarism is triggered by the advances in IT per se? This question led me to explore *Plagiarism* from socio-historical perspectives in order to shed a light on its wide concepts and usage. For this reason, this chapter deals with two topics, plagiarism and plagiarism detection. Section 2.1 presents plagiarism in socio-historical perspective, plagiarism scenario and taxonomy of plagiarism. A review on various methods and approaches of plagiarism detection systems including approaches in state-of-the-arts will be presented in section 2.2.

2.1 On Plagiarism

2.1.1 Plagiarism in Socio-Historical Perspective

The practice and concept of plagiarism have existed long before the term plagiarism itself came into being, as can be seen in the study on plagiarism in ancient times [101, 148]. Even, it possibly exists since human being starts the activity of writing [184], and thus it has nothing to do with the rise of information technology and the Internet. The practice of plagiarism has been quite common in Latin literature dated from first century BC to the first century CE as it was claimed by Vitruvius, Pliny the Elder, Seneca the Elder, Manilius, and Martial [101]. Surely, these writers used different terms to address the concept of plagiarism as we understand nowadays.

The earliest accusation of 'plagiarism' was raised by Vitruvius, a Latin author, in 20s BC on the preface of his 7th book *De Architectura* [101]. The term used is *furtum*, meaning 'to steal'. Later on, *surripere*, which denotes the same meaning as furtum, was used more frequently to address the plagiarism practices [101]. It was Martial, a Roman poet, who introduced the root of plagiarism by using *plagiarius* to accuse his patron, Fidentinus, of stealing his verses in his book published in 85 CE [93, 101, 148]. The word *plagiarius*, which refers to a "kidnapper or plunderer, a man who kidnaps a child or slave of another[3], is derived from *plagiare* which means 'to kidnap'. Another Latin poet in 4th century CE, Ausanius, used *Laverna* referring to 'goddess of thieves' for a plagiarist, while Macrobius

[3]Etymology Online Dictionary `http://www.etymonline.com/`

used *alieni usurpatio*, a legal term for property theft [101]. Further, McGill's study notes that other terms used in Latin are *sumere* and *transferre* which signify to 'imitate' and 'translate'. The term used by Martial disappeared and remained unused till the medieval period [148]. It reappeared in 1601 when Ben Johnson introduced the term, *plagiary*, to describe a literary theft in the English society and Samuel Johnson confirmed it by defining it in his Dictionary of 1755 as "A thief in literature; one who steals the thoughts or writings of another" [93].

One thing in common is that most words addressing plagiarism practices before the 18th century associate their meanings to a crime either as stealing or kidnapping. This shows that plagiarism is undetachable from authorship, a concept which views a piece of writing as a property of its writer. Zebroski argues that both plagiarism and authorship are a construct of social formation at a particular moment in its development [192]. Supporting Zebroski's idea, Randall claims that the existence of plagiarism depends on an act of reception of the authoritative readers [140]. This implies that stealing an intangible authorial property could be labeled as plagiarism in one culture but not in another. This depends, in my perspective, on the interpretation of the ownership concept within the concept of authorship. In a society where the ownership of a work of writing is individually attributed to its author, there exists plagiarism. On the contrary, the plagiarism accusation is unknown in a society where a piece of writing is owned collectively and shared for the benefit of its members. As an example within the Roman culture itself, the notion of plagiarism became inapplicable to texts known as scripture describing Jesus movement and biography [192], though these texts were written around the time when Martial declared as a victim of plagiarism .

The concept of originality introduced by Edward Young in 18th century took part in shaping our current definition of plagiarism. Stearns includes 3 main concepts: intent, attribution, and copy, as he defines plagiarism as "**intentionally taking** the literary property of another without attribution and passing it off as one's own, having failed to add anything of value to the copied material and having reaped from its use an unearned benefit" [162]. In later references, plagiarism definition is extended to link up with ideas, the imitation of structure, research, and organization as well as language. The definition given by Institute of Electrical and Electronics Engineering (IEEE) reflects this extension as it is defined as "the **reuse** of someone else's prior ideas, processes, results, or words without explicitly acknowledging the original author and source"[4]. No matter how many concepts are conveyed in a definition of plagiarism, the fact shows that manual and automatic plagiarism detections are still heavily based on the recognition of words, phrases, or sentences. To end the historical perspective, the nowadays definitions of plagiarism which are represented by Stearns and IEEE is much more polite as they describe plagiarism as an act of 'taking' or 'reusing' texts; and for Yilmas plagiarism is an act of 'borrowing' texts from others [190].

[4]http://www.ieee.org/publications_standards/publications/rights/plagiarism.html

2.1.2 Plagiarism Scenario

In plagiarism scenario, parameters used to judge a work as a plagiarized one should be clear. Ironically, there have been no references that clearly state this matter since there has been no common platform concerning it. In order to summarize some basic factors used to determine plagiarism, searches on word collocations were conducted on some corpora [5] in addition to literary study. These factors are summarized as follows:

a **Intent**
Some definitions of plagiarism include intention as one characteristic of plagiarism conduct [25, 93, 162]. In contrast, IEEE and Meuschke & Gipp argue that plagiarism might occur unintentionally [103]. Some searched corpora [42–44] contrast *subconcious* with *deliberate plagiarism* which support Meuschke & Gipp's argument. Subconscious plagiarism may happen due to many reasons such as psychological memory bias, cryptomnesia, or lack of knowledge on doing citation [43, 44, 103][6] .

b **Author**
The source for reusing the ideas, structure, or words should not be necessarily written by other authors but it could be one's own writing if an author reuses substantial parts of his or her own published writings without providing proper references [22, 103].

c **Consent**
Someone can be still accused of doing plagiarism even if he gets a consent from another author who collaborates with him or her, in addition when he fails to acknowledge the source [46, 103]. This defines a collusion which describes the behavior of authors who write collaboratively or copy from another, although they are required to work independently [103].

d **Level of Writing Unit**
The amount of writing unit whether it is a full paper, sections of a paper, a page, a paragraph, sentence, or phrases could be used to justify plagiarism. Bouville in [22] suggests a different treatment for plagiarism coverage. If the length of reused texts reaches less than or equal with two lines, this text needs correction and editing for its reference. Thus it is free from charge of plagiarism. But for some cautious writers, the amount or quantity does not play a part in identifying plagiarism. This is an extreme scenario in identifying plagiarism. It leaves nothing for novice writers and I think no writer is able to avoid plagiarism, since there is no limitation on the level of writing unit.

Factors listed under the points a and c (intent and consent) are traceable manually if we have contact or communication with the suspected authors, such as students in submitting

[5]The corpora which were searched are those hosted by BYU university: CoCa, CoHa, Time, Wiki, BNC, Google Books

[6]`https://en.wikipedia.org/wiki/Source_amnesia`

Table 2.1: Percentage of plagiarism per document and its category
Sources [125, 126]

Category	**Percentage**
hardly	5%–20%
medium	20%–50%
much	50%–80%
entirely	$\geq$ 80%

their term papers. But it will give difficulties if we use an automatic plagiarism checker. Concerning this matter, IEEE in its guidelines for handling plagiarism complaints defines plagiarism scenarios into five levels or degrees that range from the most serious into the least serious one. The followings are the five scenarios summarized from IEEE guidelines[7] and from [66]:

1. Uncredited copying of a single or more than one full paper whose total percentage of discovered plagiarism sums to or greater than 50%.

2. A large portion of uncredited verbatim copying within a paper whose sum of copying percentage is between 20%-50%.

3. Uncredited verbatim copying of individual elements such as paragraphs, sentences, illustrations, etc which results in significant portion up to 20%.

4. Improper paraphrasing of pages or paragraphs with no notice of reference on it.

5. Credited verbatim copying of a major portion of a paper without delineation. The use of quotation marks are expected here as a clear boundary between verbatim copying and author's own expression.

The IEEE plagiarism scenario implies that a document whose 20% of its content is copied from other sources could be addressed as a work of plagiarism, given no indication of attribution on its sources. The percentage of plagiarism in one document to its length is used to categorize the level of plagiarism, whether it is a hardly or entirely plagiarized. Table 2.1 describes the plagiarism rate per document as described in [125, 126].

2.1.3 Taxonomy of Plagiarism

Ironically, the more rigorous plagiarism scenarios are defined and more attention is paid to using plagiarism checkers, the more sophisticated also the methods applied by plagiarists in concealing their copied material. Based on the literary research and searches on corpora, a taxonomy on academic plagiarism types is proposed here. This taxonomy is composed by adding and restructuring some plagiarism types that have not been mentioned in [9]

[7]Taken from IEEE's Identifying Plagiarism : `http://www.ieee.org/publications_standards/publications/rights/ID_Plagiarism.html` retrieved in February, 17, 2015.

or in [103]. The schema of plagiarism taxonomy can be found in figure 2.1, which groups plagiarism into three categories: literal, concealed, and pseudo-plagiarisms. The following sections will explain each category of plagiarism.

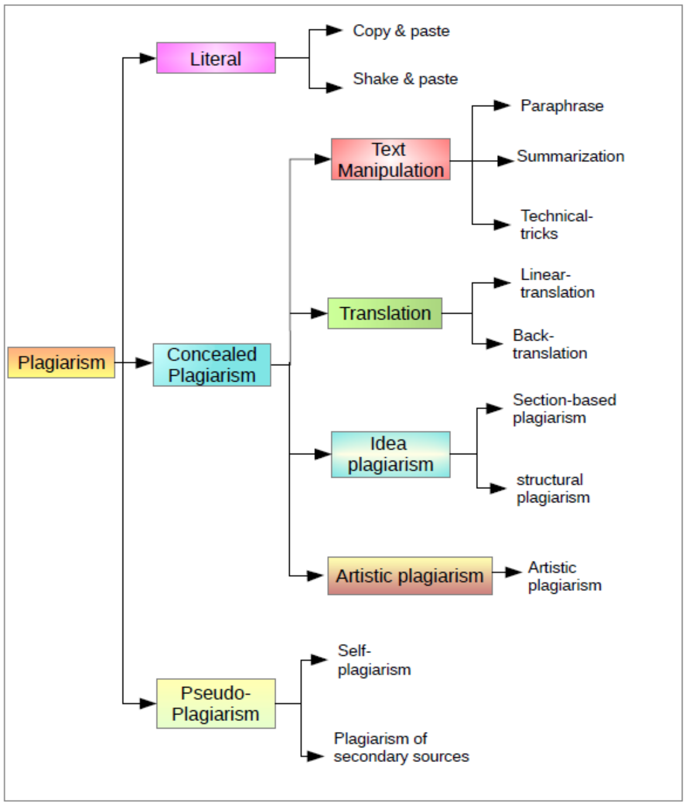


Figure 2.1: Taxonomy of Plagiarism

1. Literal Plagiarism
   In Literal or verbatim plagiarism, the authors copy the source text exactly, or does a few alteration [9]. Thus, two types of plagiarism in this category include 'exact duplicate' (exact-copy) and 'near-duplicate' (near-copy) which are commonly found in the area of detecting similarity of web pages. Near-copy is addressed also as 'shake and paste', whereby the slight alteration is done by copying text segment, paragraphs or sentences from various sources and then assembling them into a text or under a subheading, or by changing word orders, substituting words with their synonyms, or by adding and deleting [58, 184].

2. Concealed Plagiarism
   In concealed plagiarism, a great deal of effort is committed to hide the instances of plagiarism. These efforts are conducted intelligently and may take forms by manip-

ulating the text, translate it into another language, being done artistically, or by adopting the idea of the source texts.

(a) Text Manipulation
There is a fine line between plagiarism and doing text manipulation. Someone may slip into doing plagiarism when he is manipulating a text and simply forgets to provide a proper citation. Types of text manipulation take into three forms: paraphrasing, summarizing, and technical tricks.
Paraphrasing may occur on lexical and syntactic level [9]. Paraphrasing on the lexical level is done normally by adding, removing or replacing characters or words, or by replacing words with their synonyms, or hypernym; while paraphrasing on the syntactic level will be conducted by adding deliberate grammar mistakes, reordering sentences and phrases and some obfuscation effecting changes to grammar style [107]. The difference between paraphrasing on lexical level and the near-duplicate is set on the number of modification.
Summarizing texts in plagiarism cases follows the same principles on how to summarize a text. It can be conducted through sentence reduction, sentence combination, or restructuring [9]. The amount of material in summary is surely much shorter. Summarizing ideas from another text without any acknowledgment of its original source will be considered as an act of plagiarism.
Technical Trick is a type of plagiarism that emerges in response to the use of automatic plagiarism detection. It refers to the techniques that exploit the weaknesses of current APD methods so that the copied material will be undetected [103]. Mozgovoy et al. found out that some technical tricks done by students to deceive computer-aided plagiarism detection could be in form of insertion of similar-looking characters from foreign alphabets such us changing O with Greek Omicron or Cyrillic O, insertion of invisible white-colored letters to the blank spaces, or insertion of scanned text pages as images since the APD is incapable of detecting images [107].

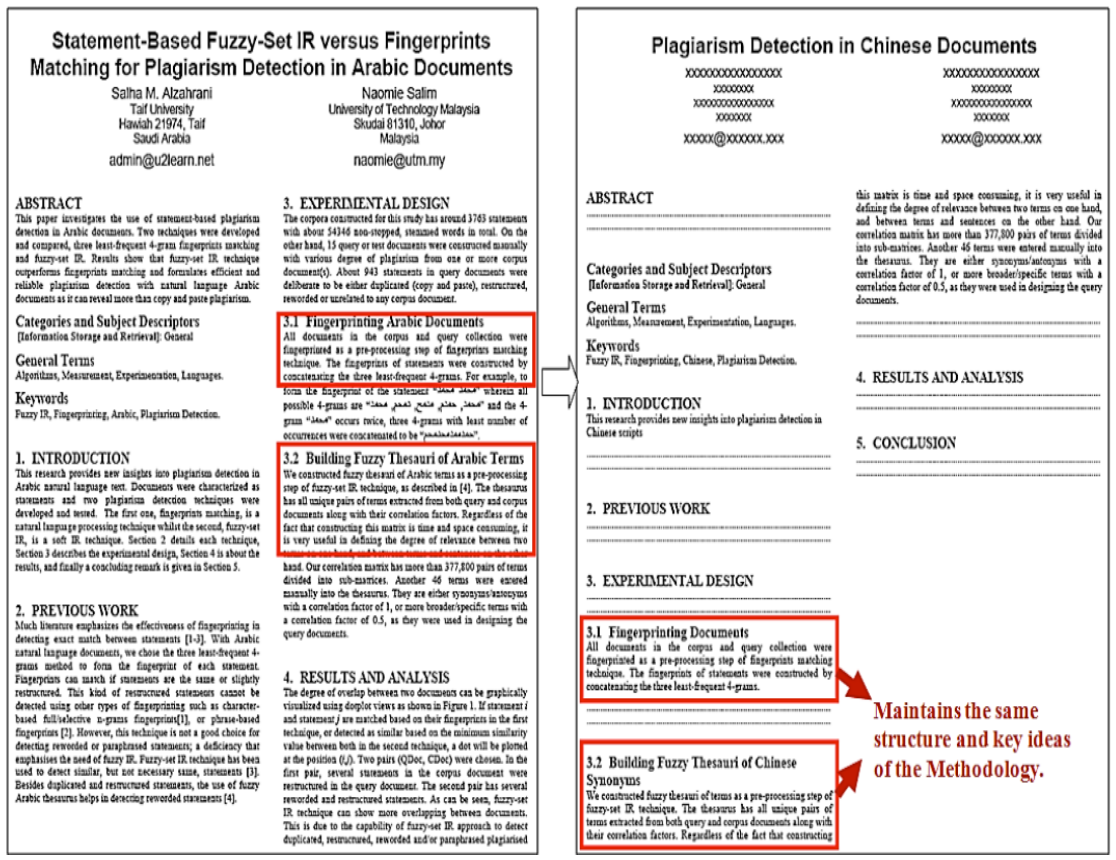
(b) Translation
Passing a written work from a text written in foreign language and translate it into one's own language without proper citation will be considered also as committing a plagiarism. Text obfuscation through translation can be distinguished into two types, that is linear-translation and back-translation. Both linear- and back-translation can be performed manually or automatically [9].
Linear-translation refers to direct translation from a source language to a target language. One may use the available tools on the Internet such as Babelfish or Google to perform automatic linear-translation, by cutting-and-pasting the words or passages, feeding them to these tools and cutting-and-pasting back the output to his text.
Back-translation in plagiarism detection field was introduced by Jones in [Jones, 2009]. Back-translation is actually a common method in the field of translingual translation and is performed with an aim to improve the quality of

the translation or to avoid using diction that might result in a fatal risk when it is applied to clinical and medical texts [11]. It becomes a serious matter if it is used to conceal a cheating act. It refers to a technique employed by students to disguise their cheating by translating a text from a source language to a target language and translating it back to the former source language. For example, a text in English is translated into French and translated back to English using machine translation tools [78]. Recognizing back-translation case in plagiarism is still beyond the reach of some cross-lingual plagiarism detection methods.

(c) Idea Plagiarism
Plagiarism of ideas is defined as Appropriating an idea such as an explanation, a theory, a conclusion, a hypothesis, a metaphor in whole or in part, or with superficial modifications without giving credit to its originator [142]. Based on its scope, Alzahrani et.al. classifies plagiarism of ideas into three types: semantic-based meaning, section-based importance, and context-based adaptation [9].
Semantic-based meaning is a form of idea plagiarism viewed from narrow perspective [9]. The obfuscation is done by either paraphrasing or summarizing, but the idea remains as it is in its original text (cf. Text manipulation section). With the reason that these types of plagiarism have been covered in text manipulation, they will not occur on the diagram shown in figure 2.1.
Section-based importance is a type of idea plagiarism that copies the idea on the level of segments of a scientific text, such as introduction, discussion, results, conclusion or even contributions [9]. The writer may change the words or language of the original texts but the idea remains the same.
Context-based adaptation is addressed also as **structural plagiarism** [58]. In structural plagiarism, one plagiarizes the outline of ideas of a source text taking the form in the compositional element in a broader scale than the section-based importance [9, 58]. Though the ideas are wrapped into different words or language but if the ideas jotted down in its outline remain the same as its source text, then it can be identified as context-based or structural plagiarism. Figure 2.2 illustrates an example of idea plagiarism in context-based adaptation. Gipp argues that structural or context-based adaptation belongs to extreme plagiarism cases, since its presence is an indicator of quality rather than from its originality [58]. It concerns mostly with works that will be published in outstanding journals or publication and thus requires highly subjective justification.

(d) Artistic Plagiarism
Artistic plagiarism is done by presenting someone's else work in a different medium [107]. A good example for this type of plagiarism is an act of converting a novel into a movie script without any appropriate acknowledgment on the novel as its source.

3. Pseudo-Plagiarism
There is still no clear consensus on some types of plagiarism such as self-plagiarism.



Statement-Based Fuzzy-Set IR versus Fingerprints Matching for Plagiarism Detection in Arabic Documents

Salha M. Alzahrani
Taif University
Hawiah 21974, Taif
Saudi Arabia
admin@u2learn.net

Naomie Salim
University of Technology Malaysia
Skudai 81310, Johor
Malaysia
naomie@utm.my

ABSTRACT
This paper investigates the use of statement-based plagiarism detection in Arabic documents. Two techniques were developed and compared, three least-frequent 4-gram fingerprints matching and fuzzy-set IR. Results show that fuzzy-set IR technique outperforms fingerprints matching and formulates efficient and reliable plagiarism detection with natural language Arabic documents as it can reveal more than copy and paste plagiarism.

Categories and Subject Descriptors
[Information Storage and Retrieval]: General

General Terms
Algorithms, Measurement, Experimentation, Languages.

Keywords
Fuzzy IR, Fingerprinting, Arabic, Plagiarism Detection.

1. INTRODUCTION
This research provides new insights into plagiarism detection in Arabic natural language text. Documents were characterized as statements and two plagiarism detection techniques were developed and tested. The first one, fingerprints matching, is a natural language processing technique whilst the second, fuzzy-set IR, is a soft IR technique. Section 2 details each technique, Section 3 describes the experimental design, Section 4 is about the results, and finally a concluding remark is given in Section 5.

2. PREVIOUS WORK
Much literature emphasizes the effectiveness of fingerprinting in detecting exact match between statements [1-3]. With Arabic natural language documents, we chose the three least-frequent 4-grams method to form the fingerprint of each statement. Fingerprints can match if statements are the same or slightly restructured. This kind of restructured statements cannot be detected using other types of fingerprinting such as character-based full/selective n-grams fingerprints[1], or phrase-based fingerprints [2]. However, this technique is not a good choice for detecting reworded or paraphrased statements; a deficiency that emphasises the need of fuzzy IR. Fuzzy-set IR technique has been used to detect similar, but not necessary same, statements [3]. Besides duplicated and restructured statements, the use of fuzzy Arabic thesaurus helps in detecting reworded statements [4].

3. EXPERIMENTAL DESIGN
The corpora constructed for this study has around 3763 statements with about 54346 non-stopped, stemmed words in total. On the other hand, 15 query or test documents were constructed manually with various degree of plagiarism from one or more corpus document(s). About 943 statements in query documents were deliberate to be either duplicated (copy and paste), restructured, reworded or unrelated to any corpus document.

3.1 Fingerprinting Arabic Documents
All documents in the corpus and query collection were fingerprinted as a pre-processing step of fingerprints matching technique. The fingerprints of statements were constructed by concatenating the three least-frequent 4-grams. For example, to form the fingerprint of the statement "[illegible]" wherein all possible 4-grams are "[illegible]" and the 4-gram "[illegible]" occurs twice, three 4-grams with least number of occurrences were concatenated to be "[illegible]".

3.2 Building Fuzzy Thesauri of Arabic Terms
We constructed fuzzy thesauri of Arabic terms as a pre-processing step of fuzzy-set IR technique, as described in [4]. The thesaurus has all unique pairs of terms extracted from both query and corpus documents along with their correlation factors. Regardless of the fact that constructing this matrix is time and space consuming, it is very useful in defining the degree of relevance between two terms on one hand, and between terms and sentences on the other hand. Our correlation matrix has more than 377,800 pairs of terms divided into sub-matrices. Another 46 terms were entered manually into the thesaurus. They are either synonyms/antonyms with a correlation factor of 1, or more broader/specific terms with a correlation factor of 0.5, as they were used in designing the query documents.

4. RESULTS AND ANALYSIS
The degree of overlap between two documents can be graphically visualized using dotplot views as shown in Figure 1. If statement *i* and statement *j* are matched based on their fingerprints in the first technique, or detected as similar based on the minimum similarity value between both in the second technique, a dot will be plotted at the position (*i*,*j*). Two pairs (QDoc, CDoc) were chosen. In the first pair, several statements in the corpus document were restructured in the query document. The second pair has several reworded and restructured statements. As can be seen, fuzzy-set IR technique can show more overlapping between documents. This is due to the capability of fuzzy-set IR approach to detect duplicated, restructured, reworded and/or paraphrased plagiarised

Plagiarism Detection in Chinese Documents

xxxxxxxxxxxxxxxx
xxxxxxx
xxxxxxxxxxxxxxx
xxxxxx
xxxxx@xxxxxx.xxx

xxxxxxxxxxxxxxxx
xxxxxxx
xxxxxxxxxxxxxxx
xxxxxx
xxxxx@xxxxxx.xxx

ABSTRACT
....

Categories and Subject Descriptors
[Information Storage and Retrieval]: General

General Terms
Algorithms, Measurement, Experimentation, Languages.

Keywords
Fuzzy IR, Fingerprinting, Chinese, Plagiarism Detection.

1. INTRODUCTION
This research provides new insights into plagiarism detection in Chinese scripts
....

2. PREVIOUS WORK
....

3. EXPERIMENTAL DESIGN
....

3.1 Fingerprinting Documents
All documents in the corpus and query collection were fingerprinted as a pre-processing step of fingerprints matching technique. The fingerprints of statements were constructed by concatenating the three least-frequent 4-grams.
....

3.2 Building Fuzzy Thesauri of Chinese Synonyms
We constructed fuzzy thesauri of terms as a pre-processing step of fuzzy-set IR technique. The thesaurus has all unique pairs of terms extracted from both query and corpus documents along with their correlation factors. Regardless of the fact that constructing this matrix is time and space consuming, it is very useful in defining the degree of relevance between two terms on one hand, and between terms and sentences on the other hand. Our correlation matrix has more than 377,800 pairs of terms divided into sub-matrices. Another 46 terms were entered manually into the thesaurus. They are either synonyms/antonyms with a correlation factor of 1, or more broader/specific terms with a correlation factor of 0.5, as they were used in designing the query documents.
....

4. RESULTS AND ANALYSIS
....

5. CONCLUSION
....

Maintains the same structure and key ideas of the Methodology.

Figure 2.2: An example of structural plagiarism, The left is supposed to be an original article and the right is the simulated plagiarism case
Source: ([9], p.4)

Some references claim that self-plagiarism and plagiarism of secondary source are definitely a type of plagiarism [23, 58, 142, 184], while others such as [22] argue that these belong to another activity, and hence the label of plagiarism cannot be attached to them. Examining how plagiarism is defined, the phrases such as "...someone else's work.." or "...property of another..." (cf. Pp 8-9) occur in most of plagiarism definition, then it is clear that it does not refer to taking one's own writings or works. Roig's question "if plagiarism is conceptualized as a theft, is it possible to steal from oneself?" is relevant to exclude this activity from the label of plagiarism [142]. For this reason and for the fact that many references identify such activity as self-plagiarism, we put self-plagiarism and plagiarism of secondary source into category of Pseudo-Plagiarism.

(a) Self-Plagiarism
Self-plagiarism is defined as the reuse of someone's own previously written work or data without any proper citation [142]. It may denotes another unethical activity since the available references relate self-plagiarism with these other 4 activities:
Redundant and duplicate publication, that is the activity of publishing a paper whose content is essentially almost similar in more than one journal without proper citation or acknowledgment [142].
Salami Slicing s an act of segmenting a broad topic of research or study into several topics that should be published in one single paper [142]. This type of misconduct is usually done to have as many publications as possible. It is actually an unethical issue for the writer rather than a plagiarism.
Copyright infringement. The redundant publication and the salami slicing may cause copyright infringement. This is due to the fact if someone send his work to be published in a journal or scientific periodicals, one agrees to transfer his copyright to the publisher of journal or periodicals to publish and reuse his work [142]. And if he sends his work to another publisher, this automatically violates the copyright of the publisher with whom he has signed the agreement.
Text Recycling. Roig notes that the pressure to publish for researches may cause text recycling. It is defined as the writer's reuse of portions of text that have appeared previously in other works [142]. Text recycling is done by writing another paper describing entirely different empirical investigations but using nearly identical or similar methodologies. The line between text recycling and redundant publication is really fine here.

(b) Plagiarism of secondary sources
Bouville's disagreement on Martin's claim that someone is committing plagiarism when he quotes the secondary source without looking up the first or original source, leads to the term plagiarism of the secondary source [22]. Personally, I stand for Bouville with a reason that there are many factors for someone not to look up the first or original source. It may deal with the availability and accessibility of the first source. As long as someone gives proper citation, he is

not necessarily to read or look up the first source. This becomes an extreme plagiarism scenario if it is agreed and accepted.

2.2 Automatic Plagiarism Detection

Though it seems just yesterday that we have started relying on the use of automatic plagiarism detection systems, its concept has been thought long enough. Chong claimed that automatic plagiarism detection started off as a detection tool for multiple-choice tests proposed by Angoff [35]. In fact, Angoff developed a statistical method called A Variant Index to detect efforts to copy during test administration [12]. But it was done manually by setting up three different groups of sample chosen from every odd-numbered computer tape, the samples' answered sheet were then compared and analyzed using this variant index. Thirty three years later, McMagnus, Lissauer, & William examined the performance of Angoff's A Variant Index and implemented it into a computer program called Acinonyx to detect answer copying on postgraduate medical examinations [20].

One true aspect of Chong's claim is that plagiarism detection started in 1970s. In 1970s to 1980s, plagiarism detection was used to detect and prevent programming code plagiarism in Pascal, FORTRAN and C by keeping tract of metrics such as number of lines, variables, statements, subprograms, class to subprogram and other parameters [9, 115]. During 1970s, Ottenstein developed an algorithm to detect code plagiarism for FORTRAN source code [115], while a tool for detecting plagiarism in Pascal was developed some years later by Sam in 1981, and at the same time John et al. elaborated the work of Ottenstein in detecting duplication of FORTRAN source codes [9]. In those decades, a tool was capable only to detect a single programming language. Just in 1990, a system called Plague was able to detect code clones in two or more programming languages: Pascal and Prolog source codes [9]. During 1990s-2000, Chong noted that most plagiarism detections were developed for detecting code duplicates, there were only a handful of PD which focused on written texts [35].

The plagiarism detection for natural language, as noted by Alzahrani et al., was initialized by the work of Brin et al. and Shivakumar & Garcia-Molina. Brin and his colleagues developed a detection system named COPS to register documents and detecting copies for the sake of building a digital library so that only the original documents will be registered [24]. A prototype called SCAM was developed using word frequency occurrences [152]. It was then elaborated to find near-replicas of web pages for the sake of improving web crawling, improving ranking search function in search engines and for archiving applications [153]. By the end of 2000, Chong [35] reported that there were only a handful of commercial system available, eg. EVE2 and iParadigm that was well known later as TurnItIn.

From 2000 onwards, the plagiarism detection has successfully caught the attention of many scholars and as its consequence, various methods and approaches could be easily found during this period. In early period of 2000s, there were two distinct approaches in comparing the duplicate documents. This involves comparison of every pair of documents in corpus, or comparing a suspicious document against others in corpus, which in [74] is referred as n-to-n or one-to-n comparison. The main approach dominating the information to represent text during this period was fingerprinting which was specifically developed for this purpose [74]. The use of fingerprinting for comparing similar documents was pioneered by Manber [95] and Shivakumar in [152]. Some outstanding variant of fingerprinting techniques that were applied to detect near-duplicate or partial duplicate are locality sensitive hashing or SimHash [32], Winnowing Algorithm [146] or fuzzy fingerprinting [164].

The International competition on plagiarism detection that was initialized since 2009 has contributed very much to the improvement on the various methods and added to this, the concept realization on how to detect plagiarized documents. This shared task provides not only definition, framework on conducting research on plagiarism detection, but also sets up the corpus, the evaluation concepts and measurements. So far, we have discussed on the history of automatic plagiarism detection, but what is actually meant by plagiarism detection? And what are the tasks of plagiarism detection?

2.2.1 Types of Automatic Plagiarism Detection

Plagiarism detection is defined as a form of a quadruple $s = \langle s_{plg}, d_{plg}, s_{src}, d_{src} \rangle$ where s_{plg} is a passage in document d_{plg} which is a plagiarized version of s_{src} in the source document d_{src} [27]. The task of plagiarism detector, as noted in [126], is to detect s by reporting a corresponding plagiarism detection $r = \langle r_{plg}, d_{plg}, r_{src}, d'_{src} \rangle$ where r_{plg}, is a passage identified by the detector as a plagiarized version of r_{src}. r is said to detect s if and only if $s_{plg} \cap r_{plg} \neq \emptyset$, $s_{src} \cap r_{src} \neq \emptyset$, and $d_{src} = d'_{src}$. The so called passage here could be in the form of sentences, segment of tokens in specific length or a sliding window of (non-) overlapping tokens or ngrams. Thus, the focus of plagiarism detection goes further till the level of passage. This differs greatly from former systems that was used to detect the duplicates of web sites documents and which measured similarity on the whole document levels [164, 193]. Figure 2.3 illustrates technical definition of plagiarism detection and plagiarism task.

The definition of plagiarism detection and its task presented before suggests that there are source documents from which the plagiarized passages are taken. In its development, some plagiarism detection approaches need no source documents and hence analyses a given document suspected as a plagiarized version. On the field of Automatic Plagiarism Detection, a document containing plagiarized passages is addressed as a *suspicious document*. Following the terminologies on this field, from now on, this thesis will use this term, suspicious document, to refer to one containing plagiarism sections. Based on its use of document collection as corpus, the approaches on plagiarism detection are categorized in two types, namely external plagiarism detection (EPD) which bases its detection on finding the source document [27], and intrinsic plagiarism detection (IPD).

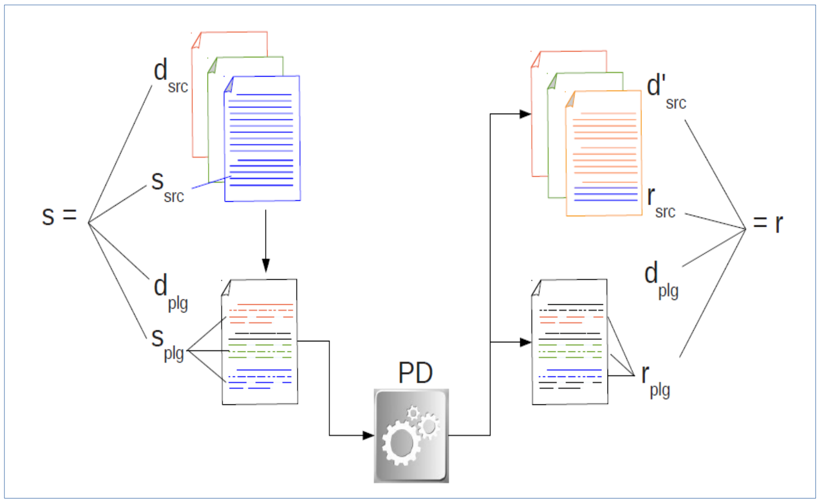


Figure 2.3: Technical definition of external plagiarism detection and its task

2.2.1.1 External Plagiarism Detection

The mechanism of external plagiarism detection is based on the fact that the sources of a plagiarism case is hidden in a large collection of documents [124]. For this reason, the EPD algorithm requires the availability of a corpus containing preprocessed documents that has been indexed correspondingly, and it works by comparing heuristically a suspicious document with each document in the source document corpus. In order to reduce the computational cost, Stein, Meyer zu Eissen, and Potthast [164] introduced a three-stage process of EPD that is illustrated in figure 2.4. Most of nowadays EPD algorithms follow this process which comprises retrieval process, detailed analysis, and post-processing.

The first step, heuristic retrieval, is meant to retrieve a small number of documents which are highly probable to be the source documents. Since the retrieval step compares a large number of documents, the retrieval models which reduce the retrieval dimensionality and computationally inexpensive are commonly applied. So far, most of External Plagiarism Detection (EPD) apply fingerprinting or sub-string matching in this stage [58]. The candidate documents outputted from the retrieval process are then extensively analyzed by performing passage-to-passage comparison between suspicious and candidate documents. The aim of this stage is to identify pairs of the possibly similar passages and to discard the rest of passages that are highly dissimilar. The knowledge-based post-processing analyzes whether the identical passages identified on the former step have been properly quoted [164]. This is to avoid the false positive detection on one hand, and the plagiarism offenses, on the other hand.

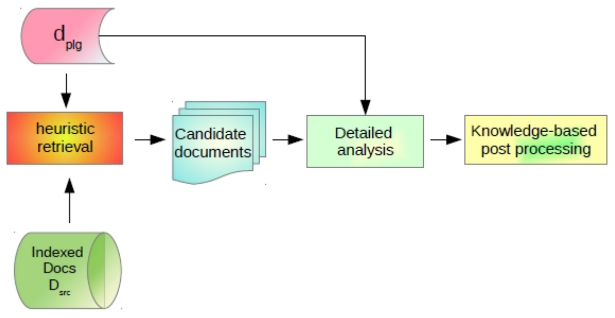


Figure 2.4: Three-stage process of external plagiarism detection
Source: [164]

2.2.1.2 Intrinsic Plagiarism Detection

The term *Intrinsic Plagiarism Detection* (IPD) was coined by Meyer zu Eissen & Stein in [48] to introduce a method of detecting plagiarism that does not require reference collection of potential original documents. The mechanism of this algorithm is based on the idea of portraying human skill in recognizing the copied parts of a text which are marked by a drastically or undeclared change on writing styles. The emergence of IPD is strongly related to Authorship Verification(AV). It can be viewed as a generalization of authorship verification and attribution [9], since the input to the IPD system is a document in isolation, and its task is to find the suspicious sections within that single document [48, 165]. Unlike IPD, an authorship verification system is given some pieces of writing examples of an author, for example author X, and its task is to determine whether or not a text is written by X. In term of similarities and differences between IPD and AV, Halvani in [69] summarizes that IPD is not addressing who the writer is as AV, but rather the suspicious sections. Besides, the context for IPD and AV is different, but they share slightly similar technical background.

The strategies in IPD approaches typically include the analysis of suspicious document, dplg's writing, that is well known as stylometry analysis. According to Stein et al. [165], the appropriate stylometric features for IPD fall into one of the following categories:

- **character-based lexical features (cblf)**: which deal with text statistics such as character n-gram frequency, frequency of special characters and compression rate.
- **Word-based lexical features (wblf)**: such as word length average, sentence length average, average number of syllable or words and term frequency word n-gram frequency.
- **Syntactic features (SynF)**: Part of Speech (POS), POS n-gram, frequency of function words and frequency of punctuation.
- **Structural features (StrF)**:average paragraph length, indentation, use of greetings and farewell, and use of signatures.

To make these features work, each feature category is assumed to be a set containing a finite number of distinct features, and the writing style is a union of all features sets [69] as seen below:

$$Style := cblf \cup wblf \cup SynF \cup StrF \tag{2.1}$$

In order to work out these features more systematically, Potthast et al [126] define a building block of IPD into four stages which comprise chunking strategy, writing style retrieval model, an outlier detection algorithm and post-processing. The chunking strategy is meant to define a boundary for feature extractions. The chunk length should be chosen in approximately equal size [69], otherwise it would influence the accuracy of the end result. The retrieval model is a model function that maps feature representations and their similarity measure. In some references, the retrieval model is also called as feature extraction [69]. The outlier detection attempts to identify chunks that are noticeably different from the rest. This is done either by measuring the deviation from the average document style or chunk clustering [126]. Most participants in the third international competition of plagiarism detection merged overlapping and consecutive chunks that have been identified as outliers in post-processing stage of IPD [126]. Undoubtedly, the end result of detections will not take form in quadruple as in EPD (see p. 23), but rather in a tuple of $r = \langle r_{plg}, d_{plg} \rangle$.

In comparison to External Plagiarism Detection, Halvani argues that Intrinsic Plagiarism Detection (IPD) is more difficult since there is no available reference document except the suspicious one. This leaves no further possibilities to uncover plagiarism case except to detect suspicious sections, and even if suspicious sections are found, there is still no guarantee that these sections are truly plagiarized [69]. But the emergence of IPD approach is to anticipate a case where the reference material is not always available or the amount of reference is tremendously large [48]. This makes IPD approaches increasingly important. However, researches on IPD have attracted less attention than EPD whose number and its method varies greatly. Since IPD approaches are beyond the scope of this study, the succeeding section will focus on reviewing methods applied in the state-of-the-art of EPD.

2.2.2 Outstanding Approaches on External Plagiarism Detection

So far, there have been two institutions which continually evaluate plagiarism detection systems, i.e. a research center at Hochschule für Technik und Wirtschaft (HTW), Berlin, and PAN [8] competition. HTW Berlin focuses on the evaluation of commercial plagiarism detection using their hand written test corpus [185], while PAN is aimed to conduct a benchmarking activity on uncovering plagiarism detection and authorship for the sake of promoting research and innovation in these fields. Most works submitted to PAN competition are in the form of research prototypes. PAN competition is held annually, provides standardized corpora for both source and suspicious documents, and evaluation measures as well. Softwares submitted to PAN competition include notebooks reviewing their applied approaches or methods. Unlike software submitted to PAN, commercial Plagiarism

[8]PAN is an acronym for **P**lagiarism Analysis, **A**uthorship Identification, and **N**ear-Duplicate detection.

Detection Systems do not usually reveal their methods, except the size of their databases. Thus, their approaches remain unreviewable. The following study on EPDs is mostly based on works submitted to 1st to 6th PAN competitions and will be organized as a three-stage process of most EPDs mentioned in section 2.2.1.

2.2.2.1 Source Retrieval for External Plagiarism Detections

Based on reports of PAN competitions and some commercial plagiarism detection systems, there have been two ways of retrieving the potential source document candidates, i.e. by comparing the suspicious document online or against the Web and offline or against in-house database. The online retrieval is not meant literally by performing real-time comparison of suspicious document against the web, but rather it is a simulation of online environment as found in 4th - 6th PAN competition [127–129], or by crawling the websites whose IP addresses are indexed in servers and identified as Internet Sources as in the case of TurnItIn [102]. The choice on comparing the suspicious document affects the building blocks of the retrieval subtask. The building blocks of retrieval process in a system which check suspicious document against its local database comprise: choosing document representations, indexing, measuring similarity or distance between suspicious and source document, and filtering. The building block of retrieval approaches for the so-called online EPD are defined in [127] and consists of five steps. They are chunking, keyphrase extraction, query formulation, search control, and download filtering. Surely not all online or offline EPDs follow rigidly these building blocks, some steps are sometimes skipped or merged for the sake of efficiency or its unnecessity due to the applied approaches.

2.2.2.1.1 Document Representation

In the retrieval subtask of EPDs, the comparison strategies affects greatly the option of document representation which inherently inlcudes the strategy of selecting document features and feature weighting. Several types of document representations have been proposed and most EPDs rely on one of the followings: Vector Space Model, fingerprinting, suffix data structure, or sets [58, 60, 163, 166].

2.2.2.1.1.1 Vector Space Model

In the field of Information Retrieval, vector space model (VSM) has become a widely known standard of document representation. In VSM, documents are represented as vectors of features. These features which characterize each document have values and correspond to a dimension in VSM. The process of encoding document as vectors brings a consequence of losing the relative order of terms. For this reason, VSM is included in the bag of words model which ignores the exact ordering of terms [97]. The idea is based on an assumption that two documents having almost similar bag of words have the same content and hence share similar topics. Sidorov et al. in [155] argue that the construction of VSM is somehow subjective because the researchers should decide which features or terms should be used, and which scale their values should have. The decision which leads to term selection strategy and term scaling will influence the performance of VSM in retrieving the potentially source documents in EPDs.

The strategy of selecting terms includes document preprocessing techniques and term unit selection. The standard preprocessing steps are case folding, eliminating non-readable characters, tokenization, stopword removal and stemming. In their implementation, some EPD systems ignore several steps of preprocessing and apply only tokenization as found in [19, 79], or combine some of these steps with the custom ones such as removing diacritics and converting all characters into US ASCII [80], normalize synonyms and abbreviations as in [63], or consider preprocessing as an unnecessary step as in [194]. In determining the term unit, there lies two main considerations: how to reduce the computation time in parallel to increase the recall of the potentially source documents. This consideration leads to various term units which basically could be classified into four groups, character n-grams, word n-grams, meta-term, and sentences. The character n-grams may take various lengths, such as 8 to 16 characters [65], while n in word n-gram may vary from 1 to 16 as the longest as found in [108], but word 4-6 grams are most widely used [79, 80, 114]. The use of metaterm as a feature unit can be found in [19] which converts a token into integers by using token lengths. All term lengths greater than 9 were cut to 9, so that the document becomes simply a sequence of numbers between 1-9. Then the metaterm 8-grams are extracted, weighted and indexed. The use of metaterms as a term unit has proved that they could increase precision and recall rates, and reduce the computation time on retrieval process [19]. The summary on the usage of term units could be seen in table 2.2.

Table 2.2: Term units in the VSM-based retrieval subtask

Term Unit	**Found in**
Char N-grams	[65]
Word N-grams	[63], [108], [114], [104], [28]
Metaterm N-gram	[19]
Sentences	[112], [54], [118]

The common representation scheme of scaling term in VSM is either weighted or non-weighted [26]. In weighted scheme, the weight of each term is based on the computation of its term frequency. At least there are two variants of frequency-based term weighting which are applied in Retrieval subtask of EPDS. The first is the well known tf-idf weighting which favors rare terms in the collection, and gives a high weight to key terms of a document. *tf* refers to the number of raw term occurences in a particular document, and *df* to the number of documents in which term t occurs. The *idf* weight is defined [97] as:

$$idf_t = log_{10} \frac{N}{df_t} \tag{2.2}$$

Tf-idf weighting is not only well known but it is also widely used as it can be found in [19, 65, 82, 108, 120]. Another weighted vector of terms is Relative Frequency Model proposed by Shivakumar and Garcia-Molina [152] which makes use of relative term frequency to compute the closeness set.

In non-weighted terms, the term is assigned a value either 1 or 0 depending on the term existence or non-existence, and thus it is identified as a binary vector. In this model, a documentis is represented as a sequence of terms or a binary vector $\langle e_1,...,e_M\rangle \in \{0, 1\}^{|V|}$, where |V| stands for the size of vocabulary in the document. Compared to the weighted vector representation, the binary vector is less popular in Retrieval subtask of EPDs as it was applied only in [63]. One advantage of using VSM for representing both source and suspicious documents is its flexibility to the mode of comparison, where suspicious document could be checked against the web directly or online simulation and against local database.

2.2.2.1.1.2 Fingerprinting

Fingerprinting has been the most popular model of representing document and applied both in retrieval and detection subtasks as well. In earlier applications, fingerprinting has been used to compute the overlap between pairs of web documents as found in the work of Shivakumar & Garcia-molina [153]. The idea behind fingerprinting is to perform efficient comparison by using a set of document features called fingerprints [26] rather than the whole features as in the case of string matching or VSM. In fingerprinting model, some document chunks are selected and converted into a set of integers or byte strings depending on its fingerprint function. Each element of this set is called minutia and a pointer to it is then saved in a hash table. A hash collision indicates redundancy of the minutia, and hence they are similar chunks. The concept of document fingerprinting is depicted in picture 2.5. According to Hoad and Zobel, the strength and efficiency of fingerprinting model lie on the tuning of its main parameters that cover four areas: substring size, substring encoding, substring number, and substring selection strategy [74].

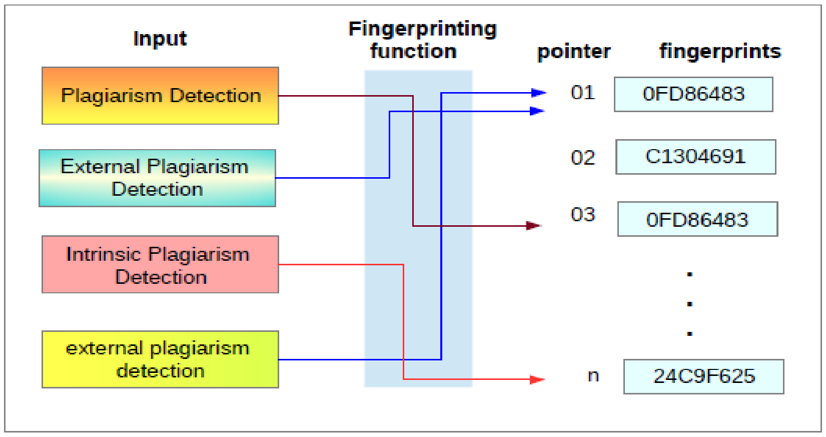


Figure 2.5: The concept on Fingerprinting

The substring size which has significant impact to the accuracy of fingerprints defines the fingerprint granularity [74]. The selection of fingerprint granularity should be considered carefully as fine granularity is sensitive to false matches, while rough granularity tends to generate fewer matches [24] and be vulnerable to change [74, 166]. In retrieval subtask of EPD, the granularity of fingerprint could be specified by the number of characters in a string and thus its unit takes the form of character n-grams [95]. Shcherbinin and Butakov used 50 characters as a unit size [150]. Other possibilities are by using word and sentences as units and their granularity is simply defined by the number of words or sentences. The word 5-grams becomes the most frequently used chunk unit [79, 194], though some systems apply variation between word 3- and 4-grams [10] or word 4- to 6-grams [80]. A sentence as a unit chunk is rarely found in current systems but in 1990s, Brin et al. hashed a sentence as the smallest chunk in their system called COPS [24].

The process of encoding the selected chunk unit into a minutia should satisfy the principle of fingerprint uniqueness [163] to address the problem of collision and the issue of reproducibility [74], that is every time a given string is processed, its output should be the same integers. For this reason, selection on hashing algorithm plays a significant role in affecting efficiency and effectiveness of fingerprinting methods [74, 137]. The popular MD5 hashing method is often applied in EPDs as it could be found in[79, 80], followed by Winnowing algorithm [145, 194], 64-bit Rabin fingerprinting [72], and shingling with 64-bit hash [10].

The substring number defines the fingerprint resolution, that is the number of minutia used to represent a document. In deciding the fingerprint resolution, it needs to consider the space required to store the index as there is a trade-off between fingerprint quality, processing time, and storage requirement [163]. Schleimer et al. in [145] noted that there are two methods of specifying the fingerprint resolution: fixed-size and variable-size fingerprints. The advantage of fixed-size fingerprints is that the system is more scalable as large documents have the same number of fingerprints as short documents. The disadvantages of this strategy are that firstly, only the near copies could be detected, and the comparison between two documents having totally different sizes hardly shows meaningful result [145]. In variable-size set of fingerprints, its number is determined mostly by the document length. By this technique, large documents have more fingerprints, and result in having greater possibility to match more queries [74]. As the idea on fixed-size set of fingerprints was proposed by Heintze, it was applied in his system with 100 chunks per document in database [71]. Most current EPDs apply variable-size set of fingerprints.

Defining the number of fingerprints to represent a document leads to a strategy on how to select them to this amount. Theoretically, Hoad and Zobel classified the fingerprint selection strategy into four: the full fingerprint, the positional selection, frequency-based, and structure-based selection [74]. In the full fingerprint, all fingerprints are saved and used to represent a document as found in [10, 80]. The positional selection strategy select fingerprints based on their positions in a document. In its implementation, it could be applied by choosing non-overlapping fingerprints of n-chunk size. The frequency-based selection strategy makes use of the number of fingerprint occurrences, while the structural-based selection strategy discriminates fingerprints on the basis of their occurrences in some specific string patterns at some specific positions. The frequency-based selection strategy is hardly found in current systems of EPDs, and the structural-based selection strategy was found in a system dated in 2003 [74]. Most current EPDs systems apply selection strategies that does not fall into one of categories mentioned before, or combine two or more aforementioned strategies. For example, Kasprzak and Brandejs [79] use the most significant 30 bits of the hash to identify the chunk, while hash with the least significant values in a chunk size could be found in [72, 150, 194].

2.2.2.1.1.3 Suffix Data Structure

Modelling document as vecors and fingerprints is commonly found in current EPD systems due to its less expensive computation and efficiency. Another model represents a document as an index structure that contains all suffixes of a document string. This model is closely associated with suffix tree which is constructed by considering all suffixes of a given text string as keys (or nodes) and the starting position of the suffix as values or leaves in a tree [52]. The comparison is done by matching the query pattern to these stored suffixes with leaves, and thus it allows an efficient matching in a linear time complexity. One of the arguments against suffix trees is the space requirement and structure that could occupy $O(n^2)$ space, if it is stored in a naive way [106]. A more simple and space-efficient alternative of suffix tree is suffix arrays proposed by Manber and Myers in [96]. Basically, it is an alphabetically sorted list of all suffixes of a string. The start position of the smallest suffix in the set is stored along with its string as in suffix tree. As Suffix Array stores n integers from the range [1:n] where n stands for the text length, it takes n words or n log n bits to store suffixes, and hence in its practice, it has proved to be competitive to suffix tree [52]

Both Suffix trees and suffix arrays make use of a set of characters as their unit, that is the pure document string without any manipulation. Unlike suffix tree and suffix array, suffix vector represents all substrings of a text t in a vector V(t). The vector is a mapping of the depth of a node in a suffix tree that is the number of characters being encountered from the root, the start position of the given node and its successive node [106, 134]. Suffix tree along with its alternatives (Suffix arrays suffix vectors) have been applied in detecting both source code and natural language plagiarisms as found in [16, 96, 106]. All of these systems were dated before 2002, and the efficiency they claimed did not match our nowadays concept of efficiency. Compared to fingerprint and VSM, all models belong to the suffix trees suffer from the storage space and searching time and result in an expensive computation. In a large document collection, the retrieval of potentially source document candidates requires document representation and algorithm that are computationally less complicated and more efficient. For this reason, this type of representation is unsuitable for Retrieval substak anymore but they are still useful in the text alignment or detailed analysis substask of EPD.

2.2.2.1.1.4 Stopword N-grams

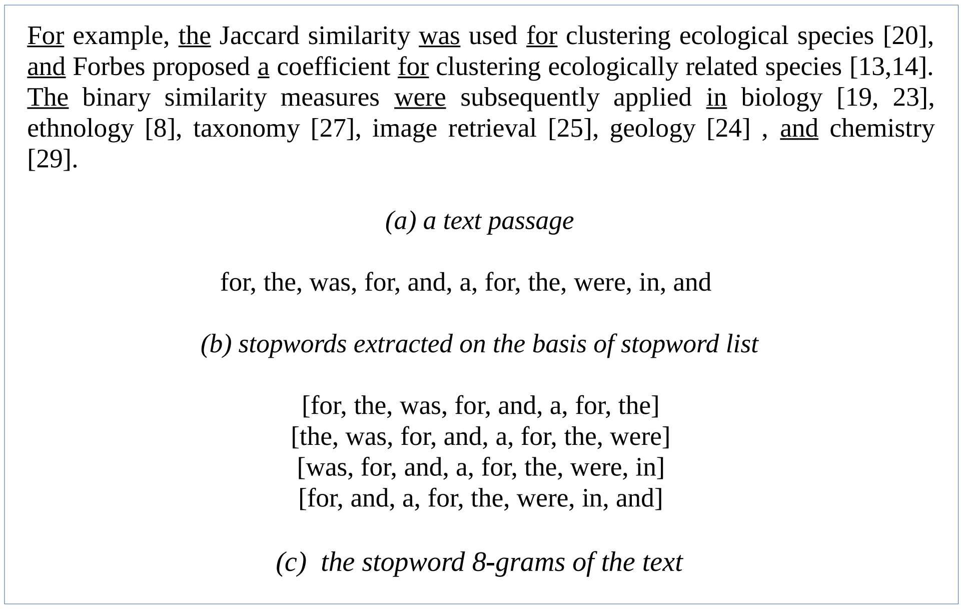
Stopwords are function words whose occurrences in a text are frequently high but they have a little value in a process of selecting documents [97]. In text processing, stopwords are often discarded to reduce the number of posting in the index. Somehow, Stamatatos in [160] saw a great potential of using stopwords to find passage similarities. His idea of utilizing stopword n-grams as document representations is based on the fact that stopword n-grams capture similarities on syntactic structure [160]. In a heavily disguised passage where words could be replaced by their synonyms or paraphrased, the stopwords would remain unaltered. Therefore, Stamatatos argues that stopword n-grams could be used as structural features of a document. Besides, an analysis on stopword n-gram pattern may reveal a writing style. So, stopword n-grams have been very useful in attributing authorship [160, 161] and detecting style for intrinsic plagiarism as well. Another reason of using stopword n-gram is that it is a reliable method for identifying the exact passage boundaries of two documents sharing similarity [160].

The first step in implementing this idea is to define a set of stopwords, how many stopwords are included in this set. Stamatatos defined 50 most frequent words to be included in this set [160]. The second step is to define the scope of text segment and N in n-gram. The text segment defines the length of context in which the Stopwords would be extracted, whether it is a paragraph, a section, or the whole document as a segment. Given a document, stopword list and a segment length, all stopwords defined in the stopword set are extracted. Finally, stopword n-grams with a length n are generated to construct a document profile P. Given a document d, the profile $P(n, d)$ comprises all the stopword n-grams. The stopword n-grams in P(n,d) are ordered according to their first appearance in the document [160]. The procedure of transforming a text passage into stopword 8-grams is displayed in figure 2.6 which is an adaptation illustration in [160]. The text passage is copied from [34]:

Stopword n-gram (SWNG) as document profile introduced by Stamatatos has gained researchers' attention as it has been applied in some research prototypes. In EPD, SWNG is mostly experimented along with other representations as can be found in [154] which applied SWNG along with name entity n-grams and word n-grams. Shrestha and Solorio [154] simply used a list of 50 stopwords defined in [160]. Kong et al. experimented multi-features in detecting high obfuscation plagiarism and inlcuded SWNG among other features such as character n-grams, word n-grams, and POS n-grams [85]. Different from [154, 160], Kong et al. used the top-7 stopwords in the list. As in former researches, Abnar et al. also used SWNG as an alternative features to other n-grams variations: word n-grams, expanded word n-grams, contextual word n-grams in Text alignment subtask [1].

2.2.2.1.1.5 Citation Patterns

Another possibility of representing a document in Eeternal Plagiarism Detcetion (EPD) is by using citation patterns which was proposed by Gipp [58, 60, 61]. Citation which is commonly addressed as *in-text citation* consists of two essential parts, the quotation and the source. The quotation could be in a form of direct copying from another piece of writing, paraphrased, or translated versions. The citation sources are very often written directly after the quotation or after the writer names. Mentioning citation source is one of requirements in academic and scientific writings which is aimed to give credits to the author of original concept [76]. For this reason, citation is always present in academic writings.



For example, the Jaccard similarity was used for clustering ecological species [20], and Forbes proposed a coefficient for clustering ecologically related species [13,14]. The binary similarity measures were subsequently applied in biology [19, 23], ethnology [8], taxonomy [27], image retrieval [25], geology [24] , and chemistry [29].

(a) a text passage

for, the, was, for, and, a, for, the, were, in, and

(b) stopwords extracted on the basis of stopword list

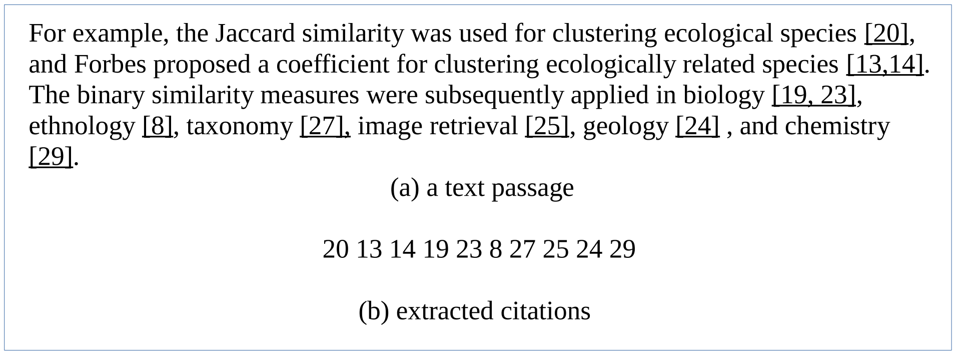
[for, the, was, for, and, a, for, the]
[the, was, for, and, a, for, the, were]
[was, for, and, a, for, the, were, in]
[for, and, a, for, the, were, in, and]

(c) the stopword 8-grams of the text

Figure 2.6: A text passage transformation into stopword n-gram profile. Firstly (a), all stopwords belong to the top 50 frequent words are extracte, then stopword 8-grams are generated (b).

The citation which is used to represent a text refers to the source of quotation and not to quotation itself. Gipp mentioned at least four reasons for using citation as document features, which are a) citation availability in academic texts enables its extraction as other features such as word n-grams [58], b) citation could be used as language-independent features [59], c) citation allows inferring semantic content or information [60, 103], and d) citation pattern indicates structural similarity [58].

This method which uses citation and references as document representations to determine similarities between documents is coined as Citation-based Plagiarism Detection (CbPD)[58, 59]. The framework of CbPD consists of four components: a document parser, a relational database - MySQL, a detector, and a web-based front end [58]. The document parser scans the text, identifies citation data, and extracts two different sets: a set of references or bibliography, and a set of citations. The citation parsing is done through an open-source tool called *ParsCit* [58]. Both references and citations are saved in a relational database. To illustrate, the citations extracted from a text passage is displayed in figure 2.7. In detecting passage similarity, these citations which are segmented in smaller chunks will be matched with those of another document. Preceeding the citation matching process, the probability of shared references of compared documents is computed using the extracted references. The computation of shared reference probability is based on the idea that two documents sharing the same references have a probability of the same topic, and hence they are related [61]. Only a pair of documents sharing references on a certain degree will undergo the comparison of citation pattern.


For example, the Jaccard similarity was used for clustering ecological species [20], and Forbes proposed a coefficient for clustering ecologically related species [13,14]. The binary similarity measures were subsequently applied in biology [19, 23], ethnology [8], taxonomy [27], image retrieval [25], geology [24] , and chemistry [29].

(a) a text passage

20 13 14 19 23 8 27 25 24 29

(b) extracted citations

Figure 2.7: A transformation of a text passage into citation pattern

As it is widely known, there are various styles of writing in-text citation. Unfortunately, there is no available information in all papers discussing CbPD on which citation styles are extracted by its parser, whether the parser is able to extract all, many or specific writing styles only. So far, there have been only one research paper reporting the implementation of citation pattern as document features besides those written by CbPD innovator and his colleagues. This paper reports the comparison of content and citation-based approaches with "the goal of evaluating whether they are complementary and if their combination can improve the quality of detection" [121]. Further it concludes that the combination of the methods can be beneficial.

2.2.2.1.2 Indexing

In Information Retrieval (IR), indexing is a process of building an index, which is a logical view where documents in a collection are represented through a set of index terms or keywords [29]. The goal of indexing is to speed the retrieval process of the needed information [97] by searching the index instead of the content of documents. The indexing process consists of three steps: defining the data source, transforming content of documents, and building an index [29]. The data source definition is done by a database module which specifies documents, and the operations to be performed on them [29]. The transformation of document content into their logical views was done through text operation which is based on the chosen document representation (cf. section 2.2.2.1.1). The index is created on the basis of this representation. Ceri et al. noted that there are different kinds of indexing structure, but the most widely used is the *inverted index* [29]. The inverted index consists of two basic parts: an index and a posting list [97]. The index which is also called as dictionary takes the form of terms or any other text representations such as fingerprintings or citations. Each term in the index has a list which states the document IDs where that term is found. Such list is well known as a posting list.

In External Plagiariam Detection (EPD), the need of indexing process is much influenced by the chosen strategy of retrieval task, whether the retrieval process is done offline by searching the database or online by searching the web. In an EPD system which does online retrieval process, indexing could be skipped in its architecture, since indexing has already done by the search engine. Indexing becomes necessary in a system doing offline retrieval. Unfortunately, there are many research papers which do not report their indexing process. In EPD systems, the indexing process cound be distinguished into systems which build their own indices from scratch and those which use tools for indexing. Among those which build their own indices, the inverted index still dominates the index structure as found in [79, 80], [19] which uses n-gram dictionary, or [58] which uses citation as its index and MySQL as its database management. Some EPD systems use the available tools for indexing and searching such as Apache Lucene [54, 108], SOLR Lucene [38], Indri [119], or Terrier IR system [112]. In $4^{th} - 6^{th}$ PAN PCs, the indexing is done by ChatNoir which indexed the ClueWeb09 corpus [127–129]. In some systems using fingerprinting as document representation, the source documents will be chunked first, then instead of document, the chunks will be indexed [79, 80, 108].

2.2.2.1.3 Query Formulation

If indexing is a process done to source documents, queries are formulated from a suspicious document to be matched against the index in the database. Like source documents, the content of suspicious document will be transformed into one of the document representations reviewed in previous section. Unlikely, queries will not be formulated simply from all features of suspicious document. The number and the length of queries will affect both the computation time and retrieval results in term of recall and precision rates. Similar to document representation, query formulation plays an important role in the retrieval subtask since it determines whether all or only some source documents are retrieved. Different from query formulation in Information Retrieal (IR) which is to match the availability of words or phrases in a source document, the queries in EPD is to match similar passages which have a broader scope. The challenges of query formulation lie on how to select features which will represent the hidden obfuscated passages in a suspicious document, to keep the smallest number of queries as possible, but they are able to match all potentially source documents. These challenges led to a query fromulation strategy which generally consists of 3 steps: Chunking strategy, keyterm selections, and query formulation. However, PAN's definition on retrieval building blocks for online EPD system breaks up query formulation process into four steps: chunking strategy, keyphrase selection, query formulation, and search control [128, 129].

The chunking strategy is meant to set a boundary for selecting keyterms which will be used to form queries. The chunking strategies applied in EPD system vary from word-based chunking which defines a chunk on the basis of the number of words such as 40 [108], 100, 200 words [132] to no chunking which sees the whole document as one chunk [49]. Other chunking strategies applied in EPD retrieval subtask are sentence-based, line-based, paragraph-based, heading-based, and text tiling with different length for its chunk unit. Some applications combine these chunking strategies such as Suchomel et al. in [170] which combine document-based chunk, sentence-based chunk, and headings which are used both as chunk delimiter and the basis of keyterm extraction. Haggag and El-Betagy use text-tiling to divide a document into topically-related chunks, and then segment each chunk into some pseudo-sentences [68]. Table 2.3 presents a summary on systems applying these chunking strategies.

Table 2.3: A summary on chunking strategies and systems applying them

Unit of Chunk	Found in
Document chunk	[170], [49], [83], [169], [38]
Line-based chunk	[49], [50]
Text-tiling	[68], [70]
Paragraph-based chunk	[83], [90], [169]
sentence-based chunk	[68], [188], [189], [84], [170], [169], [112]
Word-based chunk	[38], [108], [132], [169], [183]
Heading	[170], [169]

Strategies for selecting keyterms, addressed also as keywords or keyphrases [128, 129], could be grouped into two strategies: those which rely on weighting scheme and those that use available tools. Among the weighting schemes, tf-idf is quite often used as it is applied in the work of Elizalde which select top-10 words scored by tf-idf to be a query per 50-line chunk [50], or Kong et al. who combine tf-idf, tf, BM25, and Enhanced Weirdness (EW) to select top-10 phrases in a chunk [49]. Using no weighting scheme, Muhr et al. simply choose all words in a block of 40 tokens to feed as queries [108]. The open source tools applied in choosing keyterms are Python NLTK lemmatizer, KP-miner, and NLTK sentence detector. Suchomel and Brandejs combine NLTK lemmatizer and tf-idf weighting scheme to choose top-scored six lemmas [169], while Nawab and Clough use NLTK sentence detector to split documents into sentence, and use each sentence as a query [112]. Haggag and El-betany use KP-miner which returns the topmost keyterms which are supposed to be chunk characteristics and consists only 1 to 3 words [68]. A query is then formulated from these selected keyterms or sentences by defining number of terms per query. For example, Costajussa et al. formulate a query from 30 top-ranked terms for short suspicious documents, and 20 top-ranked terms for long documents. [38]. Prakash and Saha formulate 4 queries for each chunk, whereas each query consists of maximal 10 terms which are selected through their *documement level tf* and *paragraph level tf* [132]. Jayapal formulates a 10-word query from a chunk consisting of 4 sentences [77].

There is a slight technical difference for online and offline retrieval subtasks in term of query formulation. Due to a search engine constraint in accepting long query, a process of tailoring keyterms into an acceptable query length and feed it into an Application Programming Interface (API) of a search engine need to be done. Thus, a suspicious document may be represented by several short queries. Unlike the online system, the offline retrieval subtask is able to process a long query at once. However, no EPD systems uses all document features as a query. Based on the research reports, query formulation is mostly found on systems which retrieve candidate documents online or systems which use word and character n-grams, sentences, or any length of subtrings of a document as documement features. EPD systems using fingerprinting, citation paterns, stopword n-gram and metaterms generally skip the query formulation process, since mostly they base their candidate document retrieval on the computation of the shared common features (SWNG, fingerprints, etc) between those documents [19, 58, 79, 80, 160, 194]. To be exact, they develop diferent methods and algorithms for selecting and matching document features in a specific segment of documents.

2.2.2.1.4 Similarity Measures

The next step in an offline retrieval subtask is to measure similarity between a pair of source and suspicious documents by matching queries to indexed document features. In order to measure similarity, Bao et al. make a distinction between local and global information [18]. The local information could have been strings, substrings, word n-grams or any other forms of features which cover a specific area or segment of a document, while global information covers any kind of features whose scope covers the whole document such as word frequency or word vector [18]. Concerning the information scope conveyed in document features, Stein and Eissen introduce the distiction between local and global similarity [163]. Local similarity assessment approaches analyze matches of confined text segments by relating directly to its number of identical features [58, 163, 166]. An explicit example of local similarity assessment would be Jaccard coefficient which measures the identical features as the quotient between the intersection and union of features among two regions.

On the contrary, global similarity assessment approaches do not depend on the identical regions. They analyze characteristics of longer text segment, or the complete document, and express the degree of a document pair's similarity in their entirety [58]. Vector Space Model (VSM) along with Cosine similarity measure could be categorized as global similarity assessment because it quantifies the term frequency of the entire document and neglects the word order [163]. However, this local-global distinction is not fixed rigidly. The global similarity assessment such as Cosine similarity could be transformed into local similarity by changing its scope from a document into a section, a paragraph or a sentence. Similarly, Jaccard coefficient could be adjusted into a global similarity assessment by encoding the whole document as one segment.

The choice of the similarity measures is highly correlated with the choice of document representations. The global similarity such as Cosine Similarity is commonly applied for VSM, and most fingerprinting approaches tend to use local similarity such as Jaccard or their custom similarity measures, but fingerprint variants such as *simhash* computes a global vector of its variables for its feature weights [158]. In the retrieval subtask of EPD, some systems which expand the existing document representations tend to apply the available similarity or distance measures such as in [3, 178, 181] which apply Cosine similarity. However, systems introducing new concepts on document representations tend to introduce custom similarity or distance measures, or make some adaptations to the available measures. For example, Basile et al., who introduce the use of one single interger for a token representation and T-9 Match concept, adapt the Canberra distance normalized by the number of n-gram feature profiles in both documents [19]. Basile's n-gram distance measure is applied also in [168]. Stamatatos proposes to compute stopword n-gram profiles of two document regions by considering the number of stopword occurrences, stopword membership to its defined set, and the maximal sequence of words in which a stopword occurs [160]. Gipp uses *Bibliographic Coupling* which measures the number of similar references in both documents [58, 60]. There are systems which simply rely on the absolute number of common features such as in [79, 80] which require only 20 similar fingerprints. Besides the absolute number of similar fingerprints, Zou et al. define additional requirements such as the similar fingerprints should be successive and within the defined valid interval [194].

2.2.2.1.5 Filtering Source Candidate Documents

The last step in Retrieval subtask is to filter the computation outputs of document similarity or distance. The aim of filtering is to reduce the number of candidate documents and to save computation time during the detailed analysis process in the Text Alignment subtask by discarding documents which are not worthwhile being compared [128]. The filtering approaches applied in External Plagiarism Detection (EPD) systems could be differentiated into two groups based on the retrieval strategy applied whther it is online or offline. In systems doing online retrieval, the filtering is closely related to download strategy which is done for each query submitted to search engine. One suspicious document could be represented by several queries, where each query consists of 10 words or the maximal number of words a search engine could process in one session.

Several EPD systems select their candidate documents by ranking the result of similarity computation and selecting documents on the first n-rank, where n varies from 3 documents for each query submitted to a search engine as in [23, 84], 10 documents [19, 111] for the whole query, 10 documents for each submitted query [49], to 51 documents [65]. Other systems set up the minimum number of similar features found on source-suspicious document pairs to filter the candidate documents, such as having minimal 20 similar n-gram chunks [79, 80], 5 similar n-grams where n covers a large chunk as in [77], N similar and successive features where N is unspecified [114, 194]. Few systems use the similarity value as a filtering threshold as in [108] which discards documents having similarity values less than 8.0, or [63] which takes documents having the ratio of matching words over 0.5. The interesting filtering strategy applied in [189] compares the outputs of similarity computation with meta-file containing the annotated information on source-suspicious document pairs. If the outputted document IDs are listed in this metafile, then these documents would be selected as candidate documents. This is a tricky strategy that works for the sake of competition. Such strategy definitely will not work in a real case, because there will be no metafile informing source documents for a given suspicious one.

The filtering process marks the end of a Retrieval subtask in an EPD system. The filtered documents will be fed to the analysis module known as Text Alignment. To summarize the Retrieval subtask, table 2.4 presents a summary of Retrieval strategies for the first winners of $1^{st} - 6^{th}$ PAN competitions and some state-of-the-arts in EPD systems.

Table 2.4: Summary on the Retrieal Strategies of EPD State-of-the-Arts

Strategies/Found in	[65]	[19]	[79]	[63]	[56]	[68]	[84]	[160]	[58]
Doc Representation									
Term Units	Char 16-grams	meta-worn 8-grams	word 5-grams	word uni-gram	word uni-gram	word uni-gram	word uni-gram	top 50 stop-words	citation & refer-ences
Representation	VSM	VSM	fingerprint	VSM	VSM	VSM	VSM	stopword n-grams	citation pattern
Query formulation	NA	all fea-tures	all finger-prints	all fea-tures	top-10 terms	3 terms from KPMiner	top 10 key-words: tf-idf, tf, BM25, EW	top 50 stop-words	all refer-ences
Measures									
Term Weighting	Kernel Function: Linear & RBF	NA	Boolean weight	NA	Relative freq	NA	NA	-	-
Similarity/distance	Minkowski & Can-berra	Custom: n-gram ditance	Jaccard	Custon: binary similarity	custon: EW	ChatNoir	ChatNoir	custon: stopword n-gram profiles	BC
Filtering	top-51 docs	top-10 docs	docs$\geq$ 20 similar finger-prints	matching word ratio $\geq$ 0.5	NA	50% queries found in 500 char snippet of d_{src}	top 3 docs per query	min member $\geq$ 10, maxse-quence $\geq$ 10	similar references above thre

2.2.2.2 Text Alignment

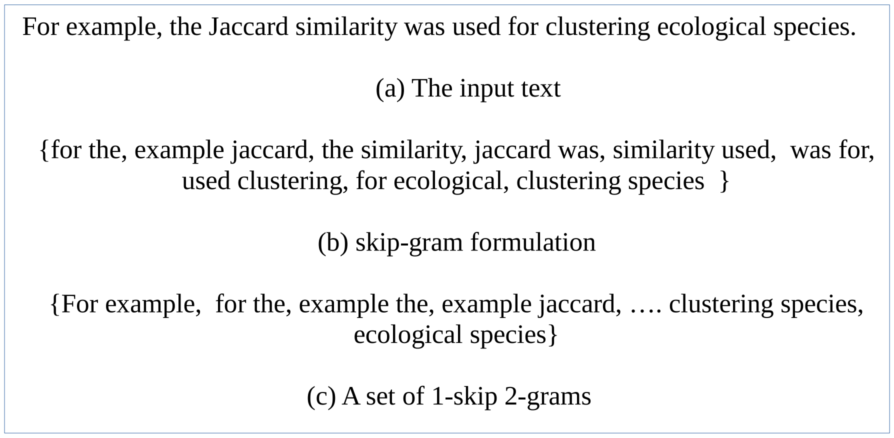
The term Text Alignment, referred as *detailed analysis* or *comparison* in former references [124, 164, 170], has been borrowed from Bioinformatic field which is used to match gene sequences. In EPD, Text Alignment (TA) analyzes further whether a suspicious document d_{plg} contained plagiarized passages from source candidate documents D_{Ret} which are outputted from retrieval subtask [127]. Given a suspicious document and a set of source candidate documents, the main task of Text Alignment is to identify all contiguous passages of reused texts between them [128]. The challenges of Text Alignmnet lie on how to identify passages of a text that have been obfuscated. Besides obfuscation types (cf. section 2.1.3), the obfuscation levels whether a passage is lightly or heavily modified intensify these challenges. In order to detect obfuscated passagess maximally, Text Alignment subtask is defined to be a three-step process: seeding, extension, and filtering [129].

2.2.2.2.1 Seeding

Being consistent in using Bioinformatics terminology, *seeding* refers to "matches" between d_{plg} and $d_{src} \in D_{src}$ using seeds. Seed heuristics, which are akin to document features in former references, are used to identify the match or similar passages either by exact matching or creating matches by changing the text in a linguistically motivated way [65]. In matching process, the aim is to match as many seeds as possible in order to build up larger similar text sequences. But the number of excessive seeds to match could turn out to be an ineffective strategy as the matching algorithm will fail to recognize the obfuscated passages in d_{plg}. Therefore, seeding strategy needs heeding as it determines the plagiarism types being recognized.

Preceeding seed computation, some EPD systems apply standard text normalization such as lower casing, removing non-readable characters, and stopword elimination. Text preprocessing such as stemming, lemmatization, parsing, POS-tagging, or sentence segmentation will be executed depending on units chosen as seeds. In the field of detecting Web page duplication, there are two family methods for feature computation: content-based and non-content-based methods [99]. The content-based methods use features found in document contents such as words, sentences, or paragraphs, while non-content-based methods rely on metadata such as HTML or XML structures. In seed computation, most EPD systems submitted to PAN PCs apply content-based methods. However, a handful of Text Alignment subtasks rely on pseudocontent-based methods. We call it pseudocontent-based menthod, as it uses a small list of features from document content, but they are hardly considered as part of document content in text processing. These features are stopwords and citations.

On the content-based method family, seed heuristics could be created purely from text strings such as character 16-grams [65], word 1-gram [62], word 2- to 5-grams [1], word 5-grams [154]; or they are sorted as in sorted word 3-grams [179], sorted word 4-grams [169], sorted word 5-grams [116]. Seed heuristics could take the form of word 5-grams containing at least one name entity referred as name-entity 5-grams [154], or they are selected according to its Part of Speech (POS) as in POS 3-grams [85]. Another technique for creating seeds is skip-gram which is a generalization of n-grams. In skip-gram, the components need not to be consecutive. A set of k-skip n-gram includes all consecutive n-grams in addition to k-skip grams. Figure 2.8 illustrates a building process of word 1-skip 2-grams. Examples for seed heuristics in skip-grams are word 1- to 4-skip 2-grams [64], 1-skip 3-grams [179], and k-skip n-grams while k and n are not clearly specified [116]. Moreover, seeds are also created using sentence pairs that exceeds a certain similarity threshold as in [83, 144]. Fingerprints could be used as seeds as in [57], or [7] which uses Rabin karp algorithm for matching the hash value of character 20-grams.


For example, the Jaccard similarity was used for clustering ecological species.

(a) The input text

{for the, example jaccard, the similarity, jaccard was, similarity used, was for, used clustering, for ecological, clustering species }

(b) skip-gram formulation

{For example, for the, example the, example jaccard, …. clustering species, ecological species}

(c) A set of 1-skip 2-grams

Figure 2.8: Example of skip-gram formulation and a set of word 1-skip 2-gram

Under the pseudo-content-based methods, seeds are created from top 50 stopwords, and the seeds take the form of unsorted or sorted stopword 8-grams [154, 160, 171] or where n is not clearly reported as in [85]. Citation pattern could be considered as seed heuristcis as it is used to do the matches. Gipp developed three different algorithms to create seeds from citation and to evalute their matches [60]. These algorithms are Longest Common Citation Sequence, Greedy Citation Tiling, and Citation Chunking which consider whether the seed order is preserved or ignored and whether the match is done locally or globally [58].

2.2.2.2.2 Seed extension

The next building block in Text Alignment subtask is *seed extension* whose aim is to merge previously found seeds into aligned passages. The basic idea is to present the whole passage rather than multiple chunks of separate seeds [180]. For example, word 3-grams would be extended to a sentence, some sentences, a paragraph or even to a section. So far, there have three approaches applied in seed extension algorithms of EPD systems. These approaches are rule-based approaches, clustering-based approaches, and dynamic programming.

Rule-based approaches become the most widely used in seed extension as it could be found on the the following works [57, 83, 85, 154, 160, 179]. In rule-based approaches, the algorithm encodes seeds along with their start and end offsets, and combines them under certain rules such as seed adjacency or gap among seed distances. Some extension algorithms do a two-step merge heuristics as in [7, 171]. The rationale behind the second merging process is to merge the overlapped or repeated matches which could occur as a result of the matching algorithm. For example, Suchomel et al. in [171] merge adjacent seed matches that are less than 4000 characters apart on the first step and merge again the resulting passages to another seed passage by checking if the gap between them contains at least 4 seeds. Alvi et al. execute the second step merging by defining relations between previously matched seed chunks [7]. If the relation stands between these passage chunks are overlapping and containment, then they will be definitely merged into one larger passage. Stammatatos uses rule-based approach by considering the SWNG profiles to set suspicious passage boundaries which are associated with big changes in consecutive values of matches of SWNG profiles [160].

Clustering-based Approaches have become an alternative approach to rule-based one lately. In general, clustering is applied to detect suspicious and source passages. Practically, each system applies clustering algorithm differently. Glinos applies 3-step clustering based on topic related words. The first step is *basic clustering* which is a hybrid of clustering and ruled-based approach, then *word clustering* which is used to determine whether the susppicious passages is a summary, and the last is *bigram clustering* which is to detect a pair of suspicious and source passages [62]. Gross and Modaresi use aglomerative single-linkage clustering to merge a pair of passage references that have minimal distances [64]. Palkovskii and Belov employ angled-ellipse-based graphical clustering algorithm to define clusters of shared fingerprints [117], while Sanchez-Perez et al. apply an algorithm relating to divisive clustering [144], and Abnar et al in [1] apply density-based clustering.

Dynamic programming is still a minor approach for seed extension in EPD systems, but at least there have been two systems applying this approach. One is proposed by Glinos as an alternative for the clustering-based seed sextension reviewed before. He employs Smith-Waterman algorithm which is extended and modified by providing a mechanism for detecting multiple alignments, methods for handling large documents and joining adjacent subsequences, and a similarity measure for comparing document features [62]. Another system uses algorithm from BLAST family which is borrowed from Bioinformatics field commonly used to align gene sequences [113].

The last building block in Text Alignment subtask is filtering. Based on our literary research, many EPD systems demonstrate almost similar techniques in filtering step of Text Alignment and post-processing stage in EPD pipeline. For this reason, the review on filtering process is presented under Post-processing section. To conclude, the summary on strategies and methods employed by systems in Text Alignment subtask are displayed in table 2.5, which summarizes only the methods and techniques applied in winning algorithms of $1_{st} - 6_{th}$ PAN competition on Plagiarism Detection.

Table 2.5: Summary on Text Alignment methods of EPD state-the-arts

phases/ in	[65]	[19]	[79]	[63]	[82]	[179]	[117]
Seeding	fingerprints from char 16-grams	T-9 Matches	MD5 hash of word 5-grams	word 1-gram	sentences whose cosine $\geq$ 0.42 & Bry-curtisian score $\geq$ 0.32	Contectual n-grams	fingerprints for word n-gram, SWNG, name-entity n-grams
Extension	rule-based: Monte-carlo optimization	rule-based: tuning-up 4 parameters	rule-based: valid interval	degree of concordance between tested passages	rule-baed: Bilaterla Alternating Sorting algorithm	rule based: merge-Sort algorithm	Angled ellipse based graphical clustering
Filtering	passage pair whose length $\leq$ 256 chars contiguity score $\leq$ 0.75	NA	overlapping detection removal	overlapping detection removal	passages with word overlap $\leq \theta$	NA	NA

2.2.2.3 Post-processing

In its introduction of a three-stage process for plagiarism detection analysis, Stein et al. conceptualize post-processing as a stage to analyze whether identical detected passages have been properly quoted to avoid plagiarism offense [164]. In its implementation, many systems consider post-processing as a filtering process whose task is to remove all passages which do not meet its criteria, as it is aimed to deal with overlapping passages in order to reduce false positives [7, 125].

Most filtering strategies rely on rules which are based on one of three approaches: minimum character lengths, number of words, or a threshold value based on the similarity or distance scores. Some filtering strategies combine these approaches. Under character-based filtering approaches, some systems set up different minimum character lengths for source passage s_{src} and suspicious passages s_{plg} such as in [7] which discards aligned passages, if $s_{src} \leq 200$ characters or $s_{plg} \leq 100$ characters. Some approaches simply discard aligned passages whose character length $\leq$ 150 characters [144], 300 characters [171], or 190 characters [116]. Another applied filtering approach excludes aligned passages whose number of words is less than 40 [62], or 15 words [64]. Some systems combine the similarity score with the minimum number of word to set up their filtering rule such as in [56] which removes passages containing less than 50 words and whose cosine score $\leq$ 0.75. However, Kong et al. rely on the word overlap score calculated by jaccard index to discard passage

Table 2.6: Summary om the characteristics of feature-based EPD systems

	language dependency	similarity dimension	efficient computation	Plagiarism cases		
				(near-) copy	paraphrase	summary
Content-based approaches:						
VSM	no	lexical	fair	good	fair	poor
Fingerprint	no	lexical	good	good	fair	poor
suffix-data struct.	no	lexical	poor	good	fair	poor
Pseudo-content-based approaches:						
SWNG	yes	structural, semantic	good	good	fair	poor
citation	no	structural, semantic	poor	good	fair	poor

pair, though the score threshold is not implicitly reported [83]. To summarize, the filtering step would be seen as an unnecessary step to the algorithm, but it is needed to present the output nicely. That is why in some systems, the filtering step is integrated in the extension phase [180].

2.3 Conclusion

The historical review on *plagiarism* proves that plagiarism conduct has existed since more than 2 thousand years ago and there has been a shift of its central meaning from copying text in literary field to the academic field in the 21^{st} century. In plagiarism scenario, the concession on the length of text reuse to be considered as plagiarism remains unclear. For this reason, each External Plagiarism Detection (EPD) system sets up its own definition on the length of aligned passages to be considered as a pair of source-plagiarized passages. The acceptable shortest pair is defined to have length of 15 words [64] or 100 characters [7]. Unfortunately, filtering the identified passage pairs by the citation sources has remained a challenge for EPD systems. Many of them simply ignore this process, despite the common view which states that the difference between plagiarized passages and non-plagiarized passages lies on the presence of citation.

Our study on the detection approaches shows that most EPD systems, even the state-of-the-art algorithms, measure similarity of documents and extract similar aligned passages on the dimension of lexical similarity rather than semantic or structural similarities. There have been efforts to capture document similarity on the semantic dimension [18, 35], but the trade-off between computational effort and detection accuracy resulted in the development of EPD algorithms which deal with the detection performance on the lexical level. In both Retrieval and Text Alignment subtasks, most EPD systems rely on the content-based approaches in selecting their document features and seeds such as substrings, string vectors or fingerprints. Few systems attempted to rely on pseudo-contend-based methods such as citation patterns or Stopword n-grams (SWNGs). Each of these representations or features has strengths and weaknesses. Table 2.6 summarizes the characteristics of each representation in terms of language dependency, dimension of similarity captured, computation efficiency, and their strengths and drawbacks.

The suffix-data structure and string matching turn out to be very accurate in detecting literal copy and near-duplicates, but its performance will decrease as the obfuscation level of copied texts increases. Other drawbacks of string matching are that they are attributable to exact macthing [58] and require high computational effort. Similar to suffix data structure, fingerprints and VSM are also very good at detecting verbatim and near-copies. Some fingerprint algorithms are capable of detecting moderate obfuscated texts. Besides, fingerprinting methods and other meta-strings such as T-9 match prove to be the most efficient features to compute. However, they become unsuitable features for systems applying online retrieval subtask. On the contrary, VSM which needs more computational effort is applicable for both offline or online retrieval subtasks. VSM strengths and weaknesses lie on its use of bag of word models. It turns out to be good at measuring similarity on the global level which signifies a major copy from one or two specific sources. But it shows low performance in detecting partial duplicates in which only a small portion of document is copied from a source. Another VSM drawback is that it is unable to handle medium to heavily obfuscated texts.

As a language-dependent feature, stopword n-gram (SWNG) is capable of capturing the lexical, structural, and even semantic similarity dimensions without applying semantic analysis. It is reported that SWNG is able to detect an extensive modification of a passage where most words are replaced but the structure remains [160]. However, it becomes apparently unable to detect plagiarism cases that have high word shuffling or in cases where the structure is highly obfuscated, unless the number of n is really small, but it will lead to a high false positive [125]. So far, there has been no researches reporting its performance when it is applied to detect texts in languages whose top frequent words have little role in forming the well-formed sentences such as in western Austonesian language family.

It has been reported that the strength of citation pattern as document features lies on its ability to detect disguised plagiarism, given the documents shared sufficient citations [103]. Its drawback is that it requires longer text segments containing more shared citations. Moreover, if the sources of copied texts are not listed in references and no citations referring to the sources, CbPD algorithm will definitely fail. Another challenge of CbPD is the extraction of various citation writing styles. Due to its limited scope of detection, it is better to use CbPD as a complementary method instead of the main one.

In conclusion, the literary research on External Plagiarism Detection systems has led to systems capable of detecting passage similarity between document pairs rather than detecting plagiarism. The similarity between these passages are supposed to be a sign of plagiarism presence. A human role is still needed to give judgment whether a text will be considered as a plagiarized version or not, since most EPD systems do not filter similar-detected passages through the presence of citations.

Chapter 3

An Overview on Indonesian and EPD for Indonesian

Since this thesis deals with a plagiarism detection system for Indonesian texts, the introduction on Indonesian language, its morphological and syntactic characteristics will be summarized in section 3.1. Section 3.2 reviews the previous works on the so-called plagiarism detection for Indonesian texts.

3.1 History of Bahasa Indonesia

As an official language of Indonesia, *Bahasa Indonesia* is spoken both as first and second language by approximately 240 million people. With this large number of speakers, Bahasa Indonesia becomes the 6^{th} most widely spoken language in the world [92, 136]. On its earlier phase, Bahasa Indonesia was partly an artificial language as its existence and formation have been established through some agreements in some national congresses. But in its development, Bahasa Indonesia changed to be a purely natural language due to its wide acceptance from people living in the archipelago called Indonesia today and its natural assimilation with vernacular languages. Furthermore, it has replaced some vernacular languages for the last 3 decades as more young generation become Indonesian native speakers and incapable of speaking their mother tongues anymore. The following overview on History of Indonesian proves this argument.

Triggerred by the need for a unifying language in the independence movement, the youth's vow, (*Sumpah Pemuda*) held in 28 October 1928, declared to have one national language, *Bahasa Indonesia*. The next question was "What is Indonesian?" since the archipelago had no common language before and the congress refused to use Dutch which signified a colony relation [41]. The congress agreed to choose Riau-malay as the root of Indonesian with two considerations:

- **Lingua Franca**: Riau-Malay which is the native tongue of people living in both sides of Straits of Malacca has been used as a *lingua Franca* for trading and commerce among the islands widespread in Indonesia, Malaysia, Singapore and Philippines for over than a millenium [110, 167, 176].

- **The simplicity of Malay Grammar,** Compared to other vernacular languages, had the pontential to unite people living under the Dutch colony into one nation.

Thus, Bahasa Indonesia was aimed to fulfill two functions: for building national identity and unity [110].

Following-up the youth vow, the first congress on Bahasa Indonesia which was held in 1938, decreed to form a Language Commission whose task was to create terms, to define the normative grammar of Indonesian and to systematically develop Indonesian as a nation-wide language of administrative and modern technology [123, 136]. In 1943, the Language Commission has successfully composed a list of 7000 new terms that were published in the Dictionary of Indonesian terms I & II during 1945-1947 [86, 123]. Among these entries, the contribution of the vernaculars, such as Javanese, Balinese, and Sundanese could be easily traced. Then, *Bahasa Indonesia* was proclaimed as the formal language of the new republic on the day of its independence, 17 August 1945. Being used in political and scientific affairs, Bahasa Indonesian could not rely much on the contribution of the vernacular languages any more, and thus it turned to borrow terminologies from Dutch, English, and Arabic. The significant achievement of the Language Commission is the success of composing 321.719 terms from various scientific fields in 1966 [86].

Another significant milestone of Bahasa Indonesia History occurred in 1972, when the Ministry of Education revised the spelling system and issued a book well known as 'perfected spelling' for Indonesian. Following the spelling revision, a book entitled *General Guidance on the Word and Terminology Building* was issued in 1975 which has become a guidance on how to build new terms not only for Bahasa Indonesia but also for Malaysian and Brunei-Malay [86]. Revised in [174], the book stated that the allowed sources for building new words and terminologies for Indonesian are:

- **Indonesian itself**. The common and archaic terms root in Riau-Malay. Some examples of Malay archaic words but have turned to be familiar again due to the enormous emergence of computer-related terminologies:

mangkus	effective (*English*)
sangkil	efficient (*English*)

- **Languages from the same family**. Belonging to Austonesian family, Indonesian is closely related to Javanese, Sundanese, Balinese, Buginese, Tagalog or Filipino, Maori, etc. bUt priority is given to the vernacular languages of the archipelago. Some examples of terminologies in Information Technology are:

Unduh (*Javanese*)	download (*English*)
unggah (*Javanese*)	upload (*English*)

- **Foreign languages**. The process of building new terms from foreign languages is referred as Indonesianization, which has been done through two processes: Adoption and Adaptation. In adoption, the terms are taken for granted as in *model, data, tutor, semester.* In adaptation, the spelling system is customized to Indonesian syllabification and Morphology such as in:

buku	book (*English*)
gereja	igreja (*Portuguese for church*)

> The adoption and adaptation process of terms are allowed only if the denotative meaning of foreign terms cannot be found in Indonesian and vernacular languages, and if the chosen foreign terms are more concise in form compared to its translation in Indonesian or local languages.

Kridalaksana in [86] noted that the tendency in choosing and accepting the new terms for building Indonesian vocabularies can be classified into the following processes:

1. **Nationalization** which is a process of enriching Indonesian terms by digging up the vocabularies of vernacular languages and archaic words of Riau-Malay.

2. **Internationalization** refers to a process of enriching Indonesian vocabularies which are accomplished by adopting and adapting vocabularies from foreign languages.

3. **Eastern classicism** is a process of enriching Indonesian vocabularies by adopting and adapting vocabularies from Sanskrit-rooted old Javanese, Sanskrit, and Arabic.

4. **Western classicism** refers to a process of enriching Indonesian vocabularies done by adopting and adapting vocabularies from Greek and Latin.

3.2 A brief Overview on Indonesian Morphology

Based on Indonesian history, Indonesian morphology unavoidably integrates the morphology of its vernacular languages as well. Basically, Indonesian words are formed through two morphological processess: concatenative and non-concatenative morphology operations [122]. The concatenative morphology regulates word building through affixation process, which is a process of glueing affixes or bound morphems to a free morpheme or stem. This process characterizes Indonesian as an agglutinative language [73, 176], though it is not as agglutinative as Turkish. The non-concatenative morphology operations combine morphems in a more complex way by reduplication and combination between affixation and reduplication [73].

3.2.1 Structures of Concatenated Morphemes in Indonesian

Basically there are two distinctive processes in concatenating morphemes in Indonesian. The first is through affixation process and the second process is to concatenate clitics and particles. Affixes in Indonesian could be classified into four categories: prefixes, suffixes, infixes, and circumfixes. These four affixes are distinguishable on the basis of their concatination positions. Indonesian clitics and particles are concatenated also into a base, but unlike affixes, they undergo slightly different concatenating rules. Figure 3.1 displays a rough structure of a word formed by concatenative morphology processes.

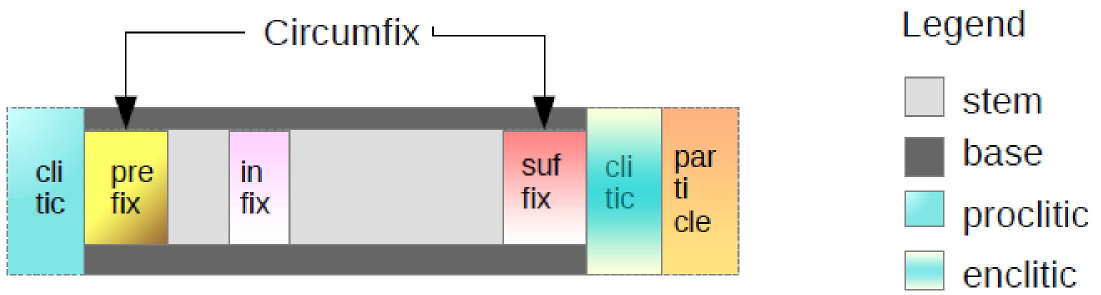


Figure 3.1: A word building structure through concatenative processes
Adapted from [139]

3.2.1.1 Affixes in Indonesian

Morphologically, a morpheme which is the smallest meaningful unit in the grammar of a language [9] is dintinguished into free and bound morphemes. A bound morpheme is a morphem that cannot stand alone, eg. affixes and clitics, while free morpheme is one that is able to function independently as a word. A free morpheme could be a stem which is a root. If a bound morpheme is attached to a stem, they form a word which could be a base for other affixes to concatinate. A base could consists of a root, or a root with its affixes. As implied before, all 4 types of affixes occur in Indonesian. Prefixes are bound morphemes which precede the base form or a root, for examples *ber-, di-, ke-, meN-, peN-, per-, se-ter-*. Infixes are bound morphemes which occur inside a root, i.e. *-el-,-em-, -in-*. Suffixes follow either a root or base form, eg. *an, -kan, -i*, and circumfixes, also known as confixes, wrap around the base. Circumfixes take a form of inseparable pair of a prefix and a suffix. Figure 3.1 displays also the position of each of these affixes. A complete list of Indonesian affixes is presented in table 3.1, which has been summarized from [109, 138, 156].

How these affixes occur in a word is governed by morphotactics which is akin to a syntax of a morpheme. Morphotactic rules represent the ordering restrictions in place on the ordering of morphemes. So far, morphotactics rules for Indonesian can be classified into 13 classes [8, 122]. Ten of these classes belong to concatenative morphology which outcast clitics and particles. These morphotactic rules regulate which affixes occur on the first or second order, and if they appear as second order affixes, which first order affixes are allowed to be their combination. Figure 3.2 illustrates the depth structure of affixes in a word building. Some example cases that belong to 10 classes mentioned before are as follows:

- Prefix *per-* which implies meaning as an intensifier verbs (VI) may appear either as 1st or 2nd order prefixes as in:

[9]definition by Glossary of Linguistic terms `http://www-01.sil.org/Linguistics/GlossaryOfLinguisticTerms/WhatIsAMorpheme.htm`

Table 3.1: List of affixes

Affixes	Noun Affixes	Adj Affixes	Adv & Adjunct affixes	Verb Affixes	Derivation of numbers	inflectional verb affixes
Prefixes	peN- per- maha-	PeN- se- tuna- antar-		ber- meN- per- ter- ke-	se- per- ke- ber-	ber- di-
Infix	-el - -er- -in-	-em-	-ah-	-em-		
suffixes	-an -wan -wati -man -anda -nda	-i -wi -iah -an	-an	-kan -i	-an	
Circumfixes	ke-..-an peN-..-an per-..-an	ke-..-an	se-..-an se-..-nya	per-..-kan per-..-i ber-..-an ber-..-kan ke-..-kan	ber-..-an	di-..-i meN-..-kan

per+tajam	pertajam (to sharpen)
VI + stem	tajam (sharp)
meN+per+tajam	mempertajam (to make sth sharper)
AV + VI + stem	AV: Active voice prefix

- The active voice (AV) prefix *meN-* appears always as a first-order prefix. If it precedes another prefix, it can be combined only with intransitive verb (ItrV) prefix *ber-* or imperative circumfix (IC) *per-..-kan, ke-..-an* and may be combined with imperative suffix (IS) *-kan* as in:

meN+sapu	sapu (a sweeper)
AV + stem	menyapu (to sweep floor)
meN+per+satu+kan	satu (one)
AV + IC + stem + IC	mempersatukan (to unite)
meN+ber+henti+kan	henti (stop)
AV+ItrV+stem+IS	memberhentikan (to fire sb from a job)

The concatenating rule of most affixes are simple and done by merging these affixes into a stem or a base with little exception rules. However, the prefixes and circumfixes with variable *N* undergo a morphophonemic processes. It is a process of change which is

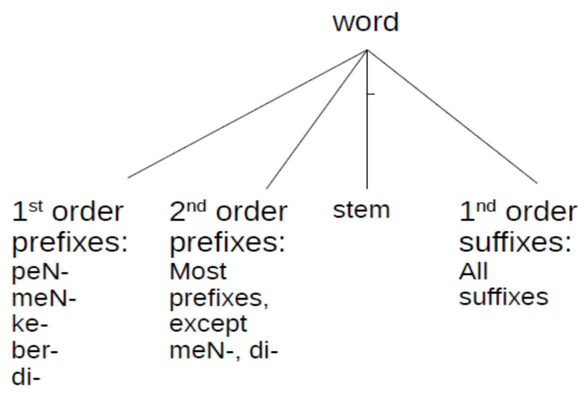

word
1st order prefixes: peN- meN- ke- ber- di-
2nd order prefixes: Most prefixes, except meN-, di-
stem
1nd order suffixes: All suffixes

Figure 3.2: Affix orders in a word building process

conditioned by the initial sound or phoneme of a base [41]. In Indonesian, morphophonemic rules can generally divided into two: rules modeling phonetic changes in stems and rules which model phonetic changes in affixes [122]. Darjowidjojo in [41] listed 9 morphophonemic rules while Pisceldo et al. [122] defined 11 morphophonemic rules, 4 rules belong to the first group and 7 rules deal with the second group. Two of morphophonemic rules belonging to the first group are:

- If meN- or peN- are attached to a base started with /k/, replace /k/ by /ng/ and drop N in peN- or peN-. Example: meN+kantuk → mengantuk.
- If meN- or peN- are attached to a base started with /t/, replace /t/ by /n/ and drop N in peN- or peN-. Example: meN+tertawakan → menertawakan.

The complete morphophonemic rules could be found either in [167] or in [8, 138], while two examples of morphophonemic rules representing the second group are as follows:

- /N/ is replaced by /n/ if there is meN- followed by /d/, /c/, /j/, /sy/ or there is peN- followed by /d/, /j/, /c/. Examples: meN+duduk+i → menduduki; peN+jual → menjual.
- /N/ is replaced to /nge/ if before one-syllable base. meN+cat → mengecat.

Suffixes and infixes remain uninfluenced by the morphophonemic processes. There is only one-order position for suffixes: the first-order suffix or if it poses the second-order-ending, the first-order ending will be posed by clitics or particle. The infixes are inserted after the first consonant of a base. Some examples are:

-in-+ kerja	Noun Infix (NI)	kerja (to work)
NI + stem		kinerja (performance)
-ah- + dulu	formalizing-infix (FI)	dulu (past/ago)
FI + stem		dahulu (formal form of *dulu*)
-an + makan	noun suffix (NS)	makan (to eat)
NS + stem		makanan (food)

Morphologically, affixes as bound morphemes are categorized into dervivational and inflectional morphemes. Derivational morpheme is defined as morphemes which create a new word or a word having different grammatical category from its stem, while inflectional morphemes are used to indicate aspects of the grammatical function of a word and never used to produce new words [191]. There are only a handful of inflectional morphemes in Indonesian as displayed in table 3.1. Most of Indonesian affixes could be classified into derivational morphemes. However, there is no clear cut distinction between inflectional and derivational morphemes in Indonesian as argued by Pisceldo et al. [122] in the following cases: from the stem *pukul* we could derive words like *pemukul, memukuli* and *pukulan.* The formation of *memukuli* seems to be 'inflectional', and the formation of *pemukul, pukulan* is derivational because the derived words are nouns and have different meaning from their stem. However, *memukuli* is argued to be derivational [122] as it has quite different lexical properties from its stem, though both *pukul* and *memukuli* are from the same category, i.e verb.

3.2.1.2 Clitics and Particles

A clitic is a morpheme that has syntactic characteristics of a word, but shows evidence of being phonologically bound to another word [10]. Most clitics are syntactically free, have grammatical rather than lexical meaning and are usually attached at the edges of words. however, Indonesian clitics show slightly different characteristics as they occur both at front of a base that is called proclitic, and at the edge of words as enclitic. They have different lexical meanings and grammatical functions. The proclitics *ku-, kau-* are replaceable with their free morphemes if they are used in formal discourses. Anyhow, the enclitics *-mu, -nya* as possessive pronouns will produce peculiar sense of meaning if they are replaced by their free morphemes. The summary of Indoensian clitics and their functions are given in table 3.2.

There are four particles only in Indonesian *-lah, -kah, pun, per.* Two of them are recognised as foregrounding particles, i.e. *-lah, -kah.* Both particles are always attached to the preceeding words. Article *pun* will be attached to the word only to the following 12 words: *adapun, andaipun, ataupun, biarpun, kalaupun, kendatipun, maupun, meskipun, sekalipun, walaupun, sungguhpun* [156]. These words are considered as one word with one meaning. The occurrence of-*pun* in other words will be written separately. Particle *per* which means start, every, and for the sake of is written separately. This article is an adoption of English preposition *per.* Table 3.3 summarized the function of these particles [40, 139, 156].

3.2.2 Non-concatenative Word Building

The non-concatenative morphological process of building a new word in Indoensian takes the form of a reduplication which is a productive process in Bahasa Indonesia as it is

[10]This definition is taken from GLossarx of Linguistics terms, `http://www-01.sil.org/linguistics/GlossaryOfLinguisticTerms/WhatIsACliticGrammar.htm`

Table 3.2: A List of Indonesian clitics and their functions
source [139]

Functions	Proclitics	Free morphems	Enclitics	Functions
subjective pronoun, 1st person singular (1st sing)	ku-	saya, aku	-ku	possessive pronoun (PP), 1st sing objective pronoun, 1st sing
subjective pronoun, 2nd sing	kau-	kamu, anda	-mu	PP, 2nd sing objective pronoun, 2nd sing
		ia, dia	-nya	pp, 3rd sing subjective pronoun of passive verb object pronoun of active verb definite article

Table 3.3: A list of particles and their functions

-kah	question marker
-lah	imperative marker predicative marker predicate negation cooccurrence with *pun*
(-)pun	focusing adjunct balance & antithesis marker
per	meaning: resume, start every, each for the sake of

readily applied to many stems [105]. It is used in inflections to express various grammatical functions sunch as plurality, intensifier, etc. and in lexical derivation to create new words [51]. In reduplication, a root or stem of a word or even the whole word is repeated exactly or with a slightly morphological change. There are three types of reduplication in Indonesian, i.e. full reduplication, partial reduplication and imitative reduplication [156].

A full reduplication involves repeating the entire word where two parts are separated by a hyphen. The productive process of full reduplication can be distingushed into four types [51, 105, 138]:

- *Reduplication of free bases* in categories of noun, verbs, adjective, pronoun, and numbers. These types of reduplication expresse plurality for nouns, or action done carelessly for verbs, and *although* for adjectives under certain contexts. eg:

baca-baca	sakit-sakit	dua-dua
read-read	sick-sick	two-two
"reading for fun"	"though being sick"	"each two"

- *Reduplication of stem with affixes*:

membunuh-bunuh	bunuh-membunuh
AV+kill-kill	kill-AV+kill
"killing"	"kill each other"

- *Affixed reduplication* in which confixes are attached to the reduplicated words. In the following examples, the intransitive-circumfix (ItrVC) *ber-..-an* and adjective circumfix *ke-..-an* change both the semantics and the word categories of their stems:

bersakit-sakitan	kekanak-kanakan
ItrVC+sick-sick+ItrVC	AdjC+child-child+AdjC
"work very hard"	"childish"

- *Reduplication without corresponding single bases.* Sometimes the reduplicated words have no unreduplicated counterpart to which they can be related [51]. These words are treated as bases in dictionary:

kupu-kupu	mega-megap
butterfly	to pant

Partial reduplication occurs only with bases which begin with consonant. It involves placing before the base a syllable consisting of the first consonant of the base followed by vowel *e* [156]. This type of reduplication is no longer productive in the language. The meaning of partially reduplicated word cannot be generalized, but it is connected with its stem. In some cases, it has no relation at all as in:

tangga	tetangga	luhur	leluhur
ladder	neighbour	noble	ancester
tua	tetua	tapi	tetapi
old	elders	but	but (formal form)

In **imitative reduplication**, two parts of the word are not identical, though they are similar [156]. The variation between two parts of the word can involve either consonants or vowels. Frequently the first component of the word occurs as a simple stem. Nouns, adjectives, and verbs can undergo this type of reduplication. The first sets of examples given below show variations in consonant, while different variations in vowels are given on the second set examples:

lauk	lauk-pauk	cerai	cerai-berai
dish	side dishes	separated	scattered
balik	bolak-balik	tindak	tindak-tanduk
return	to and fro	action	behaviour

3.3 A Brief Overview on Indonsian Syntax

While the core issue in morphology of a language is how to build a word, syntax deals with how to compose these words into longer sequences which are called sentences. A sentence is a grammatical unit that is composed of basic constituents completed with its intonation [31]. With this definition, the scope of syntax is really wide. Since this section is aimed to give an overview on Indonesian syntax and not to analyse it, the syntax description presented here is restricted to one topic only: word order which becomes a main feature to align Indonesian to English or any other languages. The fact that Indonesian word order is inseparable with its voice, a short overview on the up-to-date theories on Indonesian voice system would be presented also.

3.3.1 Word Orders and Grammatical Relations

As in many languages, the basic sentence structure in Indonesian generally consists of two immediate constituents, i.e. one subject and one predicate. Subject position is usually occupied by a noun phrase or a pronoun phrase which precedes the predicate immediately. However, some sentences consist only predicates as immediate constituents, which Sneddon et al. [156] call subjectless clauses as shown in (1). Some linguists such as Furihata in [53] address sentence (1) to have a verb phrase as its grammatical subject. Disregarding different theories addressed to it, sentence (1) poses a verb marked by a passive prefix (PV) *di-* and the Adjective *dapat* is marked by a predicative particle *-lah*.

(1)	Dapat-lah	di-simpul-kan	bahwa	serang-an	itu	telah	di-rencana-kan.
	able-Pred	PV-conclude-kan	that	attack-N	the	PastADV	PV-plan-kan.

'It can be concluded that the attack has been planned'.

Predicates have an important role in Indonesian basic structure. Unlike western European languages, a predicate in Indonesian is formed not only by a verb or verb phrase but also by adjective, nominal, prepositional or numeral phrases [31, 53]. The predicative phrase category is used for naming the sentence, and thus there are verbal, adjectival, nominal or prepositional sentences in indonesian. There are copulas such *adalah* or *ialah* which is comparable to *to be* in English: is, am, are, and which can be combined with non-verbal predicates. But the use of these copulas is optional. In contrast to western European languages, the absence of verbal predicate, including copulas, does not turn these sentences into ungrammatical ones. Sentences in (2)a-d exemplify cases of those with non-verbal predicates. Thus, a sentence completed with a copula and an indefinite article such as "Ibunya adalah seorang dokter gigi di puskesmas itu" can also be expressed as in (2)a. Both versions are grammatically acceptable, well form, and correct.

(2) a. Ibu-nya dokter gigi di puskesmas itu.
Sbj Pred
Mother-3POSS doctor tooth in health center that.
'Her mother is a dentist in that health center.'

b. Kucing-mu kurus sekali.
Sbj Pred
cat-2POSS skinny very.
'Your cat is very skinny.'

c. Gaji-nya se-juta se-bulan
Sbj Pred
salary-3POSS one-million one-month
'His/her salary is one million a month.'

There are two opposing views concerning the subject-predicate order in Indonesian sentences. The first group represented by Chaer [31] and Müller-Gotama [109] views that Indonesian sentences are characterized by stringent word order. Disapproving such opinion, the second group proves that a kind of scramblings exists in Indonesian. Among those in second group are Chung [36] and Gil [55]. Müller-Gotama argues that Indonesian is a consistent head-initial language with a basic Subject-Verb-Object (SVO) word order. Topicalization and passivation may give variation on the word order, but they will not affect the order of grammatical subject (gr-subject) and its verb [109]. He proves his arguments by providing the following sentences and claimed that (4) is unacceptable:

(3) Saya mau beli pakai-an di pasar baru minggu depan.
1sg want buy wear-N in market new week front.
'I want to buy clothes in Pasar Baru next week.'

(4) ?? Saya mau beli pakai-an minggu depan di pasar baru.
1sg want buy wear-N week front in market new.
'I want to buy clothes next week in Pasar Baru.'

However, Gotama bases his arguments on sentences with verbal predicates only, the verbs are in the stem form with no affixation at all, and thus they are unchangeable. Furthermore his perspective is driven by his concern on Indonesian-English sentence alignment. This makes him fail to see that Indonesian sentences can occur without subjects and non-verbal predicates as presented earlier. The case will be different if affixes or particles are attached to the verbal predicates, and the perspective is centered to Indonesian sentences per se, without bothering their equivalences in English. As native speaker, I do agree with Chung and Gill that the subject-predicate order could be switched. Just one example is to attach particle -lah to a verb stem. In such cases, verbal predicates are allowed to precede gr-subject as in sentences (5) a-b.

(5) a.

Di sini	hatiku	hancur.	Me-nangis-lah	saya	dengan	sangat	sedih.
Here	heart-1POSS	broken.	AV-cry-lah	1sg	with	very	sad.

'Here my heart was broken. I cried bitterly.'

b.

Pada	hari	itu	terciptalah	suatu	negara	Indonesia	merdeka.
On	that	day	PV-create-lah	a	country	Indonesia	independent.

'On that day Indonesia has become an independent country.'

In sentence (5)b, the English equivalence would be expressed better in active voice, though in its original version, it takes a form of a passive voice. Without particle-lah, the sentences above have the normal subject-predicate order as shown in (6). The indefinite article suatu refering to *'a'* could be eliminated when it precedes verbs such as shown in (6)b.

(6) a.

saya	menangis	dengan	sangat	sedih
1sg	AV-cry-∅	with	very	sad.

'I cried bitterly'

b.

Negara	Indonesia	merdeka	tercipta	pada	hari	itu.
Country	Indonesia	independent	PV-create-∅	on	that	day.

'Indonesia has become independent on that day'

The order of basic constituents which consists of subject and verbal predicates in Indonesian could not be simply defined as S-V-O or V-S-O. This could be seen on Gil's comment as follows: "If it had verbs, one might say that it was a verb-initial language, though word-order is probably more flexible than many other verb-initial languages. If it has subject and objects, one might wonder whether verb-subject or subject-verb order. Object may occasionally precede the verbs, though much less frequently than subjects" [55]. The case of word order becomes more complicated if it deals with voice aspects. In solving problems of analyzing Indonesian voices, Arka and Manning in [14] shed light implicitly on the problems of word orders which will be presented in the following section.

3.3.2 Voices in Indonesian

In Lingusitic terminology, *voice* is used to refer to a grammatical category that expresses the semantic functions attributed to the referents of a clause. It indicates whether the subject is an actor, patient, or recipient [11]. Recent studies in Linguistics show that Austronesian

[11]definition by Glossary Of Linguistic Terms available at `http://www-01.sil.org/linguistics/GlossaryOfLinguisticTerms/WhatIsVoice.htm`

languages are renown for their highly developed voices which are generally richer than those encountered in Indo-European languages such as English which shows only two-way systems: active-passive alteration [13, 141]. As part of western Austronesian languages, Indonesian poses the unusual nature of voice systems which have led to controversy in linguistics. Let us consider the following sentences:

(7) a. Aku akan menanam pohon mangga itu.
1sg FUT AV-plant mango tree that
'I will plant that mango tree.'

b. Pohon mangga itu akan ku-tanam.
tree mango that FUT 1sg-plant
'The mango tree, I will plant.'

c. Pohon mangga itu akan di-tanam (olehnya).
tree mango that FUT PV-plant by-3sg
'The mango tree will be grown (by him).'

In example (7)a, it is clear that the gr-subject is the actor. Hence, it can be labelled as an active voice. sentences in (7)a & b exhibit non-actor gr-subjects. Arka in [13] claimed that one of the non-actor voices, marked with *di*-verb plus a pronominal (PP) agent as exemplified in (7)c, can be analysed as a true passive equivalent to the English passive voice, because its patient argument appears as gr-subject and the agent or actor is grammatically optional (marked by the brackets). As for sentence (7)b, there is a vagueness on how to address it. Some grammarians would align and translate such sentence into English with active sentence, but analyze it as a passive one. Among these are Chung [36] and Alieva [6]. The traditional grammarians would address sentence (7)b as an active-voice, as an alternative form of (7)a with a proclitic *ku-* as its gr-subject.

Based on binding theory, Arka and Manning in [14] analysed sentence (7)b to expose a specific voice which cannot be categorized as both active or passive voices. The rationale is that firstly *di-* verbs, which is a passive marker in Indonesian, cannot bind a non-third-person agent. Transforming (7)b into a passive form will make it ungrammatical as shown in (8). Secondly, in transforming AV to PV, it is required that the actor as gr-subject in AV becomes an oblique object or well known as logical subject (l-subject) [12] in PV. This requirement could not be applied also to (7)b, since proclitic and enclitic cannot be an oblique l-subject but still be a 'term/core arguments' [14]. This is proved by (8)b and (8)c.

(8) a. * Pohon mangga itu akan di-tanam oleh-ku.
tree mango that FUT AV-plant by-1sg
'The mango tree will be planted (by me).'

b. Andi me-nyapa-ku/-mu/-nya.
Name AV-greet-1/2/3sg

[12]A logical subject is the constituent which is the 'doer of the action', the constituent that actually carries out the process, but not as gr-subject [47]

'Andi greeted me/you/him.'

[*] 'I/You/He greeted Andi.'

c. * Obat itu di-minum-ku/-mu .
medicine that PV-drink-1/2sg

The medicine is taken by me/you

Arka and Manning further show that the 3^{rd}-person pronoun suffix in the *di*-V-*nya* is not oblique. For that reason, they rejected cases shown in (7)b to be analysed as a passive or active voice, instead labelled it as an undergoer voice (UV). Adopting Arka and Manning's view, Riesberg provides important evidence that pronouns and proclitics immediately preceeding stem verbs are indeed undergoer voice construction [141]. She suggests that the *di*-V constructions do not form a uniform class but belong to two different voices. Further, she summarizes that at least Indonesian exhibits three voices as shown in table 3.4.

Table 3.4: Voice marking in Indonesian
source [141]

Active Voice	**Undergoer Voice**	**Passive**
meN-V	di-V-nya	di-V-PP
meng-V	pro-V	di-V-NP

As Indonesian voice system is beyond the scope of this study, for further study on the unique cases of Indonesian voices, please see [141] or [13, 14]. Coming back to the topic of word order for basic constituents in Indonesian, we could add a word order pattern which have not been mentioned either by Müller-Gotama [109] or Gill [55] as reviewed in section 3.3.1. Adopting Arka and Manning's view on cases (7)b which addresses proclitics as undergoer, we can then assign a syntactic function as object to these proclitics and pronouns. The gr-subject and verbs are quite clear in case (7)b, in which the gr-subject 'Pohon manga' occurs at the begining and the verb at the end of sentence. Then, we have a subject-object-verb (SOV) constituent order here. Thus, it can be concluded that the undergoer voice contributes a S-O-V order for Indonesian sentences.

3.4 Former Works of Plagiarism Detection for Indonesian Texts

Researches on Plagiarism Detection for Indonesian texts have not been well developed as those done for western European languages such as English, German or Spain as presented in the former chapter in section 2.2.2. The situation is worsened by the fact that some Indonesian researchers prefer experimenting their algorithms on English texts rather than on Indonesian for many different reasons. One reason is its unavailability of standardized corpus for evaluating the algorithm. This section concerns more on methods and techniques applied in researches on Plagiarism Detection done by Indonesians independent of languages of the texts. Most papers surveyed here deal with External plagiarism detection (EPD), only one deals with cross-language plagiarism detection (CLPD). The review on the evaluation corpus building will be presented separately and could be found in section 5.1.1.1.

Researches on EPD systems done by Indonesians could be distinguished into two groups:

- researches which detect plagiarism by applying Stein's three-stage architecture, or at least which try to find and locate the supposedly plagiarized parts, and
- researches which perform document comparison directly.

Researches in group 2 tend to compare and measure similarity on the document level, rather than to find and locate the common passages or sections of compared documents. For this reason, researches on group 2 will be addressed as researches on near-duplicates instead of plagiarism detection. From 16 surveyed papers on PD systems, 31.25% of them belong to the first group, while the majority, 68.75%, belong to the second group or near-duplicates. The review on following sections is based on this group division.

3.4.1 Researches on Near-Duplicates

Duplicate and near-duplicate documents are practically a form of literal plagiarism. In section 2.1.3, duplicate is addressed as *copy and paste*, while near-duplicates is renowned also as *shake and paste*. However, there are slightly different methods and algorithms for detecting duplicates and near-duplicates in one group and plagiarism detection in another group. The algorithms for detecting plagiarism are required to be able to find, locate and extract the common passages or sections between two documents being compared. In duplicates and near-duplicate systems, the algorithm tends to measure the similarity of the compared documents globally. It must not refer to the exact location of similar passages, insteads, it provides simply the similarity percentage between source-suspicious document pairs. Another generalization that is derivable from researches on duplicates and near-duplicates is that many of them use various fingerprinting techniques as the document representation [91, 97]. Using these as basis criteria, 10 out of 16 papers report to detect duplicates and near-duplicates.

3.4.1.1 Document Representation

In term of document representations, researches dealing with near-duplicates could be distinguished into two groups: fingerprints and token-based document representations. six out of ten papers in near-duplicates reported to use fingerprints as document representation, while the rest four employ token-based features in the forms of binary vectors [2, 94], strings, substrings as token [45, 89], and weighted substrings [94].

3.4.1.1.1 Fingerprinting Techniques

Fingerprinting becomes a favorite technique of representing document for Indonesian researchers as it dominates the document representation in detecting both near-duplicates and Plagiarism Detection. Interestingly, five out of six papers reported using the same techniques for fingerprint generation, that is Rabin-Karp fingerprint or rolling hash [100, 133, 135, 143, 175]. All of these five researches used ASCII characters to convert each letters in a document into byte strings. The difference between them is set on the feature units, feature length, and the prime number for computing the hash value. Mardiana et al. use word n-grams as its feature unit with unspecified n value, and 25 as its prime number [100]. The caracter n-grams are more commonly used as feature unit, where n is set to 7 as found in [133], n varies from 2-10 characters [175], and n represents a quite long sequence of characters, i.e. 30 characters [135]. Unlike 5 researches mentioned before, Wibowo et al. used word unigram as feature unit and MD5 function for generating fingerprints of a document [186].

With the question of fingerprint resolution, which is the number of fingerprints used to represent a document, comes the question of the features or substring selection strategy. Certainly, there are many strategies on how to select these features to be fingerprints which could be classified into four, namely full fingerprinting, positional strategies, frequency-based strategies, and structure-based strategies [74]. Whether it is coincidence or not, winnowing algorithm becomes the favorite strategy in fingerprinting selection as it is applied in 5 out of 6 papers reviewed in this group [100, 133, 135, 175, 186]. Winnowing algorithm is a fingerprint selection strategy which combines the positional strategy with the minimum hash value in a window. It works by firstly segmenting the hashes into a window length, then selected the minimum hash value. If there is more than one hash with the minimum value, it selects the rightmost occurrence [146]. The winnowing algorithm needs at least a parameter that is the length of hash window. Window length applied in these 5 systems are 4 hashes [133, 186], 30 hashes per window [135], and unspecified window length [100, 175]. The illustration on winnowing algorithm is presented here in figure 3.3 with a reason that it is used also as fingerprint selection strategy in Plagiarism Detection (see section 3.4.2.1). The only one applying full fingerprinting selection strategy is Salmuasih and Sunyoto [143].

3.4.1.1.2 Token-based Document Representations

Compared to fingerprinting, there are more variation of strategies in formulating document representations which use token or string as their feature unit. At least, there are three different document representations applied in four papers detecting near-duplicates: binary vectors, weighted vectors, and strings which include also substrings. Two papers reported using pure string after normalizing them [45, 89]. Adam and suhardjito used binary vectors from word unigram for their document representation [2], while Mahathir applied three different strategies which are based on three different document representations: binary vectors from word unigram, non-weighted substrings for the longest common subsequent,

```
kuku-kuku kaki kakakku kaku-kaku
```
(a) a text

```
kukukukukakikakakkakukaku
```
(b) The text after preprocessing

```
kukuk   ukuku  kukuk  ukuku   kukuk   ukuka
kukak   ukaki  kakik  akika   kikak   ikaka
kakak   akakk  kakka  akkak   kkaku   kakuk
akuka   kukak  ukaku
```
(c) the sequence of char 5-grams derived from the text

```
77 72  77  72  77  42  35  98  50  63  50
98 39  37  8   88  45  83  25  35  91
```
(d) A hypothethical sequence of hashes of char 5-grams

```
[77 72 77 72]  [35 98 50 63] [39 37 8  88]
[72 77 72 77]  [98 50 63 50] [37  8 88 45]
[77 72 77 42]  [50 63 50 98] [8  88 45 83]
[72 77 42 35]  [63 50 98 39] [88 45 83 25]
[77 42 35 98]  [50 98 39 37] [45 83 25 35]
[42 35 98 50]  [98 39 37  8] [83 25 35 91]
```
(e) a window of hashes of length of 4

```
72 42 35 50 39 37 8 25
```
(f) Fingerprints selected by winnowing

Figure 3.3: How Winnowing algorithm works. The first step is to normalize the text in (a) into one continuous string as seen in (b). The next step is to generate character n-grams, or 5-grams as it is exemplified in (c). Using rolling hash function, the n-grams are converted into hashes whose hypothethical values are shown in (d). The hash values are segmented into a defined window length which is 4 in this example, and in each window, select the minimum hash value. If there is more than one hash with minimum value, the right most will be selected (e). The selected hash values become the document fingerprints (f). Adapted from [146]

and weighted substrings [94]. Mardiana et al. applied Vector Space Model for word unigram as a comparative method to the other two fingerprinting methods [100]. Table 3.5 summarizes document representation used in near-duplicate researches.

3.4.1.2 Comparison Methods and Similarity Measures

3.4.1.2.1 Comparison Methods with Fingerprints

In doing comparison between a test document (d_{plg}) and source documents (D_{src}), systems using fingerprints as document representation tend to measure document similarity in terms of the number of common fingerprints. All of these systems do comparison on the scope

Table 3.5: Summary on document represenations used in near-duplicates

Methods	Found in
Fingerprints	
Fingerprint generation:	
Rabin-Karp	[100], [133], [135], [143], [175]
MD5 function	[186]
Fingerprint selection:	
Winnowing	[100], [135], [133], [175], [186]
full-fingerprinting	[143]
VSM	[94], [2], [100]
Strings and Substrings	[94], [89], [45]

of document level and none hinted on segmentation or chunking techniques. In measuring the number of shared fingerprints, Dice coefficient is more favoured than Jaccard as it is applied in [100, 143, 186]. Mardiana et al. compared the performance of the system by using two similarity measures: Jaccard and Dice [100]. Syahputra [175] and Pratama et al. [133] hinted no information on how they compared the document similarity. However, Pratama saved the offset of fingerprints in a tuple consisting of a set of fingerprint and its offset, ⟨selected fingerprint, offset⟩, but he did not specify further on its usage.

The comparison method reported by Purwitasari et al. [135] is worth reviewing here, as it detects the cross-check plagiarism among student assignment in one particular class. Based on the idea that plagiarism occurs among documents having similar topics, Purwitasari et al. did clustering first as a preprocess of comparison. Hartigan Index is used to determine the number of clusters. This is aimed to get an ideal number of clusters and to avoid the undervalue or overvalue resulted from user's manual input. The next step is to cluster all documents in their corpus using K-means++ algorithm. The post-clustering process is to calculate the common subsequence between documents under the same cluster. This is done by measuring the authenticity of each document to another document in one cluster by dividing the number of different hashes with the total number of hashes in both documents. A pair of Documents having low authenticity value will be regarded as a pair of source and copied documents.

3.4.1.2.2 Token-based Comparison Methods

All four systems using token-based document representations exploit different comparison methods. Kurniawati et al. applied Jaro-Wrinkle which is a type of string edit distance algorithm [89].However, no further information is provided on how to shift a string distance into a document level. Similar to Kurniawati et al., Djafar et al, employed also a string edit distance of a dynamic programming type, Smith-Waterman algorithm [45]. Unlike Kurniawati et al., Djafar et al. measured the distance between 2 compared documents by summing up the costs of deletion, insertion, and transposition operations between each token in those compared documents. In their third strategy, Mardiana et al. compared documents on the basis of their vectors by using Vector Space Model and Cosine as similarity measure [100].

Unlike the other systems, a PD system proposed by Adam and suharjito tries to incorporate shallow NLP techniques by using POS tagger of Stanford NLP toolkits [2]. It segments both D_{src} and d_{plg} into paragraphs and sentences, then applies the POS tagger on the level of sentences. Only adjectives, nouns, adverbs and verbs are selected. Using Wordnet, the synonyms of these words are searched and used to transform the selected tokens into meta-tokens to represent a paragraph, though it is unclear which meta-token is chosen. The comparison is done on the level of paragraph using Jaccard index [2]. Similar to Adam and Suharjito, Mahathir, in one of its methods, segments both documents into sentences, and tokenize each sentence [94]. Basically, he employs three methods of ROUGE algorithm which is a method to determine the quality of summary by comparing it to other summaries created by human [33]. The three ROUGE methods are ROUGE-N which computes similarity of two documents on the basis of the shared n-grams, ROUGE-L which applies Longest Common Subsequence (LCS) algorithm, and ROUGE-W which is a weighted LCS and which gives more weight to the contiguous sequences [33, 94]. Each method defines Precision, Recall, and F-measure on the basis of its features. After measuring document similarity using 3 ROUGE methods, Mahathir computes the correlation of each d_{plg} to the 5 topics using Pearson Correlation. Lastly, he applies Naive Bayes classifier to classify each d_{plg} into 5 assigned topics which turn to be D_{src}. Unfortunately, both Adam-Suharjito and Mahathir experimented their algorithms on English corpora. Table 3.6 presents the summary on comparison methods used in near-duplicates.

Table 3.6: Summary on comparison methods on near-duplicates

Methods	**Found in**
Comparison Methods:	
Document vectors	[100], [143], [151], [2], [186]
Clustering	[135]
Classification	[94]
String edit distance	[89], [45]
Dynamic programming	[45]
Similarity Measures:	
Dice	[100], [143], [151], [186]
Jaccard	[100], [2]
Rouge	[94]
Cosine	[100]
Authenticity measure	[135]

3.4.2 Researches on Plagiarism Detection

Using the main criteria whereas an External Plagiarism Detection (EPD) system should be able to find, locate, and extract the similar passages, we found out 6 researches belonging to this category. Four out of these six systems applied the tree-stage process proposed by Stein et al. [4, 157, 173, 181], one is designed to deal with Text Alignment task instead of the whole process [5], and another one did comparison and analysis on the whole document directly [147].

3.4.2.1 Document Representations

In terms of document representation, some systems employ the exact reprentations for both Heuristic Retrieval (HR) and Text Alignment (TA) stages [147, 181], the same representations with different features and strategy [4], and several different document representations for HR and TA [5, 157, 173]. In Heuristic retrieval, all these systems implemented either fingerprinting or Vector Space Model as their document representations. The fingerprinting generation techniques show no difference from the first group reviewed earlier, that is Rabin-Karp Fingerprinting and Winnowing algorithm for fingerprint selection strategy [4, 173]. The VSM implemented in HR is the generalized VSM which uses tf-idf as its weighting process, and the extended VSM model which incorporates the contextual-usage meaning of words for its vectors [88], i.e. Latent Semantic Analysis (LSA). Vania and Adriani applied the generalized VSM with tokens as its feature unit [181], while Soleman et al. compared the generalized VSM and LSA with tokens and phrases as their features [157].

In matching process or TA stage, more document representations other than VSM and fingerprinting could be found. Sediyono proposed a model for processing suffix-array data structure efficiently by generating a triangle graph for each paragraph of a source document [147]. In a system designed to execute a TA task only, Alfikri and Purwarianti [5] implemented 3 different document representations: binary vectors with word bigrams as its features, Two models of VSM, generalized and LSA, and fingerprinting. On their former system which is aimed to detect a cross-language plagiarism [4], they applied rolling-hash for fingerprinting generation, and full-fingerprint selection strategy. Unlike in their retrieval stage which uses fingerprints, Suryana et al. made use of normalized substrings for their matching algorithm: Longest Common Subsequence (LCS) algorithm. Unfortunately, there is no explanantion on how to extract the longest common sequences, whether it is done through suffix-array data structure or simply through string-matching [173]. The summary on the use of document representation in this category is presented in table 3.7.

3.4.2.2 Comparison Methods and Similarity Measures

In general, the methods employed to measure similarity between d_{plg} and d_{src} both in Retrieval and Text Alignment could be grouped into four methods: string matching, frequency-based comparison, document vector-based comparison, and classification. Vania and Adriani made use of Apache Lucene for indexing, Retrieval, and Alignment [181]. Lucene scoring uses a combination of the Boolean model and the generalized VSM to de-

Table 3.7: Summary on document represenations used in Plagiarism Detection

Document Representation	**Found in**
Heuristic Retrieval	
generalized VSM	[181], [157]
LSA	[157]
Fingerprints	[4], [151]
Text Alignment	
generalized VSM	[181], [5]
LSA	[5]
Fingerprints	[5]
Suffix-array	[147], [151]

termine the relevance of an indexed document to a user's query [13]. The top-10 documents outputted by Lucene are selected to be the source candidates, which are then segmented into paragraphs and reindexed to Lucene. The segmentation into paragraph is applied also to a d_{plg}, whereas each paragraph is used as a set of queries. Lucene does the comparison and the top-5 ranked paragraphs are selected to be source candidates of each paragraph in a d_{plg}. The post processing is done by removing passages that have low similarity score whose threshold is not explicitly specified. The last filtering is done by removing pairs of paragraphs having less than 3 overlapping word 6-grams. The remaining pairs of paragraphs are considered as a pair of source and plagiarized paragraphs.

In their comparison strategy, Soleman et al. segmented d_{plg} into chapters, paragraphs, sentences, and no segementation which means the whole document as one segment in Heuristic Retrieval (HA) [157]. In Text Alignment task, only the first three segmentation types are applied to both d_{plg} and d_{src}. In HR each segment of d_{plg} is compared to an unsegmented d_{src}, but in TA each segment of d_{plg} is compared to a segment of d_{src}. Cosine similarity is used as a similarity measure for both generalized VSM and LSA models. A note worth mentioning here is that the segments applied in HR are not used to formulate queries, but rather it is treated as an independent unit of d_{plg} as it is compared to the whole document of d_{src}.

Alfikri and Purwarianti [5] applied two classification methods, Naive Bayes and Support Vector Machine (SVM) on their system designed to execute TA only. Each classification method is run on four different features generated from word unigram, word bigrams, full fingerprints from rolling-hash, and weighted vetors computed through LSA. In their former system which is designed to compare a d_{plg} in Indonesian to a set of D_{src} in an English corpus, Alfikri and Purwarianti [4] inlcuded phrase chunking, synonym analysis and removing sentences containing citations in their preprocessing stage in addition to standard preprocessing. The citation is matched through a pattern matching whose pattern consists of parentheses, author's name and a publication year. The phrase chunking and synonym analysis which uses Wordnet 2.1 are used to choose which words best fit the translation. The Indonesian-English translation is done by devising Google translate. The next phase

[13]Information on Lucene is available on `https://lucene.apache.org/core/2_9_4/scoring.html`

is to transform the translated features into fingerprints. Dice coefficient is used to measure the similarity between compared documents both in Retrieval and Alignment phases. The difference between fingerprints used in Retrieval and Text Alignment subtasks lies on the n-gram length in fingerprint generation and fingerprint selection strategy.

The EPD system reported by Suryana et al. [173] proposes a peculiar method in selecting source candidates in HR task. Instead of measuring similarity between d_{plg} and d_{src}, a fingerprint index of 2-3 tree is generated from inverted index to eliminate the irrelavant documents. The 2-3 tree saves the fingerprints along with their posting list which consists of information on DocId and a frequency of matched fingerprint in that docment. If a fingerprint of d_{plg} matches a fingerprint in the tree, the matched frequency in a d_{src} will be incremented. This frequency value is used as a parameter to eliminate the irrelavant documents, though the threshold frequency is not clearly stated. In TA task, Suryana et al, use the longest common string algorithm for matching the source candidates and a d_{plg}.

The Longest Commonly Consecutive Word (LCCW) algorithm proposed by Sediyono and Mahmud [147] locates and extracts the similar passages from a source and suspicious documents by a means of a triangle tree. Firstly, the algorithm segment both documents into paragraphs, but a triangle graph will be generated only for each paragraph in a source document. The graph is built level by level; the first level nodes contain a word unigram, the next level nodes contain a word n+1-gram from their base nodes. the comparison is conducted paragraph per paragraph by binary search: diagonal or vertical search. The diagonal search is applied if the start node is in the source. The vertical search is applied if the diagonal search find the CCW. By using this technique, the sequential check node by node can be avoided [147], and the longest common consecutive words could be located, even if the length of these common consecutive words is less than the paragraph length. It is reported that LCCW performance outperforms the suffix-tree [147]. Table 3.8 summarizes the comparison techniques and similarity measures for reviewed Plagiarism Detection systems.

Table 3.8: Summary on comparison methods in Plagiarism Detection Systems

Methods	**Found in**
Comparison Methods:	
Document vectors	[181], [4], [157]
Classificiation:	
Naive Bayes	[5]
SVM	[5]
Tree of graphs	[147], [151]
Similarity Measures:	
Dice	[5], [4]
Cosine	[157], [181], [5]
Custom measures	[147], [151]

3.4.3 Experiment Scenarios

The experiment scenario for 16 surveyed papers could be distinguished into 2 groups, those which use both source and test documents from available corpora, and those which build their own evaluation corpora. Those using available corpora do not need to design any experiment scenario as it has been already defined and will not be reviewed here. Among those which build their own evaluation corpus, the number of documents tested varies from 2-4 [89, 100, 133], 12 documents [4], 25 documents [2], 28 documents [175], 60 docs which are compared against each other [135], and 70 documents [5]. The test documents are mostly a literal copy from one or more than one source documents with no obfuscation at all or with an obfuscation which is done by shufling the order of paragraphs or sentences [2, 4, 73, 100, 143, 186]. The obfuscation types done on a literally copied d_{plg} from one d_{src} are synonym replacement on the 50% of document length [100], paraphrasing some sequences with paraphrase percentage of 20% [186] or 50% [100] of document length, partial paraphrase on the sentence level [4], summary obfuscation in a small portion [157], sentence structure alterations such as changing the voices from active to passive [5, 186].

Many systems do experimentation on short test documents with a length of 14-58 words or with documents which consist of maximally 2 paragraphs only [45, 89, 100, 135, 143], and on medium documents with the length of 200-1100 words [173, 175]. Most systems compare the whole d_{plg} to a d_{src} as one document segment. The exceptions are found in [157] which compares each segmented chunk of d_{plg} to d_{src} as one document segment in HR, but each segmented chunk of d_{plg} to each chunk of d_{src} in TA task, and in [147] which does its comparison on the level of paragraph chunks. The applied evaluation measures are precision, recall and f-measure which are expressed in percentage [2], or in a value ranges from 0-1 [157]. Another evaluation measure is *accuracy* which is expressed in percentage [5], or in value of 0-1 [135]. Systems which measure text similarity or detecting near-duplicates simply take for granted the similarity scores. Most papers reported that the performanc of their systems are good and very good with score of evaluation measures above 0.7 or 70%.

3.5 conclusion

The historical review of *Bahasa Indonesia* shows that it inherits the agglutinative characteristic from Riau-Malay as its origin. In its growth, Indonesian becomes partly an isolated language which can be seen on how it builds phrases and compound words which is much influenced by the loan words taken from vernacular languages within Indonesian archipelago itself as well as from foreign languages. The voices and affixation process in Indonesian influence the syntactical structure whether a sentence will have S-V-O, V-S-O, or S-O-V word order, with a note that V stands for predicate which is not always in the form of a verb or copula.

The review on published researches on Plagiarism detection done by Indonesians, independent of the fact whether those researches solve the problem of Plagiarism Detection for

Indonesian texts or not, shows that most systems deal with duplicate and near-duplicate detection, even though detecting near-duplicates has not become challenges in EPD any more (cf. section 1.2). Secondly, it could be concluded that most systems are still trapped on doing exact matching. This could be seen from their proposed methods, strategies, and algorithms. There are efforts to detect obfuscated texts by incorporating semantic analysis such as LSA or substituting some words with their synonyms. However, the application of synonym substitution is still limited to obfuscating the test documents. It would be more beneficial if the synonyms are used to expand queries in HR or to match seeds in TA subtask. LSA proves to be useful in recognizing near-copy, given a text with synonyms replacement as its obfuscation type. But, detecting obfuscated texts which include several obfuscation types such as near-copy, paraphrase, and summary demands not only providing test documents containing these types of obfuscation but also designing comparison methods which allow matching such texts. So far, the only research detecting indonesian texts with such methods and algorithm is the one proposed by Alfikri and Purwarianti in [5].

As fingerprint dominates the document representations both in Retrieval and Text Alignment subtasks, Rabin-Karp algorithm becomes the favorite method of fingerprint generation. This might be caused by two possibilities: either by Rabin-Karp's efficiency in computing the hash value of a string or by its computation simplicity. In generating hash values of a sequence of tokens, Rabin-karp computes fully only the hash value of the first token or gram, the next token's hash value is computed by subtracting the sum of multiplication between the base and the first and the last character of the next token from its former token's hash value [151]. The LCCW algorithm presented in [147] proposes an efficienter method to compute suffix-array as document representation. LCCW proves to very good to detect exact copy, but it is unable to cope with obfuscated copy (cf. 2.6). Another drawback from this algorithm is that its time and space complexity is quadratic as reported in [147].

Among systems applying Retrieval subtask, none applies query formulation. Most systems tend to use all features of a d_{plg} as a set of queries. The advantage of this strategy is that the possibility of finding the source documents is quite high. The drawback lies on its computation effort in a use case where a medium or long size of d_{plg} is compared against a large corpus of source documents. Another drawback of such strategy is that it gives no possibility of online comparison which requires limited number of queries. One system proposed by Soleman et al.[157] applied segmentation on the level of document, chapters, paragraphs and sentences. However, the segmentation has been used improperly, whereas each segmented chunk is compared against the whole content of d_{src} as one segment in HR subtask. This is an unbalanced comparison which will surely leads to the result where the unsegmented d_{plg} gives the highest recall compared to any retrieval strategy with segmentation.

In Text Alignment subtask, the concept of *seed extension* is unknown. Most systems stop at matching process which is included in seeding phase (cf. section 2.2.2.2) with the exception of LCCW algorithm, if the diagonal search could be parallelled to *seeding* and the vertical search on the tree of graphs is considered to be the *seed extension* in finding the longest common words. It could be summarized that the methods of matching process in TA is more diverse, including document vector-based similarity computation, classification methods, and tree-based string matching. This matching process is done either globally – on the document level or locally – on the segmented chunks. To conclude, most systems working on Indonesian texts have not considered to incorporate linguistic analysis in their Text Alignmnet as well as Heuristic Retrieval subtask.

Chapter 4

A Framework for Indonesian Plagiarism Detection

This chapter describes a proposed framework in detecting plagiarism for Indonesian texts. This framework is wrapped up into 4 sections. Section 4.1 discusses the system workflow, the top-down approach applied in the system and its three main subtasks. The various methods for retrieving the potential source documents will be described in section 4.2. In this section, the various preproccessing strategies and document representations will be discussed, in addition to strategies for query formulation, measuring document similarity and filtering. Section 4.3 presents the various strategies on text alignment subtask which includes how to select seeds, seed matching and extension. The post-processing is presented in section 4.4.

4.1 The Proposed System Workflow

In section 2.2.2, we could see that the majority of the available External Plagiarism Detection (EPD) systems do computation on the document level in the retrieval phase, and on the heuristic comparison phase or text alignment, they use the smallest units such as character n-grams, token, word n-grams, or sentences to be matched and merged under certain defined conditions into larger sequences and then into passages. The disadvantages of this method are that firstly, exhaustive comparison on smaller units is computationally expensive. Secondly, many matches whose lengths are under the defined criteria will be discarded, which again signifies the waste of some computation efforts. Different from these EPD systems, the proposed framework in this study utilizes the top-down approach in the context of document structures. This means it does computation firstly on the document level, paragraph level, then to smallest units, keywords and key phrases to determine the passage boundaries in the identified similar paragraphs only. This top-down approach which ignores computation on the sentence level is based on the presumptions that:

1. Plagiarism often takes place at larger sequences than a sentence.
   Based on the plagiarism scenario (cf. sections 2.1.3 & 2.2.2), the criteria that are commonly used to define the existence of plagiarism are the minimum number of similar characters, words or lines in a broadly-defined chunk, or even the percentage of similar passages compared to the document length.

2. Manipulation in plagiarism cases often has a major effect on sentences.

A sentence conveying a single idea, could be reworded by unnecessary sub-ideas which may result in more than one sentence. In another case, ideas conveyed in several sentences could be packed into a single sentence.

3. Keywords of a passage are unlikely objects to obfuscate
   Keywords are part of content words but intuitively could be distinguished from them, as keywords convey the main ideas of a passage. Unlike content words that have more probability to be paraphased and modified, it is assumed that the probability of modifying keywords is relatively lower in doing text modification.

4. In Indonesian academic texts, the keywords or significant terms are mostly loanwords or borrowing words.
   As reviewed in section 3.2, vocabularies from vernacular languages in the archipelago and foreign languages enrich modern Indonesian language. Thus, taking keywords as smallest units to detect similar passage boundaries is presumed to be more effective than a consecutive sequence of strings or tokens.

Through this approach, a system prototype called *PlagiarIna* has been implemented. It is based on a three-stage process introduced by Stein et al. in [164]. The system architecture of PlagiarIna could be seen in figure 4.1 which displays its three main processes: source candidate document retrieval, text alignment, and post-processing. The top-down approach is applied firstly in source retrieval which selects source candidates by computing similarity on the document level. The similarity on the paragraph level is applied on the text Alignment stage. Only pairs of paragraphs from suspicious-source document pairs having similarity values above threshold will be exhaustively compared to determine the passage boundaries.

From figure 4.1, it could be seen that the evaluation is run at two different stages. Firstly, it evaluates the performance of the retrieval subtask by assessing its outputs taking the form of the candidate documents. Secondly, the evaluation is done to the detection results of text alignment subtask. Performing evaluation for the back-end output of detection and considering it as the performance of the whole system could be misleading. The reason is that in a system workflow such as PlagiarIna, both retrieval and text alignment subtasks contribute equally to the high performance of the system. Since both subtasks are interdependent, no matter how good and efficient the detection algorithm (Text Alignment) is, if the real source documents are not retrieved during the retrieval phase, the end performance will be disappointing. On the contrary, there might be possibility that some source documents would have been retrieved, but due to plagiarism types or obfuscation level, the text alignment algorithm fails to recognize the source passages. For this reason, the evaluation for each subtask will help reveal which method in which stage needs improvement. The following sections present the strategies of this study per subtask.

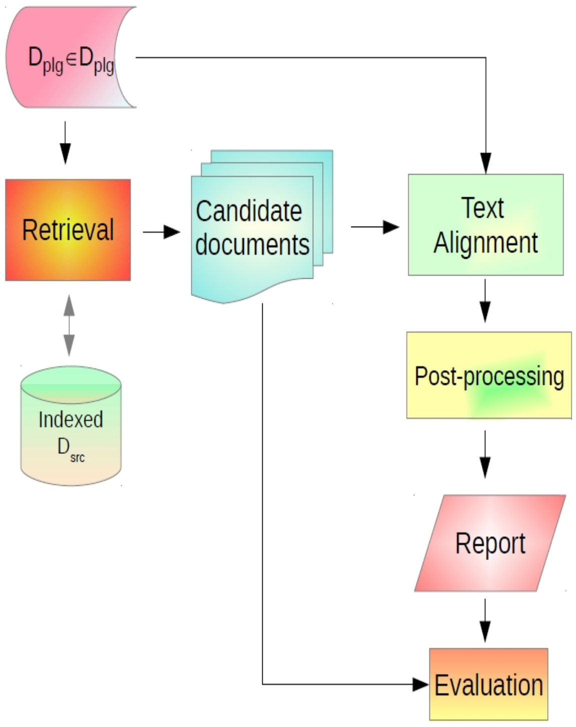


Figure 4.1: System architecture of PlagiarIna

4.2 Candidate Document Retrieval

The task of Retrieval phase in a plagiarism detection system as defined in PAN competition is 'to retrieve all source documents while minimizing the retrieval cost' [14]. The source documents referred to in this task include all documents whose content might be fully, partially or even slightly reused or plagiarized. The challenge of this task is how to find such source documents out of thousands even millions of documents. The main challenge of source retrieval subtask could be specified and broken down into the followings:

- How does one maps suspicious and source documents into a document representation which enables searching and matching similar long sequences of document content as found in the case of verbatim or shake and paste text reuse, while giving possibility for capturing alteration and obfuscation in those long contiguous sequences of words pertaining to cases of paraphased and summarized ones?

- How does one formulate effective queries for retrieving all of these source documents? The query formulation includes the keyword selection strategy which needs to consider that queries selected from the keywords should represent unknown plagiarized

[14]`http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/plagiarism-detection.html`

passages, instead of representing the topic or information relatedness between suspicious and source documents. It is true that text reuse occurs among documents sharing similar topics and information but they should not be used as parameters for the occurrence of plagiarism or reused texts.

- How does one measure the similarity between suspicious and candidate documents? How does one select highly probable candidate documents among any other documents? Source documents whose contents are slightly reused and heavily obfuscated in suspicious document tend to have low similarity value. This makes filtering a challenging task in retrieval phase.

These challenges form the building blocks of the retrieval phase of this prototype which comprise document representation, query formulation, similarity measurement, and filtering. As PlagiarIna's retrieval phase is designed to do searching offline, the weighting and indexing process of source documents are included in the document representation. Strategies applied in each retrieval building blocks are presented in the following sections.

4.2.1 Text Preprocessing

In PAN competition setting, text preproccessing could be excluded from the system's building block since the training and testbed corpus is already available. In real setting, preproccessing text is significantly needed to reduce data dimensionality. The first stage of text preprocessing done in this study is to convert various document formats into plain text. Then, the shallow Natural Language Processing (NLP) method is applied to perform text normalization, token extraction, and token normalization. The text is normalized by lower case folding, converting non-readable characters and numbers into a white space, and reducing the number of white spaces into single one. Tokenization is done to extract tokens which are then normalized with stemming and stopword elimination.

4.2.1.1 Stopword Elimination

Stopword elimination is a language-dependent process which requires a stopword list. The common strategy in building a stopword list is by sorting token according to its collection frequency [97], that is the total number of term occurrences in all documents in a corpus. This is called a frequency-based stopword. In cases of Indonesian text retrieval, a semantic-based stopword list is needed and its application together with frequency-based stopwords have proved to increase the performance of an Information Retrieval system [15, 177, 182]. A Semantic-based stopword list takes account of semantic functions of a word in a sentence [15]. Such words could take the form of verbs, adverbs, or adjectives, but semantically have little value for retrieval process and their low frequencies prevent them to be included in the frequency-based stoplist.

This study applies two types of stopword lists mentioned before. Instead of using the available frequency-based stopword, this study created its own stoplist by selecting tokens having high document frequency (DF) and collection frequency (CF). Tokens having high

CF values and which occur in more than 40% of documents in the corpus were selected to be stopwords. The frequency-based stoplist consists of 233 words. This list includes characters which are commonly used to mark preliminary pages in thesis such as i, ii, ix, etc. As for semantic-based stopwords, there are two sets of readily available semantic stoplists: Tala- and Vega-stoplists. Tala-stoplist comprises 758 unique words [177], while Vega-stoplist is classified into two groups. The first group consists of 169 words and the second one comprises 556 words [15, 182]. The semantic-based stoplist that is used in this study is Tala-stoplist which combines the frequency-based and the semantic-based ones. Both frequency-based stoplist dereived specifically from our corpus and Tala-stoplist are available in the Appendix A.

4.2.1.2 Stemming

Stemming is a normalization process which allows token conversion into a morphologically less invariant form. Like stopword list, stemming is also a language-dependent process. Basically, there are two types of algorithm for Indonesian stemming process, the linguistically-motivated stemming, and the rule-based stemming. In linguistically-motivated stemming algorithms, the process of stripping affixes (prefixes, suffixes, infixes and circumfixes) is based on complex morphological rules and the stemming results will be checked against a dictionary of root words. If the stemmed token is found in the dictionary, it will be delivered as output. If the dictionary look-up process fails, the stemmer returns the original unstemmed tokens. In her study on evaluating the performance of 6 different stemming algorithms for Indonesian, Asian's experiment in [15] shows that the performance of linguistically-motivated stemming algorithms outperform the rule-based one. Further, she demonstrates that CS-stemmer turns out to be the best stemmer wich achieves an accuracy rate of 96.4%. The high accuracy of CS stemmer is resulted from a strategy which allows the algorithm 'to evaluate each step and to test if a root word has been found, and to recover from errors by restoring affixes to attempt different kinds of combinations' [15].

However, the tradeoff between computational effort and stemming accuracy in a preprocessing stage makes this study turn to the rule-based stemmer such as the Porter stemmer. Since Porter's algorithm can only do suffix stripping, the modified version of Porter Stemming for *Bahasa Indonesia* defines five affix-rule clusters which are processed according to the following order: removing particle, removing possessive pronouns, removing 1^{st} order prefix. Following first order prefix removal, suffixes will be removed first, if a rule is fired then followed by removing the second order prefix [177]. If the rule fails, the second order prefix is removed first, then followed by suffix removal. Tala has evaluated the performance of modified Porter stemmer for *Bahasa Indonesia* and reported that it produces 11.8% non-comprehensible words [177].

For the stemming process, this study makes use of IDNstemmer written by A.F. Wicaksono and B. Muhammad (2009), which is available in GNU as an open source software. IDNstemmer is a variant of Porter stemmer for *Bahasa Indonesia* which allows recursive affix removal and enables removing prefixes to their third order. The drawback of IDNstemmer is that the affix stripping rule is designed to be recursive, so that it results in greedy affix and non-affix removal. Though there has been no study which evaluates the performance of IDNstemmer, intuitively we perceived that the non-comprehensible words outputted by IDNstemmer are tolarable. In order to reduce the algorithm greediness and to decrease the number of non-comprehensible stems, we modified IDNstemmer algorithm by adding the following rules:

1. Defining restrictions on the minimum length of a token to be stemmed. The minimum token length for second order affix removal is set to 6 characters and 8 characters for the first order affix removal.
2. Eliminating the recursive rule on removing affixes, and redefine the depth of prefix removal from third order into second order prefixes, and the suffix removal is reduced to the first order.
3. Annihilating the rule for removing prefix for first person singular subject *ku-* with the reason that in a formal and scientific written discourse, the first person singular subject will be expressed by using specific token instead of prefix *ku-*.
4. Defining the most frequent circumfixes to be removed such as *me-..-kan, me-...-i.*
5. Defining additional rules for elimination of suffix -i by checking the occurrences of most frequent circumfixes such as *me-...-i; di-...-i.* The reason is that most variants of Porter stemmers ignore circumfixes and treat circumfix as a separate prefix and suffix. The consequence is the greediness in stripping all characters defined as suffixes including those which are not.

Since evaluation of stemmer's output is beyond the scope of this study, the performance test of this modified IDNstemmer was conducted by running it on a handful of source documents and observing specific words which were potentially stemmed into non-comprehensible words. The output of this modified IDNstemmer is much better than the original one in term of the number of incomprehensible words. Besides, it is computationally less expensive than linguistically-motivated stemmers. Thus, this modified IDNstemmer contributes positively to the preprocessing stage.

4.2.2 Document Representation

There are three considerations that motivate this study to experiment on different kinds of features for representing documents. Firstly, the document features in the state-of-the art plagiarism detection systems (see section 2.2.2.1.1) are still dominated by string-based approaches in spite of their deficiency in retrieving reused text with medium to heavy obfuscation level. The stopword n-gram features are inappropriate for Indonesian texts. In Indonesian, the function words such as articles or preprositions could be discarded in constructing a well-formed sentence. The citation pattern is used better as complementary method [61]. Secondly, including semantic analysis is computationally too expensive for practical plagiarism detection task as shown in Bao's experiment which took account of using synonyms and hypernyms. His findings showed that the detection performance increased by factor two, but the processing time increased by factor 27 [17, 58]. Thirdly, Asian in her study on Indonesian text retrieval has experimented various techniques by combining 3 different kinds of stopword lists, 6 stemming algorithms, language identification (English, Indonesian, and Malay)[15]. Her study showed that combining stopping and stemming increased precision and recall, although the increases were not significantly different from no stopping and no stemming [15]. Learning from Asian's research and the fact that retrieving source document in PDS is a more challenging task, this study will examine the application of three different features for representing documents: phraseword, character n-grams, and token.

4.2.2.1 Phraseword

Phraseword is a metaterm for n-tokens that is designed to capture phrases and consecutive words which have been modified morphologically or lexically. It represents each token in two characters only. Phraseword building process depends on two parameters which practically define its types, i.e. a token length, and either the only first or the first two characters of a token. The text normalization process mentioned in section 4.2.1 determines the variation number in each type. Suppose, we have a short document consisting only the following sentences:

(a) *Saya menyerahkan diri saya ke polisi.*

(b) *Mereka menanyai saya tentang uang yang dirampok Amir kemarin.*

The first type of metaterm transforms a token into two character-length terms by its length (1-9) and first character. Any token whose length is greater than or equals to ten will be represented by a star sign (*). Then, n-grams of this coded term will be formed. Literally, the sentences above will be coded into: 4s *m 4d 2k 6p 6m 8m 4s 7t 4u 4y 8d 4a 7k. The text normalization results in four variations according to its setting whether it applies frequency-based stopword removal only (var1), stopword and stemming (var2), tala stopword removal (var3), or Tala stopword and stemming (var4). The example of text conversion into **phraseword 3-gram** is illustrated in table 4.1.

The second type of metaterms are formed by slicing the first two characters of a token. This second variation of metaterm creation requires stemming in its preprocessing. The stemmed tokens are assumed to represent root words which are morphologically less invariant. With the preprocessing steps, there will be 2 variations for this metaterm: var2 which applies both stemming and stopword removal and var4 which applies Tala-stopword removal and semming. The example on how to convert token into metaterm in the second type is displayed also in table 4.1.

This metaterm is coined as *phraseword*. The name is based on its form which resembles a word, and on its function which capture phrases or word sequences. By using string length and the first two characters of a token, this representation has more possibilities to match. This is the purpose of using phrasewords, that is to take an advantage of its

Table 4.1: Phraseword building and its variations

Pre-proccess	Preprocessed Token	Preprocessed metaterm	Phraseword 3-grams
		Type I	
Var1	menyerahkan diri polisi menanyai uang dirampok amir kemarin	*m 4d 6p 8m 4u 4d 4a 7k	*m4d6p 4d6p8m 6p8m4u 8m4u4d 4u4d4a 4d4a7k
var2	serah diri polisi tanya uang rampok amir marin	5s 4d 6p 5t 4u 6r 4a 5m	5s4d6p 4d6p5t 6p5t4u 5t4u6r 4u6r4a 6r4a5m
Var3	menyerahkan polisi menanyai uang dirampok amir	*m 6p 8m 4u 4d 4a	*m6p8m 6p8m4u 8m4u4d 4u4d4a
var4	serah polisi tanya uang rampok amir	5s 6p 5t 4u 6r 4a	5s6p5t 6p5t4u 5t4u6r 4u6r4a
		Type II	
var2	serah diri polisi tanya uang rampork amir marin	se di po ta ua ra am ma	sedipo dipota potaua tauara uaraam raamma
var4	serah polisi tanya uang rampok amir	se po ta ua ra am	sepota potaua tauara uaraam

inexact matching characteristics so that any modified consecutive words or phrases could be matched. Furthermore, the coded version of texts in phrasewords are on average 67,96% shorter than texts coded in token or word unigrams. This practically reduced the storage space during the indexing process.

4.2.2.2 N-grams

Along with phrasewords, the character 4- to 7-grams are used as features to represent documents. The rationale of using the short chunk is to make possible the capturing of the morphological modification within a word level. A function which streams texts into n-grams was created. This function takes the normalized text as its input, and the steps of n-gram building process are as follows: the array of tokens of a text is imploded into a single string with an underscore (_) as token delimiter. Starting from the string offset, the overlapping character 4- to 7-grams are sliced recursively till the end offset of the string.

Practically, the n-gram features underwent two kinds of stopword removal, the first was the removal of frequency-based stopword during token normalization phase, and the second was to remove stop-character n-grams which occurred right after their building process. As in frequency-based stopword, the stop-character n-gram lists were constructed by considering both character n-gram's collection frequency and document frequency as well. Therefore, the results show that the stop-character 4-gram list consists of 585 4-

character tokens, the stop-character 5-gram list contains 320 tokens, 164 tokens are in stop-character 6-grams and 104 tokens are listed in stop-charater 7-grams. The stop-charater n-gram removal is aimed to remove n-grams containing affixes which might span to a length of 7-characters if two prefixes occur simultaneously such as in case *_memper-*. Besides the average length of a root word, it is the length of prefix combinations which motivates us to define n in character n-grams as document features.

4.2.2.3 Word Unigram

Another document feature takes the form of a word which undergoes different kinds of token normalization. The text normalizations applied in word unigram are exactly the same as those applied for phrasewords in section 4.2.2.1, and they define 4 methods of word unigrams which undergone the following processes: frequency stopword removal, frequency stopword removal plus stemming, Tala-stopword removal, and Tala-stopword removal combined with stemming. The rationale for using word unigrams instead of word n-grams is its possibility to represent each “corner” of passages in a set of document queries is higher than word n-grams. Besides, word unigrams have potential to be applied in an online source retrieval subtask.

4.2.2.4 Indexing and Weighting

In a system that does comparison offline, indexing is a crucial process which associates a document with a descriptor represented by a set of features which are automatically derived from its content [21]. The purpose of indexing is to optimize speed and performance in finding relevant documents for a search query. The construction of the inverted index in PlagiarIna was performed through a function which steps through the entire documents in the collection. If a feature or term is encountered, it will be checked whether it has been encountered before. If it has, then a counter which is set to count its frequency is then increased. A hash function was created to locate term in an array, a collision caused by hash function was resolved via an array which assigned Document ID as its key and term frequency as its value. The value of collided hash was simply pushed to the end of array element. The inverted index output looks like $t_j \rightarrow \{d_1 \rightarrow tf_{1j}, d_2 \rightarrow tf_{2j}, ..., d_i \rightarrow tf_{ij}\}$, where i indicates document identifier and j stands for the term identifier in document d_i. Thus, it can be seen that instead of a linked list, an array is used to create a posting list. This is because array does not need extra storage for references as in a linked list.

The output of index construction algorithm is a set of files which are as follows:

1. **Index file** contains a tuple of posting list for each index. The index takes the form of a list of features or terms, while the posting list stores information on Document IDs and term frequencies as described in the previous passage. This index file is saved on the disk.

2. **DF file** contains an index of terms and its document frequency. This information is needed in the weighting process, and this file is also saved on the disk.

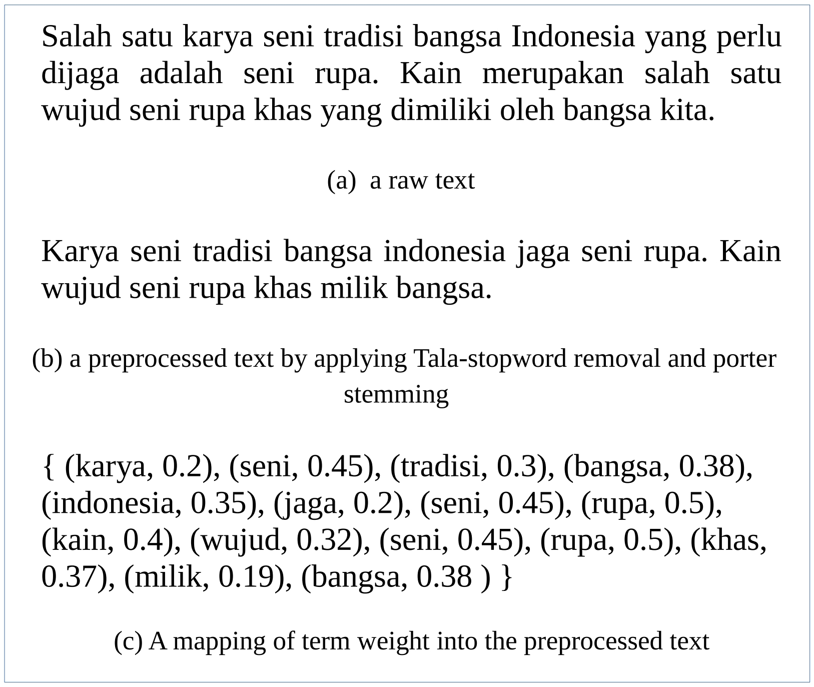
3. **Weight file** contains term weights for each document which will be needed later in the similarity comparison process. Unlike index and DF file, the weight file is saved in a relational database, MySQL. The tuple on Document ID and the term weight is stored as a text string under the field of DocID and Weight. The weight table looks like $t_j \rightarrow \{(d_1, d_5, ..., d_i), (tw_{j1}, tw_{j5}, ..., tw_{ij})\}$, where tw_{ij} refers to a weight of term t_j in d_i. This strategy was taken to avoid using matrix with DocIDs as fields which result in taking too much space for storing the zero weight values of terms which do not occur in some documents. In this design, the number of source document has no influence on the number of posting list's fields. It remains having two fields only, no matter how many source documents are indexed. Furthermore, this strategy saves only the document IDs in which the term occurs.

The term weighting applied for each document feature described in sections 4.2.2.1, 4.2.2.2, and 4.2.2.3 is tf-idf weighting. *tf-idf* is considered to be a global term weighting, because it considers term frequency not only on the whole document but also its occurrences across all documents in corpus. This is the strength of tf-idf weighting for retrieval process. Preceding the indexing process, a document file which stores information on each source documents such as Document IDs, names, and content was saved as a table in the database.

4.2.3 Query Formulation

Query formulation becomes one of challenging techniques in source document retrieval subtask. The challenge lies on the fact that firstly, the plagiarized passages are unknown and hidden inside the suspicious documents; secondly plagiarism types in those passages vary. The strategies for query formulation should consider on how to select keywords which include terms representing these supposedly unknown suspicious passages on one hand, without overloading the number of keywords selected as queries on the other hand. One important thing to note is that the queries should not be a summarized version of a suspicious document content. Such set of queries proves to be effective in retrieving documents having similar topics, which could not guarantee any presence of text reuses or plagiarism cases.

Based on challenges mentioned before, the query formulation strategy in this study considers the suspicious document length and the distribution of keyword selection. Therefore, the query selection is based on segments of a suspicious document. The first step of this segment-based query formulation is to apply the same text normalization processes and feature generation as applied in indexed source documents. This means 4 text preprocesses using two kinds of stoplits and their combination with stemming are applied also to determine the methods of each generated document features. In computing the tf-idf weight of suspicious document terms, we use term document frequencies (df) provided in DF file which is resulted from the indexing processes (See the former subsection). The weight is then mapped to each term or feature in order of term occurrence, which practically turns the suspicious document into an array containing tuples of terms and their weights. The next step is to segment the suspicious document into non-overlapping chunks. For each chunk, the terms are sorted in the descending order according to their weights. Terms are then selected according to its highest and lowest rank. Figure 4.2 illustrates the weight mapping process.



Salah satu karya seni tradisi bangsa Indonesia yang perlu dijaga adalah seni rupa. Kain merupakan salah satu wujud seni rupa khas yang dimiliki oleh bangsa kita.

(a) a raw text

Karya seni tradisi bangsa indonesia jaga seni rupa. Kain wujud seni rupa khas milik bangsa.

(b) a preprocessed text by applying Tala-stopword removal and porter stemming

{ (karya, 0.2), (seni, 0.45), (tradisi, 0.3), (bangsa, 0.38), (indonesia, 0.35), (jaga, 0.2), (seni, 0.45), (rupa, 0.5), (kain, 0.4), (wujud, 0.32), (seni, 0.45), (rupa, 0.5), (khas, 0.37), (milik, 0.19), (bangsa, 0.38) }

(c) A mapping of term weight into the preprocessed text

Figure 4.2: Weight mapping to each feature of a suspicious text in order of term occurrences

Three parameters are designed to decide terms to be query candidates. They are the length of document segment, the number of top n-highest and m-lowest rank of terms for each chunk. The first two parameters are designed to be compulsory while the last is optional. The length of segment is based on the number of weighted terms, and not on the raw text. Thus, the segment length in a query formulation covers a wider chunk than a segment with the same length in a raw text. The segmentation is aimed to get queries evenly from different 'corners' of the suspicious document to deal with the problem of getting representation for the hidden plagiarized passages.

The number of terms per chunk as well as n-highest and m-lowest ranks for query candidates are left open and become the subject of experiment. This applies also to the length of segment. The only one predefined is the number of terms to be selected from the last document segment whose length is possibly less than the defined segment length. These shorter segments are represented by 10% of its features plus m lowest-ranked terms if m is defined. Selecting query candidates from the lowest-ranked terms is quite uncommon. This technique is designed to be applied to phrasewords, assuming that phrasewords having lowest weights represent common phrases. Through this technique, it is assumed that the common and terminological word sequences and phrases could be selected as query candidates.

The selected query candidates per chunk are then merged into an array of document queries. The possibility of having redundant terms in these query candidates is great, since they are selected from different chunks. For this reason, a filetering process to check query uniqueness is applied to these query candidates. The filtered unique terms are then submitted as queries for a suspicious document to a function which measures similarities between queries and source documents.

An effort to expand queries semantically for word unigram feature using Wordnet Bahasa [15] has been attempted. The Wordnet Bahasa is a Wordnet version for Malay language which covers Indonesian and Malaysian. The problems encountered in using Wordnet Bahasa covers the need of disambiguation process of a query in order to assign a right synset out of different synysets belonging to the same part of speech (POS), and to select a word or term out of several terms classified in the same synset. Figure 4.3 illustrates this problem by displaying words *seni* as Adjective and the number of words in those synsets. If all words in a specific synset are included in a query set for an expanded term, the total number of queries will increase sharply. As its result, the recall drops as the queries become very general and large. Considering that recall rate is very important in retrieving source documents and Bao's experiment which turns out to be true in this case (see section 4.2.2), the query expansion function was detached from this study.

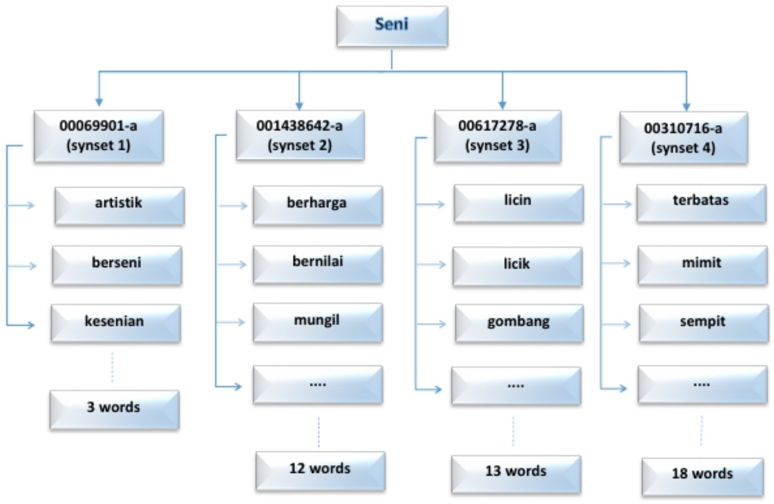


Figure 4.3: An example of synsets for the word *seni* as Adjective in Wordnet Bahasa

4.2.4 Similarity Measurement

In most cases, similarity measures quantify the similarity between the symbolic representations of two objects and map them into a single numeric value. This value depends on two

[15]available as a free resource in `http://wn-msa.sourceforge.net`. Wordnet Bahasa was constructed by research team at Nanyang Technological University (NTU), Singapore

factors, i.e. the properties of the object and the measure itself [75]. The high similarity value signifies that two objects share most of their properties and hence they are closely similar. Since each measure takes account on different aspects of object properties, they will result in different values, even if they are applied to the same objects. Considering this fact and the comparability of similarity values, the same similarity measure, i.e. Cosine similarity, is applied to measure similarity between three different representations of queries and source documents.

There lies alternatives to apply binary vector-based measures for character n-gram representations such as the well known Jaccard coefficient or Containment measure introduced by Clough and Stevenson, which calculates the intersecting n-grams and normalizes them with respect to the n-gram in suspicious document only [37]. Despite these alternatives, Cosine similarity (CS) is applied in the Retrieval task with the following considerations:

- **Cosine Similarity (CS) belongs to a global similarity measure**. CS takes account on the importance of a term in a document through its term frequency (*tf*) and its occurrences in documents in the corpus (*df*). This results in CS having a better performance for measuring large text in a large corpus.

- **It favors rare terms**. CS combined with tf-idf weighting gives higher weight to rare terms in general, especially to those having high frequency within a document but having a low document frequency.

- **It compensates the effect of document length** by computing the *dot product* of both document vectors: $\vec{P}(d_1)$ and $\vec{Q}(d_2)$, where $\vec{P}$ stands for source document vectors and $\vec{Q}$ refers to the suspicious document vectors as queries. The dot product of these document vectores are then normalized by the product of their *Euclidean length* [97]. This makes our documents (data) to have the same magnitude of vectors and the CS value lies between 0 and 1. The Cosine similarity measure could be seen on equation 4.1. The Cosine numerator, which is well known as an inner product, is also addressed as the number of matches or overlap if it is applied to binary vectors.

$$S_{Cos} = \frac{\sum_{i=1}^{d} P_i Q_i}{\sqrt{\sum_{i=1}^{d} P_i^2}\sqrt{\sum_{i=1}^{d} Q_i^2}} \tag{4.1}$$

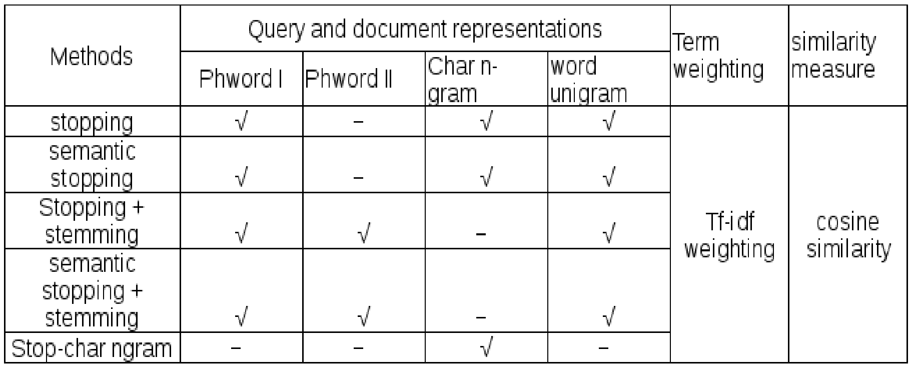
The similarity function in PlagiarIna takes the queries as its input and compares them against the inverted index of documents (see figure 4.1) which are based on the document representations. Whenever one query is matched in the inverted index, the document ID of the matched feature or term is retrieved, stored in a temporary list, and the similarity computation will then start between queries and every document in the list. During the query matching or computation of the Cosine numerator, a counter is set to count the number of matched queries in a source-suspicious document pair. The value of this counter is used to filter the retrieved documents. The outputs of the similarity function take the form of a list of tuples of Document ID and their Cosine values. These outputs are then ranked by sorting them descendingly by their cosine values.

The documents outputted from similarity function are not practically considered as source document candidates, for they are documents which match queries no matter if only one query is matched. In fact, the number of documents matching 1-2 queries is quite high. Unlike in Information Retrieval which ignores the number of matches, the candidate documents in a Plagiarism Detection System should have a reasonable number of matches. For this reason, we apply a two-step filtering method in order to reduce the number of false positive rates. The first filtering step is to discard documents having a minimum number of matches. This process is executed along with the computation of Cosine similarity. Before computing the denominator of Cosine similarity, the value in this counter is compared to the filtering parameter. If the value of counter is less than the defined parameter value, then the process of computing Cosine denominator is cancelled, and the algorithm starts computing the similarity between the next matched document in a queue and the queries. This practically discards this document from being saved in the list of candidate documents and saves a computation time.

The second step of filtering is based on cosine similarity value instead of using the top n-ranked documents. The reason is to include as many source documents as possible into a list of source document candidates. The fact that the level of obfuscation and the portion of reused passages influence the similarity value is unavoidable. If the reused pasages are heavily obfuscated or only a small portion of source passages is reused, the cosine similarity value will be low, consequently, it will be assigned a low rank too. This is one of weakneses of relying on document rank as a filtering parameter. Using document rank as a threshold such as the top 20- or 30-ranks is more practical but it may result in excluding the already retrieved-source documents from the candidate document list because of their low ranks. This leads to an undesirable result, since the task of retrieval phase is to retrieve all possible source documents. In contrast, using cosine similarity as filtering parameter may lead to a low precision rate of retrieval phase. Considering the main task of source retrieval subtask, we put weight more on the success of retrieving source documents. The precision rate will be worked out in the later phase, i.e. the text alignment. However, considering the tradeoff between precision and recall rates, the filtering threshold of cosine similarity value and the number of matched queries become the subject of experiment.

To sum up, table 4.2 displays the summary of our retrieval methods which are built by combining text preprocessing techniques with different types of document representations. It displays that there are 4 query and candidate document representations: phraseword I, phraseword II, character n-grams and word unigram. The application of those methods to each representation results in 12 method variations for phraseword type I, 6 variations for phrasewords type II, 4 variations for character n-grams, and 4 variations for word unigram.

Table 4.2: Summary of Retrieval methods applied in query and candidate representations



Methods	Query and document representations				Term weighting	similarity measure
	Phword I	Phword II	Char n-gram	word unigram		
stopping	√	–	√	√	Tf-idf weighting	cosine similarity
semantic stopping	√	–	√	√		
Stopping + stemming	√	√	–	√		
semantic stopping + stemming	√	√	–	√		
Stop-char ngram	–	–	√	–		

4.3 Text Alignment

Text Alignment, formerly known as detailed analysis, has been declared as a subtask of external plagiarism process since PAN 2012 [127], but the term itself was introduced in PAN 2013 [128]. The task of Text Alignment is to find real-world instances of text reuse, and annotate them[16]. The so-called real-world instances of text reuse refers unnecessarily to real cases of plagiarism but it could be simulated through a corpus containing source and suspicious documents which contain reused or plagiarized passages. Thus, the text alignment subtask implicitly includes a process of building such corpus. The general challenge in this subtask is how to find pairs of reused passages at one time during its comparison process. This challenge implies firstly, on building strategies for locating pairs of similar passages, and secondly on determining the similar passage boundaries.

The strategies for locating the similar passage pairs include strategy of selecting features to do exhaustive comparison. The selected features are intended to represent these pairs of passages so that various types of text reuses (cf. section 2.1.3) with their levels of obfuscation which range from light to heavy are detectable. To complicate the task challenge, this similarity or relatedness takes not only in lexical forms but also in concepts, semantics, and grammatical structures [1]. These are really broad and challenging tasks. Meanwhile, the strategy for determining the similar passage boundaries includes defining the length of relatedness and the strategy for feature extension. In Text Alignment, features are commonly addressed as seeds. Following the terminology in this field, *seeds* will be used to refer to text features from now on.

Based on the scope of this study which concentrates on aligning monolingual text reuse with paraphrase and summary obfuscation as its highlight (see section 1.2), a framework which enables us to customize different methods and tune up parameters on a GUI surface was developed. This framework uses paragraph-based comparison in locating the similar pairs of passages and rule-based approach in determining the similar passage boundary. Figure 4.4 illustrates the general framework proposed for Text Alignment subtask. It

[16]cited from `http://pan.webis.de`

starts by extracting the contents of source documents whose ID are listed in the retrieval outputs. The next steps cover: text normalization, paragraph similarity measure which are preceeded by seed selection and generation of paragraph and seed index tables, seed processing which includes seed matching, extension, coupling, and merging, and filtering as the last step. The next sections discuss these steps in detail.

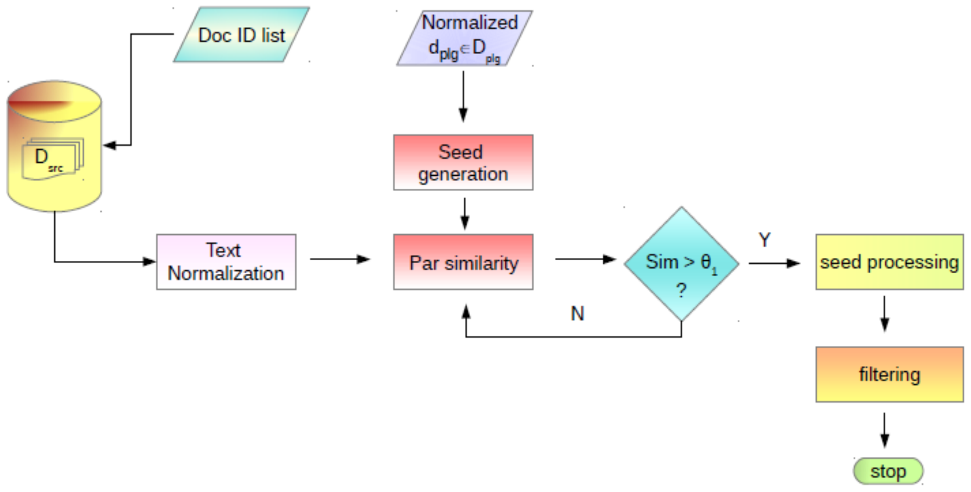


Figure 4.4: The general framework for our Text Alignment process

4.3.1 Text Normalization

In alignment phase, text normalization is applied to candidate documents, while suspicious document undergo this process during the retrieval phase. Preceeding retrieval process, all documents in corpus are normalized to construct an inverted index. The text normalization applied to candidate documents outputted from retrieval process is not a repetitive process because in retrieval, the suspicious document queries are compared to the indexed terms instead of the real content of source documents. However, in text alignment, the comparison is performed directly to the small number of candidate documents. The text normalization of candidate documents inlcudes eliminating non-readable characters, punctuations, numbers, lower-casing, and replacing multiple white spaces into a single space. Newlines and paragraph breaks are preserved and their successive occurrences will be reduced into a single newline. It is then used to segment a candidate document into paragraphs. In specific cases, newlines or paragraph breaks are used within a sentence such as in cases of wrapping text in columns, tables, etc which return short paragraph as its results. To anticipate such problems and to cope with titles, subtitles, captions of figures or tables, short paragraph segments that consist of less than 100 characters are merged to their successive paragraph.

For each paragraph segment, two different processes of text preprocessing are applied. The first one is to remove all white spaces which turns a paragraph into a long string of successive characters. This is done to generate a paragraph offset table which is used to store the information on document IDs, paragraph IDs, the start and end offsets of each paragraph. This table is saved as an array and it is generated dynamically, as the candidate documents change depending on the retrieval outputs. The paragraph offset table for suspicious document is created ealier before source retrieval task under the same process. In the second preproccessing, the white spaces are preserved to perform token normalization such as tokenization, stopwords removal, and stemming. The variation of token normalization process applied to candidate documents takes the same patterns as those applied to suspicious document during the retrieval process. The same treatment of token normalization is done in order to avoid repetition of preproccessing the suspicious document.

4.3.2 Seed Generation and Paragraph Similarity Measure

Given a set of retrieved candidate documents for a suspicious document, the next phase in EPD workflow is to identify the match using seed heuristics which 'either identify exact matches or create matches by changing the underlying texts in a linguistically motivated way' [128]. In seed generation techniques (cf. section 2.2.2.2.1), it is quite common to come up with as many reasonable seeds as possible, so that their merging enables the algorithm to build up larger aligned passages. Unlike these techniques, the seed generation in PlagiarIna is aimed to serve dual functions, i.e. as paragraph queries in measuring segment similarity and as a heuristic match. For this reason, seeds are generated on a paragraph basis.

4.3.2.1 Seed Generation

The model used to generate seeds in this study are based on some facts and assumptions. Based on the facts that each paragraph is a collection of sentences dealing with a single theme which builds a distinct section of written text, and the fact that this single theme is expressed through several keywords, we assume that these keywords are rarely altered. The context or words surrounding these keywords have higher possibilities of becoming objects of alteration. These assumptions apply mostly to academic texts loaded heavily with terminologies, which in Indonesian texts are marked by calques or loan-words. Yet, in paragraphs conveying a general theme these assumptions partly apply, meaning that some keywords become unavoidably objects of modification.

Considering facts and assumptions mentioned previously, this study borrows the scoring method employed by Kiabod et al. in [81] for selecting keywords which are akin to seed generation. To get keywords or significant words of a document, Kiabod et al. compute firstly the word local scores. The 'significant' words will be selected by a word local score threshold which is the average of all text word local scores multiplied by a Pruning Factor (PF) [81]. PF is a number ranging between zero to one (0-1). Since this scoring method is applied to summarize a document, the scoring continues on computing the word global score on the second computation phase; a total score of a word is then calculated by using its local and global score.

Applied for generating seeds in the local scope (paragraph), this study borrows only the word local scoring along with its pruning method, and adapts its equation by changing the locality scope to a paragraph as a segment. As in Kiabod's word local scoring, two statistical criteria are used. The first statistical criteria is the term frequency of the word which is normalized by total number of words (represented by TF) [81]. Yet the second criterion which is a sentence count is adapted to be a paragraph count (ParCount). It refers to the number of paragraphs containing the word normalized by the number of total paragraphs in a document. The relative term frequency computation is also adapted to term frequency in a paragraph normalized by the total number of words in that particular paragraph. The adapted word local score is then defined as in equation 4.2.

$$word\_local\_score = \alpha\ ^{*}\ TF + \ (1-\alpha)^{*}\ ParCount \tag{4.2}$$

where α is a constant for parameter weight in the range of (0, 1) which was determined empirically, and ParCount stands for paragraph count.

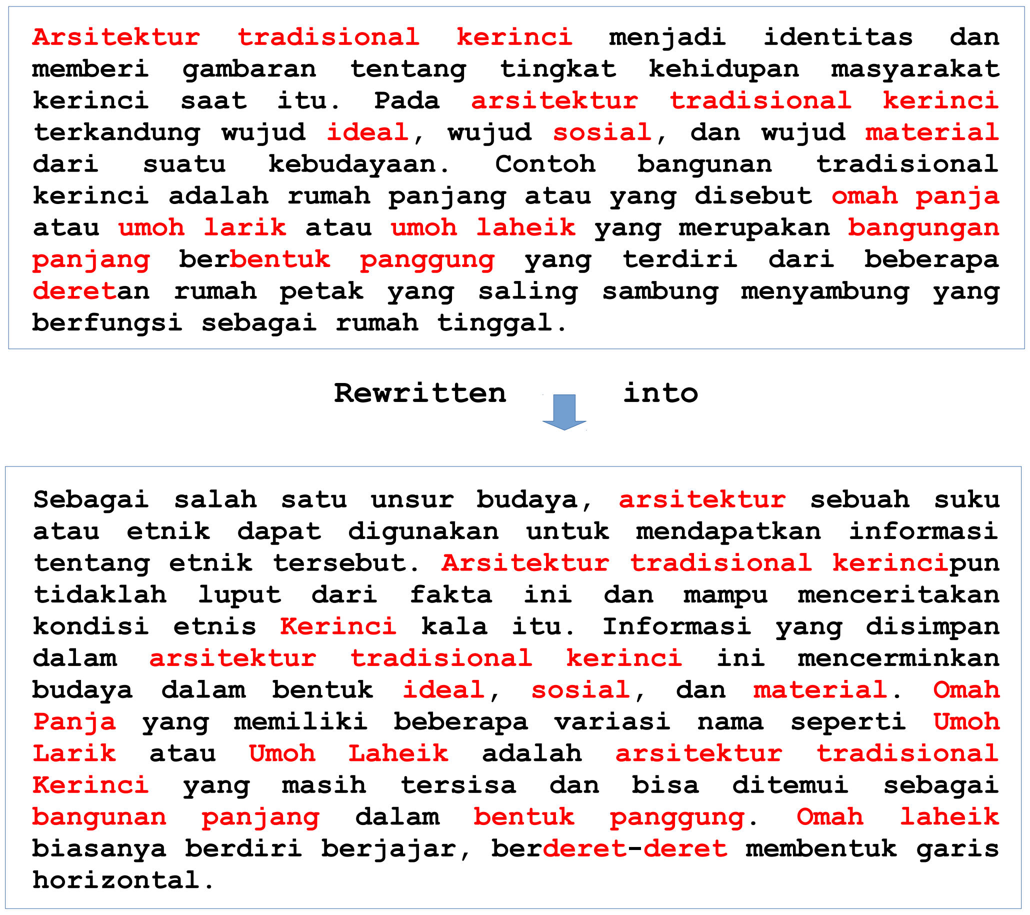
After calculating the word local score, the algorithm proceeds by removing 'insignificant' terms and save only terms whose scores are above a threshold as paragraph seeds. The word local score threshold is defined exactly as in [81]:

$$word\_local\_score\_threshold = \frac{\sum_i \ word\_local\_score(i)}{number\ of\ text\ words} * PF \tag{4.3}$$

where i represents the word index and PF stands for Pruning Factor. PF could be defined intuitively to decide how many percentage of terms will be used as seeds in a passage or paragraph. By increasing Pruning Factor, less words will be selected. Less number of seeds is good at matching heavily obfuscated paragraphs, but it results in a high false positive detection also. To make a balance and get a better result, an empirical test for defining PF is administered to 2 persons. Given short documents, they were asked to rewrite each of its paragraph by using the same paragraph themes. The unaltered words were then annotated as seeds chosen by human writers. These unmodified words and their number were used as a standard in tuning-up the PF value. Given the same documents as input, the algorithm was run by inputting different PF values within the range of 0-1. Then, the outputted seeds were compared to the unmodified words in rewritten paragraphs by human writers. It turns out that a PF value of 0.5 gave outputs of seed number and seeds which closely resemble the samples. Figure 4.5 displays an example of a rewritten paragraph with heavy terminologies in this test. The first paragraph is the source version, while the second is the rewritten version [17].

It needs to be noted, that seeds generation is a separate process from local word scoring. The seeds are generated for each parapgraph of a suspicious document by pruning the

[17]This is one of paragraphs written by Edy Hadisaputro sent through email in January 12, 2015



Arsitektur tradisional kerinci menjadi identitas dan memberi gambaran tentang tingkat kehidupan masyarakat kerinci saat itu. Pada arsitektur tradisional kerinci terkandung wujud ideal, wujud sosial, dan wujud material dari suatu kebudayaan. Contoh bangunan tradisional kerinci adalah rumah panjang atau yang disebut omah panja atau umoh larik atau umoh laheik yang merupakan bangungan panjang berbentuk panggung yang terdiri dari beberapa deretan rumah petak yang saling sambung menyambung yang berfungsi sebagai rumah tinggal.

Rewritten into

Sebagai salah satu unsur budaya, arsitektur sebuah suku atau etnik dapat digunakan untuk mendapatkan informasi tentang etnik tersebut. Arsitektur tradisional kerincipun tidaklah luput dari fakta ini dan mampu menceritakan kondisi etnis Kerinci kala itu. Informasi yang disimpan dalam arsitektur tradisional kerinci ini mencerminkan budaya dalam bentuk ideal, sosial, dan material. Omah Panja yang memiliki beberapa variasi nama seperti Umoh Larik atau Umoh Laheik adalah arsitektur tradisional Kerinci yang masih tersisa dan bisa ditemui sebagai bangunan panjang dalam bentuk panggung. Omah laheik biasanya berdiri berjajar, berderet-deret membentuk garis horizontal.

Figure 4.5: An Example of a rewritten paragraph for seed generation

word local scores, while local-word scoring is applied to paragraphs of both suspicious and candidate documents. Both local-word scoring and seed generation will be needed in the next process, that is to measure paragraph similariry.

4.3.2.2 Paragraph Similarity Measure

The next step is to measure similarity between each paragraph in a suspicious document and each paragraph in every candidate document. This involves selection of similarity measures which is based on three considerations. Firstly, the similarity measure should be capable of accomodating local comparison on the scope of a paragraph. Secondly, paragraph pairs outputted from this process should cover pairs of source and reused texts with different kinds of obfuscation types. Thirdly, the order of reused terms should be ignored. In other words, the similarity metrics applied should accommodate the Bag-of-word model. To achieve these goals, every compared paragraph will be represented as both binary and weighted vectors, and two different similarity measures were used to complete each other.

Dice coefficient was selected to be one of similarity measures as 'Dice and Cosine are some of the best corpus-based measures' [159]. Besides, Dice coefficient is a flexible measure which could be applied to compute both binary and weighted vectors in local or global environment setting. In this task, Dice coefficient was implemented as a local similarity metric which was aimed to capture text reuses containing obfuscation on the level of paraphrase and summary. Assuming that matching paraphrased and summarized text reuse needs only a handful of significant keyterms, Kiabod's local word scoring is applied to weigh terms, and his method of significant word selection is used to generate suspicious paragraph queries. Having weight for each term and queries, the similarity between paragraph queries and paragraphs in source document could be computed using equation 4.4 which was borrowed from [30].

$$S_{Dice} = \frac{2\sum_{i=1}^{d} P_i Q_i}{\sum_{i=1}^{d} P_i^2 + \sum_{i=1}^{d} Q_i^2} \tag{4.4}$$

where P_i stands for a candidate paragraph vector, $\vec{P}(par)$, and Q_i represents the paragraph query vector, $\vec{Q}(par)$. In applying Dice coefficient, queries representing a suspicious paragraph are formed from seeds which are weighted through local-word weighting as shown in equations 4.2 and 4.3.

The second similarity metric is meant to capture as many similar terms as possible. This is to anticipate text reuses with obfuscation from the types: *copy and paste*, *shake and paste* or near-duplicate. The simple but famous Jaccard coefficient was used to serve this purpose. As a binary similarity metric, Jaccard coefficient computes similarity of two sets by the size of their non-zero shared values (or overlapped seeds) divided by the size of the union of both sets as seen in equation 4.5. The strengths of Jaccard coefficient lie on its simplicity and its nature that penalizes a small number of shared terms by lower values [98]. In External Plagiarism Detection, Jaccard coefficient is commonly applied in applications using fingerprinting method as document representation as it could be found in [79, 80, 108, 145, 194].

$$S_{Jaccard}(par_{dsrc},\ par_q) = \frac{|\ par_{dsrc}\ \cap\ par_q\ |}{|\ par_{dsrc}\ \cup\ par_q\ |} \tag{4.5}$$

where par_{dsrc} refers to a set of unique terms in a paragraph of a candidate document, and par_q refers to the set of unique terms in a suspicious paragraph.

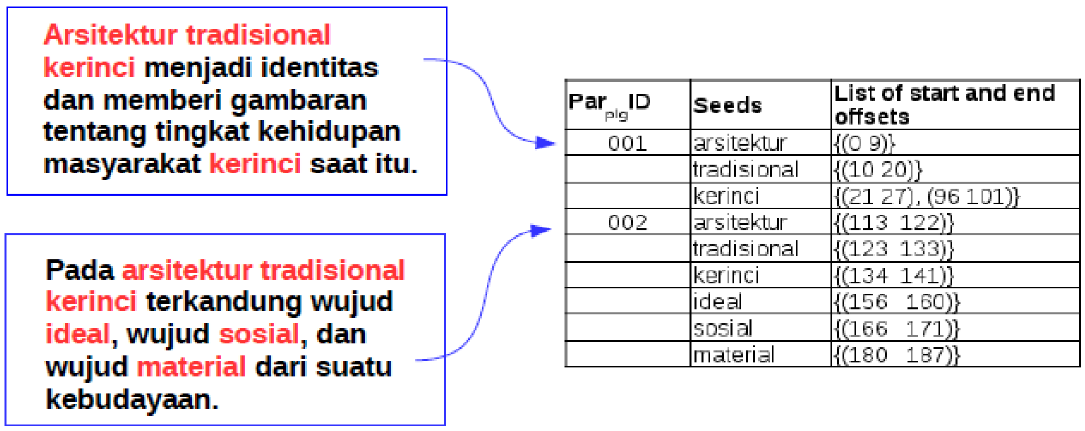
The outputs of paragraph similarity from both coefficients are formulated into array of arrays where the information on the paragraph ID in suspicious document (par_{plg}ID), source document ID (d_{src}ID), and paragraph ID in source document (par_{src}ID) are mapped as array keys and the similarity score as values. the ranked similarity scores were obtained by sorting these arrays according to their values. The paragraph pairs were then filtered by setting up threshold for each similarity coefficient. The threshold values become subjects of experiment and we came to the constants: 0.35 for Jaccard and 0.4 for Dice coefficient thresholds. Only pairs of paragraphs whose score above these thresholds would be saved and filtered for their uniqueness since it was highly probable that Jaccard and Dice outputted the same pairs of paragraphs. As the similarity score is not needed any more, it is discarded and the information that is saved for further process is the triple of par_{plg}ID, d_{src}ID, par_{src}ID which is as follow:

$$\{\ [0] \rightarrow (2, 1984, 5),\ [1] \rightarrow (3, 1756, 1),\ \ldots\ ,\ [9] \rightarrow (10, 1875, 22)\ \}$$

4.3.2.3 Seed Processing

On the next step, seeds are processed for matching similar parts of paragraph to build larger passages and to set up the boundary of these similar passages. The seed processing covers seed matching, seed merging and seed extension. For the sake of seed matching, a seed index is created right after the seed generation process. The seed index is generated in real-time and stored as an array of arrays as it is used repeatedly for matching seeds of different candidates documents. However, it is deleted when different suspicious document is inputed to the system. It needs to be noted that this seed index is created for suspicious document only.

A specific function which finds all occurrences of seeds and computes their start and end offsets within each paragraph was constructed. Then, the start and end offset of every seed on the level of document was computed by adding these offsets to the start of paragraph offset in which these seeds occur (see section 4.3.2.1 for paragraph offset generation). The final information saved in the seed index comprises the suspicious paragraph ID (par_{plg}ID), seeds, seeds start and end offsets on the level of document. Table 4.6 illustrates this real-time seed index where par_{plg}ID and seeds are mapped as array keys and tuples of start and end offsets are saved as array values.



Arsitektur tradisional kerinci menjadi identitas dan memberi gambaran tentang tingkat kehidupan masyarakat kerinci saat itu.

Pada arsitektur tradisional kerinci terkandung wujud ideal, wujud sosial, dan wujud material dari suatu kebudayaan.

Par_{plg}ID	Seeds	List of start and end offsets
001	arsitektur	{(0 9)}
	tradisional	{(10 20)}
	kerinci	{(21 27), (96 101)}
002	arsitektur	{(113 122)}
	tradisional	{(123 133)}
	kerinci	{(134 141)}
	ideal	{(156 160)}
	sosial	{(166 171)}
	material	{(180 187)}

Figure 4.6: An example of seed index for a suspicious document with 2 short paragraphs

Seed matching is carried out by looking up the seed index and the array of filtered paragraph pairs as outputs of paragraph similarity. Only seeds from suspicious paragraphs whose IDs are listed in the array values are extracted from the seed index and are used to match seeds on the referred paragraphs of candidate documents. Using the same function for building seed index, the start and end offsets of matched seeds in referred paragraphs of a candidate document are saved in another temporary seed table. Thus, two separate temporary seed tables are generated as a result of seed matching. The first temporary table is akin to subset of the seed index as it contains any information on matched seeds only from filtered paragraphs of suspicious document. The second temporary table contains all information of matched seeds of par_{src} in candidate documents. Both tables have the same data structure as seed index displayed in figure 4.6. The difference is that the temporary seed table of candidate documents has one more dimension of array for saving documentID, $\text{d}_{\text{src}}\text{ID}$.

The computation of seed merging is performed by looking up these two temporary tables, verifying the defined rules and parameters. The rules and parameter setup are based on the following considerations:

1. **Giving space for any context modification**. The seed merging in this model should be able to capture any modification such as wording in paraphrased cases, deleting, replacing some words with others, and shuffling the word order.

2. In **defining a gap** between seeds, it should heed the scope of merging which is inside a paragraph and not a section of a text. The gap between seeds should not be to large even longer than a length of a short paragraph.

3. **Avoiding seed repetition**. Some specific seeds may occur repetitively in different passages of a document. In some cases, their repetitive occurrences unnecessarily imply a text reuse, if their context conveys different ideas. The defined rules for seed merging should be able to excludes paragraph pairs containing the seed repetition which indicates no text reuses.

Based on these considerations, seed merging was performed in a two-step merging process. The merging algorithm takes a seed table and three parameters as inputs. The three parameters are distance gap between individual seeds (α), the length of merged seeds (len), and the distance gap between the merged seeds (β). The whole merging process is shown in algorithm 1. The first to nineth line of algorithm describe the first step of merging which starts by sorting seeds in ascending order according to their start offsets. Then the distance between neigbouring seeds was calculated by substracting the end offset of the current seed ($seed_n$) from the start offset of its successor ($seed_{n+1}$). Owing to considerations mentioned before, α is set to different values, where a longer gap is allowed between individual seeds of the suspicious paragraphs. After some empirical experimentations, we came to a combination of 50 character gaps for candidate document seeds and 35 characters gaps for suspicious document seeds. The algorithm merges seeds whose gap is less than or equal to α.

Algorithm 1 Seed Merging Algorithm

```
Input:  $S \leftarrow$ seeds of a $par_{plg}$, $\alpha$, $\beta$, $len$
Output:  merged seeds
  for all S do
    $sortedS \leftarrow$ sort(S)
  end for
  for $a = 0$ to $\mid sortedS \mid -1$ do
    $gap \leftarrow$ computeGap(sorteds, $sorteds + 1$)
    if $gap < \alpha$ then
      $sorteds + 1 \leftarrow merge(sorteds, sorteds + 1)$
      $unset(sorteds)$
    else
      $MergedS \leftarrow sorteds$
    end if
  end for
  for all MergedS do
    $lenMs \leftarrow$ length(mergeds)
    $gapMs \leftarrow$ calculateGap(mergeds, $mergeds - 1$)
    if $lenMs > len$ AND gapMs $< \beta$ then
      $mergeds \leftarrow merge($mergeds-1$, mergeds)$
      unset($mergeds - 1$)
    end if
  end for
```

The α values defined on the first merging produced short sequences that needed to remerge, if longer sequences of text reuse are required as final outputs. This is intentionally done as a longer gap will result in greedy seed merging. In the second step of merging, the algorithm takes the outputs of the first merging process and remerges the short merged seeds on the basis of defined rules, i.e. only seed sequences whose lengths are above threshold (len) and whose distances are within the defined gap (β) will be remerged. The sequences which do not fulfill those two parameters are discarded. This time, the β value is set to be equal for the seed gap of suspicious and candidate documents. These parameter values become subject of experiment.

The two-step seed merging process is an internal process within a filtered paragraph in our temporary tables, which outputs longer sequences for that particular paragraph. Taking an example from table 4.6, par_{plg}ID 001 has 4 seeds, three of them are mergeable, while the gap of the 4^{th} seed to the third one is beyond the gap to merger, then it is left unmerge, and the start and end offsets of the merged seeds of par_{plg}ID 001 lie between 0-27, while the start and end offsets of par_{plg}ID 002 are 113-187

The next process is to couple these longer sequences into a pair of source and suspicious sequences. This is done by looking up the array outputted from paragraph similarity process. Assuming that $\mathrm{par_{src}}$ID 002 and $\mathrm{par_{plg}}$ID 005 are listed as a value pair in this array, then the offset sequences of $\mathrm{par_{src}}$ID 002 will be coupled to offset sequences of $\mathrm{par_{plg}}$ID 005. Considering that some paragraphs have more than one sequence, a set of coupling rules need to be defined. The rules are set to simply couple sequences within paragraph pairs, if these paragraph pairs have the same number of short sequences. If one of the paragraph pairs has only one sequence and another has more sequences, then this only one sequence will be coupled to all sequences of its paragraph pair. If both paragraphs in this pair have unequal number of sequences, only sequences whose length is over 100 characters will then be coupled to exactly one sequence with the same length criteria in its corresponding paragraph pair. This last rule functions also to filter short sequences being coupled. The outputs are saved as array of arrays consisting information on d_{src}ID, start, end offsets and sequence length of $\mathrm{par_{src}}$ID, start, end offsets and sequence length of $\mathrm{par_{plg}}$ID.

The seed extension processes further the similar-identified sequence pairs outputted from seed merging. The rational is that these merged sequence pairs identify only similar sequences within a paragraph. To capture the possibility of similar sequences over a paragraph scope, the seed sequence pairs need to be extended if their conditions fulfill the requirements defined. The seed extension algorithm is based on the relation matches defined by Alvi et al. in [7] which identify four categories of matches. These four relations of matches are:

1. **Containment** identifies a match within another match. Assuming that we have two pairs of matches or merged sequences with $\{(s1, e1, l1) \rightarrow (a1, b1, ln1), (s2, e2, l2) \rightarrow (a2, b2, ln2)\}$ where s, e, l stands for the start, end offsets and length of sequence in the source, while a, b, ln refer to the same things but for suspicious document. The second matched pair is said to be within the first matched pair if

$s2 \geq s1, e2 \leq e1, and\ l1 \geq l2$ [7].

2. **Overlap** describes a condition where only a part of a match is within another match. Two pairs of merged sequences are said to be overlapped if $e2 \geq e1 \geq s2 \geq s1$ [7].

3. **Near-disjoint** identifies pair 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.

Owing to paragraph-based merging technique, each possible variation of these four relations does not always occur on our pairs of merged sequences. Thus, the extension algorithm extends only merged sequence pairs with near-disjoint relation occurring in the source document. The extension operation depends on the relation category. For near-disjoint relation, the extension is performed by taking the start offset of the first sequence pair (s1), the end offset of the second sequence pair (e2), and subtracting the start offset of the first sequence from the end offset of the second sequence pair ($l2 - s1$). Table 4.3 describes the extension strategies by presenting the possible relations for source and suspicious merged seed sequences, the extension action, and the possible plagiarism cases covered by such a relation. The writing style in table 4.3 adapts table 1 in [7].

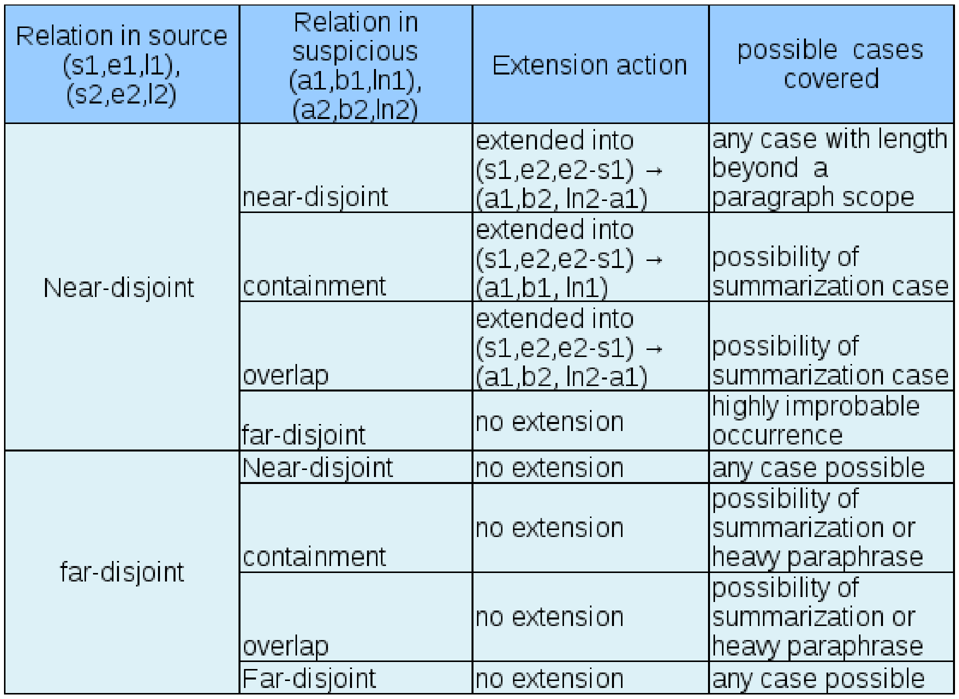
The output of the seed extension process is saved to the same array as its input. The difference is that this array has fewer number of similar-identified sequence pairs but they are much longer. This is made possible by replacing the two sequence pairs being compared with their new extended set of information. If the condition for extension is not met, no extension action would be performed.

4.4 Post-Processing

The post-processing in this system is aimed to filter any detected sequence pairs which are too short, as they often lead to a high false positive. Discarding them from the detection list could improve the precision rate. 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.

The end result of detection is formulated in xml format. An xml file consisting all detected passage pairs is generated for a single given suspicious document. The aim of writing the final output in XML file is to ease the evaluation process. The meta data of test documents consisting artificial and simulated plagiarism cases are written also in XML format. The visualization of the end output in an interactive Graphical User Interface (GUI) is left to a future work, since it needs separate and different methods to visualize the result. With an xml format of the output, the mapping of the detected sequence pairs to a visualized version in both documents will be made easier. Figure 4.7 displays the final

Table 4.3: Possible relations of matches and the seed extension strategy



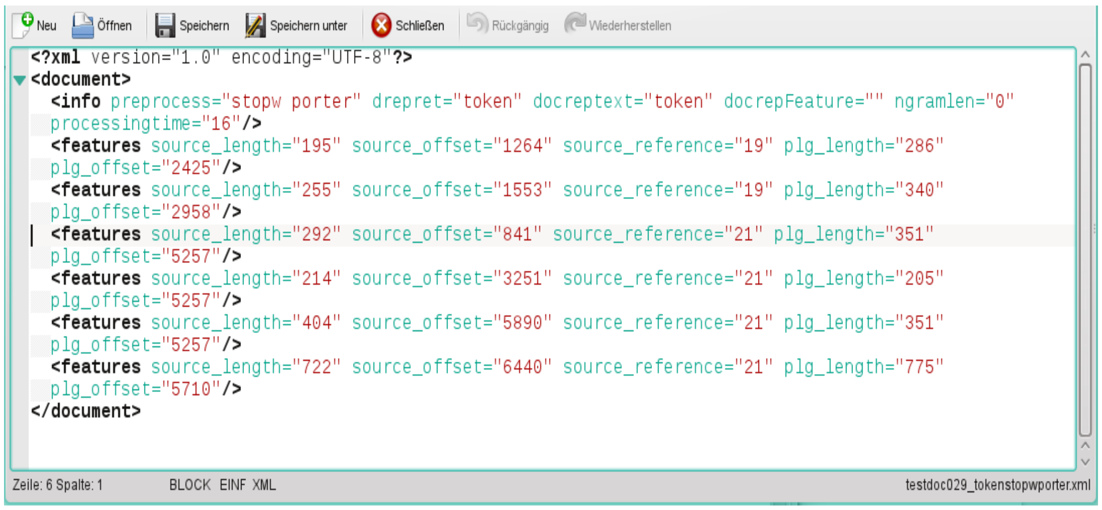
Relation in source (s1,e1,l1), (s2,e2,l2)	Relation in suspicious (a1,b1,ln1), (a2,b2,ln2)	Extension action	possible cases covered
Near-disjoint	near-disjoint	extended into (s1,e2,e2-s1) → (a1,b2, ln2-a1)	any case with length beyond a paragraph scope
	containment	extended into (s1,e2,e2-s1) → (a1,b1, ln1)	possibility of summarization case
	overlap	extended into (s1,e2,e2-s1) → (a1,b2, ln2-a1)	possibility of summarization case
	far-disjoint	no extension	highly improbable occurrence
far-disjoint	Near-disjoint	no extension	any case possible
	containment	no extension	possibility of summarization or heavy paraphrase
	overlap	no extension	possibility of summarization or heavy paraphrase
	Far-disjoint	no extension	any case possible

output of detection in the xml file format which saves not only information on the detected sequences, but also on preprocessing methods, document representations used for retrieval and detection phases, and the processing time in seconds.

4.5 Summary

This chapter describes the proposed framework for Indonesian External Plagiarism Detection. Inspired by the architecture and techniques from researches on the EPD, this framework addressed the problems of plagiarism detection in two main subtasks: the retrieval and text alignment. The third subtask of an External Plagiarism Detection system, the Post-processing, functions as filtering process of the detection outputs of text alignment subtask.

In addressing the retrieval problem referred to as the research problem number 1.1 - 1.3 (cf. section 1.2), three different document features were implemented as document representations. It introduces the use of phraseword as a document feature [87], which is meant to capture phrases and word sequences in one single feature, while giving space for any modification within these phrases. These phrasewords are implemented and experimented along with character n-grams and word unigrams. The shallow Natural Language Processing (NLP) techniques are applied for text and token normalization, in which two different kinds of stopwords are applied. The segmented-based query formulation introduces the



```
<?xml version="1.0" encoding="UTF-8"?>
<document>
  <info preprocess="stopw porter" drepret="token" docreptext="token" docrepFeature="" ngramlen="0"
  processingtime="16"/>
  <features source_length="195" source_offset="1264" source_reference="19" plg_length="286"
  plg_offset="2425"/>
  <features source_length="255" source_offset="1553" source_reference="19" plg_length="340"
  plg_offset="2958"/>
  <features source_length="292" source_offset="841" source_reference="21" plg_length="351"
  plg_offset="5257"/>
  <features source_length="214" source_offset="3251" source_reference="21" plg_length="205"
  plg_offset="5257"/>
  <features source_length="404" source_offset="5890" source_reference="21" plg_length="351"
  plg_offset="5257"/>
  <features source_length="722" source_offset="6440" source_reference="21" plg_length="775"
  plg_offset="5710"/>
</document>
```

```
Zeile: 6 Spalte: 1          BLOCK  EINF  XML                    testdoc029_tokenstopwporter.xml
```

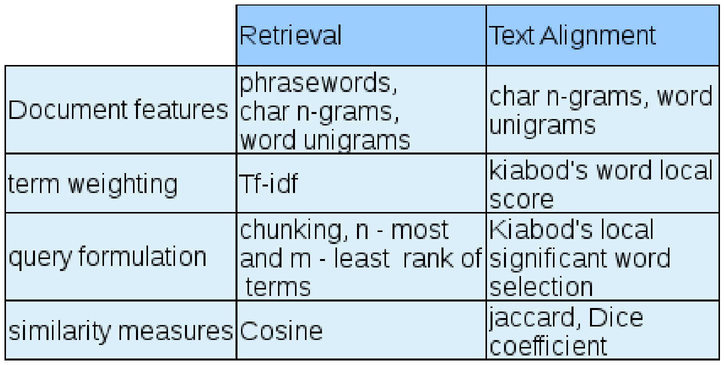
Figure 4.7: The final output in an XML file format

idea of formulating queries formed from terms with highest and lowest scores within a segment. Finally, Cosine similarity which is one of the best global similarity measures was used to compute the similarity between suspicious and source documents in the corpus.

In addressing the problem of text alignment, this framework implements paragraph-based alignment in which the seeds are constructed to perform dual functions, as seed heuristic match and as suspicious paragraph queries. Seeds are generated by implementing word local scoring method and local-word score pruning proposed by Kiabod et al. [81] for text summarization. The paragraph comparison was done by using two different represesentations, character n-grams and word unigrams which are then projected as weighted and binary vectors. Two local similarity measures were utilized for this purpose: Jaccard coefficient and Dice similariry. The seed matching process is applied only to the similar-identified paragraph pairs. The seed merging within a paragraph were performed by looking up the dynamic seed tables generated during seed matching process. The seed extension strategy is based on the folowing relations: containment, overlap, near-disjoint, and far-disjoint.

Our paragraph-based alignment techniques result in no overlapping and repetitive detections, so that the post-processing task is much easier. It removed passage pairs whose length are less than 125 charaters for source passages aligned to suspicious passages whose lengths are less than 150 characters. To sum up, table 4.4 presents the summary on the methods used in each subtask and the framework which was formerly designed to be specially applied to Indonesian texts which turns to be universal and language-independent.

Table 4.4: A summary on the methods used in this framework



	Retrieval	Text Alignment
Document features	phrasewords, char n-grams, word unigrams	char n-grams, word unigrams
term weighting	Tf-idf	kiabod's word local score
query formulation	chunking, n - most and m - least rank of terms	Kiabod's local significant word selection
similarity measures	Cosine	jaccard, Dice coefficient

Chapter 5

Corpus Building and Evaluation Framework

Evaluating the performance of an External Plagiarism Detection System requires at least two things: a set of documents as corpus and evaluation framework. Chapter 5 deals with these main issues. The process of building evaluation corpora for an External Plagiarism Detection systems will be presented in section 5.1. Firstly, this section will review corpus building in some EPD systems and PAN PC workshops, then it describes the strategies used to build PlagiarIna's evaluation corpus. Having the same structure as its former section, section 5.2 presents some measures employed to evaluate EPD systems in general especially in PAN PC workshops, then it discusses the concepts and measures used for evaluating every main phase in PlagiarIna, the retrieval and text alignment.

5.1 Evaluation Corpus Building

5.1.1 A Survey on Evaluation Corpora

Building an External Plagiarism system includes designing and building corpus for measuring its performance. Internationally, there have been two institutions only which continually evaluate Plagiarism Detection systems, and systematically issue reports on methods of their evaluation corpus building as well as the evaluation results (cf. section 2.2.2). Owing to the fact that there has been no publicly available testbed for evaluating the performance of External Plagiarism Detection (EPD) system for Indonesian texts, this study turned to these two institutions as models for comparison. A survey on corpus building strategies on EPD researches for Indonesian texts was conducted as well. In our survey, we determined four parameters to be observed, they are strategies on corpus building, corpus acquisition, comparison task, and corpus size. The following subsections present firstly the survey on EPD researches for Indonesian text, then followed by reviews on the evaluation corpus building in PAN and HTW research center, Berlin.

5.1.1.1 Evaluation Corpora for Indonesian EPD

Based on the language of texts being processed, published researches on EPD conducted in Indonesian universities or by Indonesians could be classified into two groups: researches which worked on documents in English and in Indonesian. In this section, we will not review all systems mentioned in section 3.4, instead we surveyed 8 researches only due to information sufficiency reported for building their evaluation corpus. Unfortunately, three [135, 157, 173] out of eight surveyed researches processed documents in Indonesian and the rest dealt with documents in English [2, 4, 147, 166, 181]. In term of corpus acquisition, two out of five researches which processed English documents built their own corpus, while two utilized the available standardized corpora. Vania & Adriani evaluated their algorithm by using PAN Corpus 2010 and translated non-English documents into English [181], Sediyono employed data set from TREC and RFC collection [147], while Mahathir used Clough and Stevenson corpus which were then translated into Indonesian [94]. Adam and Suharjito used articles from a journal but did not provide any information from which journals they derived their articles [2]. Purwitasari et al. who worked on Indonesian texts acquired their evaluation corpus from students' courseworks taking Socio-Ethic course [135]. Soleman Purwariati [157], and Suryono et al. [173] used articles for their source documents and provided no information on how to acquire them. These eight systems do their comparison task locally, meaning that a suspicious document would be checked against their local databases.

The size of evaluation corpora in researches mentioned before ranges from 5 to 100 documents for source documents and 4 to 95 for test documents. The exception falls to Suryono et al. who used 10.000 articles as source documents [173]. Among those who build their own evaluation corpora, the considerations for building test documents cover the number of source documents to be plagiarized in one suspicious document, the percentage of plagiarism, and the obfuscation types. The number of source documents for one test document could range from 1 to N in which the maximum number of N ranges from 2-5 documents or it was not clearly stated as in [4, 157]. The percentage of plagiarized texts in one test document is hardly comparable and reported, as some reported it qualitatively such as *few sentences are added or deleted* without providing its proportion to the length of suspicious or test documents.

The plagiarism types for doing obfuscation in test documents could be categorized into four groups: no-obfuscation, near copy, paraphrase, and summary. The obfuscation types of paraphrase and summary were found only on the work of Soleman and Purwarianti [157] among those who worked on Indonesian texts. Using different terms, Mahathir defines the obfuscation level into *light* and *heavy revisions* [94]. The light revision refers to a copied text which still resembles its original, while *heavy revision* refers to a paraphased text. Further information about the evaluation corpus of these surveyed EPD researches is presented in table 5.1.

One out of three researches working on Indonesian texts provides sufficient information on how to do obfuscation for its test or suspicious documents. Soleman and Purwarianti generated 6 types of test cases in which the test cases 1-3 are a form of verbatim copies [157]. The differences are set on the number of source document, where in test case 1, the whole test document is taken from one source only, and more than one source documents in test cases 2 and 3. In test case 3, the order of sentences and paragraphs are shuffled. Test case 4 deals with paraphrase, while test case 5 deals with summarized obfuscation. The last test case contains literal copies from documents not available in the source document corpus. Unlike Soleman and Purwarianti, Suryana et al. created verbatim copy only

Table 5.1: Comparison on evaluation corpus aspects of Indonesian EPD sytems. In this table, **NA** stands for No available information, **INA** stands for Indonesian, **Eng** refer to english, and **local** is meant to refer to offline comparison as opposed to online one. The sign # is used to refer to the word *number*.

	Found in							
	[135]	[181]	[147]	[2]	[4]	[94]	[173]	[157]
Source doc #	60	27.053	NA	100	10	5	10000	47
Test doc #		65.558 cases	NA	20	4	95	NA	6
Corpus acquisition	Student course-works	PAN 2010	TREC & RFC	a jour-nal	NA	Clough & Steven-son corpus	NA	NA
language of texts	INA	Eng	Eng	Eng	Eng	Eng	INA	INA
Experimented Task	local	local	local	local	local	local	local	local

for test documents by taking only some paragraphs from one source or from more than one source documents [157]. Suryana et al. provided no further information on the test documents, except on the source document length which ranges between 200-1100 words. Purwitasari et al. did not make any distinction between source and test documents. All 60 documents in their corpus were compared to each other. It seems that the system would be meant to do a cross-check among students' work [135].

Unfortunately, most surveyed researches did not provide information on who did the obfuscation for the test documents. It is quite possible that the test documents and their obfuscation are generated by the researchers themselves. If this assumption is true, then the possibility of bias could not be avoided, since the obfuscation complexity will definitely affect the evaluation result.

5.1.1.2 PAN Evaluation Corpus

The lack of an evaluation framework and the need of developing it for PAN workshops led Potthast et al. to conduct a systematic survey on the state-of-the-art in evaluating plagiarism detection [131]. This survey examined 275 research papers which deal with detecting plagiarism in natural language or text and programming codes, but it analysed in-depth 205 papers. A review on that survey presented here concerns with the statistical data of plagiarism detection dealing with texts in natural language only. 80% of the surveyed papers did the experiment task by comparing the suspicious document against their local database, 15% used web retrieval and 5% was not clearly stated [131]. This data correlates with the corpus acquisition which shows that 80% of them built their own database, while 20% used the availaible data. For the corpus size, most paper have $10^2 - 10^3$ documents, only 8% have a $10^5 - 10^6$ documents in their corpora. Interestingly, 11% of these researches built their corpora with 1-10 documents only [131]. Further, Pothast et al. reported that there is a tendency in which a small corpora is commonly built from student course works or term papers, while documents for a large corpora were derived mostly from news wire articles or from 'sources where text overlap occurs more frequently'[131]. Implicitly, this survey defined that a small corpus for experimenting plagiarism detection system comprises less than 10^3 documents, while corpus containing more than 10^3 documents would be considered as a large corpus.

Based on the building strategy and experiment task, PAN evaluation corpus could be categorized into two groups. Evaluation corpora used in the $1^{st} - 3^{rd}$ PAN PC workshops belong to group 1 or the first group, and the second group comprises corpora used in $4^{th} - 6^{th}$ PAN PC workshops. The corpora on group one were designed for evaluating the performance of an EPD locally as a whole system, while the corpora in the second group were aimed to evaluate separately each subtask of EPD system, i.e. the retrieval and text alignment subtasks. The corpora in the first group were built from documents derived from Gutenberg project [124]. the corpora in group 1 share the same source-to-suspicious document ratio in which 50% documents of these corpora are labelled as source documents and 50% are designed to be suspicious documents. 50% of the suspicious documents or 25% of the whole corpus are documents with no plagiarism cases at all, and the rest are documents containing plagiarism cases [124–126]. These corpora share also document length composition, in which 50% of it are short documents with 1-10 pages length, 35% of them are medium documents which are defined to have length between 10-100 pages, and the rest are long documents with $10^2 - 10^3$ pages. One more commonality among these corpora is that the plagiarism language for mono-lingual detection is English, while languages for cross-lingual detection covers German and Spain which are then translated into English.

The obfuscation strategies for suspicious documents among PAN corpora in the first group differ slightly. In the 1^{st} PAN's test document corpus, all suspicious documents were generated algorithmically by considering the bag-of-words model. Three heuristic operations were used to construct plagiarized passages S_{plg} from source passages S_{src} [124]. These three heuristic operations cover: text random operation, semantic word variation, and shuffling words in random while maintaining the order of their Part of Speech (POS) [126]. This obfuscation strategy for generating artificial plagiarism cases has been applied from the 1^{st} to 5^{th} PAN's evaluation corpora [125–128]. Started in 2^{nd} PAN, a variety of obfuscation strategies have been employed. Besides artificial obfuscation, simulated plagiarism cases and automatic translation from German and Spain to English were introduced in 2^{nd} PAN's test document corpus [125]. Simulated obfuscation refers to a technique of creating plagiarism cases by purposeful modifications which are performed by human writers. In $2^{nd} - 3^{rd}$ PAN PC, the simulated obfuscation was done through crowd sourcing in Amazon's Mechanical Turk [126, 131].

Besides the obfuscation strategies, the percentage of plagiarism per document, the corpus size, and the length of plagiarism cases varied among PAN corpora in the first group. On 2^{nd} PAN's corpus, 15% of suspicious document corpus contains documents which are almost entirely obfuscated or heavyly obfuscated, and 45% of the corpus contains documents with light obfuscation (cf. section 2.1.1). The heavy-light-obfuscation ratio in 3rd PAN corpus is around 57% to 10%. Note that the number of plagiarism cases are distinguished from the number of suspicious documents in PAN corpora. A *plagiarism case* is used to refer to a passage containing any types of obfuscation in a suspicious document. One suspicious document may contain several plagiarism cases depending on the percentage of plagiarism per document, as for an example in 2nd PAN's corpus, there are 27.073 suspicious documents with total 65.558 plagiarism cases [125].

Different strategies of corpus building were applied to the second group of PAN corpora as a consequence of an effort to mimic a real word scenario in detecting plagiarism. Started from 4th PAN, there have been two different corpora in which one corpus serves to evaluate retrieval subtask performance, and another is aimed to assess text alignment subtask. The evaluation corpus for retrieval subtask is a kind of web simulation consisting of a large scale web documents which can be searched and browsed as if it was a real web. The source documents were derived from the ClueWeb09 data set which were grouped into N source sets where N refers to the number of topics chosen randomly from TREC topics and equals to 520 in 5th PAN retrieval corpus [128]. The test documents were generated by hiring professional writers. Each writer was allowed to choose a topic once only. Based on these topics, writers searched for sources in the web corpus (or source sets) then reused texts from the retrieved source documents for composing a suspicious document [127–129]. The evaluation scenario starts by submitting queries which are formed from a given suspicious document to one of the following search engines: Indri or ChatNoir [128]. The search would be redirected to servers hosting the source document corpus instead of the real web [128].

The strategies of building corpora for 1st − 3rd PAN PC were simply transferred to building evaluation corpora for text Alignment subtask during 4th−6th PAN PC workshops. In term of obfuscation strategies, the 5th − 6th PAN corpora introduced cyclic translation and summary obfuscations whose source documents were taken from Duc 2001 [128]. Unlike in 1st − 5th PAN corpora whose artificial plagiarism cases were generated using deep NLP techniques, the artificial obfuscations for 6th PAN corpora were done by implementing a naive approach of obfuscation. The resulting passages from such approach bear no semantics and are hardly readable [127, 129]. The statistic summary on the 1st − 6th PAN corpora is presented in table 5.2.

5.1.1.3 HTW Evaluation Corpus

Since 2004, the research center at HTW, Berlin, has conducted totally 8 software tests to the commercial Plagiarism Detection systems. The last test which was conducted in 2014 was labelled as partial test and used the same test data set from the former one conducted in 2013 [18]. In all these tests, only suspicious documents were created as evaluation corpus,

[18]Further information on the software test of HTW Berlin is available in `http://plagiat.htw-berlin.de/software-en/`

Table 5.2: Comparison on PAN evaluation corpus. In this table, **NA** stands for 'no available information', while **none** is used to refer to the absence of obfuscation, or *no-plagiarism.*

	1st PAN	2nd PAN	3rd PAN	4th PAN	5th PAN	6th PAN
Corpus size:						
Total doc #	41.233	27.073	26.939	NA	NA	NA
Plagiarism Cases	94.202	65.558	61.065	3.033	6000	6000
Topic #				NA	144	144
Retrieval				300	297	297
Text Alignment				NA	8.427	8.427
Corpus acquisition	Gutenberg project	Gutenberg project	Gutenberg project	Gutenberg project Clue Web09	Gutenberg project Clue Web09 Duc01	Gutenberg project Clue Web09 DUC01
Experimented task	local	local	local	web simulation	web simulation	web simulation
Obfuscation	artificial	artificial, simulated translation	none paraphrase, translation	artificial, simulated, none, real cases	random, cyclic translation, summary	random, cyclic translation, summary, verbatim

since most commercial EPD systems have developed their own databases for source documents. Each suspicious document, known as a test case, is numbered in ascending order. The numbering system continues if new test cases are added for the following software tests. The review on building the test cases or evaluation corpus of HTW Berlin reported here is based on a report which was issued for the software test 2013.

The generated test cases have been mostly short hand-written texts with 1 to 2 page length [185]. For the software test 2013, 20 new test cases which were written by a single student were added to the test case corpus. The corpus includes two long test cases which were constructed by generating random text and inserting plagiarized text from the smaller test cases [185]. These long test cases having 40 & 80 page length are aimed to represent a Bachelor's and Master's theses in testing system performance for longer texts [185]. As for the language of the text, 5 out of 20 new test cases were written in English and the rest are in German. 2 additional test cases written in Hebrew from former test cases were inlcuded also to the 2013 test set. In total, the software test 2013 run 35 test cases [185].

The obfuscation strategies for the test cases cover *Copy and Paste*, *Shake & Paste*, disguised plagiarism, translation, structural plagiarism, and 'Pawn sacrifice' [185]. In addition, a homoglyph trick was introduced as one of obfuscation techniques which was conducted by replacing letters with charaters of non-latin alphabet which look almost similar but have different internal representations. The sources of HTW test set are more diverse as test cases are plagiarized from articles or documents downloaded or taken from

Wikipedia, Sueddeutsche Zeitung, medical journal articles, Kafka's writing, Tronixstuff, Newadvent, Google Books, and a real plagiarism case. Two test cases which were based on sources found in Google Books were aimed to test EPD system's performance in recognizing scanned texts [185]. Unfortunately, there is no information on the percentage of plagiarism per document in their test cases. Table 5.3 summarises HTW's test corpus.

Table 5.3: Summary of HTW's test document corpus

Corpus size:	
Number of test Document	72
Evaluated test document	35
Test document length	
short [1-2 pp.]	97,2%
medium [40, 80 pp]	2,8%
Language of texts	English, German, Hewbrew, Japanese
Obfuscation strategy	Copy & paste, disguised plagiarism, translation, structural plagiarism, pawn sacrifice, homoglyph trick
Corpus acquisition	Wikipedia, Google Books, medical journal articles, Tronixstuff, Newadvent, Sueddeutsche Zeitung, a real plagiarism case

5.1.2 Evaluation Corpus Building for PlagiarIna

As there has been no standardized corpus for evaluating plagiarism detection systems for Indonesian texts, the process of corpus building for evaluating PlagiarIna was influenced much by the first group of PAN PC corpora building and HTW reserach center, Berlin. This influence could be seen especially on the techniques applied to create plagiarism cases in suspicious documents. However, the corpus acquisition for source documents differs significantly since it was based on a real use case of a plagiarism detection system for academic purposes. The following subsections explain in detail the process of evaluation corpus building for PlagiarIna which comprises source and suspicious documents.

5.1.2.1 Building Source Document Corpus

Based on the scope of this study (cf. section 1.2) and the use case of PlagiarIna, texts which were selected as source documents take the form of bachelor theses, scientific articles or papers for proceedings and journal submission, summary of bachelor theses in the form of articles, popular scientific articles appearing in online magazine and outstanding newspapers, articles appearing in personal blocks, and handouts or lecture scripts. However, articles become the dominant form of the source documents. Those texts were acquired in two ways as follows:

1. **Manual aquisition**. Having full access to the archive of Duta Wacana Christian University (DWCU), some bachelor theses submitted in 2011-2012 and articles with topics about Information Technology, Architecture, and Theology were selected randomly and manually.

2. **Automatic web grabbing**. Many articles were grabbed from specifically defined websites such as article directories, portals of Gunadarma univeristy, UNS university, Indonesian national geography, and some personal blocks hosting popular articles on the topic of history which can be found in *Pendidikan Sejarah* blogspot, or *Wiryanto* blogspots which posts many articles about civil engineering.

The complete list of URL websites as sources of web grabbing activity could be found in table B.1 in Appendix B. The process of collecting source documents was done in July 2012 which resulted in 4950 documents[19].

These documents underwent a selection process which was done manually by skimming through the texts. Some considerations used for file selection are the document length and content. Short documents having less than 2 paragraphs or a page length were discarded. The assessment on the document content was based on the text genre and intuitive decision. If the grabbed articles turned out to be news on scientific or research discoveries, then they were also discarded. An intuitive decision which judge whether the texts have potential to be plagiarized in student paper assignments and theses was used to include files into the source document corpus. This explained why the lecture scripts provided in some personal blocks were included in this corpus. After the selection process, the total number of source documents in the corpus remains 2014.

The filtered and compiled source documents had various file formats. For the sake of indexing, the files needed to be converted into plain text format, and the file format conversion occurred in 2 batches. The first batch of file conversion utilized file format converter available in the Internet such as Go4convert or Zamzar applications[20]. One of drawbacks of using online file converters is that each application accepts only one document as an input, and thus it took a lot of time for doing file conversion. This drawback motivated us to write a specific PHP script for file conversion which enables a user to upload and convert several documents at once and sends the converted files directly into a server. Thus, the second batch of file conversion was done through a PHP script.

The source documents were selected on the basis of different areas of study such as Information technology, Medicine, Theology, etc. These study areas were used to label the source document names, as text classification was used to be a part of PlagiarIna's modules. When the idea of using text classification was dropped, the source document labels remain unchanged. In total, there are 21 areas of study to categorize the source documents. Each category has different number of documents. The proportion of source documents in each category is presented in table 5.4. As for document length, PAN classifies documents

[19]The process of source document collection and compilation was done by Eka Cahyandaru, an employee at Computer Lab, IT Department, DWCU, Yogyakarta, Indonesia

[20]available at `http://go4convert.com/ToTxt` and `http://www.zamzar.com/convert/pdf-to-txt/`

having 1-10 pages as short texts, medium texts have approximately 10-100 pages, while long texts were reported to have 100-1000 pages [124–126]. Our corpus contains 83.80% short documents. The high proportion of short documents is based on the fact that most source documents are in the form of articles or papers as mentioned before. The complete ratio of source document length could be seen in table 5.5 which summarizes the statistic data of the source document corpus.

Table 5.4: The proportion of source document number and classes in PlagiarIna's corpus

Classes	%	**Classes**	%
Agriculture	5%	Geography	3%
Anthropology & sociology	3%	History	7,4%
Architecture	1,5%	Information Technology	17%
Art & culture	2,8%	Languages & Literature	6,9%
Biology	3.7%	Medicine & Health	3,3%
Business, Finance, Economy	18,1%	Photography	2,7%
Civil Engineering	2%	Physics	3,4%
Communication	0,8%	Psychology	1,6%
Education, Pedagogy	3,5%	Theology	8,9%
Fischery, Aquaculture	2%	Tourism	2,5%
Forestry	0,9%		

The major language used in the original texts is Indonesian. However, there were eight articles written in English, but they were translated into Indonesian using Google translate [21]. Besides, a small portion of English text is unavoidably present in some documents in the form of abstract, as several articles provide two versions of abstracts, one is written in English and another is in Indonesian. The abstracts in English are kept as they are, and have neither been deleted nor translated. The reason is that checking, deleting or translating abstracts which are not present in all articles would be a time-consuming task.

Table 5.5: Summary on Source document statistics

Language of texts		**Document length**	
Indonesian	100%	Short (1-10 pp)	83,8%
Translated {Eng, INA}	8 docs	Medium (10-100 pp)	15,9%
		Long ($\geq 100pp$)	0,3%

Corpus size	
# indexed documents	2014

[21]I would like to acknowledge two out of these 8 articles in English were written by Prof. Titien Saraswati PhD. Generously, she gave her articles for the purpose of this project.

5.1.2.2 Building Test Document Corpus

Building test document corpus is inseparable from creating plagiarism cases, since it is not easy to get real plagiarism cases in great numbers. A *plagiarism case*, in the terminology of this field, is used to refer to a passage which is obfuscated or modified on the basis of a plagiarism type (see section 2.1.3). The obfuscation could be done either locally, that is in a passage, or globally, which considers the whole document as one passage. In PAN corpora, it seems that the obfuscation for a suspicious document, d_{plg}, is done locally and each test document contains several plagiarism cases from one specific obfuscation type only, for example, $d_{plg}01$ contains 3 paraphrased passages, while $d_{plg}02$ contains 4 verbatim copied passages. All documents sharing the same type of plagiarism cases are then saved into the same folder.

Influenced by PAN corpus building techniques, the plagiarism cases in this study were created through two methods as stated below:

1. **Algorithmic generation** refers to an act of obfuscating a text automatically. Two scripts were coded to do this task. In the algorithmic generation which resulted in artificial plagiarism cases, the obfuscation was applied globally. This states the difference between PlagiarIna test document corpus and PAN's.

2. **Simulation by human writers**. In this technique, plagiarism cases are written by human writers as if they committed a plagiarism. In this simulation, the obfuscation was instructed to be done locally.

The following sections will explore techniques used to create artificial and simulated plagiarism cases.

5.1.2.2.1 Generating Artificial Plagiarism cases

The artificial plagiarism cases were generated through three heuristic operations: random text operation, shuffling word order, and semantic word variations [131]. These operations could be implemented in two different techniques depending on the number of source documents as follows:

1. **More than one** source document. In this technique, the algorithm could be assigned to take a passage randomly from different source documents. Then it applies one of these heuristic operations to the selected paragraphs, and then compose these obfuscated paragraphs into a single document.

2. **One source document**. In the second technique, the algorithm takes randomly one specific source document, and performs one obfuscation operation to this single document.

The first technique outputs a file consisting of passages having unrelated topics, but it has several number of source documents. The second technique, which refers only to one specific source document, was applied in this study with the following considerations: firstly, this obfuscation technique keeps the topic relatedness by taking all passages within a document; and secondly, the obfuscated document is aimed to represent a near duplicate case, in which the duplicate version has a high portion of resemblance to its source document. The partial-duplicates from various source documents are tackled by simulated plagiarism cases.

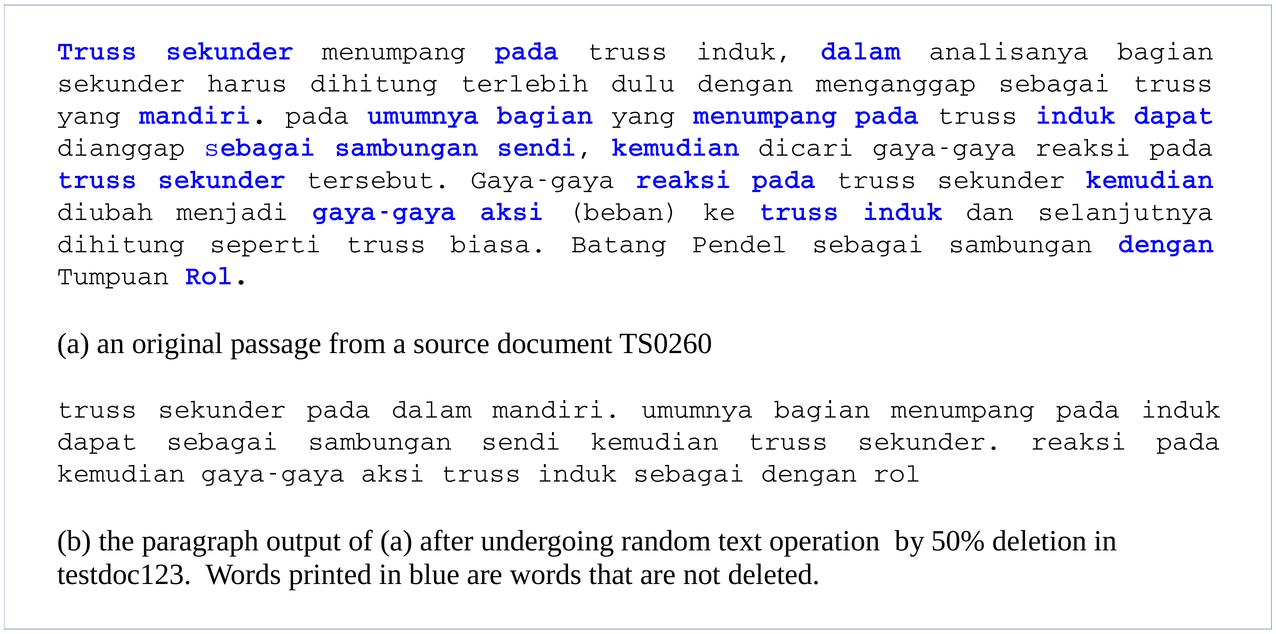
Unlike the first group of PAN corpora which preserved POS of a word in doing random text operation and shuffling the word order, the obfuscation strategy in this study applied a naive approach which considered a bag of word model. This means, in doing random obfuscation, the POS of a word and its order would be ignored. The rationale is that in real obfuscation case, a human writer would change POS, word orders and words in context. Even, in text reuses which are done intelligently, the words in context would be replaced by words or phrases having different semantics. Coincidentally, the random obfuscation strategy in 6^{th} PAN corpus building applies also a naive approach with a purpose "to test whether text alignment algorithms are capable of identifying reused passages from a bag-of-words model point of view" [129]. But at this time, our corpus has been completely built.

The random text operation was done by randomly deleting, inserting, deleting and inserting a number of words which were done on the basis of the percentage of document length. For this purpose, we defined the degree of obfuscation in terms of the percentage of document length. The light obfuscation covers less than 15% of word deletion or insertion, medium obfuscation ranges from 16%-30% of document length, and obfuscation greater than 30% of document length would be considered as heavy. The reason is that the algorithm is designed to delete any word in random position throughout the document. The algorithm does not simply delete several paragraphs as in systems reviewed in section 5.1.1.1. The deletion process of 30% words produces already many incomprehensible sentences. For this reason, the obfuscation percentage in this artificial plagiarism case is purposefully defined to be lower than the definition presented in table 2.1. Figure 5.1 exemplifies a passage which was algorithmically obfuscated with 50 % word deletion.

In the insertion process, the words to be inserted were taken from *Daftar Kata Dasar Bahasa Indonesia* which is a lexicon of Indonesian root words [22]. The insertion function works by choosing words randomly as many as the given number from this lexicon, and then inserts them in random position of the obfuscated target text. The deletion and insertion obfuscation were performed separately on separate documents, but they were also performed in one document in which deletion is performed first then followed by insertion.

Shuffling the word order was done by defining the frequency of shuffling. The shuffling frequency was implemented as the number of iteration in completing the task of shuffling. Using the built-in function provided by PHP script, 1 iteration of word shuffle produces semantically unreadable passages but the passage structure remains. 2 iterations of word shuffled results in semantically non-sense passages and structural disorder of passage boundary. For this reason, we defined te output of 1 iteration of word shuffle

[22]This lexicon was downloaded from `http://stop-words-list-bahasa-indonesia.blogspot.de/2012/09/daftar-kata-dasar-bahasa-indonesia.html` in June 2013



Truss sekunder menumpang **pada** truss induk, **dalam** analisanya bagian sekunder harus dihitung terlebih dulu dengan menganggap sebagai truss yang **mandiri**. pada **umumnya bagian** yang **menumpang pada** truss **induk dapat** dianggap **sebagai sambungan sendi**, **kemudian** dicari gaya-gaya reaksi pada **truss sekunder** tersebut. Gaya-gaya **reaksi pada** truss sekunder **kemudian** diubah menjadi **gaya-gaya aksi** (beban) ke **truss induk** dan selanjutnya dihitung seperti truss biasa. Batang Pendel sebagai sambungan **dengan** Tumpuan **Rol**.

(a) an original passage from a source document TS0260

truss sekunder pada dalam mandiri. umumnya bagian menumpang pada induk dapat sebagai sambungan sendi kemudian truss sekunder. reaksi pada kemudian gaya-gaya aksi truss induk sebagai dengan rol

(b) the paragraph output of (a) after undergoing random text operation by 50% deletion in testdoc123. Words printed in blue are words that are not deleted.

Figure 5.1: An example of an obfuscated passage by deletion process in artificial plagiarism cases.

to be heavily obfuscated. An example of a shuffled passage and its source passage are displayed in figure 5.2. For the purpose of executing the random text operation and word shuffle, a specific script was created. This script was completed with a Graphical User Interface (GUI) through which a user could select the source document from database, fill in the percentage of words to delete or insert and the frequency of the word shuffle. The algorithm for random text operation and word shuffle could be found in Algorithm 2.

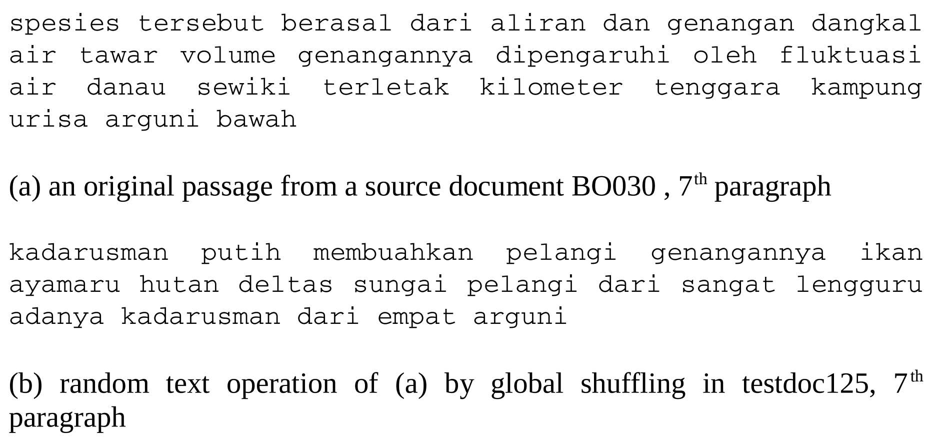
Like random text operation, **the semantic word variation** was performed by using Wordnet *Bahasa* (see section 4.2.3) with a naive approach too. It is said to be a naive approach since firstly, there were no POS-tagging process in choosing the words to be replaced; and secondly there was no disambiguation process in choosing its substitute among words having the same synsets. The replacement process runs as the function chooses words randomly according to the defined number of words to be replaced. For each chosen word, it finds their synsets in the Wordnet Bahasa. If the chosen word has more than one synsets, one snyset is chosen randomly, extracts all words having the same chosen synset, and simply randomly chooses one out of many words under this synset. The summary on the statistics of the artificial plagiarism cases could be seen in table 5.6.

5.1.2.2.2 Simulating Plagiarism Cases

The main goal of creating simulated plagiarism cases is to have test documents which mimic real cases of text reuses. For this reason, the simulated plagiarism cases were written by

Algorithm 2 Algorithm on random text operation

```
Input:  $\mathrm{d}_{src}, rootLex, nrDel, nrIns, shuflFreq$
Output:  $obfuscatedFile$
  $\mathrm{d}_{src} \leftarrow preprocess(\mathrm{d}_{src})$
  $\mathrm{d}_{src} \leftarrow tokenized(\mathrm{d}_{src})$
  $function$ Deletion($d_{src}$, nrDel)
      $nrDel \leftarrow percentageToNrOfWordCoversion(nrDel,\mathrm{d}_{src})$
      $wordDel \leftarrow randomSelectionOfWord(\mathrm{d}_{src}, nrDel)$
      for $(a = 0,\ a < countwordDel,\ a++)$ do
         if match($wordDel_a, d_{src}$) then
               $delete(\mathrm{wordDel}_a)$
         end if
      end for
      $return\ d_{src}$
  $end\ function$
  $function$ Insertion($d_{src}$, rootLex, nrIns)
      $nrIns \leftarrow percentageToNumberOfWordConversion(nrIns,\mathrm{d}_{src})$
      $wordIns \leftarrow randomSelectionOfWord(rootLex,\ nrIns)$
      for all (wordIns) do
         $\mathrm{d}_{src} \leftarrow Insert(wordIns,\ randomOffset(\mathrm{d}_{src}))$
         $return\ d_{src}$
      end for
  $end\ function$
  if $nrDel \neq 0$ then
     $d_{src} \leftarrow Deletion(d_{src},\ nrDel)$
  else if $nrIns \neq 0$ then
     $d_{src} \leftarrow Insertion(d_{src,\ rootLex,\ nrIns})$
  else if $nrDel \neq 0\ AND\ nrIns \neq 0$ then
     $d_{src} \leftarrow Deletion(d_{src},\ nrDel)$
     $d_{src} \leftarrow Insertion(d_{src,\ rootLex,\ nrIns})$
  else
     for $i = 0,\ toshuflFreq$ do
        $d_{src} \leftarrow shuffle(d_{src})$
     end for
  end if
```



```
spesies tersebut berasal dari aliran dan genangan dangkal
air tawar volume genangannya dipengaruhi oleh fluktuasi
air danau sewiki terletak kilometer tenggara kampung
urisa arguni bawah
```

(a) an original passage from a source document BO030 , 7th paragraph

```
kadarusman putih membuahkan pelangi genangannya ikan
ayamaru hutan deltas sungai pelangi dari sangat lengguru
adanya kadarusman dari empat arguni
```

(b) random text operation of (a) by global shuffling in testdoc125, 7th paragraph

Figure 5.2: An example of an obfuscated passage by shuffling process

Table 5.6: A summary on artificial test document statistics

Corpus size		**Document length**	
# artificial files	128	short (1-10 pp)	100%
Obfuscation level		**Obfuscation type**	
Light	22%	Shuffle	27%
Medium	39%	Deletion & insertion	22%
Heavy	39%	Deletion	20%
		Insertion	21%
		synonym replacement	11%

human writers. In real cases of text reuse, writers would disguise their copied texts using various types of plagiarism as mentioned in section 2.1.3. The length of text reuse varies greatly in real cases. In some smartly-written cases of text reuse, the disguised texts often cover only small fragments or one short passage from a long source document. Based on the research scope, the plagiarism types which are portrayed in the simulated test documents are *copy and paste* (copy), *shake and paste* (shake), paraphrase, and summary whose length ranges from short to medium. One plagiarism case could be taken from or summarized from different passages of various source documents. The purpose is to test whether the retrieval algorithm is able to retrieve source documents from which only small portion of text is reused, and whether text alginment algorithm is capable of recognizing various sources for a plagiarism case. The simulated plagiarism cases were done in 3 batches whereas the first batch was performed through crowd sourcing, and in the rest batches, 2 students for each batch were hired to write the plagiarism cases.

The crowd sourcing was enabled by creating an HTML page and PHP script which processed data submitted by participants. The web page was hosted temporarily on the Web. An invitation letter containing the link was sent via email and Facebook to students, ex-students and colleagues in Duta Wacana Christian University (UKDW), and some Indonesian friends living in Munich. The web page contained only three essential things for creating simulated plagiarism cases: instructions, two text fields, and questionnaire. The instructions informs participants how to do the task. It was completed with two pairs of examples, one pair demonstrated the source paragraph and its acceptable paraphrased version, and another pair exploited a set of source and its unacceptable paraphrase version. The accepted and unacceptable versions of paraphrase deal with how paraphrase is performed, whether the paraphrased version preserves the ideas but wrap them in different words or expression or it changes the ideas of source version.

The core of the tasks were displayed and done through two text fields provided. One text field displayed one paragraph which was chosen randomly from a source document by the script as a user starts a web session. On the next step, a participant could click the provided button to refresh the source paragraph if the topic was considered inappropriate. Based on this displayed source paragraph, a participant rewrote her/his own paraphrased versions on another provided text field. After completing his or her task, a participant could click the submit button which sent and saved the rewritten version into a MySql table along with the information on the source paragraph ID and source document ID from which this paragraph was taken. The questionnaire which comprises 7 questions was aimed to collect participants' demographic data. The questions in this questionnaire take the form of closed questions whose answers were opted in a drop-down menu just for saving the space of the page.

The GUI of the page for doing simulation looked a little bit cluttered, since instruction, task and questionnaire were fitted into one page. The advantage of such a page is that the participants did not need to scroll up or down the page, or click a link for completing a task. The consequence was that the page design bore no aesthetic value. This strategy was done in purpose by considering the participants' characteristics in which they are very busy though they are quite Internet-savy. Besides, their motivation in participating on this task was to support their friend's project. This led to a tendency for doing the task as fast as possible, being reluctant to follow a link or to scroll up and down a page. With such considerations, the aesthetic aspect of the web was sacrificed, the main objective was that all tasks were completed. This design has been very beneficial since most participants completed the questionnaire task, only few of them ignored it.

The crowd sourcing involved 33 persons whose demographic data could be seen in table 5.7. This data shows an interesting fact especially in answering the question whether participants have ever done plagiarism before, 55% participants acknowledged that at least they have done plagiarism once in their life, 24% declared that they never committed it, and the rest were unsure whether they have done it or not. The question on the native speaker was posed to differentiate whether Indonesian is their mother tounge or second language. Besides origins, speaking Indonesian as one's mother tongue signifies generation. The young generation under 25 years old has a greater possibility to be Indonesian native speakers.

Table 5.7: The demographic data of participants involved in crowd-sourcing

Age		**Education**	
18-22	10%	high school	9%
22-28	30%	Non-degree	12%
29-35	18%	Bachelor	33%
36-45	36%	Master	42%
45-55	6%	PhD	3%
Native speaker		**Gender**	
Yes	40%	Male	45%
2nd Language	18%	Female	55%
# paragraph submitted		**plagiarized**	
1	58%	Yes	55%
2	6%	No	24%
≥ 3	36%	n/a	21%
Writing as part of job			
Yes	55%		
2nd Language	45%		

The crowd-sourcing which was aimed to create paraphrase and summary obfuscation types of text reuse produced many *copy and paste* or *shake and paste* types. Besides, it resulted high redundancy of source paragraph selection. This is an unanticipated result from giving freedom to participants to choose source paragraphs for completing the task. Selecting only one out of redundant paragraphs randomly, the rewritten paragraphs having the same source document ID were then combined into a test document. In total, the first batch of simulated plagiarism case resulted in 70 test documents. However, many of these documents have only 1-2 paragraphs. The test document length and the complexity of obcuscation types led us to do the next batches of simulation by hiring students who were studying in Munich and at UKDW [23]. Figure 5.3 presents an example of a paraphrased passage resulting from a crowd-sourcing process.

In the second batch, two Indonesian students studying in Ludwig-Maximilian University, Munich were hired. They produced 10 test documents which have plagiarism types of summary and paraphrase with light to medium obfuscation levels. Based on the goal of achieving qualified test documents in an expected number and the limitation of research fund, 2 UKDW students were hired in the third batch for creating simulated plagiarism cases. The remote communication and file transfer were done through Web-based media. This third batch of simulation resulted in 25 test documents whose obfuscation level vary from light, medium to heavy. One plagiarism case could have 1-5 source passages from

[23]UKDW stands for Duta Wacana Christian University located in Yogyakarta, Indonesian

`selain itu lang juga mengatakan bahwa layout lingkungan mempengaruhi pola interaksi sosial antar manusia. ada beberapa ciri lingkungan yang bisa membuat orang saling berinteraksi: cara perabotan diletakkan dalam ruang lobby hotel kantor atau street furniture di ruang terbuka menjelaskan interaksi yang diharapkan antar manusia`

(a) an original passage in source document AR020A, 21th paragraph

`ada pendapat yang mengatakan bahwa perilaku mempengaruhi rancang bangun begitu juga sebaliknya. lang berpendapat bahwa rancangan lingkungan mempengaruhi pola interaksi sosial antar manusia. ada beberapa ciri lingkungan yang bisa membuat orang saling berinteraksi cara perabotan diletakkan dalam ruang lobby hotel kantor atau street furniture di ruang terbuka menjelaskan interaksi yang diharapkan antar manusia`

(b) a paraphrased passage of (a) resulted from crowd sourcing saved as testdoc003, 6th paragraph

Figure 5.3: An example of a paraphrased passage from crowd-sourcing

different source documents, and one test document may contain more than one type of plagiarism. The complexity of plagiarism case in test documents resulted from the third batch is much higher than those produced from the former batches. The percentage of plagiarism per document (cf. table 2.1) in all test documents produced by simulation method is greater than or equals to 80%. The length of simulated test document varies from 300-1200 words. The summary on the statistics of test document corpus is available in table 5.8. An example of an obfuscated passage with a summary obfuscation type from batch 3 is presented in Appendix B, figure B.1[24].

5.1.2.2.3 No-Plagiarism Cases

Besides artificial and simulated plagiarism cases, our test document corpus is completed also with documents containing no plagiarism cases. Instead of simulating test documents with no-plagiarism cases, we selected articles whose topics and subject areas have not been covered in source document corpus. We assumed that these documents share no commonality with all documents which become the source objects of modification and copy. We address these test documents as no-plagiarism cases. We simply selected research articles on plagiarism detection reviewed in section 3.4. Since several articles are written in English, we used Google translate tool to translate them into Indonesian and then labeled these test documents as no-plagiarism cases on their meta-files.

[24]The example was written by Manila kristin. The selection of paragraph was done on the basis of paragraph length which is quite short compared to others, but due to its length, it is better to be presented in appendix.

Table 5.8: A summary on simulated test document statistics. The sign # refers to the phrase *the number of.*

Corpus size		**Plagiarism per document**	
# simulated files	105	entirely ($\geq$ 80%)	100%

Doc per batch		**Obfuscation type**	
Batch 1	67%	Copy	14.1%
Batch 2	10%	Shake	20.3%
Batch 3	24%	Paraphrase	58.8%
		Summary	6.8%

Document length	
Short	100%

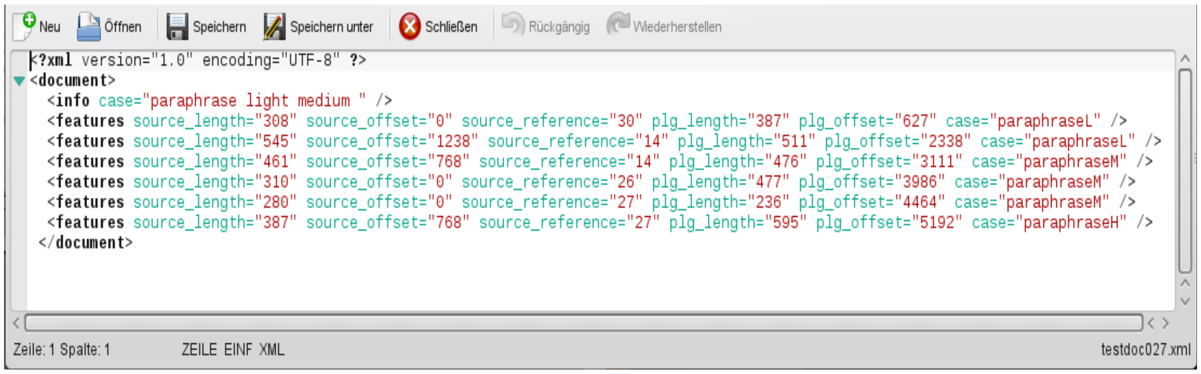
5.2 The Evaluation Framework

Having surveyed 275 papers dealing with text as well as code plagiarism, Potthast et al. stated that "authors proposing PDS often use non-standardized evaluation methods" [131]. For this reason, PAN PC proposed an evaluation method for EPD systems which in its development has undergone some elaboration for its concepts. Basically, the evaluation framework proposed by PAN PC could be distinguished into two approaches. These approaches were based on PAN's retrieval strategy whether the evaluation was carried out to the whole EPD system, or either to the retrieval subtask or text alignment subtask only. The first approach which was applied during the $1^{st} - 3^{rd}$ PAN PCs evaluated the performance of an EPD system as a whole, and thus the performance assessment was carried out on the end outputs of EPD systems [124–126]. The second approach carried out a separate evaluation for Retrieval as well as Text Alignment subtasks [127–129].

Independent of PAN's changing policies, an evaluation which assesses the end outputs of an EPD system without evaluating each of its subtask suffers from, at least, two drawbacks. Firstly, there is no way to know which subtask performs well. Secondly, the maximum performance of Text Alignment subtask is hardly measured in cases where not all source documents are retrieved. The reason lies on the fact that Text Alignment module processes candidate documents which are outputted from Retrieval process. These shortcomings could be overcome by conducting an oracle experiment for evaluating the performance of each subtask separately. However, evaluating each subtask separately may lead to another drawback, that is, the trickle-down effect in the system performance is hardly captured and measured. Considering these drawbacks and PlagiarIna as a workflow system whose every component contributes to the whole system performance, its evaluation will be carried out in three stages as follows:

a) Evaluating the retrieval subtask independently

b) Evaluating the text alignment by conducting an oracle experiment



```xml
<?xml version="1.0" encoding="UTF-8" ?>
<document>
  <info case="paraphrase light medium " />
  <features source_length="308" source_offset="0" source_reference="30" plg_length="387" plg_offset="627" case="paraphraseL" />
  <features source_length="545" source_offset="1238" source_reference="14" plg_length="511" plg_offset="2338" case="paraphraseL" />
  <features source_length="461" source_offset="768" source_reference="14" plg_length="476" plg_offset="3111" case="paraphraseM" />
  <features source_length="310" source_offset="0" source_reference="26" plg_length="477" plg_offset="3986" case="paraphraseM" />
  <features source_length="280" source_offset="0" source_reference="27" plg_length="236" plg_offset="4464" case="paraphraseM" />
  <features source_length="387" source_offset="768" source_reference="27" plg_length="595" plg_offset="5192" case="paraphraseH" />
 </document>
```

Zeile: 1 Spalte: 1 ZEILE EINF XML testdoc027.xml

Figure 5.4: A metafile containing an annotation data of a source-test document pair

c) Evaluating the whole system performance

For the sake of evaluation, a metafile in XML-format for each test or suspicious document was generated. The metafile contains gold-standard annotation which is manifested in 6-tuple information $\langle s_{src}$ length, s_{src} offset, $d_{src}ID$, s_{plg} length, s_{plg} offset, case $\rangle$, where s_{src} refers to a source passage in an annotated source document d_{src}, and s_{plg} refers to a plagiarized version of s_{src} in a suspicious document. Note that this 6-tuple information is almost similar to the output of PlagiarIna as displayed in Figure 4.7, the difference is set on the *case* attribute whose value informs the obfuscation type and level. Figure 5.4 displays a capture of one annotated metafile conveying a set of these 6-tuples, whereas d_{src}ID is transformed into an xml-attribut called *source_reference*.

The metafile was generated semiautomatically through the aid of a script called *parMerge*. Given a document and their passages labelled as plagiarism cases, the script computes its task and outputs the start, end offsets, and the length of the given passages. The same procedure was applied to the referred source passage s_{src} in a d_{src}. However, the writing process of the outputs of this script into an XML file was done manually. The information in this metafile would be compared to the information in the xml-file outputted by PlagiarIna during the evaluation process.

5.2.1 Evaluation Measures for Retrieval Subtask

A specific evaluation measure for the retrieval subtask was introduced in 5^{th} PAN PC. Due to its online retrieval strategy, the possibility of retrieving a document d_{ret}, which is a duplicate or near-duplicate of a source document, d_{src}, is unavoidable in this scheme, since the web is full of documents which are (near-) duplicates to each other [128]. The system would count such d_{ret} as a false detection, though manually a human evaluator would consider this d_{ret} as a true detection [128]. For this reason, the 5^{th} PAN evaluation method introduced an idea of devising a near-duplicate detector to check the existence of (near-) duplicate among the source document corpus. The duplicate documents, D_{dup},

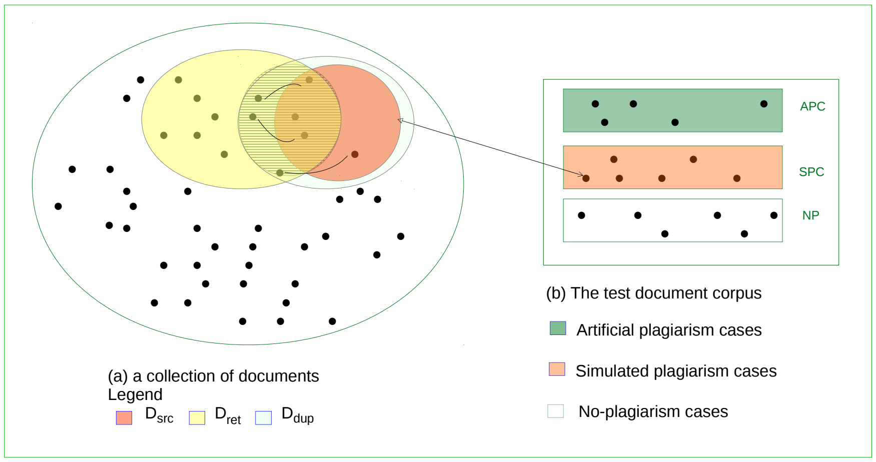

APC
SPC
NP
(b) The test document corpus
Artificial plagiarism cases
Simulated plagiarism cases
No-plagiarism cases
(a) a collection of documents
Legend
D_src
D_ret
D_dup

Figure 5.5: The inclusion of duplicate documents in the true positive detection for computing precision and recall. (a) decribes a collection of documents where $D_{src}(d_{plg})$ are hidden, and (b) describes our test document corpus. In the intersection set of $D_{src}(d_{plg})$, $D_{ret}(d_{plg})$, and $D_{dup}(d_{plg})$, the line connecting two dots (documents) stands for a (near-) duplicate relation, i.e. a d_{ret} has been detected to be a (near-) duplicate of a source document d_{src}

which are outputs of this detector are included in measuring the precision and recall of the retrieval process [128].

In measuring the performance of our retrieval process, we adopted the idea of using the near-duplicate detector from PAN PC, but we did some adjustment on its measures with reasons that the retrieval process is done offline and the nature of our source document corpus is much simpler than the nature of web documents. Though the number of duplicate documents among our source documents is not as high as those on the web, we found out, during the pilot experiment, that the translated articles from English to Indonesian turn out to be near-duplicate versions of some articles written by the same author but they were published on different media. Thus, it is very probable that there is one or a set of unknown (near-) duplicate documents D_{dup} for a source document. Recall that the source document, d_{src}, is a document whose content is reused partially in a suspicious document d_{plg}. To anticipate the occurrence of duplicate documents, a near-duplicate detector script was written using word unigram as its features and Jaccard coefficient as its similarity measure.

Since (near-) duplicate documents for a source document are unknown and hidden, the only way to find them is by checking the retrieved documents of a specific suspicious document d_{plg}. For this reason, we run our near-duplicate detector to the retrieved documents only. The near-duplicate detector compares each $d_{ret}(d_{plg})$ to each $d_{src}(d_{plg})$. We defined a Jaccard similarity threshold θ to be 0.7 for a retrieved document $d_{ret}(d_{plg})$ as a (near-) duplicate document (d_{dup}) of a $d_{src}(d_{plg})$. The similarity threshold should be relatively high to ensure the assumption that a d_{dup} contains the plagiarized passage as d_{src} does. We argue that the threshold value of 0.5 is risky enough, though an human assessor would agree that two texts with Jaccard similarity value 0.5 are quite similar in content. But the probability of the absence of a plagiarized passage in such documents is also high, that is almost 50%. This could happen since the Jaccard coefficient applied in this detector computes similarity on a global document level while the plagiarized passage occurs on a local one. A retrieved document would be considered as a (near-) duplicate document, if its similarity to a d_{src} is greater than or equal to threshold θ and if it is not listed in the annotated file as a $d_{src}(d_{plg})$. These retrieved duplicate documents would be added to the source documents, $D_{src}(d_{plg})$, to form a new set of larger source documents referred to as $\hat{D}_{src}(d_{plg})$. This new set of source documents will be used to compute the true positive of retrieval. Figure 5.5 illustrates the effect of including $D_{dup}(d_{plg})$ in computing the true-positive for precision and recall measures. The following defines $D_{src}(d_{plg}), D_{ret}(d_{plg}), \hat{D}_{src}(d_{plg})$ in a context where the EPD system is given only a single input of a d_{plg} :

$D_{src}(d_{plg}) = \{d \mid d \text{ is reused partially in } d_{plg} \text{ and is mentioned in the anotation of } d_{plg}\}$
$D_{ret}(d_{plg}) = \{d \mid d \text{ is a retrieved document for a given } d_{plg}\}$
$\hat{D}_{src}(d_{plg}) = D_{src}(d_{plg}) \cup \{d \in D_{ret}(d_{plg}) \mid d \text{ is a (near-) duplicate of some } d \in D_{src}(d_{plg})\}$

Based on these set definitions, the precision and recall for one given d_{plg} as input are then defined as follows:

$$Prec(d_{plg}) = \frac{\mid \hat{D}_{src}(d_{plg}) \cap D_{ret}(d_{plg}) \mid}{\mid D_{ret}(d_{plg}) \mid} \tag{5.1}$$

$$Rec(d_{plg}) = \frac{\mid \hat{D}_{src}(d_{plg}) \cap D_{ret}(d_{plg}) \mid}{\mid \hat{D}_{src}(d_{plg}) \mid} \tag{5.2}$$

The F-measure that weights equally the precision and recall would be used also to measure the performance of the Retrieval subtask. The F-measure used is the tradional one which is the harmonic mean of precision and recall:

$$F1(d_{plg}) = 2 \ . \ \frac{Prec(d_{plg}) \ . \ Rec(d_{plg})}{Prec(d_{plg}) + Rec(d_{plg})} \tag{5.3}$$

The Macro-Average Precision, Macro-Average Recall, and Macro-Average F1 scores which measure the average of precision and recall of all given D_{plg} in the experiment are then defined as follows:

$$Precision_{macro} = \frac{1}{n} \sum_{i=1}^{n} Prec(d_{plg}) \tag{5.4}$$

$$Recall_{macro} = \frac{1}{n} \sum_{i=1}^{n} Rec(d_{plg}) \tag{5.5}$$

$$F1_{macro} = \frac{1}{n}\sum_{i=1}^{n} F1(d_{plg}) \tag{5.6}$$

where n stands for the number of total test document in $\mid D_{plg} \mid$ in a test set category, and d_{plg} refers to a given test case or document.

The following example will illustrate better the inclusion of near-duplicate documents in measuring precision and recall. Suppose d_{plg1} is annotated as a plagiarized version of the following source documents $D_{src}(d_{plg}) = \{d_1, d_2, d_3, d_4\}$. Given this d_{plg1} as a query, the retrieval module of PlagiarIna retrieves $D_{ret}(d_{plg}) = \{d_2, d_3, d_4, d_7, d_9, d_{10}, d_{12}\}$. Conventionally, the true positive detection is computed on the basis of the intersection between $D_{src}(d_{plg})$ and $D_{ret}(d_{plg})$ which covers a set of {d$_2$, d$_3$, d$_4$}. Thus, the computation of precision turns to be 3/7 and recall = 3/4. The occurrence of D_{dup} among $D_{ret}(d_{plg})$ will change this computation. Suppose the near-duplicate detectors finds out that d_7 is a near-duplicate of d_1, while d_9 is a duplicate of d_2. Thus, $D_{dup} = \{d_7, d_9\}$, $\hat{D}_{src}(d_{dplg}) = \{d_1, d_2, d_3, d_4, d_7, d_9\}$. The computation of Prec(d$_{plg}$), Rec(d$_{plg}$), and F1(d$_{plg}$) is as follows:

$$\begin{aligned} Prec(d_{plg}) &= \frac{\mid \{d_1, d_2, d_3, d_4, d_7, d_9\} \cap \{d_2, d_3, d_4, d_7, d_9, d_{10}, d_{12}\} \mid}{\mid \{d_2, d_3, d_4, d_7, d_9, d_{10}, d_{12}\} \mid} \\ &= \frac{5}{7} \\ &= 0.71 \end{aligned}$$

$$\begin{aligned} Rec(d_{plg}) &= \frac{\mid \{d_1, d_2, d_3, d_4, d_7, d_9\} \cap \{d_2, d_3, d_4, d_7, d_9, d_{10}, d_{12} \mid}{\mid \{d_1, d_2, d_3, d_4, d_7, d_9\} \mid} \\ &= \frac{5}{6} \\ &= 0.83 \end{aligned}$$

In this example, the inclusion of near-duplicate documents results in higher rates of recall and precision than the conventional computation. However, our experiments show that such increase occurs only when the number of retrieved D_{dup} is quite significant, which occurs quite seldom. The idea of including near-duplicates documents in this evaluation measure is not to increase either recall or precision rates, but to include as many *source documents* as possible, including those which are not annotated as $D_{src}(d_{plg})$.

5.2.2 Evaluation Measures for Text Alignment

Different from its former measures, three level performance measures for Text Alignment were introduced on PAN PC 2014. These measures assess the performance of detection on the character, plagiarism case, and document levels. The character-level measure, which has been used since 1st PAN PC (2009), is meant to capture the completeness of a detection of a given plagiarism case [129]. Another reason of measuring the plagiarism detection on the character level is that many plagiarism detection algorithms extract a plagiarism case by its overlapping substrings which impacts on the multiple detections [129]. Thus, precision and recall are measured on the levels of character, case, and document in PAN shared task 2014. Besides these measures, *Granularity* which checks the overlapping detection and *Plagdet* which measures the overall performance of detection were introduced as part of text alignment measures.

In PAN'14, a case-based measure was introduced to address the shortcoming of the character-level measure which could not inform which plagiarism case is detected and which is not. A detected plagiarism case would be reported as a plagiarism case, if the values of its character precision $prec_{char}$ and character recall rec_{char} are greater than or equals to the defined thresholds for each $prec_{char}$ and rec_{char} [129]. The recall threshold adjusts the minimal detection accuracy in regard to passage boundaries, and a precision threshold adjusts how accurate a plagiarism detection to be [129]. Unlike in case level, measures on the document level assume that a detected source-suspicious document pair would be regarded as a true positive detection, if this pair contains at least one plagiarism case whose length is greater than the minimum threshold. Suppose the text alignment subtask detects 3 plagiarism cases in a pair of $d_{src} - d_{plg}$, whereas only 1 out of 3 plagiarism cases having length beyond the defined threshold. This pair of source-suspicious documents (d_{src}, d_{plg}) would be considered as a true positive detection [129].

In assessing PlagiarIna's performance, we adopted the three levels of abstraction offered by PAN: character, case, and document level measurements. However, these measures are still unable to inform which obfuscation types are well detected and which are poorly recognized. To address this shortcoming, we introduced a measure to assess the recognition on the obfuscation type. This measure assesses the detection accuracy in a context where a test document, d_{plg}, may contain several distictive obfuscation types as in our test documents resulted from simulation process. Such context states the difference between our test document corpus from PAN's one [129], in which one d_{plg} is designed to contain one obfuscation type only. Recall that *obfuscation type* refers to the text manipulation done to a passage of a d_{plg} such as deletion, shuffle, copy, paraphrase, etc. The following subsections will present these four measures in details.

5.2.2.1 Character-Level Measures

As in PAN'14 [129], the computation of character-level measures is based on the plagiarism case (cf. definition on section 2.2.1), in which **S** is used to refer to a set of source-suspicious passage pairs defined in an annotated meta-file of a given suspicious document, and **R** denotes a set of detected source-suspicious passage pairs outputted by our prototype, PlagiarIna. In character-level measure, a plagiarism case $s \in S$, where s $= \langle s_{plg}, d_{plg}, s_{src}, d_{src} \rangle$, is used as references to characters of d_{plg} and d_{src} specifying passages s_{plg} and s_{src}. Correspondingly, r $= \langle r_{plg}, d_{plg}, r_{src}, d_{src} \rangle$ is used to represent a reported detection of a plagiarism case $r \in R$ [125]. r is said to detect s iff $s \cap r \neq \emptyset$, $r_{plg} \cap s_{plg} \geq 150$ characters, and $r_{src} \cap s_{src} \geq 125$ characters. Note, that we used the same minimum character threshold applied in the post-processing of PlagiarIna (cf.section 4.4): 125 characters for a r_{src}, and

150 characters for the r_{plg}. Thus, the macro-averaged precision and recall which we applied are defined exactly as in [125]:

$$prec_{char}(S,R) = \frac{1}{\mid R \mid} \sum_{r \in R} \frac{\mid \bigcup_{s \in S}(s \sqcap r) \mid}{\mid r \mid} \tag{5.7}$$

$$rec_{char}(S,R) = \frac{1}{\mid S \mid} \sum_{s \in S} \frac{\mid \bigcup_{r \in R}(s \sqcap r) \mid}{\mid s \mid} \tag{5.8}$$

where $s \sqcap r$ 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.

Multiple detections for a plagiarism case are possible as a consequence of algorithms which extract a plagiarism case by its overlapping substrings. Unfortunately, precision and recall are unable to address such overlapping detections [124, 125]. To cope with this problem, PAN introduced a granularity measure which assesses the frequency of detection for a case. We applied this granularity measurement in our evaluation. The granularity value 1 refers to an ideal detection, while granularity value over 1 refers to the repetition of a plagiarism case detection. The granularity is then defined as in [124]:

$$gran(S,R) = \frac{1}{\mid S_R \mid} \sum_{s \in S_R} \mid R_s \mid \tag{5.9}$$

where $S_R \subseteq S$ denotes the detected passages or cases by detections in R, $S_R = \{s \mid s \in S \wedge \exists r \in R :$ r detects s $\}$ and $R_S \subseteq R$ denotes the detections of s, that is $R_S = \{r \mid r \in R$ and r detects s $\}$. Figure 5.6 illustrates this evaluation concept for assessing text alignment performance, and gives an example on the concept of S_R and R_S.

The overall performance of a plagiarism detection algorithm is measured through *Plagdet* which combines all these three measures: precision, recall, and granularity. The idea of introducing Plagdet measure is to get a single value of a system performance to be comparable to other systems, since precision, recall, and granularity do not allow for a unique ranking among different approaches of plagiarism detection systems [129]. For this reason, the measures are combined into a single overall score. The precision and recall measures are combined into F1 score which turns to be their harmonic mean. Thus the F1 and plagdet measures are defined as follows:

$$F1 = 2 \ . \ \frac{prec_{char}(S,R) \ . \ rec_{char}(S,R)}{prec_{char}(S,R) + rec_{char}(S,R)} \tag{5.10}$$

$$plagdet(S,R) = \frac{F1}{log_2(1 + gran(S,R))} \tag{5.11}$$

The following example demonstrates how these measures work. Suppose we have an annotated file of a d_{plg1} which contains the following pairs of information in **S** as shown in example (1[a]). We also have the detection results **R** as seen in 1[b]:

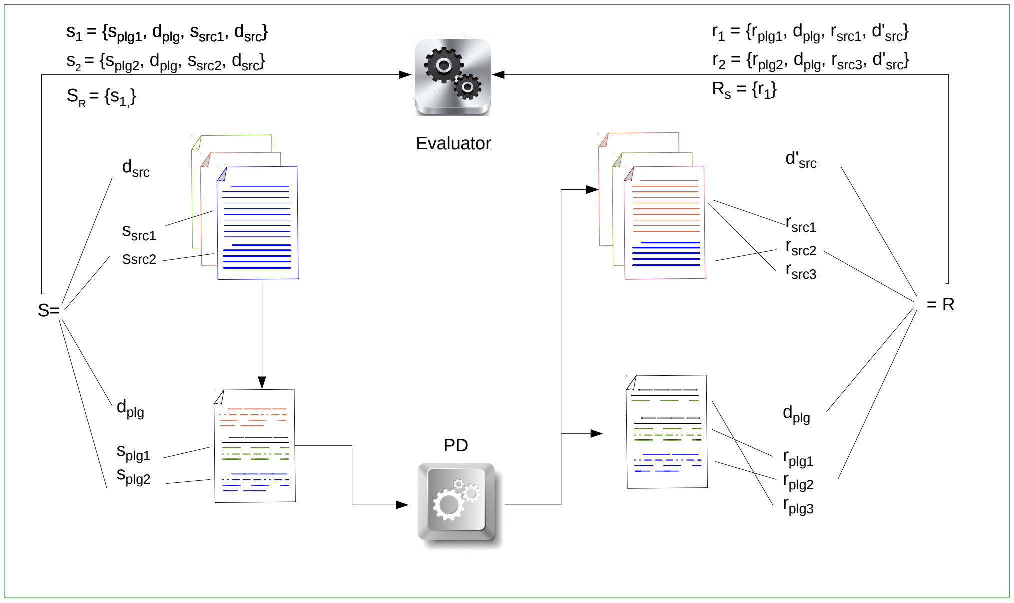

s1 = {splg1, dplg, ssrc1, dsrc}
s2 = {splg2, dplg, ssrc2, dsrc}
SR = {s1,}
Evaluator
r1 = {rplg1, dplg, rsrc1, d'src}
r2 = {rplg2, dplg, rsrc3, d'src}
RS = {r1}
dsrc
Ssrc1
Ssrc2
S=
dplg
splg1
splg2
PD
d'src
rsrc1
rsrc2
rsrc3
= R
dplg
rplg1
rplg2
rplg3

Figure 5.6: an illustration on basic concepts for evaluating Text Alignment performance. In this Figure, S refers to a set of source-suspicious passage pairs defined in a golden annotation file for a suspicious document d_{plg1}, while R refers to a set of detected source-suspicious passage pairs for d_{plg1}. In this examples, S = {s$_1$, s$_2$} and R = {r$_1$, r$_2$}. S_R is a subset of S which denotes the detected passages by detection R, that is S_R = {s$_1$}. R_S, which is also a subset of R, denotes the detection on S, i.e. R_S = {r$_1$}.

Example 1. *Pairs of plagiarism cases*
Cases defined in S for d_{plg_1}:
[s1] $\langle$ srcLen=400, srcOffset=0, d_{src}=1094, plgLen=450 plgOffset=0 case=paraphrase $\rangle$
[s2] $\langle$ srcLen=300, srcOffset=780, d_{src}=1094, plgLen=275 plgOffset=1200 case=paraphrase $\rangle$
[s3] $\langle$ srcLen=400, srcOffset=250, d_{src}=2005, plgLen=400 plgOffset=200 case=copy $\rangle$.
Cases reported in R:
[r1] $\langle$ srcLen=380, srcOffset=0, d_{src}=1094, plgLen=400 plgOffset=0 $\rangle$
[r2] $\langle$ srcLen=400, srcOffset=250, d_{src}=2005, plgLen=400 plgOffset=200 $\rangle$

Note that the length refers to the length of a plagiarism case or passage in characters and not in words.

In computing precision and recall on the character level, the so-called *similar characters* do not literally refer to the same characters but to the strings located in the range of similar offsets. Thus, a passage in r, or r_{plg} will be regarded as a true detection of s_{plg} in s, if it is located within the range of the defined offsets in s, though this r_{plg} has undergone any modification in lexical, semantic or syntactic level. In our example above, we have $\mid S \mid= 3$, $\mid R \mid= 2$ which refer to cases in S and in R. The number of s detected by r, or $\mid S_R \mid= 2$, while the number of r that detects s, $\mid R_S \mid= 2$, since each case is detected once only. In character-based measures, the intersection of s and r is computed by the union of intersected characters between its source and suspicious passages, i.e. $s \sqcap r =\mid s_{src} \cap r_{src} \mid + \mid s_{plg} \cap r_{plg} \mid$. The recall, precision, granularity and Plagdet are then computed as follows:

$$\begin{aligned}
prec_{char} &= \frac{1}{2} \times (\frac{780}{780} + \frac{0}{0} + \frac{800}{800}) \\
&= \frac{1}{2} \times (1 + 0 + 1) \\
&= \frac{1}{2} \times 2 \\
&= 1 \\
rec_{char} &= \frac{1}{3} \times (\frac{780}{850} + \frac{0}{575} + \frac{800}{800}) \\
&= \frac{1}{3} \times (0.91 + 0 + 1) \\
&= 0.33 \times 1.91 \\
&= 0.63
\end{aligned}$$

Using equation 5.3, we get F1 equals to 0.77, then the granularity and Plagdet score are computed as follows:

$$\begin{aligned}
gran &= \frac{1}{2} \times 2 \\
&= 1
\end{aligned}$$

$$\begin{aligned} plagdet &= \frac{0.77}{log_2(1+1)} \\ &= \frac{0.77}{1} \\ &= 0.77 \end{aligned}$$

5.2.2.2 Case-level Measures

In applying measurement on the case level, we did not see the point of defining a different minimum length threshold for a case to be considered as a true positive detection. We used the same threshold defined in the post-processing phase as in character-based measures, i.e 125 characters for r_{src} and 150 characters for r_{plg}. This means any plagiarism case outputted by our prototype, PlagiarIna, will become the assessment object. To clarify this idea, assume that we have a source passage s_{src1} in **s** having length of 600 characters. Assume also that PlagiarIna detects this s_{src1} and outputs the recognized passage, r_{src1}, which has a 301 character length only. r_{src1} would be evaluated as a detection of s_{src1}, since its length is greater than the defined minimum threshold. Let S, R, S_R and R_S denote to the same references as mentioned earlier in character level measures, but s and r refer to a pair of passages or plagiarism cases instead of a set of passages' characters. Thus, we define the precision and recall on the passage level as follows:

$$prec_{case}(S,R) = \frac{\mid R_S \mid}{\mid R \mid} \tag{5.12}$$

$$rec_{case}(S,R) = \frac{\mid S_R \mid}{\mid S \mid} \tag{5.13}$$

where S_R refers to cases in S which are detected by R, and R_S refer to cases in R which detect S.

The computation examples presented below use the plagiarism cases described in the example 1[a] and [b]. Remember that in this example, we have $\mid S \mid = 3$, $\mid R \mid = 2$, $\mid S_R \mid = 2$ and $R_S = 2$. The precision and recall in case level are then computed as follows:

$$\begin{aligned} prec_{case} &= \frac{2}{2} \\ &= 1 \\ rec_{case} &= \frac{2}{3} \\ &= 0.66 \end{aligned}$$

5.2.2.3 Document-level measures

The measures on document level try to assess the detection performance on a wider scale and to see whether all source documents for a given suspicious document are detected. Thus, it disregards whether all plagiarism cases present in a specific pair of source-suspicious document are detected or not. The minimum requirement for a detected source document d_{src} in R to be regarded as a true positive detection is that this document contains at least one accurate detection of a plagiarism case. Let D_S denotes pairs of source-suspicious documents defined in S, and D_R denotes detected pairs of source-suspicious documents in R. Based on these sets, the document-level precision and recall are defined as follows:

$$prec_{doc}(S,R) = \frac{\mid D_S \cap D_R \mid}{\mid D_R \mid} \tag{5.14}$$

$$rec_{doc}(S,R) = \frac{\mid D_S \cap D_R \mid}{\mid D_S \mid} \tag{5.15}$$

Looking back at plagiarism cases presented in example 1, the cardinality of both D_R and D_S are equal to 2, and they refer to the same document IDs. The computation of precision and recall on the document level could be seen as follows:

$$\begin{aligned} prec_{doc}(S,R) &= \frac{\{1094, 2005\}}{\{1094, 2005\}} \\ &= \frac{2}{2} \\ &= 1 \\ rec_{doc}(S,R) &= \frac{\{1094, 2005\}}{\{1094, 1094, 2005\}} \\ &= \frac{2}{2} \\ &= 1 \end{aligned}$$

5.2.2.4 Measure for the Obfuscation Type

The case-level measures evaluate the accuracy and relevancy of detected cases in general. In a test document corpus whose each of its test document contains a single obfuscation type only, the case-level measures function also as a measure for the obfuscation type recognition, when they are computed to a set of test documents containing a specific obfuscation type. Let us assume that we have 10 test documents containing paraphrased cases and 10 documents containing copied plagiarism cases. If the case-level measures are applied to assess 10 test documents having paraphrased cases separately from a set of documents containing copied cases, then we get the precision and recall scores on the case level as well as on the level of obfuscation type. Regarding the nature of our test document corpus, in which one test document may contain various onfuscation types in its plagiarism case set, we could not simply apply the dual-functions of the case-level measures. Therefore, we introduced a recognition measure for the obfuscation type which is abbreviated as obtype recognition. Let S_C denotes a set of plagiarism cases (or pair of passages) having a specific obfuscation type in **S**, and R_C denotes a set of plagiarism cases from a specific obfuscation type in **R**, where S and R refer to the same sets used in the former measures. S_C and R_C are then defined as follows:

$S_C = \{s_c \in S \mid \text{s}_c \text{ refers to a case with a specific obfuscation type in s }\}$
$R_C = \{r_c \in R \mid \text{r}_c \text{ is a detected case referring to a specific obfuscation type defined in s }\}$

We perceived that micro-average obtype is more suitable to measure the recognition of obfuscation type, since the number and types of the obfuscation in each test document vary significantly. Note that the computation of obtype recognition is inseparable from the plagiarism cases in S and R. The rationale is that the obfuscation type is a label attached to each case (or a pair of source-suspicious passages). Therefore, the obtype recognition of a single obfuscation type is defined as follows:

$$reco_{obtype}(S, R) = \frac{\sum_{i=1}^{|D_C|} \mid S_C \cap R_C \mid}{\sum_{i=1}^{|D_C|} \mid S_C \mid} \tag{5.16}$$

where D_C refers to the total number of documents containing one specific obfuscation type, eg. paraphrase or copy. To illustrate how the obtype recognition works, we need to consider cases in example 1 with d_{plg1} given as test document and d_{plg2} as a test document in the example 2.

Example 2. *Pairs of plagiarism cases*
Cases defined in S for a d_{plg_2}
[s4] ⟨ srcLen=650, srcOffset=100, d_{src}=199, plgLen=700 plgOffset=75 case=paraphrase ⟩
[s5] ⟨ srcLen=789, srcOffset=500, d_{src}=251, plgLen=225 plgOffset=987 case=summary ⟩
[s6] ⟨ srcLen=400, srcOffset=1298, d_{src}=251, plgLen=400 plgOffset=1237 case=copy ⟩
Cases reported in R
[r3] ⟨ srcLen=642, srcOffset=108, d_{src}=199, plgLen=590 plgOffset=103 ⟩
[r4] ⟨ srcLen=190, srcOffset=679, d_{src}=251, plgLen=190 plgOffset=1068 ⟩
[r5] ⟨ srcLen=400, srcOffset+1298, d_{src}=251, plgLen=400 plgOffset=1237 ⟩

Remember that in our example 1, the total number of cases in S is 3 with 2 paraphrased cases and 1 copied case, while the total number of cases in R is equal to 2 with 1 paraphrased case and 1 copied case. In example 2, both R and S have 3 cases with different obfuscation types, namely paraphrase, summary, and copy. From both examples, we have $\mid D_{src}(D_{plg}) \mid= 4$, with 2 documents being the source of paraphrased cases, i.e $\mid D_C(para) \mid= 2$. Consequently, we have also $\mid D_C(copy) \mid= 2$ and $D_C(smry) = 1$. The micro-average obtype recognition, $reco_{obtype}$, is then computed as follows:

$$\begin{aligned}
reco_{para} &= \frac{\mid \{r1\} \mid + \mid \{r3\} \mid}{\mid \{s1, s2\} \mid + \mid \{s4\} \mid} \\
&= \frac{1+1}{2+1} \\
&= \frac{2}{3} \\
&= 0.66
\end{aligned}$$

$$\begin{aligned}
reco_{copy} &= \frac{\mid \{r2\} \mid + \mid \{r5\} \mid}{\mid \{s3\} \mid + \mid \{s6\} \mid} \\
&= \frac{1+1}{1+1} \\
&= \frac{2}{2} \\
&= 1
\end{aligned}$$

$$\begin{aligned}
reco_{smry} &= \frac{\mid \{r4\} \mid}{\mid \{s5\} \mid} \\
&= \frac{1}{1} \\
&= 1
\end{aligned}$$

where the index x in rx refers to the index number of r given in example 1 and 2.

It can be clearly observed that the obtype recognition introduced before is based on a recall measure. we perceived that recall could be used to address the system's recognition to the obfuscation types. The precision rate on the level of obfuscation type could not be evaluated with a reason that the cases reported by the system, R, have no label of obfuscation types. Besides, the computation of obtype recognition is based on the pairs of source-suspicious cases (S, R), and their precision and recall are already measured by the case-level measures. The obtype recognition measure is introduced with a goal to address the drawback of precision and recall on the case level which cannot inform the obfuscation types of cases reported by the system, but not to have a redundancy in measurement.

5.2.2.5 An Accuracy Measure for No-plagiarism Case

In this study, we were also challenged to evaluate system performance on detecting documents which contain no cases of plagiarism. Since test documents containing no-plagiarism cases have no references to any source document, we perceived that precision and recall measures become inappropriate assessment for no-plagiarism case detection. For this reason, we tend to measure the detection accuracy by taking the advantage of boolean function. For the convenience of notation, we shortened this measure into *noPlagDet*.

In order to compute a noPlagDet rate, we created a single case in a gold label for a d_{plg} with no-plagiarism case label. Thus, the case set for a test document containing no-plagiarism case, S, is defined to consists of a single element only, and the values of its attributes for source documents (*srcLen, srcOffset,* d_{src}) are defined to be empty. Correspondingly, the whole suspicious document, d_{plg}, is considered to be a single case or passage with an obfuscation type of *no-plagiarism.* The consequence is that the values of d_{plg} attributes such as *plgLen, plgOffset* need to be defined in S. The pair of cases defined in S is demonstrated in the example 3 under d_{plg_3}. Unlike in S, the set of cases reported by the system (R) has a possibility to have more than 1 element. If R has a single element only, for example r_1, there are only 2 prossibilities whether all atrributes of its d_{src} and d_{plg} are assigned or not. If the cardinality of R is greater than 1, it is highly probable that all atrributes in r_i have been assigned a value. There is no possibility that only the attributes of d_{plg} in R are assigned as in S. This is caused by the filtering techniques applied on the post-processing phase which filter out all pairs of cases in R whose length are less than 125 characters for source passages aligned to suspicious passages whose length are less than 150 characters. 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 $\langle 0,0,0,0,0\rangle$ or $\langle 1,1,1,1,1\rangle$. Unlike in r, The tuple attributes in s have only the following boolean values $\langle 0,0,0,1,1\rangle$ as demonstrated in example 3.

Example 3. *Pairs of cases*
A case defined in S for a d_{plg_3}
[s7] ⟨ *srcLen=' ', srcOffset=' ',* d_{src}*=' ', plgLen=32487, plgOffset=0 case=noPlag* ⟩
A case reported in R
[r6] ⟨ *srcLen=' ', srcOffset=' ',* d_{src}*=' ', plgLen=' ' plgOffset=' '* ⟩

A case defined in S for a d_{plg_4}
[s8] ⟨ *srcLen=' ', srcOffset=' ',* d_{src}*=' ', plgLen=59864, plgOffset=0 case=noPlag* ⟩
A case reported in R
[r7] ⟨ *srcLen='419', srcOffset='600',* d_{src}*='2279', plgLen='425' plgOffset='83'* ⟩
[r8] ⟨ *srcLen='205', srcOffset='164',* d_{src}*='2281', plgLen='211' plgOffset='625'* ⟩

The boolean value of a pair of sets s and r, $bol(\bar{s},\bar{r})$ is computed by adding each attribute value of s and r, and the $bol(\bar{s},\bar{r}_i)$ is assigned 1 if the addition operation results in 1 for all of its tuple elements, i.e. $\langle 1,1,1,1,1\rangle$. If the addition operation results in $\langle 0,0,0,1,1\rangle$, then the $bol(\bar{s},\bar{r}_i)$ is assigned a value 0; where i refers to the index in R cardinality. The Boolean value of (S, R) of a given d_{plg} is defined to be 1 if at least there is one $bol(\bar{s},\bar{r}_i)$ which has value 1, otherwise its value is 0. The boolean value of $bol(S,R)$ of a given d_{plg} is defined as follows:

$$bol(S,R) = \begin{cases} 1, & \text{if } \exists bol(\bar{s},\bar{r}_i) \in bol(S,R) \text{whose value is 1} \\ 0, & \text{otherwise} \end{cases} \tag{5.17}$$

Based on equation 5.17, the noPlagDet score which is actually a macro-average of bol(S, R) is then computed as follows:

$$noPlagDet(S,R) = 1 - \frac{\sum_{j=1}^{N} \; bol(S_j,R_j)}{N} \tag{5.18}$$

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

To illustrate how equations 5.17 and 5.18 work, let us see the example 3. Given d_{plg_3} and d_{plg_4} as test documents, we have the followings:
For the d_{plg_3}:

$$\begin{aligned}
s_0 &= \langle 0,0,0,1,1\rangle \\
r_0 &= \langle 0,0,0,0,0\rangle \\
bol(\bar{s}_0,\bar{r}_0) &= \langle 0+0,\ 0+0,\ 0+0,\ 1+0,\ 1+0\rangle \\
&= \langle 0,0,0,1,1\rangle \\
&= 0
\end{aligned}$$

Since $\mid R_{dplg3} \mid = 1$, the boolean value of bol(S, R) of d_{plg_3} equals to its $bol(\bar{r},\bar{s})$ which is 0. The bol(S, R) for d_{plg_4} will be computed as follows:
For the d_{plg_4} :

$$\begin{aligned}
s_0 &= \langle 0,0,0,1,1\rangle \\
r_0 &= \langle 1,1,1,1,1\rangle \\
r_1 &= \langle 1,1,1,1,1\rangle \\
bol(\bar{s}_0,\bar{r}_0) &= \langle 0+1,0+1,0+1,1+1,1+1\rangle \\
&= \langle 1,1,1,1,1\rangle \\
&= 1 \\
bol(\bar{s}_0,\bar{r}_1) &= \langle 0+1,0+1,0+1,1+1,1+1\rangle \\
&= \langle 1,1,1,1,1\rangle \\
&= 1
\end{aligned}$$

The bol(S, R) for $d_{plg_4} = \{1{,}1\} = 1$ since it has at least one bol($\bar{s}_0$, $\bar{r}_0$) whose value is 1. The computation of noPlagDet score could be seen on the followings:

$$\begin{aligned}
noPlagDet(S,R) &= 1 - \frac{\sum_{j=1}^{N}\ bol(S_j,R_j)}{N} \\
&= 1 - \frac{0+1}{2} \\
&= 1 - \frac{1}{2} \\
&= 1 - 0.5 \\
&= 0.5
\end{aligned}$$

5.3 Conclusion

This chapter described two main issues supporting a construction of an External Plagiarism Detection: Corpus building and evaluation measure framework. The corpus building section starts with a survey which was conducted to 3 different groups: individual researches for Indonesian texts, PAN PC, and HTW research center, Berlin. It can be concluded

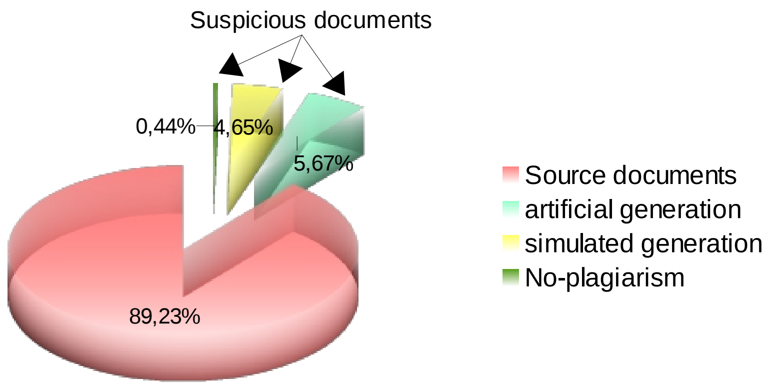


Figure 5.7: Distribution of suspicious and source documents in PlagiarIna's corpus

that there has been no standardized corpora for evaluating EPD systems for Indonesian texts, therefore each research either builds its own corpus or uses the available English corpora. Further, the test documents experimented in Indonesian EPD still deal with literal or verbatim plagiarism with a high percentage of copy from one source document into one test document. It can be assumed that most test documents were generated by the researchers themselves, as there have no information on who were involved in developing the test document corpus. The last two groups surveyed are two organisations which have been doing continual evaluations on Plagiarism Detection Systems as research prototypes as well as commercial systems.

The corpus building strategies from these two organisations influenced much the process of building evaluation corpus for PlagiarIna. This was deliberately done as an effort to standardize PlagiarIna's corpus with an aim that it would be developed further in future and be the standard corpus of evaluating EPD for Indonesian texts. The influence can be seen clearly on the obfuscation strategy applied in creating suspicious documents through algorithmic generation and simulation by human writers. However, the nature of suspicious documents resulted from simulation in PlagiarIna is much closer to a real plagiarism case, in which one suspicious document contains different kinds of obfuscation types. This nature marks its difference from PAN's test document corpus, in which one suspicious document contains one obfuscation type only. The implication is that it increases the complexity of the suspicious document, and the recognition process of such documents becomes more challenging for an EPD algorithm. To wrap up the section of corpus building, Figure 5.7 displays the distribution of suspicious documents (resulted from simulation & artificial generation) and source documents in PlagiarIna's corpus.

The only standardized evaluation measures for EPD systems are those defined for PAN competitions which introduce three levels of measurement: performance measures on the character, case, and document levels. We adapted three levels of measurement proposed by PAN. To address the drawbacks of these three measures, we introduced a measure for recognizing an obfuscation type (obtype recognition) and the accuracy of detecting a no-plagiarism case (noPlagDet rate). We applied macro-averaged precision and recall for the first three levels and a micro-average recall for obtype recognition. The computation of noPlagDet score is based on the boolean function. Besides, we borrowed also granularity and Plagdet score from PAN'14 measures. Lastly, it would be interesting to observe the results of an evaluation framework with a bottom-up approach applied to our prototype, PlagiarIna, which was constructed through a top-down approach (cf. section 4.1).

Chapter 6

Experiments and Quantitative Evaluation

This chapter presents the system evaluation and experiment results performed on the evaluation corpus described in Chapter 5. The aim is to evaluate the proposed methods and to identify which method shows the best performance. As there are different methods applied on each main subtask of the system, the evaluation is organized to assess each of these subtasks separately, i.e. the source retrieval subtask and text alignment subtask. However, the evaluation on PlagiarIna's performance as a workflow system needs also to be assessed. Therefore, Section 6.2 describes the evaluation, experiment results, and discussion on methods applied in retrieval subtask, while the experiment results on text alignment will be presented in Section 6.3. Section 6.4 discusses the evaluation strategy and experiment results of PlagiarIna as a workflow system. Preceding all of these, a short description on document test set is presented in Section 6.1.

6.1 The Test Set

One of major challenges in evaluating an External Plagiarism Detection (EPD) system is to provide a representative corpus of test documents that emulates the real situation [128]. However, the interpretation on *a representative test document corpus* differs among research groups and institutions involved in this field. The rationale is that a test document portraying a real situation of plagiarism case presents another challenge in measurement process as it requires specific evaluation strategy and measures. For this reason, most EPD systems reviewed in Chapters 2 and 3 performed experiments with a scenario of one obfuscation type per test document. Such scenario makes the evaluation process simpler, but it contradicts the former goal of emulating a real situation of a plagiarism case, in which various types of obfuscation are found in one plagiarized text or test document. In evaluating the performance of PlagiarIna, we performed experiments with texts containing various obfuscation types per test document as well as one obfuscation type per document.

This situation led us to evaluate some performances of Text Alignment subtask in micro scale and some performances in macro scales (cf. section 5.2.2). For this reason, we decided to select 70 documents as a test set which comprises 30 documents or test cases from simulation process, 30 test cases from algorithmically obfuscated texts, and 10 test documents for no-plagiarism cases. The test set selection is based on the composition of obfuscation types, level of obfuscation, and simulation batches. Thus, this test set is meant to represent simulated, artificial, and no-plagiarism cases. 33% of test documents representing simulated plagiarism cases contain one obfuscation type only, while the rest, or 67%, contain more than one obfuscation type. The detail information on each test case selected from simulation process is described on Table C.1 in Appendix C.

As there are five types of obfuscation in the artificial plagiarism cases, each obfuscation type, which takes the form of *deletion, insertion, deletion plus insertion, synonym replacment*, and *word shuffle*, is represented by 6 test cases. The information on test cases selected from artificial plagiarism cases is presented on Table C.2 in Appendix C.

6.2 Experiments on Retrieval Subtask

Using the test set explained earlier, we performed experiments on source retrieval methods that are basically composed of three main building blocks: document representation, query formulation, and filtering techniques for selecting the potential candidate source documents. Both source and suspicious documents in this system are represented by weighted vectors of terms, which are formed from three features: phrasewords, token, and character n-grams. These features become the basic objects of evaluation, as they determine and influence the tuning up parameters in query formulation as well as in filtering techniques. Two parameters in query formulation are the length of a text segment or window and the number of queries per window. Due to different characteristics of these three features, the window length was set at a different value for each document feature. Otherwise, the system would retrieve zero candidate source documents. The same strategy was applied also in tuning up the filtering parameters which comprise the minimum number of similar queries, the minimum cosine value, and the top n-ranked candidate documents.

In evaluating the retrieval subtask performance, we applied Macro-average Precision, Macro-average Recall, and Macro-average F-score (F1) as measures. The application of these measures is made possible since the plagiarism case, obfuscation types and levels are not the object of measurement as in Text Alignment subtask. Though they influence the retrieval outputs, they have no direct influence on the measurement process. The computation of Macro-Average precision, Macro-average Recall and Macro-Average F1 are displayed on equations 5.4, 5.5, and 5.6 in the former chapter.

6.2.1 Source retrieval Using Phrasewords

The idea of using phraseword as features for representing documents is based on the characteristics of Indonesian which are prone to any modification on the morphological and syntactical levels as being described in Sections 3.2 and 3.3. Phraseword is meant to capture consecutive words in an inexact matching so that any modified consecutive words or phrases could be matched too [87]. In this experiment, the phrasewords which use a token length in their codes are referred to as phraseword type 1, and the one that use the first two letters of a token to code a term is referred as phraseword type 2. Figure 6.1 exemplifies phraseword type I and II. We evaluated the performance of these two types of phrasewords and explored the granularity of phraseword by varying its size which ranges from 2 to 4-grams. We did not run an experiment on longer size than 4, since a greater size of consecutive substrings is good at matching the exact copy but they tend to be detrimental for matching the obfuscated texts.

Masalah kebakaran dan asap di Indonesia mengalami peningkatan yang cukup serius. Bahkan baru-baru ini dilaporkan oleh beberapa media massa bahwa terjadi masalah kebakaran dan asap di Kalimantan Barat dan Riau.

(a) a raw text

masalah bakar asap indonesia alam tingkat serius. Lapor media massa jadi masalah bakar asap kalimantan riau.

(b) preprocessed text of (a) by applying stopword removal and stemming

7m 5b 4a 9i 4m 6t 6s. 6L 5m 5m 4j 7m 5b 4a *k 5r.

(c) metatokens for building phraseword type I

7m5b4a 5b4a9i 4a9i4m 9i4m6t 4m6t6s 6t6s6l 6s6l5m 6l5m5m 5m5m4j … 4a*k5r.

(d) Phrasewords 3-grams of type I

ma ba as in al ti se. La me ma ja ma ba as ka ri

(e) metatokens for building phrasewords type II

mabaas baasin asinal inalti altise tisela selame lamema memaja.... askari

(f) Phraseword 3-grams of type II

Figure 6.1: An example of how to generate phrasewords type I and II

We took observation also on the effects of applying stopwords and stemming. As it has been mentioned before, our stopword lists are of two types: the frequency-based stopwords, and the semantic-based stoplist which was composed by Tala [177] and referred to as Tala-stopword in this context. The combination of using stopping and stemming results in 4 methods for phraseword type I, and two methods for phraseword type II. In phraseword type II, stemming becomes a necessary process to apply. Otherwise, the code for phraseword would merely represent prefixes. Table 6.1 summarizes the methods and its abbreviations which will be used in reporting the experiment results. In its notation, we use two digit letters to abbreviate the feature names, 1 numeric digit for the feature types, then followed by another numeric for the preprocessing techniques. For an example,

Table 6.1: The notation convention on the applied methods

Code	Description
1	frequency stopword
2	frequency stopword + stemming
3	Tala-stopword
4	Tala-stopword + stemming
PW	Phraseword
TK	Token
NG	N-grams

PW11 stands for Phraseword type 1 which is generated after applying stopword elimination on the preprocessing, and TK2 refers to the use of stemmed token which undergoes the removal of frequency-based stopwords on the text normalization process.

In order to get the ideal parameter values of query formulation, we run some pilot experiments which tested the first two methods of phrasewords (PW11, PW12). In these pilot experiments, we varied the window length of 50, 75, and 100 phrasewords, and query number of 10, 15, and 20 phrasewords per window. Based on the pilot experiment results, we set up the window length value to be 100 phrasewords with 10 query candidates per window. The 10 queries are selected from the top 8-highest scored phrasewords and the 2-least scored phrasewords in a window. Before submitting queries to the comparison function, a redundancy filtering technique is applied to all accumulated queries to make sure that queries representing a test document is a unique phraseword. Using the same approach as in query formulation, we came up to the following threshold values for filtering the candidate source documents: minimal similar queries are set up to 2, the threshold value of cosine score is set up to 0.05 for phraseword 2-grams in simulated plagiarism cases, 0.1 for phraseword 2-grams in artificial plagiarism cases, 0.007 cosine threshold for phrasewords 3-4 grams both in simulated and artificial plagiarism cases. Table 6.2 presents the results of the experiment using these parameters for the test set in simulated plagiarism case.

Table 6.2: Retrieval results using Phrasewords for simulated plagiarism cases. In this table, MAP is an acronym for 'Macro-averaged precision', while MAR stands for 'Macro-averaged Recall'. PW11 refers to Phraseword type I from method 1 (cf. Table 6.1), and PW24 refers to Phraseword type II from method 4.

Methods	**2-grams**			**3-grams**			**4-grams**		
	F-1	MAP	MAR	F-1	MAP	MAR	F-1	MAP	MAR
PW11	.31	.23	**.73**	**.32**	**.24**	.49	.51	.50	**.52**
PW12	.33	.26	.69	.25	.20	.60	.50	.53	.47
PW13	**.65**	**.64**	.66	.29	.21	.46	.48	.44	.50
PW14	.34	.27	.68	.20	.12	**.66**	.42	.39	.50
PW22	.38	.28	.60	.28	.20	.53	**.55**	**.66**	.46
PW24	.39	.39	.39	**.29**	.20	.49	.46	.48	.44

Two different treatments are applied in measuring the retrieval performance. In simulated plagiarism cases, the macro-average measures were simply applied to all test cases, while in articial plagiarism cases, the macro average precision and recall are reported for each test set category which is based on the obfuscation type. Table 6.3 reports the experiment results on artificial plagiarism cases for phraseword type I, while the results of source rerieval using phrasewords type II are presented in Table 6.4.

Table 6.3: Retrieval results using Phraseword type I for artificial plagiarism cases.

Obfuscation	**Methods**	**2-grams**			**3-grams**			**4-grams**		
		F-1	MAP	MAR	F-1	MAP	MAR	F-1	MAP	MAR
Deletion	PW11	.44	.39	.50	.22	.17	.33	.51	.46	.66
	PW12	**.52**	**.47**	**.83**	.33	.22	.66	.11	.05	.83
	PW13	.41	.33	.50	.27	.25	.33	**.58**	.55	.66
	PW14	.44	.35	**.83**	.18	.12	.50	.57	.49	.83
Insertion	PW11	.25	.18	.66	.15	.11	.33	.85	.75	1
	PW12	.04	.02	.83	.15	.10	**.83**	**.88**	**.83**	**1**
	PW13	.28	.19	.66	.08	.05	.16	.83	.72	1
	PW14	**.55**	**.48**	**.83**	**.19**	**.18**	.50	.77	.69	1
Deletion + Insertion	PW11	.34	.25	.83	.48	.45	.66	.80	.75	1
	PW12	**.63**	**.56**	**1**	**.61**	**.56**	**1**	**.94**	.91	**1**
	PW13	.42	.33	.83	.52	.42	.83	.80	.72	1
	PW14	.52	.37	.10	.44	.30	1	.72	.61	1
Synonym	PW11	.35	.31	.66	**.52**	**.51**	**1**	**.79**	**.71**	.90
	PW12	.50	.39	.83	.48	.38	.83	.72	.67	**1**
	PW13	.34	.27	.66	.51	.45	.66	.63	.55	.83
	PW14	**.51**	**.40**	**1**	.43	.34	.83	.69	.63	.83
shuffle	PW11	**.44**	**.39**	**.66**	0	0	0	0	0	0
	PW12	.11	.07	.66	.008	.005	.16	0	0	0
	PW13	.13	.09	.66	0	0	0	.008	.005	.16
	PW14	.13	.07	.66	0	0	0	0	0	0

6.2.1.1 Results and Discussion on Source Retrieval Using Phrasewords

Concerning the phraseword granularity, we hypothesized that the greater size of substring to generate phrasewords (PW) would result in higher recall and precision rates for phraseword 2- to 4-grams. This hypothesis turns out to be true partly, as the experiment results displayed in Tables 6.2-6.4 show that phraseword 3-grams have the lowest F1, recall and precision scores almost in all methods both in simulated and artificial plagiarism cases. In average, PW 4-grams show the highest scores for F1, recall and precision, except in the obfuscation category *shuffle* and *deletion* in artificial plagiarism cases (APC). Even in the obfuscation categories of insertion, deletion plus insertion, and partly in synonym, PW 4-grams are able to retrieve all source documents which is proved by its recall rate to be 1.

The interesting thing is that the highest recall rate in the simulated plagiarism case (SPC) is achieved by phraseword 2-grams with 0.73 (see Table 6.2). Besides, recall rates of PW 2-grams are in average higher than PW 4-grams for SPC. PW 2-grams proves to be

Table 6.4: Retrieval results using Phraseword type II for Artificial plagiarism

Obfuscation	**Methods**	**2-grams**			**3-grams**			**4grams**		
		F-1	MAP	MAR	F-1	MAP	MAR	F-1	MAP	MAR
Deletion	PW22	**.63**	**.58**	**.83**	.22	.15	.66	.40	.31	.66
	PW24	**.63**	**.58**	**.83**	.09	.06	.66	.50	.47	.66
Insertion	PW22	.46	.41	1	.30	.21	.66	**.76**	**.68**	**1**
	PW24	.59	.52	.83	.29	.25	.50	.74	.66	1
Deletion Insertion	PW22	.58	.48	1	.29	.17	1	**.74**	**.63**	**1**
	PW24	.69	.63	1	.28	.16	1	.74	**.63**	**1**
Synonym	PW22	.67	.56	1	.52	.42	1	.52	.42	1
	PW24	.56	.49	.83	.44	.33	1	**.83**	**.75**	**1**
Shuffle	PW22	**.08**	**.04**	**.67**	.07	.04	.05	0	0	0
	PW24	.02	.01	.50	.008	.005	.16	.01	.01	.16

the most robust size of PW for the category of shuffle, as it is able to retrieve the source documents with recall rate of 0.50 to 0.66 when the other Phraseword sizes mostly fail. These experiment results show that the finer granularity of phrasewords has better recall rates for heavily-obfuscated texts which are represented by shuffle and simulated plagiarism cases.

Comparing the performance between PW1 against PW2 is inseparable from doing comparison of using stopping and stemming. For this reason, we tend to compare the performance of PW22 against PW12, and PW24 against PW14 from Tables 6.2, 6.3, and 6.4. The constant fluctuation rates of F1, recall and precision shown in those tables hardly enable us to derive any general conclusion in regard to the performance of PW1 against PW2. However, the implementation of PW2 under the granularity of 2-grams leads to an increase in almost all measures (F1, precision, and recall) in artificial plagiarism cases, but it drops the recall rates of simulated plagiarism cases.

The results displayed in Tables 6.2, 6.3, and 6.4 show that the use of Tala-stopword per se (PW13) is competitive to the use of frequency stopword (PW11). Under the granularity of 2-grams, the use of Tala-stopwords increases both precision and F1 scores in simulated plagiarism cases (SPC) and in deletion, insertion, and deletion plus insertion in artificial plagiarism cases (APC). However, the use of frequency-stopword leads to higher recall rates but to a lower precision and F-1 for PW 3-4 grams in SPC and in APC. This experiment shows the opposite results from the former research on Information Retrieval conducted by Asian who reported that the use of frequency-based stopword leads to a decreased recall and precision, while the semantic-based stopword increases recall and precision [15]. Interestingly, the combination of stemming with frequency-stopwords hurts recall rates in SPC, but increases its recall rates in APC. The use of stemming combined with frequency-based stopword leads to a highest precision and F1-scores only in PW24. Unlike in Phraseword 2- and 4-grams, the use of stemming in Phraseword 3-grams increases recall rates, disregarding its combination with either frequency-stopwords or Tala-stopwords.

Based on the main task of source retrieval subtask, we favor methods which lead to a higher recall rate. This means that the phraseword type I (PW1) with granularity of 4-

grams is applied better for retrieving artificial plagiarism cases, while PW1 with granularity of 2-grams is more appropriate for retrieving simulated plagiarism cases. Based on recall rates, the use of stemming and frequency stopword is best applied for retrieving source documents of artificial plagiarism cases, while the use of frequency-stopword per se fits for retrieving source documents of the simulated plagiarism cases.

For artificial plagiarism cases (APC), a supplementary filtering technique was applied by selecting the top-35 ranked documents as candidate documents if the number of candidate documents outputted from the filtering technique reported in Section 4.2.4 is greater than 35. Based on our observation, this filtering technique has no effect in decreasing recall rate, yet it increases precision insignificantly. The rationale is that each d_{plg} in APC has only one annotated d_{src}.

6.2.2 Source Retrieval Using Token

Similar to retrieval using phraseword, a pilot experiment for tuning up retrieval parameters using token was conducted. In this pilot experiment, we observed the effect of using window length of 200, 250, and 300 tokens with 10, 15, and 20 queries per window. We observed also the possibility of using the top n- and the least m-ranked tokens in a window, where n ranges from 5, 8, 9, 10 and m is specified to 0,1,2,5. In order to have a more balance between precision and recall rates, we set up the window length to be 200 tokens and the number of query equals to 10 for each window. We dropped the idea of using the least m-scored token, as it hurts both precision and recall when it is applied to token. Thus, the queries per window were selected from the top-10 tokens scored by tf-idf. We set up the cosine threshold to be 0,007 as in phraseword 3-, 4-grams, and the minimum similar number of token to be 2. These parameters were applied equally to both artificial and simulated plagiarism cases.

Unlike phrasewords, the possibility of having word redundancy among these top 10-ranked tokens is much greater in tokens. For this reason, the algorithm in query formulation is assigned to check the query uniqueness within these 10 queries. If the number of unique queries is less than 10, the algorithm is assigned to select the next highest weighted token till it gets 10 unique queries per window. As in phraseword query formulation, the second stage of word redundancy filtering process is applied to the document queries, which are resulted from the union of candidate queries per segment or window.

In using token as document features, we observed the effect of using 2 kinds of stopword lists, stemming, and their combinations on recall and precision rates. The use of frequency stopword, and its combination with stemming are coded under TK1, and TK2, while TK3 and TK4 stand for methods which use Tala stopword and Tala stopword with stemming. Table 6.5 presents the experiment results on source retrieval using tokens for simulated plagiarism cases. In artificial plagiarism case, we observed these four methods under five obfuscation types. Table 6.6 describes the results of source retrieval for articial plagiarism cases.

Table 6.5: Source retrieval using token for simulated plagiarism cases. In this table, **MAP** stands for *macro-averaged precision*, and **MAR** refers to *macro-averaged recall.* The column **time** indicates time measure in seconds.

Methods	**F-1**	**MAP**	**MAR**	**Time in sec**
TK1	**.50**	.40	**.67**	4.8
TK2	.44	**.41**	.49	4.8
TK3	.31	.28	.47	5.5
TK4	.32	.36	.28	5

6.2.2.1 Results and Discussion on Source Retrieval using Token

Table 6.5 clearly shows that the use of stemming hurts recall rates but leads to an increased precision rate and stabilizes the rates between recall and precision, as it can be seen in TK2 compared to TK1 and TK4 to TK3. Independent of its use without stemming (TK1 vs TK3) or with stemming (TK2 vs TK4), the frequency stopword outperforms Tala-stopwords in all measures: F-1, precision and recall. In simulated plagiarism case (SPC), the highest recall rate, 0.67, is gained by the use of frequency stopword per se. This results show data consistency in comparison to the retrieval results using phraseword, where the use of the frequency stopword leads to the highest recall in SPC. The possible explanation for it is that the queries selected from the highest tf-idf scores are likely terminologies from loanwords or words in baseforms which are not affected very much by the stemming process.

Unlike in simulated plagiarism cases, the use of stemming leads to an increase in recall rates of artificial plagiarism cases, except in the obfuscation category of Shuffle which shows the opposite effect. From Table 6.6, it can be clearly seen that the use of Tala stopword decreases precision and F1, if it is compared to the use of frequency stopwords. However, the combination of Tala and stemming leads to the highest F1, recall and precision rates for the obfuscation types of Deletion, Insertion, and synonym replacement. The obfuscation types of Shuffle and Deletion plus Insertion display the opposite results, where the optimal recall rates of 1 are produced by methods using frequency stopwords and Tala-stopwords without stemming. Thus, the results of these two obfusctaion types correspond to those of source retrieval using phrasewords and token in SPC.

Compared to the use of phrasewords, the source retrieval using tokens generally show lower scores in almost all measures both in artificial plagiarism cases (APC) and SPC. The highest F1 scores in APC for phrasewords reaches 0.94, and precision reaches 0.91, while the highest F1 score of using token reaches 0.60 and 0.47 for the precision rate. Though the highest recall rates of both token and phrasewords reach 1, which means all sources are retrieved, phraseword outperfoms token, as phraseword methods achieve higher number of optimum recall rate, 1.00, than token's. Both token and phraseword indicate that retrieving source documents of simulated plagiarism cases is more challenging than those of artificial plagiarism cases, which is proven by the lower rates at all measures in

Table 6.6: Retrieval results of using token in Artificial plagiarism cases

Obfuscation	**Methods**	**F-1**	**MAP**	**MAR**	**Time in sec**
Deletion	TK1	.23	.17	.50	7.5
	TK2	.35	.34	.66	5
	TK3	.21	.14	.50	7
	TK4	**.45**	**.35**	**.66**	8
Insertion	TK1	.51	**.41**	.66	6
	TK2	**.54**	.40	**.83**	7
	TK3	.51	**.41**	.66	6
	TK4	**.54**	.40	**.83**	7
Deletion + Insertion	TK1	**.37**	**.24**	.83	6
	TK2	.23	.13	**1**	7
	TK3	.05	.02	.83	7
	TK4	.05	.03	1	9
Synonym	TK1	.36	**.30**	.50	7
	TK2	.21	.12	**.83**	9
	TK3	.30	.22	.50	10
	TK4	**.43**	.29	**.83**	9
Shuffle	TK1	**.60**	**.47**	**1**	6
	TK2	.33	.26	.66	5
	TK3	.15	.08	**1**	8
	TK4	.37	.29	.83	6

SPC. However, token outperforms phraseword in source retrieval for the obfuscation type of Shuffle in APC. This has been predicted, as token is unaffected by the sequence order and length as phraseword is. Besides, the bag of word model applied in this algorithm makes token a more appropriate feature for matching a text that is highly shaked and shuffled.

The effect of tuning parameter values of window length and query number per window on precision and recall rates could be clearly observed in source retrieval using token as features. The window length correlates highly with the test document length. As most of our test documents could be categorized in short texts (see Tables 5.6 & 5.8), we did not run a test with a window length longer than 300 tokens in our pilot experiments. The shorter window length will not automatically increase recall rate, as it also correlates highly to the number of selected queries. The wider window length favors only methods 1 and 2, and tends to be detrimental to shorter test cases and to methods 3 and 4 which result in much shorter texts than their original length because of the Tala-stopword removal (semantic-based stopwords).

The greater number of queries per window will not always lead to an increased recall rate. The rationale is set on the matter whether the hidden plagiarized passages are represented in the queries. In this matter, the portion of plagiarized passages plays an important role. If the plagiarized passage is too short as in many cases of summary, their possibility to be represented in a query, even in greater number of queries per window, remains low.

The combination of the highest and lowest weighted scores of tokens to be parameters of query selection is more appropriate for longer sequences of strings or metastrings as in phrasewords. For token, such combination leads only to decreased precision and recall rates, as it is presented in Table 6.7. For this reason, such combination was not applied in source retrieval using tokens.

Table 6.7: Pilot experiment results using token as features with window length=250, n={8, 5}, and m={2, 5}. N refers to the top n-ranked tokens, and m stands for the lowest m-ranked tokens.

Methods	n=8, m=2			n=5, m=5		
	F-1	MAP	MAR	F-1	MAP	MAR
TK1	.35	.27	.50	.009	.005	.66
TK2	.33	.24	.56	.002	.001	.17

6.2.3 Source Retrieval Using Character N-grams

We assumed that character n-grams are more capable in capturing the morphological modification within a lexical level, and thus we perceived that character n-grams become a more appropriate feature in processing Indonesian texts rather than word n-grams. Undoubtfully, plagiarism detection systems reviewed in Section 3.3 hypothesized the same thing, as the majority of them worked on document features at the level of characters for Indonesian texts. For this reason, we apply a slightly different method in using character n-grams as document features. The frequency stopword removal becomes a standard preprocessing in the document normalization phase. After n-grams generation, the most common n-grams were removed by means of character n-stopgram lists which are a kind of stopword list on the level of character n-grams. We assumed that most n-grams containing affixes will be removed in this process. Therefore, stemming becomes an unnecessary process for character n-grams.

In order to be consistent in performing experiments, we applied the similar approach in tuning up retrieval parameters values. The differences are set on the parameter values for the window length. Based on our pilot experiment, we came to set the window length to be 300 with 10 queries per window. The window queries are selected from the top-8 highest scored n-grams and the 2-lowest scored n-grams as in phraseword. In this experiment, we did our observation on the n-gram whose length ranges from 4-7. We tested fine-grained n-grams with an argument that the shorter n-gram sequence is more robust in global-scale matching, while the longer n-gram sequence will perform better in a local-scale matching. As retrieval subtask is a process of matching in the global scope, we favored fine-grained n-grams. Unlike in phrasewords or token, we set up the number of similar queries to be 3 minimally and the minimum cosine threshold at 0.01 as the filtering parameters.The experiment results of source retrieval using character n-grams is presented in table 6.8 for

Table 6.8: Statistical data on source retrieval using N-grams for simulated plagiarism case. In this table, *MAP* stands for macro-averaged precision, and MAR refers to macro-averaged recall.

Methods	**F-1**	**MAP**	**MAR**	**Time in sec**
NG4	.22	.14	.60	**42**
NG5	.14	.07	**.79**	54
NG6	.17	.11	.73	76
NG7	**.23**	**.14**	.63	128

simulated plagiarism cases and in Table 6.9 for artificial plagiarism cases. The granularity of n-grams is used to code the method, eg. NG4 refers to character 4-grams.

Table 6.9: The experiment results of source retrieval using N-grams for Artificial Plagiarism cases

Obfuscation	**Methods**	**F-one**	**MAP**	**MAR**	**Time in sec**
Deletion	NG4	**.18**	**.11**	**.50**	362
	NG5	.02	.01	.50	199
	NG6	.12	.07	.50	725
	NG7	.16	.10	.50	695
Insertion	NG4	.03	.02	.66	280
	NG5	.05	.03	.66	335
	NG6	**.19**	**.13**	**.66**	509
	NG7	.19	.11	.66	395
Deletion + Insertion	NG4	.09	.05	.50	974
	NG5	.06	.03	.83	974
	NG6	**.19**	**.11**	**.83**	952
	NG7	.15	.08	.83	1227
Synonym	NG4	.15	.11	.50	109
	NG5	.06	.03	.50	291
	NG6	.12	.12	.50	377
	NG7	**.45**	**.42**	**.66**	515
Shuffle	NG4	.05	.03	**1**	69
	NG5	.04	.02	.83	525
	NG6	.11	.05	.83	528
	NG7	**.14**	**.08**	.66	736

6.2.3.1 Results and Discussion on Source Retrieval using n-grams

It could be seen that the highest recall in simulated plagiarism cases is at 0.79 which is produced by character 5-grams. However, the highest precision is 0.14 which is achieved by 4- and 5-grams, while the highest F1 score is gained by using 7-grams at 0.23. The highest precision score gained by n-grams becomes the lowest one among the highest precision rates of other methods using phrasewords and token. This causes the maximum F1 score of n-grams to be the lowest among other maximum F1 scores. Interestingly, The highest recall achieved by n-grams becomes the highest recall rate if it is compared to the highest recall rates produced by phraseword or token in simulated plagiarism cases (SPC). Moreover, the recall rates among different granularity of n-grams does not fluctuate as recall rates produced by any methods using token. The lowest recall in SPC which is 0.60 is quite high. Unfortunately, this high recall rates are followed by much lower precision rates which make n-grams be a less competitive feature representation compared to token or phrasewords. Which granularity of n-gram shows the best performance in SPC is hard to tell. Based on recall rates, character 5-grams outperforms other n-grams, but based on F-1 score, character 7-grams shows the best performance.

N-grams performance in retrieving source documents for artificial plagiarism cases (APC) shows similar characteristics as in SPC, i.e. they have relatively high recall rates in all obfuscation types followed by a great gap between recall and precision rates. The highest recall rate is 1.00 which is produced by character 7-grams in the obfuscation type of Insertion and by 4-grams in Shuffle. In a heavily shuffled texts, character n-grams prove to be a robust feature representation and this makes it competitive to tokens, as the lowest recall rate in Shuffle category is 0.66. The fine granularity of character n-gram fits to be applied for retrieving d_{plg} which is shuffled, as it could be observed that the increased granularity of n-grams reduces recall rates. In artificial plagiarism cases, the correlation between n-gram granularity to a retrieval performance could be easily observed, as it could be seen that the highest recall, precision and F-1 scores are obtained by using n-grams with granularity of 6 or 7 characters. The exception falls into the obfuscation type of Deletion whose highest F1-measure rate is achieved by character 4-grams.

In general, character n-grams have a potential to be a good feature representation as it is shown by its highly stable recall rates. The great gap between recall and precision rates found both in artificial and simulated plagiarism cases is probably caused by selecting 2 lowest ranked n-grams as part of the window queries. This strategy was implemented to avoid selecting n-grams referring to the same word or string when the queries are selected only from the top n-weighted n-grams, which in our pilot experiments resulted in averagely lower recall rates. This problem could be solved by applying additional selection method which checks the minimum or maximal number of shared consecutive characters among the selected top-i ranked n-grams. The example given will clarify this idea. Suppose we have a token 'correlates'. Applying character 5-grams as feature representation, we will have the following grams: corre, orrel, rrela, ..., lates. Supposed all 5-grams from this string share the same weight scores which are selected to be queries. The additional filtering method is defined , for example, to select only n-grams sharing maximally 2 consecutive occurences of similar characters. This filtering technique will select only *corre*, and *relat*, and discards other 5-grams from this token. We assume that this selection method will help increase the precision and F1 scores. Unluckily, we have no time to implement this idea, so this filtering technique for n-gram query selection is left for future work.

Another drawback of source retrieval using character n-grams is that it needs longer processing time compared to phrasewords or token. The fastest processing time is 42

Table 6.10: Processing time of source retrieval using phrasewords. The time displayed is in second

methods		**PW11**	**PW12**	**PW13**	**PW14**	**PW21**	**PW22**
Simulated Plagiarism Cases							
2-grams		5.5	5.2	5.3	5.2	7.6	10
3-grams		20	23	15	21	37	41
4-grams		9.8	8.9	12.7	12	14.7	13
Artificial Plagiarism Cases							
Deletion	2-grams	8	8.3	12	11	8.8	9
	3-grams	26	37	32	39	32	35
	4-grams	11	8.3	12	13	8.8	10
Insertion	2-grams	11	12	12	15.6	10	14.6
	3-grams	15	37	32	39	45	46
	4-grams	16	17	19	20.6	15	19.5
Deletion + Insertion	2-grams	8.5	9.3	13	10	9.5	8.5
	3-grams	27	38	35	44	38	43
	4-grams	10	8	8.6	9.6	7.5	10.6
Synonym	2-grams	12	8.6	12	13	9.5	10.8
	3-grams	34	30	27	31	42	40
	4-grams	13.5	17	20	25	42	10
Shuffle	2-grams	9.5	9.5	10	10	8.8	10.5
	3-grams	32	55	35.5	35	41	33
	4-grams	17	7	14.5	15	11	13

seconds, and the longest one needs 1227 seconds. In term of processing time, token proves to be the most efficient feature representation, as its average processing time is less than 10 seconds. The processing time of source retrieval using phraseword both in artificial and simulated plagiarim cases is presented in table 6.10. From this table, it could be seen that the processing time of phrasewords is much more efficient than n-grams, and becomes competitive to token.

In average, source retrieval in artificial plagiarism cases (APC) gains much higher rates in all measures from different methods compared to those of simulated plagiarism cases (SPC). This signifies that firstly the obfuscations which were simulated by human writers, even in its simplest level, are much more complex than those generated algorithmically. Secondly, retrieval results correlate highly not only to obfuscation types but also to obfuscation complexity and the length of the obfuscated passages. These are the main factors that differentiate APC from SPC. In general, the source retrieval still becomes a highly challenging task in External Plagiarism Detection (EPD) when it is evaluated separately. This is proven by PAN retrieval results during the last 3 years which are presented in Table 6.11. This table demonstrates that the overall rates of source retrieval is still far from what is expected. It needs to be noted that Table 6.11 is presented here as a reference but not as a comparison to our retrieval result, since we tested our retrieval subtask on different

Table 6.11: The best PAN's source retrieval results during a three-year period (2013-2015) source [67]

Team	Year	Downloaded sources		
		F1	Prec.	Rec.
Haggag	2013	.38	**.67**	.31
Kong	2013	.01	.01	**.59**
Williams	2013	**.47**	.60	.47
Williams	2014	**.47**	.57	.48

corpus and under different retrieval strategies. Besides, it indicates that source retrieval for EPD has become an open research area that needs more exploration and study in future.

6.3 Oracle Experiments on Text Alignment Subtask

As it is explained in Sections 2.2.2.2 and 4.3, text alignment performs detailed analysis to candidate documents which are outputted from source retrieval subtask. In assessing Text Alignment performance, we run two scenarios of evaluations as follows:

a) **The oracle experiment** on Text Alignment subtask per se. In this experiment, we fed all source documents D_{src} of a given d_{plg} which enables a recall rate to achieve the score 1. This was done by plugging an *intervention function* to the retrieval subtask. The task of this function is to check whether all source documents of a d_{plg} have been retrieved. If not, this function will add the unretrieved source document IDs to the list of source retrieval outputs, on which text alignment bases its task and analysis.

b) Text Alignment as one of components of **a workflow system**. This means to evaluate the performance of Plagiarina as a whole system of an EPD. The consequence is that the intervention function is unplugged from the system, and the analysis of text alignment subtask is based on the candidate documents outputted from the source retrieval subtask only. This scenario will be presented in Section 6.4.

Unlike in retrieval subtask, experiments for text alignment need only a handful of parameters. These parameters are set empirically through a pilot study which took different test cases. Thus, we will not observe the effect of tuning up these parameters to the detection result. Insteads, we will observe the effect of using word unigram and character n-grams as seeds to the detection performance which is measured in four different levels: character, passage, document, and case levels. To see how good the performance of the proposed methods for our text alignment algorithm is, we conducted a comparison to Alvi's algorithm [7]. The reason why Alvi's algorithm was chosen among others are as follows:

a) Alvi's algorithm makes use of fingerprints as seeds, and uses Rabin-Karp algorithm for generating fingerprints.

Table 6.12: The plagdet Scores of Alvi's algorithm tested on PAN'14 corpus
Sources [7, 129]

Obfuscation types	**Plagdet scores**
Overall plagdet scores	0.6-0.7
No-plagiarism	1.0
No-obfuscation	0.9
Random obfuscation	0.4-0.5
Translation	0.5-0.6
Summary	< 0.1

Table 6.13: Results of Alvi's algorithm on test corpora of PAN'14
Sources: [7, 129]

Measures	Test corpus 2	Test corpus 3
Plagdet	0.65	0.73
Precision	0.93	0.90
Recall	0.55	0.67
Granularity	1.07	1.06

b) Alvi's algorithm has been tested to three corpora: PAN'13 test corpus 1, PAN'13 test corpus 2, and PAN'14 corpus. For PAN'14 corpus, it proves to be the second most efficient algorithm in the processing time [129]. This makes it feasible to be implemented as a commercial EPD systems rather than a mere research prototype. Tables 6.12 and 6.13 present the performance of Alvi's algorithm when it was submitted to PAN 2014.

c) Most EPD systems for Indonesian texts, as reviewed in Section 3.4, use Rabin-Karp algorithm or rolling-Hash in generating fingerprints for retrieving source document as well as for doing comparison and matching.

d) Performing comparison to Alvi's algorithm is likely to be walking and chewing gum at the same time, i.e. doing comparison with some EPD systems for Indonesian texts.

e) Compared to EPD algorithms for Indonesian texts reviewed in Section 3.4, Alvi's algorithm has advantages on the detection to the level of passages and on the identification of passage boundaries of source-suspicious passage pairs.

e) Alvi's algorithm applies rule-based approach for seed merging and extesion as our prototype, PlagiarIna, does.

In order to create the same platform for valid comparison, we specifically implemented Alvi's algorithm in the same setting as our EPD system, PlagiarIna. Alvi's algorithm which is presented in Algorithm 3 is also plugged to retrieval module and tested in two scenarios. We experimented the same test cases from the same corpus which are categorized into simulated and artificial plagiarism cases. The performance of Alvi's algorithm was measured also in 4 levels of granularity. Alvi's algorithm needs only two parameters: one for n-gram length where n equals to 20 characters and the gap between matched n-grams which is set to 30 characters. For filtering, alvi's algorithm discards a pair of detected passages whose length is less than 200 characters for the source passage, and 100 characters for suspicious passage.

6.3.1 Text Alignment Using Token as Seeds

Our method of Text Alignment which has been reviewed in Section 4.3 comprises of three-steps: seeding, seed merging and extension, and filtering. We use two kinds of seeds which are weighted and selected using Kiabod's local word score for pruning the significance word (cf. Section 4.3.2.1). One of the seed units that we used is token. In generating seeds, we needed two parameters, α which is used for weighting the local word score and β which is used as a pruning factor. We set up the values of these parameters empirically with $\alpha = 0.6$ and $\beta = 0.5$. The seeding process is used to filter source-suspicious paragraph pairs having Jaccard and Dice similarities above a threshold which is set up to 0.35 for Jaccard index and 0.4 for Dice coefficient. In extending and merging seeds using token, we set up the maximal gap for seeds to be merged is 50 characters in source candidate passages, and 35 characters in the test case or suspicious passages. The gap value is defined relatively small since our Text Alignment algorithm works within a scope of paragraph which is much shorter than a document scope. In the filtering process, we simply discarded aligned passages, if the length of its source passage is less than 125 characters and suspicious passage length is less than 150 characters.

One of challenges in measuring Text Alignment task is how to generate an automatic evaluator which computes all measures in different granularity levels at one time. These challenges are heightened with the following possibilities which we encountered during our pilot experimentation:

1. a suspicious passage has more than one source passage. Such passage is repeatedly annotated and detected as pairs of source-suspicious passages. This situation would present problems of measurement in character level if it is not handled properly, as the number of detected suspicious characters would be repetitively increased. As its result, the precision rate could be greater than 1.

2. A long source-suspicious passage pair are detected by splitting them up into several shorter pairs of source-suspicious passages. This is possible in a situation when the modification length is beyond the defined gap value. In this context, it is not an overlap or repetitive detection as it is commonly found in many detection of EPD systems, since their offsets refer to different locations within this long passage pair. This situation raises a problem of measurement on the passage level but not on the character level.

Algorithm 3 Alvi's Algorithm

```
Input:  $d_{plg}$, $\mathrm{D}_{src},\ \alpha,\ \beta$
Output:  $setsofdetectedPassage(pass_{src},\ passplg)$
  function checkRelation($match_1,\ match_2$)
      if $startMatch_2 \geq startMatch_1$ && $endMatch_2 \leq endMatch_1$ then
          $containment \leftarrow relation(match_1,\ match_2)$
      else if $endMatch_2 \geq endMatch_1\ \geq startMatch_2\ \geq startMatch_1$ then
          $overlap \leftarrow relation(match_1,\ match_2)$
      else if $(startMatch_2 - endMatch1) \geq \alpha$ then
          $nearDisjoint \leftarrow relation(match_1, match_2)$
      else
          $farDisjoint \leftarrow relation(match_1, match_2)$
      end if
  end  function
  $d_{plg} \leftarrow normalized_{plg}$
  for $a = 0,\ a <\mid D_{src} \mid,\ a++$ do
      $D^a_{src} \leftarrow normalize(D^a_{src})$
      $Ngrams_a \leftarrow generateCharNgrams(D^a_{src},\ \alpha)$
      for $b = 0,\ b <\mid Ngrams_a \mid,\ b++$ do
          $hashCode_{ab} \leftarrow rabinKarpHashing(Ngrams_{ab})$
          $hashTable_{src} \leftarrow multipleHashMap(hashCode_{ab})$
      end for
  end for
  $hashCode_{plg} \leftarrow rabinKarpHashing(generateCharNgrams(d_{plg},\ \alpha))$
  $hashTable_{plg} \leftarrow multipleHashMap(hashCode_{plg})$
  for all $d_{src}inhashTable_{src}$ do
      for $c = 0,\ c <\mid hashKey(hashTable_{src}) \mid,\ c++$ do
          for $d = 0$, ¡ $\mid hashKey(hashTable_{plg}) \mid,\ d++$ do
              if $matched(hashKey_{src_c},\ hashkeyplg_d)\ =\ true$ then
                  $matchedRel_{src_c} \leftarrow checkRelation(hashkey_{src_c},\ hashkey_{src_{c-1}})$
                  $matchedRel_{plg_d} \leftarrow checkRelation(hashkey_{plg_d},\ hashkey_{plg_{d-1}})$
                  if $matchedRel_{src_c} = containment$ && $matchedRel_{plg_d} = containment \mid\mid overlap$ then
                      $hashkey_{src_c} \leftarrow merging(hashkey_{src_{c-1}},\ hashkey_{src_c})$
                      $hashkey_{plg_d} \leftarrow merging(hashey_{plg_{d-1}},\ hashkey_{plg_d})$
                  else if $matchedRel_{src_c} = overlap$ && $matchedRel_{plg_d} = containment \mid\mid overlap$ then
                      $hashkey_{src_c} \leftarrow merging(hashkey_{src_{c-1}},\ hashkey_{src_c})$
                      $hashkey_{plg_d} \leftarrow merging(hashey_{plg_{d-1}},\ hashkey_{plg_d})$
                  else if $matchedRel_{src_c} = nearDisjoint$ && $matchedRel_{plg_d} = near - disjoint$ then
                      $hashkey_{src_c} \leftarrow merging(hashkey_{src_{c-1}},\ hashkey_{src_c})$
                      $hashkey_{plg_d} \leftarrow merging(hashey_{plg_{d-1}},\ hashkey_{plg_d})$
                  else
                      $ignore$
                  end if
                  $detectedPairs(merged_{src_c},\ merged_{plg_d}) \leftarrow aligning(hashkey_{src_c},\ hashkeyplg_d)$
              else
                  $continue$
              end if
          end for
      end for
  end for
```

Table 6.14: Results on Text Alignmnet using token for SPC

Methods	Character-based Measures				Case-based			Doc.-based		Time
	Plagdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	
TK1	**.63**	**.76**	**.60**	.97	**.76**	**.66**	**.67**	.89	.72	22
TK2	**.63**	.75	.59	1	.74	.58	.61	**.92**	**.74**	59
TK3	.62	.75	.58	.97	.71	.61	.62	.90	.7	59
TK4	.59	.70	.55	.93	.69	.6	.61	.84	.68	285
Alvi	.53	**.76**	.45	.93	.75	.52	.60	.87	.68	**0.1**

To solve such problems, we built a function which checks the presence of these two conditions. The function will not increase the number of detected characters in suspicious passages for the case 1. As for case 2, the function will detect whether the detected passage pair (r) is a case of containment in an annotated passage pair (s). If it is true, than these several passage pairs will be counted as one detection only, as long as there is no character overlap among these passages. Based on the number of detections filtered by this function, the precision, recall, granularity, plagdet measures are computed with equations presented in Section 5.2.2.

6.3.1.1 Results and Discussion on Text Alignment Using Token

The results of Text Alignmnet task using token as seeds for simulated plagiarism cases are presented in Table 6.14. From this table, it could be seen that the use of stemming decreases recall, precision, F1, and plagdet scores insignificantly on the character-based and passage-based levels. On the contrary, the combination of stemming and frequency stopword increases both recall and precision on the level of document. However, the use of Tala stopword per se and its combination with stemming leads to an insignificant drop in all levels of measures. These results show consistency to those of source retrieval.

The interesting thing to report from the performance of Alvi's algorithm is that its recall rates show insignificant differences from its rates when it was tested on PAN corpus (see Table 6.13), which lie on the range of 0.55-0.67 for PAN corpus, and 0.45-0.68 for our corpus. However, its precision rates drop from 0.90-0.93 by PAN corpus to 0.76-0.86 by our corpus. Compared to Alvi's performance, the recall rates of PlagiarIna are higher in three leves of measurement: character, case and document. In contrast, the precision rates of Alvi's algorithm is higher compared to those resulted from methods TK2, TK3, and TK4. The precision rate of TK1 is competitive to Alvi's algorithm in character-based measure, but insignificantly higher on case-level and document-level measures. Owing to PlagiarIna's competitive granularity, precision and higher recall rates, its Plagdet scores are automatically higher than Alvi's.

The obfuscation types in simulated plagiarism cases (SPC) are categorized into 6 groups: exact copy which is called *no-obfuscation* in PAN corpus (cf. Table 6.12), paraphrase which is distinguished into light, medium, and heavy paraphrases, copy and shake,

Table 6.15: Results on Text Alignment using TK1 method for APC

Obfusc.	Character-based Measures				Case-based			Doc.-based		Obfusc.
	Plagdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	types
Delete	.47	.83	.37	.83	.83	.83	.83	.83	.83	.83
Insert	.72	.98	.59	1	.91	1	.95	.91	1	1
Del+Ins	.91	.95	.85	1	.91	1	.95	.91	1	1
Synonym	.66	.83	.61	.83	.83	.83	.83	.83	.83	.83
Shuffle	.14	.57	.08	.66	.58	.66	.66	.58	.66	.66

and summary. In detecting the obfuscation types which are measured by case-based level (see Table C.5), Alvi's algorithm shows inconsistency of detection. In some test documents such as shown in testdocuments 005, 010, 011, 015, and 027, it detects heavier obfuscated cases better. For example, its recognition to heavy or medium paraphrase reaches 1 or less, but zero rate to light paraphrase. The recognition rate of PlagiarIna under TK1 in SPC is more consistent, in the context that the heavier obfuscation level for paraphrase have lower detection rates than the lighter one in those documents. It turns out that Alvi's algorithm recognizes summarized passages better than Plagiarina, as its detection rate for summary case is 0.08 higher than our algorithm (see Table 6.20).

The alignment results for artificial plagiarism cases (APC) using method TK1 are presented in Table 6.15, while Table 6.16 presents the alignment results for APC using method TK3. Table 6.17 shows the performance of Alvi's algorithm on text alignmnet task for APC. The rest of tables presenting text alignment performance for APC using TK2 and TK4 methods could be found in Appendix C. These tables show that the use of frequency-stopwords per se (TK1) leads to higher Plagdet, recall and precision rates only in character level. The use of stemming (TK2), Tala stopword (TK3), and Tala combined with stemming (TK4) increases precision and recall rates for obfuscation types of *insertion* and *deletion plus insertion* in case and document levels. For PlagiarIna, detecting globally-shuffled documents remains a challenge, as its detection scores for all measures are to be the lowest compared to those from other obfuscation types. However, TK3 shows the highest detection rates in all levels of measures for the obfuscation type of *shuffle*.

In average, Alvi‘s text alignment scores for artificial plagiarism detection as shown in Table 6.17 are lower than PlagiarIna's (cf. Tables 6.15, C.3, 6.16 and C.4). Excluding results from *shuffle* case, Alvi's recall rates on the case, document levels, and obfuscation-types lie on the range of 0,33-0,83. However, Alvi's algorithm fails to detect globally-shuffled test documents as shown by its zero scores on all measures at different levels. The interesting thing to report is that Alvi's Plagdet scores in APC range from 0.10 - 0.52, whose upper score shows similarity to its Plagdet score when it was tested on PAN'14 corpus which lies on the range of 0.4-0.5 for random obfuscation (cf. Table 6.12). In PAN'14, the APC are addressed asrtificial Plagiarism Cases are addressed as *random obfuscation*. One advantage of Alvi's algorithm over PlagiarIna lies on its processing time. It needs less than one (1) second for performing the whole process of text alignment, while PlagiarIna needs

Table 6.16: Results on Text Alignment using TK3 method for APC

Obfusc.	Character-based Measures				Case-based			Doc.-based		Obfusc.
	Plagdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	types
Delete	.43	.83	.3	.83	.83	.83	.83	.83	.83	.83
Insert	.65	.99	.5	1	1	1	1	1	1	1
Del+Ins	.87	.99	.79	1	.92	1	.96	.92	1	1
Synonym	.59	.8	.52	.83	.75	.83	.83	.75	.83	.83
Shuffle	.14	.67	.13	.67	.67	.67	.67	.67	.67	.67

Table 6.17: Results on Alvi's algorithm tested on APC

Obfusc.	Character-based Measures				Case-based			Doc.-based		Obfusc.
	Plagdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	types
Delete	.34	.83	.22	.83	.58	.83	.66	.83	.83	.83
Insert	.44	.82	.3	.83	.83	.83	.83	.75	.83	.83
Del+Ins	.52	.83	.4	.83	.45	.83	.52	.83	.83	.83
Synonym	.10	.33	.06	.33	.25	.33	.28	.33	.33	.33
Shuffle	0	0	0	0	0	0	0	0	0	0

22 to 285 seconds (cf. Table 6.14). This proves that fingerprint methods excel over any vector-based comparison methods on term of computation effort.

6.3.2 Text Alignmnet Using N-grams as Seeds

In this experiment, the seeds are formed by character n-grams whose n ranges from 4-7. The reason of using fine granularity of n-gram is that the majority length of Indonesian stems lie on the range of 4-7 characters. We assumed that n-stopgram could replace stemming, and the n-grams which remain after preprocessing are mostly n-grams derived from stems. For n-grams, we set up different parameters which are based on its length. We keep the value of α to be 0,6 as it has no influence on seed selection, but set up the β = 0,3 for character 4- to 5-grams, and $\beta = 0,2$ for character 6- and 7-grams. As β is a pruning factor for significant local word score, we wished to have more n-gram seeds by decreasing the β value. The threshold of Jaccard coefficient was also decreased to 0,3 for charater 4-5-grams, and to 0,1 for character 6- to 7-grams. The threshold of Dice coefficient was set up to the same value of Jaccard threshold. However, we applied the same maximum gap values for all n-grams. N-gram seeds occurring within 75 characters in the source passages and 100 characters in the suspicious passages will be merged.

Table 6.18: Results on Text Alignmnet using n-gram seeds for SPC

Methods	Character-based Measures				Case-based			Doc.-based		Time
	Plagdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	
4-grams	.17	.36	.12	.6	.32	.13	.17	.64	.37	**172**
5-grams	**.25**	.45	**.20**	.67	.39	.19	.24	.66	.45	375
6-grams	.23	.5	.16	.7	.42	.19	.24	**.68**	**.39**	241
7-grams	**.25**	**.56**	.18	.67	**.58**	**.22**	**.29**	.67	.38	351

6.3.2.1 Results and Discussion on Text Alignment Using N-gram Seeds

The results of using n-gram seeds for aligning source-suspicious passages of Simulated Plagiarism Cases (SPC) could be seen on table 6.18, while table 6.19 presents the results of using n-grams seeds for Text Alignmnet task for Artificial Plagiarism cases. From table 6.18, it can be observed that the character 7-gram achieves the highest F1 scores in case-based measurement, the highest Plagdet and precision scores, while 6-gram gets the highest precision and recall rates on document level. The highest recall rate at a character level is achieved by 5-grams on the rate of 0.20.

In Artificial Plagiarism Cases (APC), character 5-gram gets the highest scores for all measures in obfusctaion type of synonym, while 6-gram outperforms other n-gram lengths in detecting *deletion* case, and 7-gram proves to be the best granularity for detecting *insertion* and *deletion plus insertion* cases. The highest scores for shuffled test documents are distributed to 4-, 5- and 7-grams. Though there is no specific n-gram granularity which dominates the highest scores, it could be noted that character 5-grams are competitive to character 7-grams, as both n-gram granularities get highest scores quiet often in APC and SPC as well.

Table 6.19: Results of text Alignmnet using n-gram seeds for artificial plagiarism detection

Obfusc.	Ngrams	Character-based Measures				Case-based			Doc-based		Obfusc.
		Pldet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	types
deletion	4-grams	.14	.6	.09	.67	.67	.67	.67	.67	.67	.67
	5-grams	.21	.61	.13	.67	.67	.67	.67	.67	.67	.67
	6-grams	.22	.67	.18	.78	.78	.78	.78	.61	.78	.78
	7-grams	.14	.5	.09	.50	.5	.5	.5	.5	.5	.5
Insertion	4-grams	.18	.58	.12	.67	.58	.67	.61	.58	.67	.67
	5-grams	.34	.74	.24	.83	.72	.83	.75	.72	.83	.83
	6-grams	.3	.89	.21	1	1	1	1	.85	1	1
	7-grams	.39	.93	.3	1	1	1	1	.87	1	1
Deletion + insertion	4-grams	.17	.88	.1	1	.92	1	.94	.92	1	1
	5-grams	.5	.66	.44	1	.69	.83	.71	.65	.83	.83
	6-grams	.53	.84	.5	1	1	1	1	.76	1	1
	7-grams	.46	1	.33	1	1	1	1	1	1	1
Synonym	4-grams	.24	.65	.17	.83	.75	.83	.78	.75	.83	.83
	5-grams	.43	.99	.25	1	1	1	1	1	1	1
	6-grams	.28	.81	.19	.83	.83	.83	.83	.75	.83	.83
	7-grams	.34	.83	.23	.83	.83	.83	.83	.83	.83	.83
Shuffle	4-grams	.19	.33	.14	.33	.33	.33	.33	.33	.33	.33
	5-grams	.13	.5	.09	.67	.42	.67	.47	.42	.67	.67
	6-grams	.03	.33	.12	.33	.33	.33	.33	.33	.33	.33
	7-grams	.08	.42	.05	.50	.5	.5	.5	.42	.5	5

From table 6.18 and 6.19, we could see that the overall alignment rates using different n-gram lengths in all levels of measures are much lower than those using token seeds (cf. table 6.14). The highest Plagdet score in SPC which is achieved by 5- & 7-grams at 0,25 is still much lower than the lowest Plagdet score of token seeds under method TK4 which is on 0.59. Even, the highest scores of all measures could not exceed those of Alvi's in SPC. In contrast, the highest scores of each measure in each obfuscation type of Artificial Plagiarism Cases are relatively higher than those of Alvi's. The exception falls in the case of *deletion* under the character-based measures, whose highest *Plagdet* score achieves 0.22 only while Alvi's plagdet is on 0.34. We assumed that our proposed method by using a significant local word and paragraph-based text alignment does not fit character n-grams as seed units. The rationale is that many n-grams which are selected as seeds mostly come from the same tokens or words. If their offsets are concatenated, they form only several short passages which at the post-processing stage will be discarded (see section 4.4).

Being measured per test case, the text alignmnet scores resulted by n-gram seeds show clearly the distinct characteristics of three levels of measurement (character, case, and document levels). For example, testdoc022 on table C.6 shows that PlagiarIna fails to align source-suspicious passages on the level of characters and cases (or passages), but it detects correctly on the document level as shown by its zero scores for all measures in character and case levels but not on document level. The detection case of testdoc012 with 4-gram seeds emphasizes this characteristic where it has zero scores only for case-based detection. This is possible, due to different strategies applied for each level. In a case level, a detected pair of source-suspicious passages will be considered as true positive if both source and suspicious passages are true references of the annotated source and suspicious passages. On the contrary, the character-based measures apply Boolean operator *or* in counting a true positive detection. Thus, it is only the number of characters either in the source or suspicious passage which are considered as true positive detection, if it is only one of them which turns out to be the true detection. Another interesting point shown by per-test-case measurement is that in some test documents, n-gram seeds detect heavier level of obfuscation better than the lighter one as Alvi's algorithm did. For example in Simulated Plagiarism Cases for test documents 010, 011, 027, and 028, the heavy paraphrase and summary cases could be detected but it fails in detecting the light paraphrase cases.

Table 6.19 summarizes the scores of obfuscation type detection in Simulated Plagiarism Cases using token and ngrams seeds, and its comparison to Alvi's detection as well. From this table we could see that:

1. The n-gram detection rates on different types of obfuscation are also lower than token detection rates.

2. PlagiarIna has no problem in detecting *copy, light paraphrase, medium paraphrase*, and *shake*. Their high detection rates which are achieved by method TK3 on the rate of 0.90, 0.91, 0.81, and 0.76 prove this. In contrast, detecting heavy paraphrase and summary cases is still a challenge for PlagiarIna, as its detection rates achieve 0.53 for heavy paraphrase and 0.37 for summary.

Table 6.20: The detection rates of obfuscation types in Text Alignmnet for SPC. The highest score in each obfuscation type is printed in bold. The abbreviations used for obfuscation type are as follows: *paraL* stands for light paraphrase, *paraM* refers to medium paraphrase, while *paraH* stands for heavy paraphrase.

Methods	Copy	paraL	paraM	paraH	shake	summary
Alvi	.68	.55	.42	.42	.64	**.45**
TK1	.85	.80	.63	**.53**	.74	.37
TK2	.81	.88	.48	.48	.70	.37
TK3	**.90**	**.91**	**.81**	.44	.74	.37
TK4	.81	.78	.56	.49	**.76**	.37
4-grams	.20	.26	.29	.14	.25	0
5-grams	.37	.30	.26	.32	.43	.16
6-grams	.25	.23	.25	.32	.33	.25
7-grams	.35	.24	.20	.06	.48	.16

3. PlagiarIna's detection scores for summarized passages are relatively stable and unaffected by any method when it uses token seeds. The score keeps on the rate of 0.37 for all of these methods.

4. PlagiarIna outperforms Alvi's algorithm on detecting cases of copy, three levels of paraphrase, and shake, but Alvi's algorithm outperforms PlagiarIna in detecting the summarized passages as proven by its score which is 0.08 higher that PlagiarIna's score produced by token seeds.

6.4 Experiments on PlagiarIna's performance

The goal of this experiment is to mimic a real use case, in which the source documents of a suspicious one are hidden, unknown and thus will not be provided. For this reason, we plugged off the *intervention* function whose main task is to add the unretrieved source documents to the list of source candidates. Given only a suspicious document as input, the system retrieves source candidates, then aligns the suspicious document with each of the source candidates outputted by the retrieval subtask. This distinguishes this scenario from the former one, the text alignment subtask. Although we set up the same parameters values, it could be predicted that both PlagiarIna and Alvi's algorithm in this scenario will produce lower results than the experiments on text alignment. This is quite logical, as the source retrieval subtask has not achieved its maximal results. For this reason, we did not run this experiment on all combinations of heuristic retrieval and text alignment methods. Insteads, we selected methods which contributed to high recall on the former experiment scenarios.

In this scenario, we also performed experiments on documents which are supposed to contain no-plagiarized passages. Instead of creating test documents with no-plagiarism cases, we selected 10 articles which have been reviewed in section 3.4 and which discuss the plagiarism detection for Indonesian texts. Four of these articles are written in English, and we translated them into Indonesian using Google translate tool. The goal of this experiment is to evaluate system performance on detecting no-plagiarism cases.

6.4.1 Results and Discussion

Table 6.21 presents the detection results of PlagiarIna and Alvi's algorithm under a real setting scenario. In simulated plagiarism cases, we present two results from two different method combinations of source retrieval and text alignment. The results of using TK1-TK1 for retrieval-alignment methods are presented on the second row of table 6.21. This method represents PlagiarIna's methods whose scores are relatively higher than Alvi's algorithm, while PW22-TK1 combination[25] is selected to represent methods whose scores are relatively lower than Alvi's algorithm. Other methods show insignificantly higher and lower Plagdet, precision and recall rates than these two selected methods.

The only result from this experiment scenario which is feasibly comparable to the experiment results of source retrieval and text alignment is its recall rates on the document level. The comparison on the recall scores was performed only to those produced by the same methods. Assuming that PlagiarIna retrieves the same number of source documents as in the experiments on source retrieval subtask (see table 6.5), the possibilities for recall and precision rates resulted from this scenario are as follows:

1. If all retrieved source documents of all given test documents are successfully aligned and detected, the maximal recall rates on document level are as high as the recall rates of source retrieval subtask.
   This condition is fulfilled in the obfuscation types of *deletion + insertion* which shows exactly the same recall rate as the source retrieval: 0.83 (cf. table 6.6).

2. Most or only some retrieved source documents are aligned and detected successfuly. This leads on the decreased recall rates on the the document level as shown by almost all PlagiarIna's recall rates on table 6.21.
   In order to compare the recall rates of this experiment to those of Text Alignmnet , let us take an example of recall rate from simulated plagiarism cases using TK1-TK1 on table 6.21. Its recall rate is 0.53, while the recall rate of source retrieval under TK1 is 0.67. If 0.67 equals to 100% (given only all retrieved source documents), then 0.53 from 0.67 equals to 79% or 0.79. Compared to the recall rate of text alignment using TK1 which is 0.72 (cf. table 6.14), the recall rate of this setting scenario is actually 9.7% higher, though its nominal rate seems to be lower. Based on this calculation, the decrease or increase of recall rates in this scenario compared to recall rates of oracle experiment ranges from 6-15% for simulated plagiarism cases, while the decrease and increase of recall rates for artificial plagiarism cases ranges on 0-17%.

[25]it could be found on the $3^{rd})$ row of table 6.21

3. The precision rate on the document level in this scenario is independent from the precision rates of either source retrieval or text alignment under the same methods.
4. The precision and recall rates on case and character levels are indirectly influenced by the recall rates on the document level. This explains why almost recall, precision, plagdet scores in this scenario are averagely lower than text alignment scores.

In comparison to Alvi's algorithm, the experiment on the whole system scenario for simulated plagiarism cases shows a consistency, in which PlagiarIna outperforms Alvi's algorithm on recall rates in all measures, and also precision rate on document level. In contrast, Alvi's algorithm has higher precision rates on character and case levels. This causes Plagiarina's Plagdet score to be insignificantly higher than Alvi's. In artificial obfuscations, the precision and recall rates of PlagiarIna on the levels of document and passage are competitive to Alvi's. On the level of characters, TK1-TK1 method outperfoms Alvi's as its Plagdet, precision and recall rates of all obfuscation types are much higher. The consistent result is shown also by all measure scores on the obfusctaion type of *shuffle*, in which Alvi's algorithm fails to align any shuffled documents. In contrast, this type of obfuscation presents problem to Plagiarina only on the character-based detection. In simulated plagiarism cases, the higest recall rate of plagiarIna is 0.53 while Alvi's algorithm is 0.50; the highest precision rates of Plagiarina is 0.79, and 0.72 for Alvi's. In Artificial Plagiarism Cases, the highest recall and precision rates of Plagiarina and Alvi's algorithm are on the rate of 0.83. These rates show that under the real setting scenario, the detection rates of both systems (PlagiarIna and Alvi's algorithm) are still quite low for simulated plagiarism cases, and satisfactory for the artificial one.

Detection on obfuscation types. Table 6.23 presents the results of obfuscation type detection of both PlagiarIna and Alvi's algorithm for simulated plagiarism cases. Under the real setting scenario, both systems produce lower detection rates on all obfuscation types: copy, paraphrases, shake and summary. For an example, under TK1 PlagiarIna's detection rates for copy and light paraphrase reach 0.85 and 0.80 in the text-alignment experiment, but it reaches 0.56 and 0.68 under the real setting scenario. Alvi's detection rate on copy reaches 0.68 under text alignmnet scenario and reaches 0.35 only in this scenario. The experiment results of this scenario show the same tendency as the detection scores on obfuscation type of oracle scenario presented in table 6.20 where PlagiarIna's detection rates on copy, 3 levels of paraphrase and shake outperforms Alvi's, but its detection rate on summary which is on 0.30 is lower than Alvi's. In this scenario, Alvi's detection rate on summary remains unchanged and keeps on the rate of 0.45 as its rate in oracle scenario.

Table 6.21: The detection results of two systems on a real setting scenario, in which source documents of a suspicious one are not given.

Systems	Obfuscat.	Character-based				Case-based			Doc.-based		Time
		Pldet	Prec.	Rec.	Gran	Prec.	Rec	F1	Prec	Rec	
Alvi	SPC	.43	.70	.34	.80	.65	.38	.48	.72	.50	.13
PlagiarIna		.44	.65	.40	.83	.57	.44	.49	.79	.53	28.63
		.36	.62	.29	.83	.52	.31	.36	.77	.45	38
Alvi	Deletion	.23	.50	.16	.5	.50	.50	.50	.50	.50	.10
	Insertion	.34	.62	.24	.67	.58	.67	.61	.58	.67	.10
	Del+Ins	.5	.83	.39	.83	.83	.83	.83	.83	.83	.10
	Synonm	.37	.67	.28	.67	.67	.67	.67	.67	.67	.10
	Shuffle	0	0	0	0	0	0	0	0	0	.10
PlagiarIna	Deletion	.34	.50	.28	.50	.50	.50	.50	.50	.50	9.17
	Insertion	.47	.61	.40	.67	.58	.67	.61	.58	.67	23.83
	Del+Ins	.76	.82	.71	.83	.75	.83	.78	.75	.83	13
	Synonm	.47	.66	.40	.67	.67	.67	.67	.67	.67	6.33
	Shuffle	.25	.58	.18	.67	.58	.67	.61	.58	.67	5.6

Table 6.22: Detection rates of PlagiarIna and Alvi's algorithm on no-plagiarism cases.

	PlagiarIna								Alvi
	TK1	TK2	TK3	TK4	4-gr	5-gr	6-gr	7-gr	
Rates	.80	1	.90	1	.80	.90	.90	1	.90

Table 6.23: Case recognition rates of PlagiarIna and Alvi's algorithm for simulated plagiarism in a real setting scenario. The abbrebiations on the column headings stand for the follwowing: cp=copy, pL=light paraphrase, pM=medium paraphrase, pH=heavy paraphrase, sh= shake, sm=summary.

Systems	cp	pL	pM	pH	sh	sm
Alvi	.35	.57	.32	.42	.45	.45
PlagiarIna	.56	.68	.34	.46	.48	.30

Detection of no-plagiarism case. Table 6.22 presents the detection results of no-plagiarism cases for both PlagiarIna and Alvi's algorithm. In this experiment, we evaluated the system performance on its recognition on no-plagiarism test case only. The rationale is that given zero references for source passages and documents, recall, precision and other measures on character, case, and document levels would result in zero rates as they were computed. For Plagiarina, we run the experiments on all 8 Text Alignment methods and used TK1 for source retrieval. The results displayed in table 6.22 show that the detection rates of both Plagiarina and Alvi's algoritm on no-plagiarism documents are very high. The use of stemming in methods TK3 and TK4 leads to the optimal case detection rate: 1.00, while the use of Tala-stopword leads to an increased detection rate of 0.90 compared to the use of frequency stopwords which results in the detection rate of 0.80. For the n-gram seeds, the granularity of n-grams plays a role in increasing the detection accuracy, as 7-gram achieves the optimal detection rate, 1; and 4-gram achieves the lowest detection rate, 0.80.

The TK1-method which only achieves 0.80 case-detection rate, alignes a topically-related passage pair for each testdoc136 and testdoc140. The similar passage pair detected for testdoc136 deals with a topic on term weighting using tf-idf, while the similar passage pair detected for testdoc140 deals with bibliography referring to the occurences of 3 consecutive references which share 2 similar bibliography entries. The TK3-method detects the similar passage pair on tf-idf term weighing only. Other PlagiarIna's methods having a similar detection rate to TK1 and TK3 detect the same passages for the same test documents. Though human assessor will not judge that the aligned passage pairs detected by TK1 and TK3 as source-plagiarized passage pairs, it will be really difficult for a system to filter such passages, since they share so many common terminologies or keywords, specifically for a detected passage pair of testdoc136.

Though Alvi's detection on no-plagiarism case achieves the same rate as TK3, 5- to 6-gram seeds, it aligns a different passage pair. Alvi's algorithm aligned a passage consisting of author names, authors email addresses and some early parts of abstact of the testdoc140 with an article title, author names, author email addresses and also earlier part of abstract of a source document. This passage pair shares only the similar information on author names and university affiliatian whose abbreviation appears to be a term in English. A human assessor will judge that this passage pair shares no common content.

6.5 Conclusion

In this chapter, we have described our test set and the implementation of three experiment scenarios for testing and evaluating the proposed methods on detecting plagiarism for Indonesian texts. The first and second scenarios evaluate each source retrieval and text alignment subtask independently to see their maximal performances. The third scenario evaluates the performance of the whole system under a real use case setting.

On the first experiment scenario, we have explored several methods of source retrieval which use different kinds of features for document representation, feature lengths, stopping, and stemming. We have experimented three different document features namely phrasewords, token and character n-grams. we used two types of stopwords: the frequency-based stopwords and Tala-stopword which is a semantic-based stopword list. For phraseword features, we have experimented two types of phrasewords: PW1 uses a string length and its first character to represent a token, and PW2 which simply uses the first two characters of a stemmed token to represent it. The experiment results show that PW2 outperforms PW1 on its precision and F1 scores, while PW1 shows higher recall rates than PW2 almost in all methods for artificial and simulated plagiarism cases. The experiment on phraseword length shows that phraseword 2-grams outperform phraseword 3 to 4-grams in recall rates for simulated plagiarism cases, but phrasewords 4-grams outperform other phraseword lengths in all three measures for artificial plagiarism cases, except for shuffled obfuscation. In average, the use of stemming for phraseword features leads to increased precision and F-1. Among Phraseword methods, PW2 has potential to be robust features for retrieving source documents as their precision and recall rates are almost in balance. However, further research and experiments need to be conducted to make their recall rates be more competitive.

In experimenting token as document features, we observed the effects of using stopping, stemming, different window lengths for query formulation, and different numbers of queries per window. In average, the use of either stemming or Tala-stopwords increases precision rates only in simulated plagiarism cases, but its use in artificial plagiarism cases leads to increased recall and precision. The highest recall rate in simulated plagiarism cases is produced by the use of frequency-stopword per se. The window length for query selection correlates highly with the test or suspicious document length, but the number of queries per window does not always correlate to higher recall rates. There are factors such as the length of plagiarized passages, plagiarism types , and query selection strategies which influence the recall rates.

For n-gram features, we applied only frequency stopword removal on the preprocessing stage, then discarded the most common n-grams by using character n-stopgram lists. We run our experiments on these clean character n-grams in fine granularity with n ranges from 4 to 7 and observed their performances based on their granularity. In simulated plagiarism cases, the highest recall rate is produced by 5-grams, while 6- and 7-grams achieve the highest recall rates interchangeably on different obfuscation types of artificial plagiarism cases. Compared to phrasewords and token, n-grams features achieve lower scores of recall and precision in artificially obfuscated documents. However, their recall rates in simulated plagiarism cases are relatively higher than those produced by token, even its highest recall rate which is on 0,79 becomes the most high recall rate compared to the highest recall rates produced by phrasewords (0.73) or by token (0.67). The drawback of using character n-gram features compared to phrasewords and token lies on its processing time which needs much longer time than Phrasewords and token. Character n-grams could be potentially robust document features for source retrieval, if further filtering and selection technique are added to its method on query formulation.

On the second experiment scenario, an 'intervention' function was plugged into the source retrieval to add the unretrieved source document IDs to its outputs. Given source documents among retrieved candidate documents and a suspicious document, text alignment subtask performs its analysis to output pairs of source-plagiarized passages out of these documents. Its performance is evaluated with precision and recall measures on 4 levels: character, passage, document, and cases or obfuscation types. Plagdet score which combines precision, recall, F1, and granularity becomes a feasible score comparable to other methods or systems. In simulated plagiarism cases, the highest Plagdet score, 0.63 is produced by methods which apply stopword removal (TK1) and its combination with stemming (TK2). The use of Tala-stopword and stemming leads to the lowest Plagdet score (0.59) among methods using token as seeds. In artificial plagiarism cases, TK1 produces the highes Plagdet scores in all obfuscation types. The highest Plagdet score, 0.91 is achieved by TK1 for onfuscation type of deletion plus insertion. Compared to token seeds, n-grams produce much lower Plagdet scores. Their highest Plagdet score in simulated plagiarism cases, 0.25, is achieved by both 5- and 7-grams. In artificial plagiarism cases, the highest Plagdet score of n-grams, 0.53, is achieved by 6-grams for the case of deletion plus insertion.

In this scenario, we compared PlagiarIna's performance to Alvi's algorithm for a reason that it shares some commonalities in its methods to majority researches on plagiarism detection for Indonesian texts. Being implemented and run on the same environment as PlagiarIna, the highest Plagdet score produced by Alvi's algorithm in simulated plagiarism cases (SPC) reaches 0.53, which is in fact still lower than the lowest Plagdet score of PlagiarIna in SPC produced by TK4, 0.59. In artificial plagiarism cases, PlagiarIna outperforms Alvi's Plagdet scores in all obfuscation types, especially in shuffle. PlagiarIna's alignment performance using n-grams seeds produces lower Plagdet scores than Alvi's Plagdet scores both in artificial and simulated plagiarism cases. PlagiarIna's detection rates for copy (0.90), 3-levels of paraphrases (0.91, 0.81, 0.43) and shake (0.70) outperform Alvi's detection rates for copy (0.68), paraphrases (0.55, 0.42, 0.42), and shake (0.64) as well. In contrast, Alvi's detection rate on summary which is on the rate of 0.45 is higher than PlagiarIna's rate which reaches 0.37 only.

In the third scenario, PlagiarIna and Alvi's algorithm were experimented on a real use case of a Plagiarism detection system. For this reason, the intervention function was unplugged from the system. Being evaluated with the same measures, PlagiarIna outperforms Alvi's algorithm in all measures of artificial plagiarism cases. In simulated plagiarism cases, PlagiarIna's Plagdet score under TK1 method is insignificantly higher than Alvi's algorithm. In average, all measure scores produced by both systems are much lower in this scenario compared to their scores on text alignment experiments. The case detection results in this scenario show the same tendency as those of Text Alignment, where PlagiarIna's detection rates on Copy, 3 levels of Paraphrases and Shake outperforms Alvi's, but its detection rate on Summary is lower than Alvi's. For no-plagiarism case, PlagiarIna's detection rates range from 0.80 to 1, and Alvi's detection rate reaches 0.90. The use of stemming which is combined either with frequency stopword (TK2) or Tala-stopword (TK4) leads to the optimal rate 1.00 for no-plagiarism case. Besides TK2 and TK4, character 7-gram achieves the optimal rate, 1.00, for no-plagiarism case detection.

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, which is on the rate of 0.67 (cf. table 6.13). Its precision rates tested in our corpus 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. The insignificant difference between Alvi's scores tested on PAN corpus and ours is particularly noticeable on its highest Plagdet score for artificial plagiarism cases [26] which reaches 0.52 in our corpus, and 0.50 in PAN corpus. Alvi's detection rate on no-plagiarism case reaches 0.90, which is a very high score. However, it is less high than its score tested in PAN corpus which is able to reach the optimal rate, 1.0. Based on these rates, it could be boldly concluded that our evaluation corpus has reached an international standard level. Thus, it fulfills the fourth objective of this study (see section 1.3) which is to provide a standard evaluation corpus for Indonesian plagiarism detection systems.

The sharp decrease on Plagdet scores under a real use case shows that the source retrieval performance influences much the end results of a plagiarism detection system, and so does PlagiarIna. For source retrieval task, this research emphasizes to work on document representations and query selection through a segmentation process. For future works, methods on query selection and formulation need to be deeply explored for the sake of increasing the recall rate. Kiabod's local, global term weighting and significant word pruning, which were implemented for text alignment subtask in this research, are worth experimented for query selection. The segmentation methods designed for query formulation should consider the suspicious document structure which is segmented through chapters, headings, and subheadings. The last thought on efforts to increase recall rate of a source retrieval subtask could consider using the number of shared references between source-suspicious documents for filtering process.

The averagely high Plagdet scores of PlagiarIna in Artificial Plagiarism Cases 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 only on the level of lexemes or words.

[26] In PAN'14, the obfusction types under artificial plagiarism cases in our corpus are known as random obfuscation (see table 6.13)

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 case recognition rates on three level of paraphrases, shake, and copy which are higher than Alvi's scores prove that our paragraph-based alignment method 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 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. For future work, detection techniques of passage boundary need to be based on sentence boundary.

Chapter 7

Summary and Future Works

Being intended to conclude our study, this chapter is organized into two main sections. Section 7.1 outlines the summary of our study and its contributions, while section 7.2 presents a preview on how aforementioned contributions lead to further research directions in External Plagiarism Detection systems for Indonesian texts.

7.1 Summary and Research Contributions

Plagiarism, which is an act of taking someone else's work or ideas and passing them off as one's own, is strongly associated with academic plagiarism in the last few years. The problem setting of academic plagiarism leads to our research questions which deal with problems of retrieving sources of a plagiarized document and detection or alignment of source-plagiarized passages. To answer these questions, five objectives were set.

The first objective of this study is "*to conduct a thorough literary research on state-of-the-art algorithms on plagiarism detection systems in general and on the available plagiarism detection systems for Indonesian texts*".

A comprehensive review on models and state-of-the-art algorithms on plagiarism detection has been performed and presented in chapter 2. This review found out that most state-of-the-art systems still work on lexical levels, and are capable of detecting copied, shaked, or lighly obfuscated plagiarism cases, but still have difficulties in detecting heavily obfuscated plagiarism cases. Some state-of-the-art systems tried to capture and detect passage similarity on the structural and semantic levels without applying any semantic analysis such as the use of Stopword N-grams (SWNG) and citation-based plagiarism detecion (CbPD) model. Somehow, one drawback of devising stopword n-grams is that it is a language-dependent model. In a language whose most frequent stopwords have no roles in defining its well-form syntactic structure, stopword n-gram-based plagiarism detection model becomes impracticable. Meanwhile, citation-based plagiarism detection which is claimed to be capable of detecting heavily obfuscated plagiarism cases will fail easily, if the sources of copied texts are not listed in the references and no citations referring to the copied sources are given.

The literary study on external plagiarism detection systems for Indonesian texts, which is presented in chapter 3, reveals that former researches mostly deal with detecting duplicate and near-duplicate cases, applying exact matching strategy, measuring similarity on the document level, and therefore are incapable of referring to the exact location of the similar passage pairs. Only a handful of systems truly worked on partial duplicate or plagiarism detection. Regretably, some of them have not developed any strategy on retrieving source documents, or just stop at matching process on the alignment phase. In term of evaluation corpus, there is no public and standard corpora available to evaluate External Plagiarism Detection for Indonesian texts. These researches use either the available corpora containing texts in western European languages or develop their own corpus. Presumably, the plagiarism cases for test documents were also developed by the researchers themselves, as there is hardly any explanation on who wrote test documents. Only one research [135] acquired its both source and test documents from student coursework papers.

In addition to that literary research, this study has investigated the history of plagiarism practices, plagiarism scenario, and plagiarism taxonomy by exploring six (6) corpora hosted by Brigham Young university. So far, studies on ancient literature prove that the practice and concept of plagiarism have existed in Latin literature, which is a long time before the term *plagiarism* itself came into being. In that era, different terms were used to address *plagiarism* practices. This is to repudiate some references that simply blame the Internet and vast advancement of Information technology as the cause of plagiarism practices among students.

To address the weaknesses of former plagiarism detection systems for Indonesian texts, a workflow system which enables the execution of various methods of plagiarism detection phases has been designed. The systems comprises a three-step process: source retrieval, text alignment, and post-processing. The various methods in each step or subtask is realized into a plug and play system which enables users to switch to different methods without switching to or initializing a different program application.

This system design becomes a fulfillment of our second objective of this study which is "*to design a framework for execution of various detection methods in a system workflow*".

The third objective of the study is " *to find and implement a competitive state-of-the-art algorithm on plagiarism detecion for Indonesian texts* ".

In order to achieve this objective, we proposed to apply a top-down approach, three different document features in a source retrieval subtask, and a two-step text alignment method. The realization of top-down approach is traceable firstly on source retrieval subtask which measures similarity of source-suspicious document pairs globally, and outputs a limited number of candidate documents. Secondly, the similarity computation on the level of paragraph, which is a smaller structure unit within a document, was applied on text alignment phase. Only pairs of paragraphs from candidate documents and a suspicious document having similarity values above the defined threshold were aligned and post-processed. Chapter 5 presents the design and implementation of our proposed methods which are summarized on the following paragraphs.

In boosting the performance of source retrieval, we did not rely on one strategy only, insteads we based our retrieval methods on three different document features. We introduced the use of Phraseword, which is a metaterm for word n-grams. Its use as a document feature is aimed to overcome the weaknesses of using the exact consecutive occurrences of token or string as queries and document profiles. We applied two different strategies in forming phrasewords which result in two types of Phrasewords. Two other features are token or word unigram and character n-grams. The combination of these document features with the pre-processing techniques results in 11 methods: 6 methods for phrasewords, 4 methods for word unigram, and 1 method for n-grams. Basically, we applied a similar segment-based query formulation techniques for all of these methods, but varied some parameter values on the segment length and the number of queries per segment. The filtering techniques applied for selecting candidate documents are based on the number of shared queries, the defined minimum cosine value, and the top 35 ranked retrieved documents.

Instead of using a chunk of consecutive strings, we borrowed a term weighting method from text summarization, Kiabod's local word weighting and pruning [81], to weight and select seeds. The selected seeds are then devised to have a twofold function: as discriminators for extracting source-suspicious paragraph pairs, and for matching seeds within these extracted paragraph pairs. In computing similarity between paragraph pairs, the binary similarity metric Jaccard coefficient was applied to select paragraphs containing text reuses with obfuscation types of Copy and Shake, while Dice coefficient was used to give high scores on paragraph pairs containing obfuscation types of paraphrase and summary. The seed matching, merging, and extension are based on two-step rules. The first rules merge seeds within each of selected paragraph pairs to form a short passage pair, while the rules on the second step extend the passage pair boundaries by merging them to another passage pair from different paragraphs only if their distance is less than the defined gap values. Using Boolean operator OR, the aligned passage pairs whose source passage has length less than 125 characters or suspicious passage has length less than 150 characters will be discarded on the post-processing stage.

The fourth objective which is "*creating a standard evaluation corpus for testing Indonesian plagiarism detection systems*" is fulfilled and described in chapter 5.

The building process of evaluation corpus in this task combines the strategies applied by PAN shared tasks [128] with concepts used by HTW research center to create test cases [185]. The evaluation corpus comprises 128 test documents generated artificially through random obfuscation, and 105 test documents were created through simulation by human writers. However, we selected 70 documents to be run on the experiments which compared Alvi's algorithm and our prototype system, PlagiarIna. Being run at our test document corpus, Alvi's highest scores of recall, precision, Plagdet, and detection rate on no-plagiarism cases correspond to its scores when it was tested on PAN'14 corpus. This proves that our evaluation corpus has reached an international standard level and fullfill the fourth objective of this study.

The fifth objective of this study is "*to evaluate the performance of the proposed methods and to compare it to one of state-of-the-art algorithms*".

To realized this objectives, we developed three scenarios of evaluation which was described in chapter 6. **The first scenario** evaluates the performance of the source retrieval subtask under 11 methods mentioned earlier. The results show that all methods from three different features: phrasewords, token, and character n-grams, have higher rates on all measures for artificial plagiarism cases than simulated plagiarism cases. Methods using phraseword features, specifically PW2 4-grams, and PW1 2-grams produces higher recall rates than methods using token features. Some methods under phrasewords and token are able to achieve the optimum recall rates, 1. In average, source retrieval using character n-grams produces lower recall, precision, and F1 scores in artificial plagiarism cases. On the contrary, the recall rates produced by character n-grams in simulated plagiarism cases are more stable and higher than recall rates produced by phrasewords or token.

On the second scenario, we run an oracle experiment for text alignment performance of both systems, i.e. all source documents of a suspicious document were provided among other retrieved source candidates. The evaluation measures, precision and recall, were carried out in four levels namely, character, cases (or passage), document levels, and obfuscation types. The computation of granularity measurement is based on the character and passage levels. Plagdet which combines these three measures into a single score is a measure for overall performance of a plagiarism detection system. The experiment results show that the Plagdet scores of PlagiarIna resulted from methods using token seeds are higher than Alvi's Plagdet scores both in artificial and simulated plagiarism cases. In contrast, Plagdet scores produced by n-gram seeds are lower than Alvi's Plagdet scores.

On the third scenario, both Alvi's algorithm and PlagiarIna were evaluated on a real use case which enables text alignment module perform its analysis only to the retrieved source candidates. Some PlagiarIna's combination methods for source retrieval-text alignment, such as TK1-TK1, TK1-TK3, produced Plagdet scores that are competitive to Alvi's score. The recall rates produced by these methods are much greater than Alvi's algorithm. On document levels, our methods outperform Alvi's algorithm both in recall and precision rates. However, some combination methods, exemplified by PW22-TK1 produced Plagdet scores which is lower than Alvi's. In the obfuscation type recognition, PlagiarIna outperforms Alvi's algorithm in detecting obfuscation types of Copy, Shake, and three levels of Paraphrase. In contrast, Alvi's recognition rate on obfuscation type of Summary is higher than PlagiarIna's. To conclude, the higher Plagdet scores produced by some of PlagiarIna's methods than Alvi's show that this study has fulfilled its objectives, specifically the objective numbered 4: implementing a competitive state-of-the-art algorithm on plagiarism detection for Indonesian texts.

To recapitulate, the contributions of this study are as follows:

1. A compilation of theoritical background for Plagiarism which takes form as brief history of plagiarism conduct, plagiarism scenario, and plagiarism taxonomy.

2. A compilation of researches on external plagiarism detection systems which are presented in Chapter 2.

3. A compilation of researches on plagiarism detection conducted by Indonesians. This compilation would be very beneficial for anyone performing researches on this field for Indonesian texts in future. This compilation is presented in Chapter 3.

4. An external plagiarism detection prototype which is competitive to state-of-the-art algorithm. The implementation and evaluation of this prototype were presented in Chapters 4 and 6. This contribution could be subdivided into the following:

   a) A source retrieval algorithm which uses three different document features, one of them is phraseword.

b) A paragraph-based text alignment algorithm which relies on two different strategies of paragraph weighting. One is based on binary vectors of paragraph, and another is based on seed vectors weighted through local-word score from text summarization field.

5. A standard evaluation corpus for assessing external plagiarism detection systems for Indonesian texts, which is described in Chapter 5. Our corpus has been also used in a research on multi-lingual morphological segmentation [39].

7.2 Future Work

The implementation, evaluation, and drawbacks of our proposed methods provide ideas for future research directions which will bring improvements and task completeness for future plagiarism detection systems, specifically for Indonesian plagiarism detection systems. This section on future work is organized into four (4) subsections. subsection 7.2.1 provides an outlook on the future research directions on source retrieval, while ideas for future work on text Alignment task are presented in subsection 7.2.2. Subsection 7.2.3 describes ideas on how to improve the evaluation corpus, while general research needs for improving and completing the task of plagiarism detection system for Indonesian texts are presented in subsection 7.2.4.

7.2.1 Source Retrieval Task

Due to the fact that the outputs of source retrieval determine the high or low detection rates of text alignment, we plan to improve the performance of source retrieval subtask in all its three main building blocks. For document representation, Vector Space Model is kept using, but we consider to apply different weighting schemes between source documents and suspicious document. Tf-idf is kept applying for weighting source document features, while Kiabod's scheme on calculating word score [81] looks promising to be applied on suspicious document features. We also plan to extend one more digit to phraseword, so that each token will be represented by 3 characters. The third character might represent the last character or the middle character of a token.

As a backbone method in source retrieval, the query formulation needs an improvement for its segmentation and query selection strategies. The segmentation strategy will be projected to be variable-based and dependent on the suspicious document length. For this reason, a document length checker needs to be devised on the pre-processing stage for a suspicious document. For a long suspicious document such as a thesis, the segmentation could be based on chapters, sections, and subsections. To avoid long segment which results in a possibility of no query selected for "a hidden plagiarized passage", a chapter could be considered as a separate document.

Considering that applying the same term weighting scheme for source and suspicious documents results in different weights for a term occuring in 2 different documents, we plan to apply word global and local scores proposed by Kiabod in [81] to compute term vectors of a suspicious document. Based on this term weight scheme, the computation of significant words in a segment is executed for selecting queries per window or segment. In measuring similarity of source-suspicious document pair, we stick on cosine similarity as a global similarity measure. The number of selected query per window will be also projected to be dependent on the document length. For a document longer than 30.000 words, the number of query per window will take a half of a medium or short document. The aim is to avoid having so many queries which result in low document similarity value for a source document whose content is heavily obfuscated.

For filtering the retrieved documents, we plan to incorporate Bibliographic Coupling algorithm which computes the occurrences of shared references between two documents [58]. The high bibliography similarity indicates subject similarity and since text reuses occur on texts having the same subject, Bibliographic Coupling would be beneficial if it is used as a filtering parameter. The idea is to combine the Bibliographic Coupling score with the cosine similarity values into a total similarity score which could be used to rerank the retrieved source candidates. Using a threshold defined from this total similarity score, the top-n ranked source candidates could be selected.

7.2.2 Text Alignment Task

The drawbacks of our proposed text alignment methods lie on firstly its unsatisfactory recognition rates on sumarized and heavily paraphrased passages, and secondly its definition of passage boundaries whose start and end may not correspond to start and end of a sentence. These drawbacks led us to the following research directions for future works:

1. The weighting scheme for significant word-based seeds per paragraph should combine their local and global weights as in query selection.

2. We plan to improve seed alignment by regarding the offsets of sentences in which the seeds occur. This is to address the drawback of the passage boundary detection.

3. We plan to incorporate sentence alignment which collects contextual evidence and exploits word similarity introduced in [172] to increase system's recognition on heavily paraphrased passages. Expectantly, this alignment method will increase the recognition rate on summarized passages too.

4. We plan to investigate further our filtering techniques to increase the precision rates on the level of paragraph and characters.

7.2.3 Evaluation Corpus

We plan to enlarge our corpus by increasing the number of both source and test documents. For the source document, we plan to digitize and include full bachelor theses from different subject areas archived in the library of Duta Wacana Christian University. The size of test documents will be varied, and the corpus is projected to include long documents such as master theses.

7.2.4 General Research Needs for Indonesian Plagiarism Detection system

Since the detection output of our prototype take forms of an XML-files, we plan to visualize the information contained in these files into an interactive web-based user-interface. In this interface, the visual report is projected to be accessible only by users having an access as teachers, lecturers, or examiners. Thus, the visual report could assist these users to arrive at a right conclusion on potential plagiarism. Students will be given access to submit their papers only.

Another area that needs to be done for future work is to work on a system which performs an online source retrieval. Besides, a specific algorithm for detecting cross-lingual plagiarism for Indonesian-English needs also to be constructed.

Appendix A
Stopword Lists

A.1 Frequency-based Stopword List

Table A.1: Frequency-based Stopword list

a	ada	adalah	adanya	agar
akan	akhir	antara	apa	apakah
atas	atau	awal	b	bagaimana
bagi	bagian	bahasa	bahkan	bahwa
baik	banyak	baru	bawah	beberapa
begitu	belum	bentuk	berada	berarti
berbagai	berbeda	berdasarkan	berikut	berupa
besar	biasa	biasanya	bidang	bisa
bukan	c	cara	com	contoh
cukup	d	daerah	dalam	dan
dapat	dari	dasar	data	demikian
dengan	di	digunakan	dilakukan	diri
disebut	dua	dunia	gambar	hal
hanya	hari	harus	hasil	hidup
hingga	ia	ilmu	indonesia	informasi
ingin	ini	itu	jadi	jelas
jenis	jika	juga	jumlah	kali
karena	kata	ke	kecil	kedua
kembali	kemudian	kepada	ketika	kita
kondisi	kurang	lagi	lain	lainnya
lalu	lama	langsung	lebih	luar
m	maka	mampu	mana	manusia
masa	masalah	masih	masing	masyarakat
maupun	melakukan	melalui	melihat	memang
memberikan	membuat	memiliki	mempunyai	mencapai
mendapatkan	mengalami	mengenai	menggunakan	menghasilkan
menjadi	menunjukkan	menurut	mereka	merupakan
misalnya	mudah	mulai	mungkin	nama
namun	nilai	of	oleh	orang
pada	paling	para	penelitian	penting

perlu	pernah	pertama	proses	pula
pun	s	saat	saja	salah
sama	sampai	sangat	satu	sebagai
sebelum	sebelumnya	sebuah	secara	sedang
sedangkan	sehingga	sejak	sekarang	sekitar
selain	selalu	selama	seluruh	semakin
semua	sendiri	seorang	seperti	sering
serta	sesuai	setelah	setiap	sistem
suatu	sudah	sumber	tahun	tak
tanpa	telah	tempat	tentang	terdapat
terhadap	terjadi	termasuk	tersebut	tertentu
terus	terutama	tetap	tetapi	tidak
tiga	tinggi	tingkat	tujuan	umum
untuk	utama	waktu	yaitu	yang
iii	vii	viii	xii	xiii
xiv	xvi	xvii	xviii	xix
xxi	xxii	xxiii		

A.2 Tala Stopword List

Table A.2: Tala Stopword List

ada	adalah	adanya	adapun
agak	agaknya	agar	akan
akankah	akhir	akhiri	akhirnya
aku	akulah	amat	amatlah
anda	andalah	antar	antara
antaranya	apa	apaan	apabila
apakah	palagi	apatah	artinya
asal	asalkan	atas	atau
ataukah	ataupun	awal	awalnya
bagai	bagaikan	bagaimana	bagaimanakah
bagaimanapun	bagi	bagian	bahkan
bahwa	bahwasanya	baik	bakal
bakalan	balik	banyak	bapak
baru	bawah	beberapa	begini
beginian	beginikah	beginilah	begitu
begitukah	begitulah	begitupun	bekerja
belakang	belakangan	belum	belumlah
benar	benarkah	benarlah	berada
berakhir	berakhirlah	berakhirnya	berapa
berapakah	berapalah	berapapun	berarti
berawal	berbagai	berdatangan	beri
berikan	berikut	berikutnya	berjumlah
berkali-kali	berkata	berkehendak	berkeinginan
berkenaan	berlainan	berlalu	berlangsung
berlebihan	bermacam	bermacam-macam	bermaksud
bermula	bersama	bersama-sama	bersiap
bersiap-siap	bertanya	bertanya-tanya	berturut
berturut-turut	bertutur	berujar	berupa
besar	betul	betulkah	biasa
biasanya	bila	bilakah	bisa
bisakah	boleh	bolehkah	bolehlah
buat	bukan	bukankah	bukanlah
bukannya	bulan	bung	cara
caranya	cukup	cukupkah	cukuplah
cuma	dahulu	dalam	dan
dapat	dari	daripada	datang
dekat	demi	demikian	demikianlah
dengan	depan	di	dia

diakhiri	diakhirinya	dialah	diantara
diantaranya	diberi	diberikan	diberikannya
dibuat	dibuatnya	didapat	didatangkan
digunakan	diibaratkan	diibaratkannya	diingat
diingatkan	diinginkan	dijawab	dijelaskan
dijelaskannya	dikarenakan	dikatakan	dikatakannya
dikerjakan	diketahui	diketahuinya	dikira
dilakukan	dilalui	dilihat	dimaksud
dimaksudkan	dimaksudkannya	dimaksudnya	diminta
dimintai	dimisalkan	dimulai	dimulailah
dimulainya	dimungkinkan	dini	dipastikan
diperbuat	diperbuatnya	dipergunakan	diperkirakan
diperlihatkan	diperlukan	diperlukannya	dipersoalkan
dipertanyakan	dipunyai	diri	dirinya
disampaikan	disebut	disebutkan	disebutkannya
disini	disinilah	ditambahkan	ditandaskan
ditanya	ditanyai	ditanyakan	ditegaskan
ditujukan	ditunjuk	ditunjuki	ditunjukkan
ditunjukkannya	ditunjuknya	dituturkan	dituturkannya
diucapkan	diucapkannya	diungkapkan	dong
dua	dulu	empat	enggak
enggaknya	entah	entahlah	guna
gunakan	hal	hampir	hanya
hanyalah	hari	harus	haruslah
harusnya	hendak	hendaklah	hendaknya
hingga	ia	ialah	ibarat
ibaratkan	ibaratnya	ibu	ikut
ingat	ingat-ingat	ingin	inginkah
inginkan	ini	inikah	inilah
itu	itukah	itulah	jadi
jadilah	jadinya	jangan	jangankan
janganlah	jauh	jawab	jawaban
jawabnya	jelas	jelaskan	jelaslah
jelasnya	jika	jikalau	juga
jumlah	jumlahnya	justru	kala
kalau	kalaulah	kalaupun	kalian
kami	kamilah	kamu	kamulah
kan	kapan	kapankah	kapanpun
karena	karenanya	kasus	kata
katakan	katakanlah	katanya	ke
keadaan	kebetulan	kecil	kedua

keduanya	keinginan	kelamaan	kelihatan
kelihatannya	kelima	keluar	kembali
kemudian	kemungkinan	kemungkinannya	kenapa
kepada	kepadanya	kesampaian	keseluruhan
keseluruhannya	keterlaluan	ketika	khususnya
kini	kinilah	kira	kira-kira
kiranya	kita	kitalah	kok
kurang	lagi	lagian	lah
lain	lainnya	lalu	lama
lamanya	lanjut	lanjutnya	lebih
lewat	lima	luar	macam
maka	makanya	makin	malah
malahan	mampu	mampukah	mana
manakala	manalagi	masa	masalah
masalahnya	masih	masihkah	masing
masing-masing	mau	maupun	melainkan
melakukan	melalui	melihat	melihatnya
memang	memastikan	memberi	memberikan
membuat	memerlukan	memihak	meminta
memintakan	memisalkan	memperbuat	mempergunakan
memperkirakan	memperlihatkan	mempersiapkan	mempersoalkan
mempertanyakan	mempunyai	memulai	memungkinkan
menaiki	menambahkan	menandaskan	menanti
menantikan	menanti-nanti	menanya	menanyai
menanyakan	mendapat	mendapatkan	mendatang
mendatangi	mendatangkan	menegaskan	mengakhiri
mengapa	mengatakan	mengatakannya	mengenai
mengerjakan	mengetahui	menggunakan	menghendaki
mengibaratkan	mengibaratkannya	mengingat	mengingatkan
menginginkan	mengira	mengucapkan	mengucapkannya
mengungkapkan	menjadi	menjawab	menjelaskan
menuju	menunjuk	nenunjuki	menunjukkan
menunjuknya	menurut	menuturkan	menyampaikan
menyangkut	menyatakan	menyebutkan	menyeluruh
menyiapkan	merasa	mereka	merekalah
merupakan	meski	merskipun	meyakini
meyakinkan	minta	mirip	misal
misalkan	misalnya	mula	mulai
mulailah	mulanya	mungkin	mungkinkah
nah	naik	namun	nanti
nantinya	nyaris	nyatanya	oleh

olehnya	pada	padahal	padanya
pak	paling	panjang	pantas
para	pasti	pastilah	penting
pentingnya	per	percuma	perlu
perlukah	perlunya	pernah	persoalan
pertama	pertama-tama	pertanyaan	pertanyakan
pihak	pihaknya	pukul	pula
pun	punya	rasa	rasanya
rata	rupanya	saat	saatnya
saja	sajalah	saling	sama
sama-sama	sambil	sampai	sampaikan
sampai-sampai	sana	sangat	sangatlah
satu	saya	sayalah	se
sebab	sebabnya	sebagai	sebagaimana
sebagainya	sebagian	sebaik	sebaik-baiknya
sebaiknya	sebaliknya	sebanyak	sebegini
sebegitu	sebelum	sebelumnya	sebenarnya
seberapa	sebesar	sebetulnya	sebisanya
sebuah	sebut	sebutlah	sebutnya
secara	secukupnya	sedang	sedangkan
sedemikian	sedikit	sedikitnya	seenaknya
segala	segalanya	segera	seharusnya
sehingga	seingat	sejak	sejauh
sejenak	sejumlah	sekadar	sekadarnya
sekali	sekalian	sekaligus	sekali-kali
sekalipun	sekarang	sekecil	seketika
sekitarnya	sekitar	sekitarnya	sekurang-kurangnya
sekurangnya	sela	selain	selaku
selalu	selama	selama-lamanya	selamanya
selanjutnya	seluruh	seluruhnya	semacam
semakin	semampu	semampunya	semasa
semasih	semata	semata-mata	semaunya
sementara	semisal	semisalnya	sempat
semua	semuanya	semula	sendiri
sendirian	sendirinya	seolah	seolah-olah
seorang	sepanjang	sepantasnya	sepantasnyalah
seperlunya	seperti	sepertinya	sepihak
sering	seringnya	serta	serupa
sesaat	sasama	sesampai	sesegera
sesekali	seseorang	sesuatu	sesuatunya
sesudah	sesudahnya	setelah	setempat

setengah	seterusnya	setiap	setiba
setibanya	setidaknya	setidak-tidaknya	setinggi
sesuai	sewaktu	siap	siapa
siapakah	siapapun	sini	sinilah
soal	soalnya	suatu	sudah
sudahkah	sudahlah	supaya	tadi
tadinya	tahu	tahun	tak
tambah	tambahnya	tampak	tampaknya
tandas	tandasnya	tanpa	tanya
tanyakan	tanyanya	tapi	tegas
tegasnya	telah	tempat	tengah
tentang	tentu	tentulah	tentunya
tepat	terakhir	terasa	terbanyak
terdahulu	terdapat	terdiri	terhadap
terhadapnya	teringat	teringat-ingat	terjadi
terjadilah	terjadinya	terkira	terlalu
terlebih	terlihat	termasuk	ternyata
tersampaikan	tersebut	tersebutlah	tertentu
tertuju	terus	terutama	tetap
tetapi	tiap	tiba	tiba-tiba
tidak	tidakkah	tidaklah	tiga
tinggi	toh	tunjuk	turut
tutur	tuturnya	ucap	ucapnya
ujar	ujarnya	umum	umumnya
ungkap	ungkapnya	untuk	usah
usai	waduh	wah	wahai
waktu	waktunya	walau	walaupun
wong	yaitu	yakin	yakni
yang			

A.3 Quadstopgrams

Table A.3: A list of Quadstopgrams

kan_	_men	_pen	an_p	_ber
an_m	nya_	_per	ang_	meng
n_pe	_mem	an_k	n_me	akan
a_me	asi_	ngan	gan_	an_b
an_d	an_s	_ter	aan_	peng
rang	an_t	ian_	tan_	ngka
han_	ikan	ran_	enga	_pem
a_pe	n_ke	ukan	lah_	n_di
an_a	i_pe	_pro	atan	ata_
a_di	i_me	_mas	ing_	anga
guna	emba	n_be	aran	bang
memb	jadi	nan_	enge	adi_
ahan	san_	_mel	nggu	oran
angk	uan_	ung_	a_be	ara_
kata	menj	naka	ada_	laku
enja	lan_	engg	n_ma	sia_
_mer	erja	man_	ng_m	amba
asa_	an_i	mban	n_te	baha
ting	ya_m	iste	a_ke	kat_
tas_	ama_	s_me	unak	_ada
lang	i_ke	_dip	_kon	_ind
i_di	paka	kuka	_ora	ngga
tahu	akuk	_sis	aya_	an_h
_dib	ggun	nnya	h_me	ingk
si_p	njad	embe	angg	aman
lai_	rupa	pemb	n_ba	meny
an_l	alan	anny	ah_m	pend
beri	itas	eran	erup	si_m
arak	_dil	uran	gamb	anya
pan_	_keb	unga	_kom	mela
isi_	menu	data	kasi	nila
g_me	enda	ng_p	s_pe	_kat
a_ma	_dit	a_te	asan	an_r
mili	n_ka	asar	t_me	rkan
_man	_dis	n_se	tkan	entu
_dia	r_me	ses_	logi	ter_
olog	inya	masi	alah	_hal
pera	_tah	l_me	engu	erda

erba	_kel	meru	ahas	n_pr
berb	bera	rah_	ari_	tik_
_bis	tang	k_me	raka	anan
at_p	enye	ksi_	nal_	isa_
i_be	_tan	masa	_dik	n_ko
at_m	liha	ilak	t_pe	ntuk
pat_	dila	dapa	apat	_ker
pene	mene	upak	mend	ng_b
orma	a_ba	ikas	ng_k	ah_p
n_in	tasi	_dig	emil	erik
ya_p	an_c	_kes	rika	yara
memi	liti	n_si	baik	tian
gkat	inga	anda	_tin	memp
ende	an_n	uhan	syar	enta
alam	ana_	lkan	dasa	_ban
an_g	a_se	atka	at_k	erta
indo	tif_	esia	hal_	n_pa
ingg	ilik	an_u	h_pe	i_ma
embu	peny	empe	arka	gkan
inte	_int	taka	enca	dang
erma	si_d	bisa	_tek	n_ta
ndon	an_j	enya	pert	alis
si_k	engh	al_m	dipe	ring
ihat	nesi	ones	aik_	hasi
arti	_pel	buat	dika	k_pe
mber	_ana	n_la	i_te	iki_
sika	elak	bent	ahun	uat_
bers	ng_t	_lan	enti	inta
ngun	ya_b	ment	ersi	ng_d
rasi	tera	komp	_har	tor_
akar	mena	hat_	_kep	pers
n_da	ng_s	hun_	h_di	berk
sala	erin	ning	menc	a_ka
an_f	emen	cara	s_di	l_di
bung	liki	ah_k	terj	_pan
_kem	r_pe	g_pe	jala	an_o
n_ha	hkan	lama	tung	nter
sa_m	ya_k	a_in	t_di	sion
tur_	ya_d	iper	nis_	anta
apan	asil	_mak	onal	si_b
kura	ah_d	disi	l_pe	erke

_bai	si_s	baga	_ket	unan
k_di	ensi	isti	mper	m_me
ulan	u_me	ah_b	ener	ilan
nsi_	iona	eras	sih_	at_b
tuk_	andi	_ben	nkan	i_ba
a_ta	al_p	ampu	n_an	an_e
erbe	t_ke	si_t	berd	angs
tika	kons	mema	ia_m	erha
i_se	ah_t	penu	bagi	rjad s
ati_	ding	ta_m	sar_	atak
is_m	asih	an_w	_sem	rat_
mati	beda	r_di	engi	eri_
i_ka	_dir	kkan	ya_t	ilih
knya	ra_m	meni	dite	_car
mula	nggi	teri	_ser	muka
g_di	ar_m	i_ko	ya_s	dike
al_d	_bag	ik_m	ntin	beru
ngar	agai	erse	_ala	stra
aktu	enan	_dim	g_be	ta_k
at_d	dah_	ta_p	a_ko	i_in
mbah	ah_s	ersa	diba	perk
unju	n_ja	a_pr	at_s	bah_
ar_p	_kec	etah	as_m	m_pe
ngat	eter	skan	eman	bert
_seb	k_be	at_t	s_be	al_b
ngha	h_be	i_pr	t_be	n_sa
tar_	mbua	_did	antu	ik_p
a_pa	meli	n_su	ng_a	erka
terb	pkan	h_ke	si_a	ami_
seba	urut	sung	a_si	as_p
n_bi	ai_p	perl	sa_d	n_ti
ndap	_dih	s_ke	enun	_mul
ai_m	ntar	a_ha	apka	k_ke
gai_	na_m	ya_a	ahka	is_p
ak_m	tnya	ri_m	al_k	ungk
hnya	u_pe	di_p	dise	l_be
ri_p	r_be	al_s	berh	i_ta
p_me	a_la	g_ke	a_ku	iri_

A.4 Pentastopgrams

Table A.4: A list of Pentastopgrams

_meng	an_pe	akan_	an_me	ngan_
_peng	kan_p	n_men	ikan_	a_men
ukan_	an_ke	n_pen	an_di	angan
rang_	kan_m	_memb	kan_k	atan_
an_be	jadi_	_menj	ahan_	nakan
orang	n_per	n_ber	menja	angka
menga	an_ma	a_ber	unaka	gunak
i_men	kukan	akuka	lakuk	_oran
aran_	_ada_	nggun	mbang	nnya_
enjad	an_te	njadi	_meny	annya
_pemb	nya_m	kan_s	erupa	kan_b
enggu	siste	emban	a_pen	gguna
_mela	stem_	menge	istem	an_ba
ng_me	ya_me	itas_	tkan_	a_mem
rkan_	_pend	_sist	i_pen	kan_t
nilai	ungan	_menu	ingka	gamba
ambar	n_mem	kan_d	_berb	_kata
aman_	_data	_meru	ologi	pakan
prose	an_ka	kata_	entuk	mengg
an_se	_pros	kan_a	upaka	kasi_
_pene	inya_	ikasi	penga	dapat
rupak	merup	ang_m	_mene	anan_
anya_	an_pr	ngkat	_dila	_mend
emili	_memi	ilaku	asan_	ah_me
_memp	an_ko	atkan	lkan_	milik
ngkan	i_per	uhan_	_bera	penge
arkan	_peny	_tahu	s_men	si_pe
bisa_	gkan_	mempe	_bisa	_dipe
a_per	tian_	an_in	_indo	dilak
n_pem	si_me	an_si	h_men	dasar
lihat	memil	tahun	n_pro	alan_
membe	i_ber	an_pa	erika	_pera
uran_	_hal_	alah_	tasi_	n_ter
_masa	iliki	_bers	_ting	bentu
_menc	nya_p	_berk	liki_	rikan
ember	ahun_	an_ta	_inte	hkan_
ng_pe	baik_	angga	_terj	memba
an_da	an_la	a_ter	_komp	diper

menye	n_mas	asi_p	at_pe	g_men
erang	at_me	asi_m	terja	apat_
t_men	_baik	mengh	emper	takan
_mena	ihat_	nya_b	ional	r_men
an_ha	_pers	cara_	ang_p	nkan_
ya_pe	ntuk_	erjad	rjadi	al_me
_mema	kuran	ah_pe	berik	l_men
buat_	mberi	nya_k	dang_	_pert
k_men	elaku	hasil	melak	nya_d
onal_	membu	ang_b	apan_	sa_me
gkat_	si_di	knya_	gan_p	_berd
asih_	_dite	kkan_	_dike	bagai
ya_di	kan_i	engar	_beru	_meni
mengu	berba	s_pen	sikan	ataka
tingg	ang_k	i_mem	_diba	ting_
ng_di	ng_be	gan_m	embua	engha
ah_di	aan_m	tan_p	skan_	menda
ang_t	masih	han_p	pkan_	inggi
_bert	aan_p	mbuat	_masi	endap
ndapa	at_ke	nya_t	apkan	_meli
menca	t_pen	mengi	nya_s	kan_h
ya_be	ian_m	al_di	n_mel	g_ber
i_ter	hnya_	tan_m	tnya_	ian_p
ahkan	h_ber	_dise	_terb	kan_r
ang_d	kan_l	t_ber	s_ber	ang_s
al_pe	u_men	ng_ke	a_mel	adi_p

Appendix B

Data Related to Corpus Building

Table B.1: List of URL addresses for source document corpus

Topics	URL addresses
History	http://pendidikan4sejarah.blogspot.de/2
Business, finance, & economy	http://jurnal-ekonomi.org/
Various topics	http://www.karyatulisilmiah.com
	http://artikel.staff.uns.ac.id
	http://wartawarga.gunadarma.ac.id
	http://carapedia.com
	http://www.kompas.com/
Geography	http://jurnal-geografi.blogspot.com/
	http://www.jurnalgea.com/index.php/volume-jurnal/file/
	http://nationalgeographic.co.id
Community Health	http://setengahbaya.info
	http://health.kompas.com/
Medicine	http://www.artikelkedokteran.com
	http://jurnalkedokteranindonesia.wordpress.com
Engineering	http://wiryanto.wordpress.com
Education, pedagogy	http://edukasi.kompas.com

```
wisatawan Indonesia dan asing berwisata, sebenarnya sudah disediakan
berbagai akomodasi yang sesuai dengan cara hidup wisatawan. Meski tempat
wisata di kota sudah dapat memberikan akomodasi yang sesuai, akan tetapi
berbeda dengan akomodasi di wisatawan yang ada di dearah pedesaan.
Wisatawan yang berwisata di Desa Paga dapat menikmati akomodasi yang baik
sesuai dengan cara hidup orang-orang pribumi Paga. Berbagai upaya telah
dilakukan oleh orang pribumi dengan wawasan dan kemampuan mereka yang
terbatas, untuk dapat  memahami cara hidup para wisatawan yang tentunya
memiliki cara hidup yang berbeda dengan penduduk pribumi di desa Paga.
Dengan memahami cara hidup pribumi Paga dengan segala keterbatasannya,
maka para wisatawan dapat beradaptasi dengan baik di desa Paga.
<source>AR016A  paragraf 1</source>
<source>AR015A  paragraf 1</source>
<source>AR015A  paragraf 2</source>
<source>AR015A  paragraf 3</source>
```

(a) an obfuscated passage with a summary type in its original form from testdoc025

```
sebenarnya turis akomodasi disediakan sesuai dengan cara hidup wisatawan
baik wisatawan lokal dari turis indonesia dan asing tapi ada
pengecualian untuk akomodasi wisatawan di desa daerah pedesaan turis
akomodasi di desa paga sangat banyak sesuai dengan baik dengan cara
hidup orang orang pribumi paga dengan buidings mereka yang dapat
dikategorikan vernakular makalah ini mengeksplorasi bagaimana masyarakat
pedesaan memberi makna akomodasi bagi wisatawan dengan wawasan mereka
yang terbatas di mana para wisatawan memiliki cara hidup yang berbeda
dengan orang orang pribumi pedesaan.
```

(b) A source paragraph taken from 1st paragraph of AR016A

```
Sebenarnya akomodasi turis diberikan sesuai dengan cara hidup turis
baik wisatawan lokal dari kota kota di indonesia maupun wisatawan
asing tapi ada pengecualian untuk akomodasi turis di desa daerah
pedesaan wisatawan akomodasi di desa paga sangat banyak sesuai dengan
baik dengan cara hidup orang orang pribumi paga tetapi tampaknya ada
juga upaya penduduk asli untuk memahami dan tegas untuk turis cara
hidup.
```

(c) A source paragraph taken from 1st paragraph of AR015A

```
makalah ini mengeksplorasi bagaimana masyarakat pedesaan memberi makna
dan tegas untuk akomodasi bagi wisatawan dengan wawasan mereka yang
terbatas di mana para wisatawan memiliki cara hidup yang berbeda
dengan orang orang pribumi pedesaan
```

(d) A source paragraph taken from 2nd paragraph of AR015A

```
akomodasi wisata biasanya disesuaikan dengan keinginan atau way of
life wisatawan terutama di kota namun yang terjadi di desa justru
sebaliknya fasilitas akomodasi di desa paga tetap sesuai dengan cara
hidup orang desa meskipun juga terlihat usaha orang orang desa itu
untuk mencoba mengerti dan berempati dengan cara hidup orang kota
```

(e) A source paragraph taken from 3rd paragraph of AR015A

Figure B.1: An example of simulated plagiarism case with summary obfuscation type

Appendix C

Tables Related to Experiment Results

Table C.1: The test set selected from simulated plagiarism cases. In this table, **L** stands for *light*, **M** refers to *medium*, and **H** stands for *heavy*

Test cases	**Obfuscation types**	**Obfusc. level**	**Nr. of** d_{src}	**Batches**
testdoc001	shake, parapharase	L, M	3	1
testdoc002	shake	L	1	1
testdoc003	paraphrase	M, H	1	1
testdoc004	shake	L	2	1
testdoc005	paraphrase	L, M, H	3	1
testdoc006	shake	L, M	4	1
testdoc007	paraphrase	L	2	1
testdoc008	shake	L, M	4	1
testdoc009	shake	M	2	1
testdoc010	paraphrase, shake, summary	4	2	
testdoc011	paraphrase	L, M, H	2	2
testdoc012	paraphrase, summary	M, H	2	2
testdoc013	shake, paraphrase	L, M	3	2
testdoc014	paraphrase, summary	L, M	2	2
testdoc015	paraphrase	L, M	3	2
testdoc016	copy, shake, paraphrase	L, M	3	2
testdoc017	copy, shake, paraphrase	L, M	5	2
testdoc018	shake, paraphrase	M	3	2
testdoc019	copy, shake, paraphrase	L, M	5	2
testdoc020	copy, paraphrase	L	3	2
testdoc021	paraphrase	L, H	3	3
testdoc022	copy, paraphrase	L, H	5	3
testdoc023	shake, paraphrase	L, M	3	3
testdoc024	copy, shake, paraphrase, summary	L, M, H	3	3
testdoc025	copy, shake, paraphrase	L, M, H	4	3
testdoc026	paraphrase, summary	L, M, H	3	3
testdoc027	paraphrase	L, M, H	4	3
testdoc028	paraphrase, summary	L, M	3	3
testdoc029	shake, paraphrase	L, M, H	3	3
testdoc030	copy, shake, paraphrase	L, M, H	5	3

Table C.2: The test set for artificial plagiarism case

Test Cases	**Topic description**	**Obfuscation Types**	**Obfusc. %**	**Obfusc. level**
testdoc101	Architecture & design: interior design	Synonym replacement	23	medium (M)
testdoc102	Anthropology & sociology		40	heavy (H)
testdoc103	Photography		50	heavy
testdoc114	Theology		10	light (L)
testdoc115	Tourism & travel		15	medium
testdoc116	Civil engineering		20	medium
testdoc104	Photography	Word deletion	15	light
testdoc105	Pedagogy and education		30	medium
testdoc106	Literature, art & letters		60	heavy
testdoc122	Anthropology-Sociology		10	light
testdoc123	Civil engineering		50	heavy
testdoc124	Civil engineering		50	heavy
testdoc107	Fisheries & aquaculture	deletion & insertion	40 10	M-L
testdoc117	Civil engineering		50 20	H-M
testdoc118	Anthropology-Sociology		10 15	L-L
testdoc119	Pedagogy and education		40 15	H-L
testdoc120	Literature, art & letters		15 50	L-H
testdoc121	Photography		20 40	M-H
testdoc108	Medicine & public health	Shuffle	1	medium
testdoc109	Communication		1	medium
testdoc110	Communication		1	medium
testdoc125	Biology		1	medium
testdoc126	Business & Economy		1	medium
testdoc127	Geography		1	medium
testdoc111	Tourism & travel	Insertion	50	heavy
testdoc112	Psychology		50	heavy
testdoc113	History		10	light
testdoc128	Information Technology		40	Heavy
testdoc129	Business & Economy		100	heavy
testdoc130	Geography		100	heavy

Table C.3: Results on Text Alignmnet using TK2 for APC

Obfusc.	Character-based Measures				Passage-based			Doc.-based		Case-based
	Plagdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	
Delete	.47	.83	.36	1	.83	.83	.83	.83	.83	.83
Insert	.71	.99	.57	1	.92	1	.96	.92	1	1
Del+Ins	.88	.99	.8	1	.92	1	.96	.92	1	1
Synonym	.62	.80	.56	1	.72	.72	.83	.83	.72	.83
Shuffle	.13	.57	.08	1	.58	.67	.67	.58	.67	.67

Table C.4: Results on Text Alignmnet using TK4 for APC

Obfusc.	Character-based Measures				Passage-based			Doc.-based		Case-based
	Plagdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	
Delete	.41	.83	.28	1	.83	.83	.83	.83	.83	.83
Insert	.61	.96	.46	1	1	1	1	1	1	1
Del+Ins	.88	.99	.8	1	.92	1	.96	.92	1	1
Synonym	.58	.8	.49	1	.75	.83	.83	.75	.83	.83
Shuffle	.12	.57	.07	1	.58	.67	.67	.67	.58	.67

Table C.5: The raw result of obfuscation type recognition for Alvi & PlagiarIna using TK1 in SPC. The abbreviations used in column *case-based* stand for: **cp**: copy, **sh**: shake, **pL**: light paraphrase, **pM**: medium paraphrase, **pH**: heavy paraphrase, **sm**: summary. the sign **-** refers to absence of the case, and **0** means that the case is undetected.

Test	Alvi Algorithm						PlagiarIna TK1					
cases	cp	pL	pM	pH	sh	sm	cp	pL	pM	pH	sh	sm
testdoc001	-	1	.5	-	.5	-	-	1	1	-	1	-
testdoc002	-	-	-	-	1	-	-	-	-	-	1	-
testdoc003	-	-	0	.66	-	-	-	-	1	.66	-	-
testdoc004	-	-	-	-	1	-	-	-	-	-	1	-
testdoc005	-	0	1	1	-	-	-	1	1	0	-	-
testdoc006	-	-	-	-	1	-	-	-	-	-	.5	-
testdoc007	-	.5	-	-	-	-	-	1	-	-	-	-
testdoc008	-	-	-	-	.75	-	-	-	-	-	0	-
testdoc009	-	-	-	-	1	-	-	-	-	-	1	-
testdoc010	-	1	0	-	1	1	-	1	1	-	0	1
testdoc011	-	.5	0	1	-	-	-	1	.5	0	-	-
testdoc012	-	-	1	-	-	1	-	-	1	-	-	0
testdoc013	-	1	1	-	0	-	-	1	0	-	1	-
testdoc014	-	.5	-	-	-	.25	-	1	-	-	-	.75
testdoc015	-	0	.33	-	-	-	-	1	.33	-	-	-
testdoc016	1	0	0	-	0	-	1	0	0	-	1	-
testdoc017	1	1	1	-	1	-	1	0	1	-	.5	-
testdoc018	-	-	1	-	1	-	-	-	1	-	1	-
testdoc019	1	-	.75	-	0	-	0	-	.5	-	1	-
testdoc020	.66	1	-	-	-	-	.33	1	-	-	-	-
testdoc021	-	0	-	0	-	-	-	1	-	.5	-	-
testdoc022	.33	.33	-	0	-	-	1	1	-	1	-	-
testdoc023	-	.50	0	-	0	-	-	1	0	-	1	-
testdoc024	0	0	-	-	0	0	.50	1	-	-	0	0
testdoc025	.5	1	0	0	1	-	1	1	1	1	1	-
testdoc026	-	0	.40	0	-	0	-	.50	.40	.20	-	0
testdoc027	-	.5	0	1	-	-	-	0	.33	1	-	-
testdoc028	-	1	0	0	-	.5	-	1	.50	1	-	.50
testdoc029	-	.75	.50	1	1	-	-	.50	1	.50	1	-
testdoc030	1	1	.50	0	.66	-	1	1	.50	0	.66	-

Table C.6: Text Alignment result of PlagiarIna on SPC using 7-grams

Test	Character-based Measures				Passage-based			Doc.-based		Case-
cases	Pdet	Prec	Rec	Gran	Prec	Rec	F1	Prec	Rec	based
testdoc001	.28	.97	.17	1	1	.29	.45	1	.67	pL:1, pM:0 sh:0,25
testdoc002	.73	.95	.6	1	1	.5	.67	1	1	sh:0,5
testdoc003	0	0	0	0	0	0	0	0	0	pM:0,5, pH:0
testdoc004	.54	.66	.46	1	.33	.33	.33	.67	1	sh:0,67
testdoc005	0	0	0	0	0	0	0	0	0	pL:0, pM:0, pH:0,
testdoc006	.23	.72	.13	1	1	.25	.4	1	.25	sh:0,25
testdoc007	0	0	0	0	0	0	0	0	0	pL:0
testdoc008	0	0	0	0	0	0	0	0	0	sh:0
testdoc009	.52	.57	.48	1	.5	.5	.5	1	.5	sh:0,5
testdoc010	.48	.62	.4	1	.5	.5	.5	1	.75	pL:0, pM:0, sh:1, sm:1
testdoc011	.38	1	.24	1	1	.17	.29	1	.5	pL:0,5; pM:0, pH:0
testdoc012	0	0	0	0	0	0	0	0	0	pM:1; sm:0
testdoc013	.7	.76	.56	1	1	1	1	1	1	pL:1, pM:1, sh:1
testdoc014	0	0	0	0	0	0	0	0	0	pL:0, sm:0
testdoc015	0	0	0	0	0	0	0	0	0	pL:0, pM:0
testdoc016	.31	.84	.19	1	1	.2	.33	1	.33	cp:0, pL:0, pM:0, sh:0,5
testdoc017	.48	.99	.31	1	1	.4	.57	1	.4	cp:1, pL:0, pM:0, sh:0,5
testdoc018	.75	1	.6	1	1	.67	.8	1	.67	pM:1, sh:1
testdoc019	.43	1	.27	1	1	.33	.5	1	.4	cp:0, pL:0.25, sh:1
testdoc020	.3	.96	.17	1	1	.25	.4	1	.33	cp:0,33, pL:0
testdoc021	0	0	0	0	0	0	0	0	0	pL:0 pH::0,5
testdoc022	0	0	0	0	0	0	0	.5	.8	Cp:0, pL:0, pH:0
testdoc023	.12	.54	.07	1	.5	.2	.28	1	.33	pL::0,5, pM:0, sh:0
testdoc024	0	0	0	0	0	0	0	0	0	cp:0, pL:1, sh:0 sm:0
testdoc025	.35	1	.21	1	1	.33	.5	1	.25	cp:0,5;, pL:1, pM:0, pH:0, sh:0
testdoc026	.29	.5	.2	1	.6	.2	.3	1	1	pL:0,50; pM:0,14, pH:0,2; sm:0
testdoc027	.12	.77	.06	1	1	.17	.28	1	.25	cL:0,5, pM:0; pH:0
testdoc028	.14	1	.08	1	1	.17	.28	1	.33	pL:1; pM:0, pH:0, sm:0
testdoc029	.14	1	.08	1	1	.1	.17	1	.33	pL:0; pM:0, pH:0, sh:1
testdoc030	.15	1	.08	1	1	.12	.22	1	.2	cp:1, pL:0, pm:0, pH:0, sh:0

Bibliography

[1] Abnar, S., Dehghane, M., Zamani, H., and Shakery, A. *Expanded N-Grams for Semantic Text Alignment*. Notebook for PAN at CLEF 2014, 2014. `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html`.

[2] Adam, A. R., and Suharjito. Plagiarism Detection Using NLP based on Grammar Analyzing. *Journal of Theoretical and Applied Information Technology 63*, 1 (2014), 168–180.

[3] Akiva, N. Using Clustering to Identify Outlier Chunks of Text. In *Notebook for PAN at CLEF 2011* (Amsterdam, The Netherlands, 2011). available at `http://www.uni-weimar.de/medien/webis/events/pan-11/pan11-web/about.html`.

[4] Alfikri, Z. F., and Purwarianti, A. The Construction of Indonesian-English Cross Language Plagiarism Detection. *Journal of Computer Science and Information 5*, 1 (2012), 16–23.

[5] Alfikri, Z. F., and Purwarianti, A. Detailed analysis of extrinsic plagiarism detection system using machine learning approach (naive bayes and svm). *TELKOMNIKA Indonesian Journal of Electrical Engineering 12*, 11 (2014), 7884–7894.

[6] Alieva, N. F. *Bahasa Indonesia: Deskripsi dan teori*. Kanisius, Yogyakarta, 1991.

[7] Alvi, F., Stevenson, M., and Clough, P. Hashing and Merging Heuristics for Text Reuse Detection. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[8] Alwi, H., and Sardjowidjojo, S. *Tata Bahasa Baku Bahasa Indonesia*, third ed. Balai Pustaka, Jakarta, 2003.

[9] Alzahrani, S., et al. Understanding Plagiarism Linguistic Patterns, Textual Features and Detection Methods. *IEEE Transaction on Systems, Man, and Cybernetics Parts C: Apllication and Reviews 42*, 2 (2011).

[10] Alzahrani, S., and Salim, N. Fuzzy Semantic-Based String Similarity for Extrinsic Plagiarism Detection. In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[11] Andriessen, S. Benefiting from Back Translation. *Medilingua portal* (2008). retrieved from `http://www.medilingua.com/pdf/BackTranslationsICTSummer%202008.pdf` in february 20, 2015.

[12] Angoff, W. The Development of Statistical Indexes for Detecting Cheaters. *ETS Research Bulletin Series 1972*, 1 (1972), 1–24.

[13] Arka, I. W. Developing a Deep Grammar of Indoensian within the ParGram Framework: Theoritical and Implementational Challenges. In *26th Pacisif Asia Conference on Language, Information and Computation* (2012), pp. 19–38.

[14] Arka, I. W., and Manning, C. Voice and Grammatical Relation in Indonesian: New Perspective. In *Voice and Grammatical Relations in Austronesian Languages* (Stanford, 2008), P. K. Mustin and S. Musgrave, Eds., CSLI, pp. 45–69.

[15] Asian, J. *Effective Techniques for Indonesian Text Retrieval.* PhD thesis, RMIT University, Melbourne, Australia, 2007.

[16] Baker, B. S. A Program for Identifying a Duplicate Code. In *Proc. of the 24^{th} Symposium on the Interface: Computer Science and Statistics* (1992), ACM Press, pp. 18–21.

[17] Bao, J., Lyon, C., Lane, P. C. R., Ji, W., and Malcolm, J. *Comparing Different Text Similarity Methods.* UH Computer Science Technical Report. University of Hertfordshire, 2007.

[18] Bao, J., Shen, J., Liu, X., Liu, H., and Zhang, X. Document Copy Detection Based on Kernel Method. In *Proceedings of 2003 IEEE International Conference on Natural Language Processing and Knowledge Engineering* (Beijing, China, 2003), pp. 250–256.

[19] Basile, C., et al. A Plagiarism Detection Procedure in Three Steps: Selection, MAtches, "squares". In *Proceedings of SEPLN'09* (2009), B. S. et al, Ed., pp. 19–23.

[20] Bliss, T. *Statistical methods to Detect Cheating on Test: A Review of the Literature.* PhD thesis, Brigham Young University, 2012.

[21] Boubekeur, F., and Azzoug, W. Concept-Based Indexing in Text Information Retrieval. *International Journal of Computer Science and Information Technology (IJCSIT) 5*, 1 (2013), 119–136.

[22] Bouville, M. Plagiarism: Words and Ideas. *Science and Engineering Ethics 14* (2008), 311–322.

[23] Bretag, T., and Mahmud, S. Self-Plagiairsm or Appropriate Textual Re-use? *Journal of Achademic Ethics 7* (2009), 193–203. DOI:10.1007/s10805-009-9092-1.

[24] Brin, S., et al. Copy Detection Mechanisms for Digital Documents. In *Proceedings 1995 ACM SIGMOD International Conference of Managment Data* (New York, USA, 1995), pp. 398–409.

[25] Buranen, L., and Roy, A. *Perspective on Plagiarism and Intellectual Property in a Postmodern World.* State University of New York, New York, 1999.

[26] Cedeño, A. B., et al. Monolingual Text Similarity Measures: A Comparison of Models over Wikipedia Articles Revision. In *ICON 2009* (Hyderabad, India, 2009), S. et al., Ed., pp. 29–38.

[27] Cedeño, A. B., and Rosso, P. Towards the 2nd international competition on plagiarism detection and beyond. In *Proceedings of PAN CLEF 2010 LABs and Workshops* (Amsterdam, Netherland, 2010), V. Petras and P. Clough, Eds. Notebook Papers available at `http://www.clef2010.org/index.php?page=pages/proceedings.php`.

[28] Cedoño, A., and Rosso, P. An Automatic Plagiarism Detection based on N-gram Comparison. In *ECIR 2009 LNCS 5478* (Berlin, Germany, 2009), M. Boughanem, Ed., Springer verlag, pp. 696–700.

[29] Ceri, S., et al. *Web Information Retrieval.* Springer Verlag, Heidelberg, 2013.

[30] Cha, S. H. Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. *International Journal of MAthematical Models and Methods in Apllied Sciences 1*, 4 (2012), 300–307.

[31] Chaer, A. *Sintaksis Bahasa Indonesia: Pendekatan prosess.* Reneka Cipta, Jakarta, 2009.

[32] Charikar, M. Similarity Estimation Techniques from Rounding Algorithm. In *Proceeding of 34th Annual Symposium on Theory of Computing (STOC)* (2008), pp. 380–388.

[33] Chen, C., Yeh, J., and Ke, H. Plagiarism detection using rouge and wordnet. *Journal of Computing 2*, 3 (2010).

[34] Choi, S. S., Cha, S. H., and Tappert, C. C. A Survey of Binary and Distance Measures . *Systematics, Cybernatics and Informatics 8*, 1 (2010), 43–48.

[35] Chong, M. *A Study of Plagiarism Detection and Plagiarism Identification Using Natural Language Processing Techniques.* PhD thesis, University of Wolverhampton, 2013. Retrieved from Portal of Wolverhampton Intellectual repository and E-Theses.

[36] Chung, S. *Subject and Topic.* Academic Press, New York, 1976, ch. On the Subject of Two Passives in Indonesian.

[37] Clough, P., and Stevenson, M. Developing a Corpus of Plagiarist Short Answers. *Language Resources and Evaluation 45*, 1 (2011), 5–24.

[38] Costa-Jussá, M. R., R. E. Banchs, J. G., and Codina, J. Plagiarism Detection Using Inforamtion Retrieval and Similarity Measures Based on Image Processing Techniques . In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[39] Cotterell, R., Müller, T., Fraser, A., and Schütze, H. Labeled morphological segmentation with semi-markov models. In *Proceedings of the 19th Conference on Computational Language Learning* (Beijing, China, 2015), pp. 164–174.

[40] Cumming, S. *Functional Change: the Case of Malay Constituent Order*. Mouton de Gruyter, Berlin, New York, 1991.

[41] Darjowidjojo, S. *Sentence Pattern of Indonesian*. Hawaii University Press, Honolulu, 2004.

[42] Davies, M. *TIME Magazine Corpus: 100 million words, 1920s-2000s*, 2007-. Available online at `http://corpus.byu.edu/coca/`.

[43] Davies, M. *The Corpus of Contemporary American English: 450 million words, 1990-present*, 2008-. Available online at `http://corpus.byu.edu/coca/`.

[44] Davies, M. *The Corpus of Historical American English: 400 million words, 1810-2009*, 2010-. Available online at `http://corpus.byu.edu/coca/`.

[45] Djafar, F. B., Lahinta, A., and Hadjaratie, L. Penerapan algoritma smith-waterman dalam sistem pendeteksi kesamaan dokumen. 2013.

[46] Dobrovska, D. Avoiding Plagiarism and Collusion. In *International Conference on Engineering Education (ICEE)* (2007).

[47] Eggins, S. *Introduction to Systemic Functional Linguistics*, second ed. Continum International Publishing Group, New York, 2004.

[48] Eissen, S., and Stein, B. Intrinsic Plagiarism Detecion. *ECIR 2006, LNCS 3936* (2006), 565–569.

[49] Elizalde, V. Using Statistic and Semantic Analysis to Detect Plagiairsm . In *Notebook Papers of PAN at CLEF 2013* (2013). `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[50] Elizalde, V. *Using Noun Phrases and tf-idf for Plagiarized Document Retrieval.* Notebook for PAN at CLEF 2014, 2014. `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html`.

[51] Fang, L. Y. *Indonesian Grammar Made Easy.* Times Books International, Singapore, 1996.

[52] Fischer, J. *Data Structure for Efficient String Algorithm.* PhD thesis, Ludwig-Maximilians Universitaet, Muenchen, 2007.

[53] Furihata, M. *Prosody and Syntax: Cross Linguistic Perspective.* John Benjamins Publishing Company, Amsterdam, 2006, ch. An Acoustic Study on Intonation of Nominal Sentences in Indoensian, pp. 327–348.

[54] Ghosh, A., et al. Rule-based Plagiarism Detection Using Information Retrieval. In *LAB Report for PAN at CLEF 2011* (2011). available at `http://www.uni-weimar.de/medien/webis/events/pan-11/pan11-web/about.html`.

[55] Gil, D. *Verb first: On the Syntax of Verb-Initial Languages.* John Benjamin Publishing Company, Amsterdam, 2005, ch. Word Order Without Syntactic Categories: How Riau Indonesian Does it.

[56] Gilam, L., Newbold, N., and cooke, N. Educated Guesses and Equality Judgements: Using Search Engines and Pairwise Match for External Plagiarism Detection. In *Proceedings of PAN at CLEF 2013* (2013). available at`http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[57] Gilam, L., and Notley, S. Evaluating Robustness for 'IPCRESS': Surrey's Text Alignment for Plagiairsm Detection. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[58] Gipp, B. *Citation-based Plagiairsm Detecion: Detecting Disguised and Cross-Language Plagiarism Using Citation Pattern Analysis.* PhD thesis, Magdeburg University, Wiesbaden, 2014.

[59] Gipp, B., and Beel, J. Citation-Based Plagiarism - A New Approach to Identify Plagiarized Work Language Independently. In *Proceedings of the 21th ACM Conference on Hypertext and Hypermedia* (2010), ACM. Available online in researchGate `http://www.researchgate.net/directory/publications`.

[60] Gipp, B., and 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 11th ACM Symposium on Document Engineering (DocEng'11)* (Mountain View, CA, USA, Sep 2011), ACM.

[61] Gipp, B., Meuschke, N., and j. Beel. Comparative Evaluation of Text- and Citation-based Plagiarism Detection Approaches Using GuttenPlag. In *Proceedings of 11th Annual International ACM//IEEE-CS Joint Conference on Digital Libraries (JCDL'11)* (Ottawa, Canada, 2011), ACM Press.

[62] Glinos, D. A Hybrid Architecture for Plagiarism Detection. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[63] Grman, J., and Raavas, R. Improved Implementation of Finding Text Similarities in Large Collection of Data. In *LAB Report for PAN at CLEF 2011* (2011). avalable at `http://www.uni-weimar.de/medien/webis/events/pan-11/pan11-web/about.html`.

[64] gross, P., and Modaresi, P. Plagiarism Alignment Detection by Merging Context Seeds. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[65] Grozea, C., and Gehl, C. ENCOPLOT: Pair Wise Sequence Matching in Linear Time Applied to Plagiarism Detection. In *Proceeding of SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse* (2009), B. S. et al, Ed., pp. 10–18.

[66] HaCohen-Kerner, Y., et al. Detection of Simple Plagiarism in Computer Science Paper. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)* (Beijing, China, August 2010).

[67] Hagen, M., Potthast, M., and Stein, B. Source retrieval for plagiarism detection. In *CLEF 2015 and Workshops, Notebook-papers* (France, 2015), L. Cappello, N. Ferro, and E. S. Juan, Eds.

[68] Haggag, O., and El-Beltagy, S. Plagiarism Candidate Retrieval Using Selective Query Formulation and Discriminative Query Scoring. In *Notebook Papers of PAN at CLEF 2013* (2013), Forner et al., Eds. `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[69] Halvani, O. *Register Genre Seminar: Towards Intrinsic Plagiarism Detection.* http://www.halvani.de/math/pdf/ Retrieved in February 2015.

[70] Hearst, M. A. Text Tilling: Segmenting Text into Multi-paragraph subtopic Passages. *Computational Linguistics 23*, 1 (1997), 33–64.

[71] Heintze, N. Scalable Document Fingerprinting. In *Proc. of USENIX Workshop on Electronic Commerce* (1996). Available online on `https://www.usenix.org/legacy/publications/library/proceedings/ec96/summaries/node26.html`.

[72] Henzinger, M. Finding Near-Duplicate Web Pages: A Large-Scale Evaluation of Algorithm. In *SIGIR'06* (Seattle, Washington, USA, 2006), ACM.

[73] Himmelmann, N. P. *The Austronesian Languages of Asia and Madagascar*. ch. The Austronesian Languages of Asia and Madagascar: Typological Characteristics.

[74] Hoad, T. C., and Zobel, J. Methods for Identifying Versioned and Plagiarized Documents. *Journal of the American Society for Informtion Science and Technology 54*, 3 (2003), 203–215.

[75] Huang, A. Similarity Measures for Text Document Clustering. In *Proceedings of New Zealand Computer Science Research Student Conference (NZCSRSC'08)* (Christchurch, New zealand, 2008).

[76] Hunter, J. *The important of Citation*. avaliable online in `http://web.grinnell.edu/Dean/Tutorial/EUS/IC.pdf`.

[77] Jayapal, A. K. Similarity Overlap Metric and Greedy String Tiling at PAN 2012: Plagiarism Detecion. In *LAB Report for PAN at CLEF 2010* (2010), Braschler et al., Eds. available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[78] Jones, M. Back Translation: the Latest Form of Plagiarism. In *Fourth Asia Pasific Conference on Educational Integrity (4APCEI)* (Wollongong, Australia, Sept, 28–30 2009).

[79] Kasprzak, J., and Brandejs, M. Improving the Reliability of the Plagiarism Detection System. In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[80] Kasprzak, J., et al. Finding Plagiarism by Findng Document Similarities. In *LAB Report for PAN at CLEF 2009* (2009). avalable at `http://www.uni-weimar.de/medien/webis/events/pan-09/pan09-web/about.html`.

[81] Kiabod, M., Dehkordi, M. N., and Sharafi, S. M. A Novel method of Significat Words Identification in Text Summarization. *Journal of Emerging Technologies in Web Intelligence 4*, 3 (2012), 252–258.

[82] Kong, L., et al. Approaches for Candidate Document retrieval and Detailed Comparison of Plagiarism Detection. In *Proceedings of PAN at CLEF 2012* (2012). available at `http://www.uni-weimar.de/medien/webis/events/pan-12/pan12-web/about.html`.

[83] Kong, L., et al. Approaches for Source Retrieval and Text Alignment of Plagiairsm Detecion. In *Notebook Papers of PAN at CLEF 2013* (2013). `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[84] Kong, L., et al. *Source Retrieval Based on Learning to Rank and Text Alignment Based on Plagiairsm Type Recognition for Plagiarism Detecion.* Notebook for PAN at CLEF 2014, 2014. `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html`.

[85] Kong, L., Lu, Z., Qi, H., and Han, Z. Detecting High Obfuscation Plagiarism: Exploring Multi-Features via Machine Learning. *International Journal of u- and e-Service, Science and Technology 7*, 4 (2014), 385–396.

[86] Kridalaksana, H. *Masa Lampau Bahasa Indonesia: Sebuah Bunga Rampai.* ch. Sejarah Peristilahan dalam Bahasa Indonesia.

[87] Krisnawati, L. D., and Schulz, K. U. Plagiarism detection for indonesian texts. In *Proceedings of the 15th Int. Conference on Information Integration and Web-based Applications and Services (iiWAS2013)* (Vienna, Austria, 2013), E. Weippl et al., Eds., pp. 595–599.

[88] Kumar, C. A., Radvansky, M., and Annapurna, J. Analysis of a Vector Space Model, LAtent Semantic Indexing, and Formal Concept Analysis for Information Retrieval. *Cybernatics and Information Technologies 12*, 1 (2012).

[89] Kurniawati, A., Puspitodjati, S., and Rahman, S. Implementasi jaro-winkler distance untuk membandingkan kesamaan dokumen berbahasa indonesia. Available online in http://repository.gunadarma.ac.id/394/1/Implementasi

[90] Lee, T., Chae, J., park, K., and Jung, S. CopyCaptor: Plagiarized Source Retrieval System Using Global Word Frequency and Local Feedback. In *Notebook Papers of PAN at CLEF 2013* (2013), Forner et al., Eds. `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[91] Leskovec, J., Rajaraman, A., and Ullman, J. D. Mining of massive datasets. availabel online on `http://infolab.stanford.edu/~ullman/mmds/book.pdf`.

[92] LSA. *Asian Languages and Culture.* online article posted in LSA, University of Michigan, 2012.

[93] Lynch, J. The Perfectly Acceptable Practice of Literary Theft: Plagiarism, Copyright, and the 18th Century. *Colonial Williamsburg: The Journal of Collonial Williamsburg Foundation 24*, 4 (2006), 51–54. Available online at Writing World.

[94] Mahathir, F. Sistem Pendeteksi Plagiat pada Dokumen Teks Berbahasa Indonesia Menggunakan Metode Rouge-N, Rouge-L dan Rouge-W. In *IPB Bogor Agricultural University Scientific Repository*, 2011.

[95] Manber, U. Finding Similar Files in a Large File System. In *1994 Winter USENIX Technical Conference* (San Fransisco, CA, 1994), pp. 1–10.

[96] Manber, U., and Myers, G. Suffix Arrays: A New Method on Online String Searches. *SIAM Journal on Computing 22* (1991), 935–948.

[97] Manning, C., Raghavan, P., and Schuetze, H. *Introduction to Information Retrieval.* Cambridge University Press, Cambridge, 2008.

[98] Manning, C. D., and Schuetze, H. *Foundations of Statistical NAtural Languange Processing.* MIT Press, Cambridge, massachusetts, London, England, 1999.

[99] Mao, X., Liu, X., Di, N., Li, X., and Yan, H. SizeSpotSigs: An Effective Deduplicate Algorithm Considering the Size of Page Content . In *Advances in Knowledge Discovery and Data Mining: 15th Pacific-Asia Conferen, PAKDD 2011PART I, LNAI* (2011), Springer Verlag, pp. 537–548.

[100] Mardiana, T., Adji, T. B., and Hidayah, I. The comparison of distance-based similarity measure to detection of plagiarism. In *ICSIIT:2015, CCIS 516* (2015), Springer Verlag, pp. 155–164.

[101] McGill, S. *Plagiarism in Latin Literature.* Cambridge University Press, Cambridge, 2012.

[102] McInnis, J. R., et al. *Plagiarism Detection Software: How Effective is it?* In *Assesing Learning in Australian Universities* (2002), Australian Universities Teaching Committee, AUTC.

[103] Meuschke, M., and Gipp, B. State-of-the-art in Detecting Academic Plagiarism. *International Journal for Educational Integrity 9*, 1 (2013), 50–71.

[104] Micol, D., et al. A Textual-based Similarity Approach for Efficient and Scalable External Plagiarism Analysis . In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[105] Mistica, M., Andrews, A., Arka, I., and Baldwin, T. Double Double: Morphology and Trouble: Looking into Reduplication in Indonesian. In *Australasian Language Technology Association Workshop (ALTA 2009)* (Sydney, Australia, 2009), L. Pizzato and R. Schwitter, Eds., Australasian Language Technology Association, pp. 44–45.

[106] Monostori, K., et al. Suffix Vector: Time- and Space- Efficent Alternative to Suffix Tree. In *25th Australasian Computer Science Conference* (Melbourne, Australia, 2002), M. Oudshoorn, Ed., vol. 2.

[107] Mozgovoy, M., et al. Automatic Student Plagiarism Detection: Future Perspective. *Journal of Educational Computing Research 43*, 3 (2010), 511–531.

[108] Muhr, M., et al. External intrinsic plagiarism detection using a cross-lingual retrieval and segmentation system. In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[109] Müller-Gotama, F. *Gramatical relations: A Cross-Linguistic Perspective on their Syntax and Semantics.* Mouton de Gruyter, Berlin, New York, 1994.

[110] Muslich, M. *Bahasa Indonesia pada Era Globalisasi: Kedudukan, Fungsi, Pembinaan dan Pengembangan* . Bumi Aksara, Jakarta, 2012.

[111] Nawab, R. M. A., Stevenson, M., and Clough, P. University of Sheffield. In *LAB Report for PAN at CLEF 2010* (2010), Braschler et al., Eds. available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[112] Nawab, R. M. A., Stevenson, M., and Clough, P. External Plagiarism Detection Using Information Retrieval and Sequence Alignment. In *LAB Report for PAN at CLEF 2011* (2011). available at `http://www.uni-weimar.de/medien/webis/events/pan-11/pan11-web/about.html`.

[113] Oberreuter, G., and Eiselt, A. Submission to the 6th International Competition on Plagiarism Detecion. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[114] Oberreuter, G., et al. Approaches for Intrinsic and External Plagiarism Detection. In *LAB Report for PAN at CLEF 2011* (2011). available at `http://www.uni-weimar.de/medien/webis/events/pan-11/pan11-web/about.html`.

[115] Ottenstein, K. An Algorithmic Approach to the Detection and Prevention of Plagiarism. *ACM 8* (1976), 30–41.

[116] Palkovskii, Y., and Belov, A. Using Hybrid Similarity Methods for Plagiarism Detection. In *Notebook Papers of PAN at CLEF 2013* (2013), Forner et al., Eds. `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[117] Palkovskii, Y., and Belov, A. Developing High-Resolution Universal Multi-Type n-Gram Plagiarism detector. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[118] Palkovskii, Y., et al. Using WordNet-based Semantic Similarity Measurement in External Plagiarism Detection. In *LAB Report for PAN at CLEF 2011* (2011). available at `http://www.uni-weimar.de/medien/webis/events/pan-11/pan11-web/about.html`.

[119] Parth, G., Sameer, R., and Majumdar, P. External Plagiarism Detection: N-Gram Approach Using NAmed Entitty Recognizer. In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[120] Pereira, R., et al. URFGS@PAN2010: Detecting External Plagiarism. In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[121] Pertile, S., Moreira, V., and Rosso, P. Comparing and Combining Content- and Citation-based Approaches for Plagiarism Detecion. *Journal of the Association for Information Science and Technology* (2015). DOI: 10.1002/asi.23593.

[122] Pisceldo, F., Mahendra, R., Manurung, R., and Arka, I. A Two-Level Morphological Analyser for the Indonesian Language. In *Proceedings of 2008 Australasian Language Technology Association Workshop ALTA 2008* (2008).

[123] Poesponegoro, M. D., and Notosusanto, N. *Sejarah Nasional Indonesia VI: Zaman Jepang dan Zaman Republik Indonesia* . Balai Pustaka, Jakarta, Indonesia, 1993.

[124] Potthast, M., et al. Overview of the 1st International Competition on Plagiarism Detection. In *Proceedings of the SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse* (San Sebastian, Spain, Sept, 10 2009), B. Stein et al., Eds.

[125] Potthast, M., et al. Overview of the 2nd International Competition on Plagiarism Detection. In *Notebook Papers of CLEF 2010 Labs and Workshops* (Padua, Italy, 2010), M. Braschler and D. Harman, Eds. `https://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html`.

[126] Potthast, M., et al. Overview of the 3rd International Competition on Plagiarism Detection. In *Notebook Papers of CLEF 2011 Labs and Workshops* (Amsterdam, Netherland, Sept, 19-22 2011). `http://www.uniweimar.de/medien/webis/research/events/pan-11/pan11-web/`.

[127] Potthast, M., et al. Overview of the 4rd International Competition on Plagiarism Detection. In *Notebook Papers of CLEF 2012 Labs and Workshops* (Rome, Italy, Sept, 17-20 2012), P. forner et al., Eds. `http://www.uni-weimar.de/medien/webis/events/pan-12/pan12-web/about.html`.

[128] Potthast, M., et al. An Overview of the 5th International Competition on Plagiarism Detection. In *CLEF 2013 Evaluation Lab Workshop.* (Sept., 23-26 2013), P. Forner, R. Navigili, and D. Tulis, Eds., pp. 85–98. Valencia, Spain.

[129] Potthast, M., et al. Overview of the 6rd International Competition on Plagiarism Detection. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[130] Potthast, M., and Stein, B. New issues in near-duplicate detection. In *Data Analysis, Machine Learning and Applications: 31th Conf. of German Classification Society* (Berlin, 2008), Preisach et al., Eds., pp. 601–609.

[131] Potthast, M., Stein, B., Cedeño, A. B., and Rosso, P. An Evaluation Framework for Plagiarism Detection. In *Proceedings of 23th International Conference on Computational Linguistics (COLING 2010)* (August 2010), pp. 85–98. Beijing, China.

[132] Prakash, A., and Saha, S. K. *Experiments on Document Chunking and Query Formulation for Plagiarism Source Retrieval.* Notebook for PAN at CLEF 2014, 2014. `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html`.

[133] Pratama, M. R., Cahyono, E. B., and Marthasari, G. I. Aplikasi pendeteksi duplikasi dokumen teks bahasa indonesia menggunakan algoritma winnowing dengan metode k-gram dan synonym recognition. 2012.

[134] Prieur, K., and Lecroq, T. On-line Construction of Compact Suffix Vectors and Maximal Repeats. *Theoritical Computer Science 407*, 1–3 (2008), 290–301.

[135] Purwitasari, D., et al. The use of hartigan index for initializing k-means++ in detecting similar texts of clustered documents as a plagiarism indicator. *Asian Journal of Information Technology 10*, 8 (2011), 341–347. DOI = 10.3923/ajit.2011.341.347.

[136] Quinn, G. *The Learner's of Today's Indonesian* . Allen and Unwin, New South Wales, Australia, 2001.

[137] Ramakrishna, M. V., and Zobel, J. Performance in Practice of String Hashing Functions. In *Proc. of the International Conf. on Database Systems for Advanced Applications* (Australia, 1997).

[138] Ramlan, M. *Morfologi, Suatu Tinjauan Deskriptif: Ilmu Bahasa Indonesia.* U.P. Karyono, Yogyakarta, 1983.

[139] Ranaivo-Malaçon, B. Computational analysis of Affixed Words in Malay Language. In *International Symposium on Malay/Indonesian Linguistics* (Penang, Malaysia, 2004).

[140] Randall, M. *Pragmatic Plagiarism: Authorship, Profit, and Power*. University of Toronto Press, Toronto, 2001.

[141] Riesbergr, S. *Symmetrical Voice and Linking in Western Austronesian Languages.* Walter de Gruyter, Boston, 2014.

[142] Roig, M. *Avoiding Plagiarism, Self-plagiarism, and other Questionable Writing Practices: A Guide to Ethical Writing*. St. John University, 2006.

[143] Salmuasih, and Sunyoto, A. Implementasi algoritma rabin karp untuk pendeteksian plagiat dokumen teks menggunakan konsip similarity. In *Proceedings of Seminar Nasional Aplikasi Teknologi Informasi (SNATI)* (Yogyakarta, 2013), pp. F23–F28.

[144] Sanchez-Perez, M., Sidorov, G., and Gelbukh, A. A Winning Approach to Text Alignment for Text reuse Detection at PAN 2014. In *Notebook Papers of PAN CLEF 2014 Labs and Workshops* (Web technology and Information System, Bauhaus-Universitaet, weimar, 2014). `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html#proceedings`.

[145] Schleimer, S., Wilkerson, D., and Aiken, A. Winnowing: Local Algorithm for Document Fingerprinting.

[146] Schleimer, S., Wilkerson, D. S., and Aiken, A. Winnowing: Local Algorithms for Document Fingerprinting. In *SIGMOD, ACM* (June, 9-12 2003).

[147] Sediyono, A., and Ku-Mahamud, K. R. Algorithm of the Longest Commonly Consecutive Word for Plagiarism Detection in Text-Based Documents. In *Proceedings of Third International Conference on Digital Information Management* (London, UK, 2008).

[148] Seo, M. J. Plagiarism and Poetic Identity in Martial. *American Journal of philology 130*, 4 (2009), 567–593.

[149] Septian, Y., Krisnawati, L. D., and Santoso, G. *Plagiarism Detection on Short Segmented Texts.* A Bachelor thesis, archived in the Library of Ducta Wacana Christian University, 2012.

[150] Shcherbinin, V., and Butakov, S. Using Microsoft SQL Server Platform for Plagiarism Detection. In *Proceeding of SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse* (2009), B. S. et al, Ed., pp. 36–37.

[151] Shivaji, S. K., and Prabhudeva, S. Plagiarism detection by using karp-rabin and string matching algorithm together. *International Journal of Computer Applications 116*, 23 (2015), 37–41.

[152] Shivakumar, N., and Garcia-Molina, H. SCAM: A Copy Detection Mechanism for Digital Documents. In *Proceedings of 2nd International Conference in Theory & Practice of Digital Libraries (DL'95)* (Austin, Texas, June 1995).

[153] Shivakumar, N., and Garcia-Molina, H. Finding near-replicas of documents on the web. In *International Workshop on Web and Database* (Valencia, Spain, March 27-28 1998).

[154] Shrestha, P., and Solorio, T. Using a Variety of N-grams for the Detection of Different Kinds of Plagiarism . In *Notebook Papers of PAN at CLEF 2013* (2013). `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[155] Sidorov, G., et al. Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model. *Computaciòn Y Sistemas 18* (2014), 491–504. DOI: 10.13053/CYS-18-4-2043.

[156] Sneddon, J. N., Adelaar, A., Djenar, D. N., and Ewing. *Indonesia : A Comprehensive Grammar* . Routledge, London, 2010.

[157] Soleman, S., and Purwarianti, A. Experiment on the Indonesian Plagiarism Detection Using Latent Semantic Analysis. In *2Nd International Conference on Information and Communication Technology (IcoICT)* (2014), IEEE, pp. 413–418.

[158] Sood, S., and Loguinov, D. Probabilistic Near-Duplicate Detection Using Simhash. In *Proc. of the 2011 ACM Int. Conference on Information and Knowledge Management* (Glasgow, UK, 2011), ACM.

[159] Srinivas, G. R. J., Tandon, N., and Varma, V. A Weighted Tag Similarity Measure Based on a Collaborative Weight Model. In *SMUC'10* (Toronto, Ontario, Canada, 2010).

[160] Stamatatos, E. Plagiarism Detection Using Stopword n-grams. *Journal of the American Society for Information Science and Technology 62*, 12 (2011), 2512–2527.

[161] Stamatatos, E., Fakatotis, N., and Kokkinakis, G. Text Genre Detection Using Common Word Frequencies. In *Proceedings of 18th International Conference on Computational Linguistics* (2000), pp. 808–814.

[162] Stearns, L. Copy Wrong: Plagiarism, Process, Property, and the Law. In *Perspective on Plagiairsm Intellectual Property in a Postmodern World*, L. Buranen and A. Roy, Eds. State University of new York, New York, 1999, pp. 6–18.

[163] Stein, B., and Eissen, M. Near Similarity Search And Plagiarism Analysis. In *Proceedings of the 29th annual Conference of German Classification Society (GfKI)* (MAgdeburg, Germany, 2006), S. et al, Ed., pp. 430–437.

[164] Stein, B., et al. Strategies for Retrieving Plagiarized Documents. In *SIGIR'07, ACM* (Amsterdam, Netherland, 2007).

[165] Stein, B., et al. Intrinsic Plagiarism Analysis. *Journal of Languange Resources and Evaluation 45*, 1 (2011), 63–82.

[166] Stein, B., and zu eissen, S. M. Fingerprint-Based Similarity Search and Its Applications . In *Gesellschaft fuer Wissenschaftliche Datenverarbeitung.* (2007), pp. 85–98.

[167] Steinhauer, H. *Masa Lampau Bahasa Indonesia: Sebuah Bunga Rampai.* ch. Tentang Sejarah Bahasa Indonesia.

[168] Suárez, P., González, J. C., and Villena, J. A Plagiarism Detector for Intrisic, External, and Internet Plagiarism. In *LAB Report for PAN at CLEF 2010* (2010). available at http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings.

[169] Suchomel, S., and Brandejs, M. *Heterogenous Queries for Synoptic and Phrasal Search.* Notebook for PAN at CLEF 2014, 2014. http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html.

[170] Suchomel, S., Kasprzak, J., and Brandejs, M. Three Way Search Engine Queries with Multi-feature Document Comparison for Plagiairsm Detecion. In *Proceedings of PAN at CLEF 2012* (2012). available at http://www.uni-weimar.de/medien/webis/events/pan-12/pan12-web/about.html.

[171] Suchomel, S., Kasprzak, J., and Brandejs, M. Diverse Queries and Feature Type Selection for Plagiarism Discoveries. In *Notebook Papers of PAN at CLEF 2013* (2013), Forner et al., Eds. http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings.

[172] Sultan, M. A., Bethard, S., and Sumner, T. Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *Transactions of the Association for Computational Lingusitics 2* (2014), 219–230.

[173] Suryana, A. F., Wibowo, A. T., and Romadhany, A. Performance Efficiency in Plagiarism Indication Detection System Using Indexing Method with Data Structure 2-3 Tree. In *2Nd International Conference on Information and Communication Technology (IcoICT)* (2014), IEEE, pp. 403–408.

[174] Suwardjono. *Pedoman Umum Pembentukan Istilah*, 2004. Bahasa Departemen Pendidikan Nasional, 1988, rewritten for the academic purpose by Suwardjono.

[175] Syahputra, A. R. Implementasi algoritma winnowing untuk deteksi kemiripan text. *Pelita Informatika Budi Dharma 9*, 1 (2015), 134–138.

[176] Tadmor, U. *Grammatical Borrowing in Cross-Linguistic Perspective.* Mouton de Gruyter, Berlin, New York, 2007, ch. Grammatical Borrowing in Indonesian .

[177] Tala, F. Z. *A Study of Stemming Effects on Infdormation Retrieval in Bahsa Indonesia.* Master's thesis, University of Amsterdam, Netherlands, 2003. Master thesis.

[178] Torrejjón, D. A. R., and Ramos, J. M. M. CoReMo System: Contextual Reference Monotony. In *LAB Report for PAN at CLEF 2010* (2010), Braschler et al., Eds. available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

[179] Torrejjón, D. A. R., and Ramos, J. M. M. Text Alignment Module in CoReMo 2.1 Plagiarism Detector . In *Notebook Papers of PAN at CLEF 2013* (2013), Forner et al., Eds. `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[180] Tschuggnall, M. *Intrinsic Plagiarism Detecion and Author Analysis by Utilizing Grammar.* PhD thesis, University of Innsbruck, 2014. Retrieved from `https://dbis-informatik.uibk.ac.at/files/diss_1.pdf`.

[181] Vania, C., and Adriani, M. Automatic External Plagiarism Using Passage Similarity. In *Proceedings of PAN CLEF 2010 LABs.* (2010), `http://ceur-ws.org/Vol-1176/CLEF2010wn-PAN-VaniaEt2010.pdf`.

[182] Vega, V. B. *Information Retrieval for Indonesian Language.* Master's thesis, National University of Singapore, 2001. Master thesis.

[183] Viselý, O., Foltýnek, T., and Rybička, J. Source Retrieval via Naive Approach and Passage Selection Heuristics. In *Notebook Papers of PAN at CLEF 2013* (2013), Forner et al., Eds. `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[184] Weber-Wulf, D. *False Features: A Perspective on Academic Plagiarism.* Springer Verlag, Berlin, 2014.

[185] Weber-Wulf, D., Moeller, C., Touras, J., and Zinke, E. *Plagiarism Detection Software Test 2013*, 2013. Available online at `plagiat.htw-berlin.de/software-en/test2013/reprot-2013`.

[186] Wibowo, A. T., Sudarmadi, K. W., and Barmawi, A. M. Comparison between fingerprint and winnowing algorithm to detect plagiarism fraud on bahasa indonesia documents. In *International Conference of Information and Communication Technology* (Bandung, 2013), IEEE Publisher, pp. 128–133.

[187] Wijaya, A. C., Krisnawati, L. D., and Hapsari, W. *Deteksi Plagiasi Otomatis Berbasis N-gram.* A Bachelor thesis, archived in the Library of Duta Wacana Christian University, 2012.

[188] Williams, K., Chen, H. H., Chowdhury, S. R., and Giles, C. L. Unsupervised Ranking for Plagiairsm Source Retrieval. In *Notebook Papers of PAN at CLEF 2013* (2013), Forner et al., Eds. `http://www.uni-weimar.de/medien/webis/events/pan-13/pan13-web/about.html#proceedings`.

[189] Williams, K., Chen, H. H., Chowdhury, S. R., and Giles, C. L. *Supervised Ranking for Plagiairsm Source Retrieval.* Notebook for PAN at CLEF 2014, 2014. `http://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/about.html`.

[190] Yilmaz, I. Plagiarism? no, we're just borrowing better english. *Nature 449*, 658.

[191] Yule, G. *The Study of Language*, fourth ed. CAmbridge University Press, Cambridge, 2010.

[192] Zebroski, J. Intellectual Property, Authority and social Formation: sociohistoricist Perspectives on the Author Functions. In *Perspective on Plagiarism and Intellectual Property in a Postmodern World*, L. Buranen and A. Roy, Eds. State University of new York, New York, 1999, pp. 31–40.

[193] Zhang, Q., et al. Efficient Partial Duplicate Detection Based on Sequence Matching. In *SIGIR'10. ACM* (Geneva, Switzerland, 2010), pp. 675–682.

[194] Zou, D., Long, W. J., and Ling, Z. A Cluster-based Plagiarism Detection Method. In *LAB Report for PAN at CLEF 2010* (2010). available at `http://www.uni-weimar.de/medien/webis/events/pan-10/pan10-web/about.html#proceedings`.

Acknowledgement

Firstly, I would like to express my sincere gratitude to my Advisor Prof. Dr. Klaus U Schulz for the continuous support of my doctoral study, for his patience, motivasion, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

My sincere thanks also go to Prof. Dr. Herwig Unger, who became my reviewer from German Academic Exchange Service (DAAD). I thank him for his insightful comments and encouragement, but also for the hard questions which incented me to widen my research from various perspectives. I also thank Prof. Dr. Titien Saraswati for her generosity for consenting her articles to be included in our evaluation corpus.

I also thank my mother, brothers and sister for their moral and spiritual support. I thank my CIS PhD-fellows for the time we have worked together and for the PhD dinner that we had every Wednesday evening. I thank my sisters in women small group, MICC, for their prayers, spiritual support and simply being there in time of troubles. I thank you for all my friends who were invloved in the crowd-sourcing, especially for Eddy Hadisaputro, Esti Wardhani, Tirta Wulandari, Ade Umar Said, Sri Rahayu, and Manila Kristin who worked more for the preliminary experiments and for building our evaluation corpus.

This research was conducted with the support of a DAAD-Indonesian German Scholarship Program (DAAD-IGSP). Hardware used for experiments was provided by Center for Information and Language Processing, Ludwig-Maximilian University, Munich, Germany.