A PLAGIARISM DETECTION SYSTEM BASED ON POS TAG N-GRAMS

POS ETİKETLERİNİN N-GRAMLARINA DAYALI BİR İNTİHAL TESPİT SİSTEMİ

KADİR YALÇIN

PROF. DR. İLYAS ÇİÇEKLİ
Supervisor

Submitted to
Graduate School of Science and Engineering of Hacettepe University
as a Partial Fulfillment to the Requirements
for the Award of the Degree of Doctor of Philosophy
in Computer Engineering

2022

ABSTRACT

A PLAGIARISM DETECTION SYSTEM BASED ON POS TAG N-GRAMS

Kadir YALÇIN

Doctor Philosophy, Department of Computer Engineering

Supervisor: Prof. Dr. İlyas ÇİÇEKLİ

September 2022, 134 pages

It is a common problem to find similar parts in two different documents or texts. Especially, a text suspected of plagiarism is likely to have similar characteristics with the source text. Plagiarism is defined as taking some or all of the writings of other people and showing them as their own, or expressing the ideas of others in different ways without citing the source. Today, it is observed that there is an increase in plagiarism cases with the development of technology. Therefore, in order to prevent plagiarism, various plagiarism detection programs have been used in universities and principles regarding plagiarism and scientific ethics have been added to education regulations.

In this thesis, a novel method for detecting external plagiarism is proposed. Both syntactic and semantic similarity features were used to identify the plagiarized parts of the text. Part-of-speech (POS) tags are used to identify the plagiarized sections of suspicious texts and the original sections corresponding to these sections in the source texts. Each source sentence is indexed by a search engine

according to its POS tag n-grams to access possible plagiarism candidate sentences rapidly. Suspicious sentences that converted to their POS tag n-grams are used as query to access source sentences. The search engine results returned from the queries enable to detect plagiarized parts of the suspicious document. The semantic relationship between two given words is calculated with Word2Vec, which is a method for using word embeddings. On the other hand, the longest common subsequence (LCS) algorithm is applied to calculate semantic similarity at the sentence level.

In this thesis, PAN-PC-11 dataset, which was created to evaluate automated plagiarism detection algorithms, is used. The tests are carried out with different parameters and threshold values to evaluate the diversity of the results. According to the experimental results with this dataset, the proposed method achieved the best performance in low and high obfuscation plagiarism cases compared to the plagiarism detection systems in the 3rd International Plagiarism Detection Competition (PAN11).

Keywords: Plagiarism detection, natural language processing, part-of-speech (POS) tagging, semantic similarity.

ÖZET

POS ETİKETLERİNİN N-GRAMLARINA DAYALI BİR İNTİHAL TESPİT SİSTEMİ

Kadir YALÇIN

Doktora, Bilgisayar Mühendisliği Bölümü

Tez Danışmanı: Prof. Dr. İlyas ÇİÇEKLİ

Eylül 2022, 134 sayfa

İki farklı doküman ya da metin içindeki benzer öğeleri bulma sıklıkla karşılaşılan bir problemdir. Özellikle intihal şüphesi taşıyan bir metnin, intihal yapılan kaynak metin ile benzer nitelikler taşıması olasıdır. İntihal kavramı, başka kişilere ait yazıların bazı bölümlerinin veya tamamının alınarak, kendisine aitmiş gibi gösterilmesi veya başkalarına ait fikirlerin kaynak göstermeden farklı şekillerde anlatılmasıdır Günümüzde teknolojinin gelişmesiyle birlikte, intihal vakalarında gittikçe artış olduğuna ilişkin değerlendirmeler gözlenmektedir. Bu nedenle, intihalin önüne geçmek amacıyla üniversitelerde çeşitli intihal tespit programları kullanılmaya başlanmış, eğitim ve öğretim yönetmeliklerine intihal ve bilimsel etik ile ilgili esaslar eklenmiştir.

Bu tez çalışması ile harici intihal tespitine ilişkin özgün bir yöntem önerilmiştir. Metin içindeki intihal edilmiş bölümleri belirlemek için hem sözdizimsel hem de anlamsal benzerlik özelliklerinden faydalanılmıştır. Şüpheli metinlerdeki intihal edilmiş bölümleri ve kaynak metinlerde bunlara karşılık gelen orijinal bölümleri tespit etmek için sözcük türü (POS) etiketi n-gramları kullanılmıştır. Her bir

kaynak cümle, olası intihal adayı cümlelere hızlı bir şekilde erişilebilmesi amacıyla bir arama motoru tarafından sözcük türü (POS) etiketi n-gramlarına göre indekslenir. Sözcük türü etiketi n-gram'larına dönüştürülen şüpheli cümleler, kaynak cümlelere erişmek için sorgu olarak kullanılır. Sorgulardan dönen arama motoru sonuçları, şüpheli belgenin intihal edilmiş bölümlerinin tespit edilmesini sağlamaktadır. Verilen iki sözcük arasındaki anlamsal ilişki sözcük temsillerini kullanma tekniği olan Word2Vec ile hesaplanır. Diğer taraftan, cümle düzeyinde anlamsal benzerliğin hesaplanması için en uzun ortak sıra (LCS) algoritması uygulanmaktadır.

Bu tez çalışması kapsamında, otomatik intihal tespit algoritmalarının değerlendirilmesi için oluşturulan PAN-PC-11 adlı veri seti kullanılmıştır. Testler, sonuçların çeşitliliğini değerlendirmek amacıyla farklı parametre ve eşik değerleri ile gerçekleştirilmiştir. Bu veri seti ile yapılan test sonuçlarına göre önerilen yöntem, 3. Uluslararası İntihal Tespiti Yarışması'nda (PAN11) yer alan intihal tespit sistemlerine göre düşük ve yüksek karmaşıklığa sahip intihal durumlarında en iyi performansı elde etmiştir.

Anahtar Kelimeler: İntihal tespiti, doğal dil işleme, sözcük türü (POS) etiketleme, anlamsal benzerlik.

CONTENTS

Page

TABLES

Page

FIGURES

Page

SYMBOLS AND ABBREVIATIONS

Symbols

e_p	External Plagiarism Detection
p	Plagiarism Case
δ	Threshold

Abbreviations

API	Application Programming Interface
BERT	Bidirectional Encoder Representations from Transformers
CDSDG	Copying Detection System of Digital Goods
COA	Citation Order Analysis
COPS	Copy Protection System
CTNG	Contextual N-Grams
ELMo	Embeddings from Language Models
GB	Gigabyte
GHz	Gigahertz
GST	Greedy String Tiling
IDF	Inverse Document Frequency
I/O	Input and Output
IR	Information Retrieval
ISO	International Organization for Standardization
LSE	Logical Structure Extraction
k-NN	k-Nearest Neighbor
LCS	Longest Common Subsequence
LDA	Latent Dirichlet Allocation
MB	Megabyte
MDR	Match Detect Reveal
ML-SOM	Multi-Layer Self-Organizing Map

NB	Naive Bayes
NLP	Natural Language Processing
NN	Neural Networks
PAN	Plagiarism Analysis, Author Identification and Near-Duplicate Detection
PAN11	3rd International Plagiarism Detection Competition
PAN-PC-11	PAN Plagiarism Corpus 2011
POS	Part-of-Speech
POSNG	Part-of-Speech Tag N-Grams
RAM	Random Access Memory
RFM	Recency, Frequency, Monetary
RKR	Running Karp-Rabin
SCAM	Standard Copy Analysis Mechanism
SRL	Semantic Role Labelling
SVD	Singular Value Decomposition
SVM	Support Vector Machine
SWNG	Stop Words N-Grams
TF-IDF	Term Frequency-Inverse Document Frequency
TF-ISF	Term Frequency-Inverse Sentence Frequency
UTF-8	Unicode Transformation Format-8
VSM	Vector Space Model
WNG	Word N-Grams
XML	Extensible Markup Language
YAP	Yet Another Plague

1. INTRODUCTION

1.1. Motivation

Nowadays, people can access information easily with the increase of data on the web. It is possible to create homework, papers or reports in a very short time using the simple copy-paste method. Therefore, it has become easier to create new documents in any subject by copying sections from different sources on the Internet [1]. This situation has caused to the existence of numerous identical or multiple documents that have same or similar content in a large database [2]. The widespread use of copying without citations has increased the incidence of plagiarism. Manual detection of plagiarism has become infeasible due to the excessive quantity of information [3] and as a result, automatic plagiarism detection tools needed to be designed.

Automatic plagiarism detection is defined as finding plagiarized sections of a suspicious document and matching them with their source text fragments [4]. Automatic plagiarism detection mainly focuses on two different aspects: The first is intrinsic plagiarism detection that finds possible plagiarism passages by analyzing the document according to the writing style without using any reference data set. On the other hand, external plagiarism detection algorithms aim to identify all matching text fragments from a collection of original and suspicious documents [5]. For text documents, existing research on automatic plagiarism detection mostly proposes methods of comparing plagiarized parts of a text with the original sources. By taking this objective as motivation, a syntactic-based external plagiarism detection method, which also includes semantic similarity features, is proposed within the scope of this thesis study.

The experiments are performed on a large dataset called PAN Plagiarism Corpus 2011 (PAN-PC-11) [6], which is created to evaluate of automatic plagiarism detection algorithms. The results obtained from the experiments have been demonstrated that the proposed method is capable to detect plagiarism cases successfully.

1.2. Problem Statement

Plagiarism is basically defined as showing someone else's writings, ideas, works or other original materials as your own without giving a proper reference [7]. As can be seen in Figure 1.1, plagiarism can take many different forms.

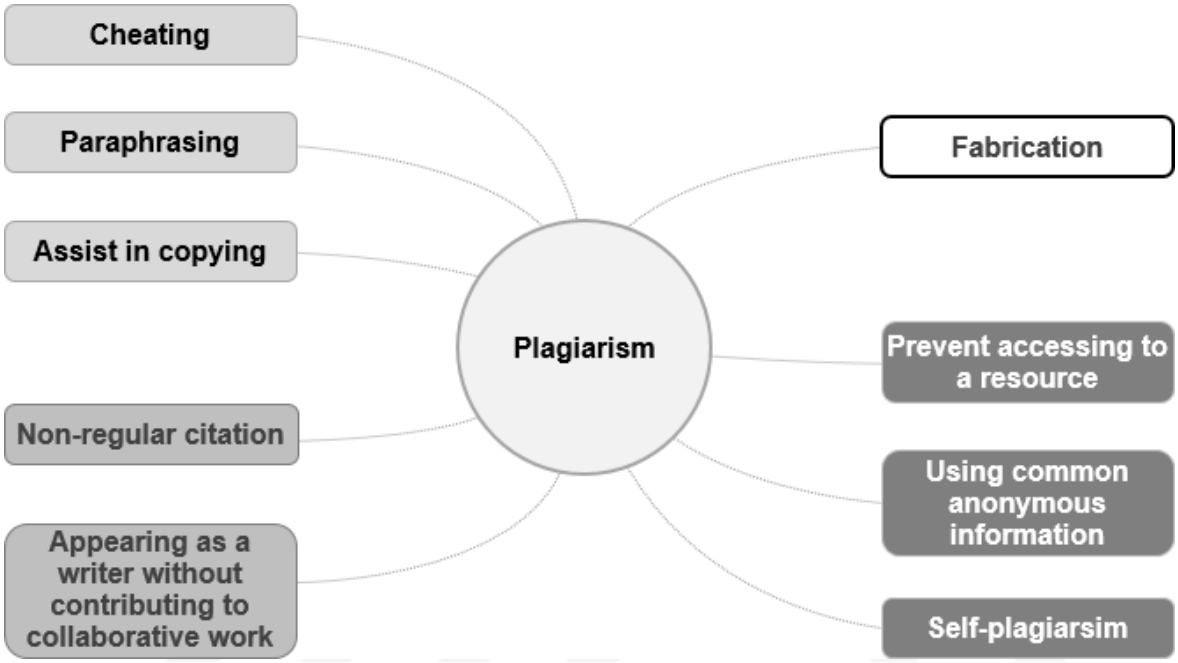

Figure 1.1. Common Forms of Plagiarism

The most common types [8-12] of plagiarism seen in student assignments or scientific papers are as follows:

- Copying all or part of a document directly without citing the source,
- Changing the linguistic structure of a document expressing someone else's ideas,
- Creating different versions of a document that have identical content given by the same author,
- Making syntactic and lexical changes on text: combining two or more sentences into a single one, splitting a sentence into several sentences,

adding or removing words or phrases, changing the order of the words etc.,

- Referring inappropriate way and reusing ideas without citations.

In addition to these different forms of plagiarism, active-passive voice conversion is one of the ways plagiarists frequently use. In this action, active sentences are converted into passive voice or vice versa based on various grammatical rules. In active voice, the subject precedes from the object. On the other hand, the object comes before the subject in passive voice. Active voice sentences are used when something is told directly, while passive voice is used when expressing actions performed by someone else.

Another problem of plagiarism is the acts made by replacing words with synonyms or antonyms in the sentences. When antonyms are changed, sentences structurally are converted into negative or positive. For example, plagiarism is tried to be hidden by using *equivalent antonyms* of “not bad” instead of “good” or “poor” instead of “not rich”. Besides, plagiarism can be done by splitting a sentence into more than one sentence or by merging multiple different or consecutive sentences into one sentence. Although there is no semantic change in any of these ways, sentence structure or words are changed in order to prevent verbatim plagiarism. Plagiarism detection becomes increasingly difficult when all of these strategies are applied in combination.

In Table 1.1, it is seen that a sentence is converted into a new sentence with a different structure by applying obfuscation strategies step-by-step. First, (1) some words are replaced with their synonyms and then (2) antonyms are used so that the meaning of the sentence remains the same. Likewise, (3) new words have been added that do not change the meaning and structure of the sentence. Finally, (4) active-passive voice conversion was made and (5) the order of the words in the sentence was changed. When comparing the original sentence with the final sentence, although two sentences are different in terms of structure and usage of the words, the final sentence completely keeps the actual sense of the original sentence. However, when comparing the current and the output sentence in any step separately, it will be easier to detect plagiarism between the two

sentences because there is less modification. However, it is obvious that there is a serious change except semantic between the original sentence and the final sentence, which is created by applying different obfuscation strategies. This shows that plagiarism detection becomes more difficult task when various different strategies are used in combination. In this case, automatic plagiarism detection basically strives to identify these obfuscation strategies.

Table 1.1. Conversion of a Sentence with Obfuscation Strategies

Obfuscation Strategy	**Result**
	Original Sentence: The researcher achieved good results in his experiments.
Synonyms Replacement	The scientist accomplished good results in his tests.
Equivalent Antonym Replacement	The scientist did not accomplish bad results in his tests.
Insert/Remove Words	The scientist did not accomplish bad results in tests of his study.
Active to Passive Voice Conversion	Bad results were not accomplished by the scientist in the tests of his study.
Change Order of the Words	In the tests of his study, bad results were not accomplished by the scientist.

Automatic plagiarism detection tools aim to detect plagiarized parts of suspicious documents and original parts in source documents that match these plagiarized parts [4]. Collection of documents can be consisted of online sources on Internet or offline sources containing the original set of documents [40]. Let $p = \{s_{plg}, d_{plg}, s_{src}, d_{src}\}$ represents a case of plagiarism; where s_{plg} is a passage in the document d_{plg} that is a plagiarized from the source passage s_{src} in document d_{src}. According to this notation, a *plagiarism detector* aims to determine p [13]. This is a very complicated process as plagiarists use various syntactic and semantic obfuscation strategies mentioned in this section.

Figure 1.2 shows an illustration of the source and plagiarized passages in the documents of d_{src} and d_{plg}. s_{src} is source passage in the d_{src} whereas s_{plg} is a passage in d_{plg} which is plagiarized from s_{src}. s_{plg} passages with the same number as s_{src} represent plagiarized sections from d_{src}. s_{src} and s_{plg} can consist

of one or more sentences. s_{plg} can be of different length and structure than s_{src}, depending on the obfuscation strategies applied to s_{src}. The plagiarism detector aims to detect p which is consisting of location and length of s_{plg} and s_{src} with d_{src} and d_{plg}.

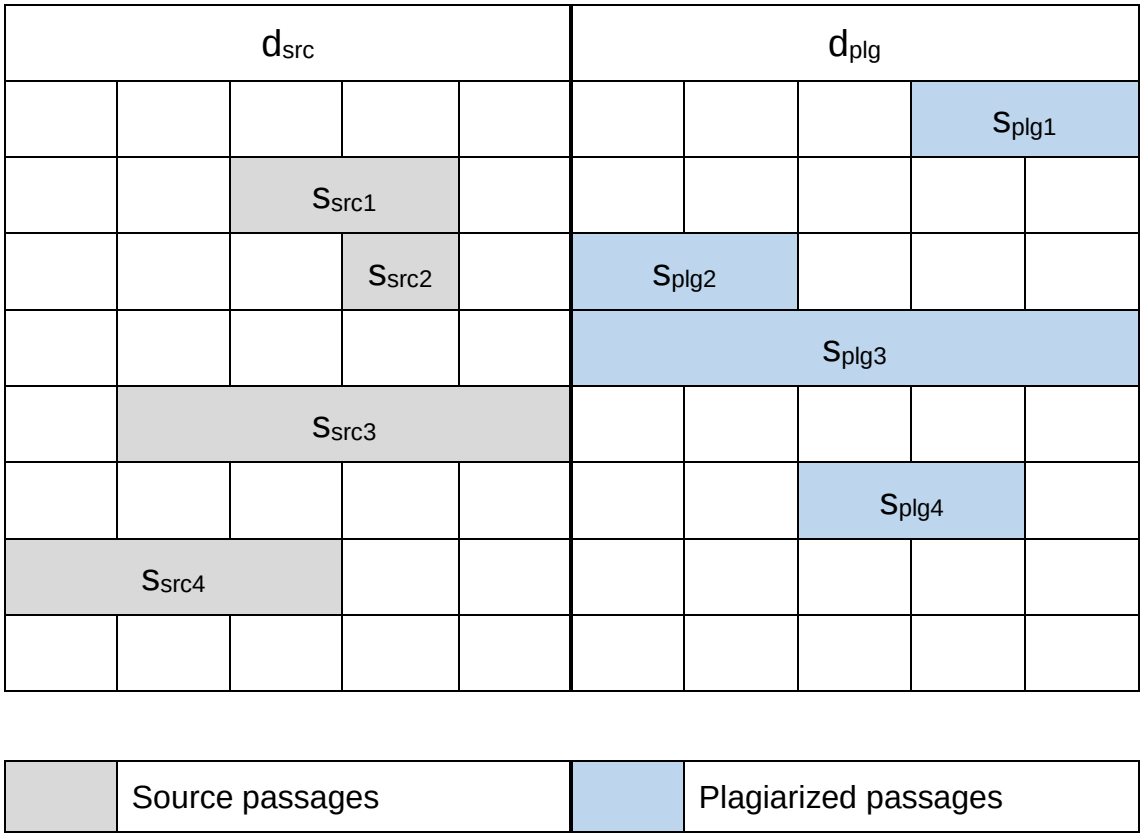

Figure 1.2. The Illustration of Source and Plagiarized Passages

The process of detecting plagiarism involves a preprocessing step to clean up the raw data of the documents using some Natural Language Processing (NLP) techniques such as lowercasing, removing stop words and punctuation, tokenization, stemming, lemmatization, synonym replacement, chunking and part-of-speech (POS) tagging. With the removal of Unicode characters, the text is cleaned and normalized at the end of these operations. Thus, parts of the text except meaningful words that are called noisy data are eliminated. Then, a set of preprocessed documents is analyzed by a comparison algorithm [14].

1.3. Aims and Objectives

The essential research objectives within the scope of this study are listed as follows:

- Develop an external plagiarism detection system focused on mono-lingual perspective.
- Develop techniques to identify candidate source sentences that are compared with the suspicious sentences.
- Make searching operation faster and provide rapid access to the candidate sentences.
- Detect syntactic and semantic similarities between source and suspicious sentences.
- Evaluate the effect of semantic similarity in the detection of plagiarism cases of different types.
- Evaluate the performance of the system using various threshold values and parameters.
- Improve the detection performance of the evaluation metrics obtained from the competition of PAN11.

1.4. Research Method

Plagiarism detection consists of two main approaches as internal/intrinsic and external [136]. Intrinsic plagiarism detection seeks to identify changes in writing style, also known as stylometry, without performing comparisons with external documents [37]. The purpose of external plagiarism detection is to find all matching text fragments between original and suspicious documents in a collection of documents [66]. Let p be a plagiarism case and e_p be external plagiarism detection, the simple formal definition of e_p is as follows: $e_p = <$

$s_{plg}, s_{src} > | s_{plg} \cap s_{src}| > \delta$, where δ is a threshold and $|s_{plg} \cap s_{src}|$ indicates a similarity between s_{plg} and s_{src} which is greater than δ.

In this thesis, an external plagiarism detection system based on POS tag n-grams (POSNG) is proposed, which uses both syntactic and semantic features to detect plagiarized sections in the documents. Because POS tags help to analyze and reveal syntactic similarities, the proposed plagiarism detection approach uses *part-of-speech tag n-grams* to determine syntactic similarities between original and suspicious sentences and also quickly access to plagiarism candidates, each of which is the source sentence. In the proposed system, the words and punctuation in each sentence are shown with an annotation called token. Then, these tokens are tagged with their POS tags. Finally, n-grams of these POS tags are generated and the sentences are represented with their part-of-speech tag n-grams. After converting the sentences to the POS tag n-grams representation, the source sentences are indexed by a search engine. Then, a query is generated from the part-of-speech tag n-grams of a suspicious document and this query accesses the source sentences to find the candidate sentences. Search engine results returned from the queries are used to identify plagiarized sections of the suspicious document.

The performance of the proposed system is further improved by using semantic similarities between sentences. The *Word2Vec* model [15] is used to find the similarity degree between the two words semantically. It is a word embedding learning technique for language natural processing used to produce vector representations of words. In this model, word associations are taken from a large corpus and a vector space is generated as output. The score of semantic similarity between words is measured by the cosine similarity calculation of their vectors [16]. Whether there is plagiarism between a suspicious sentence and its candidate source sentence is ultimately decided according to the result calculated by the *Longest Common Subsequence (LCS)* method of the semantic similarity between these sentences.

For this thesis study, a large dataset named PAN Plagiarism Corpus 2011 (PAN-PC-11), which was created to evaluate automatic plagiarism detection

algorithms, was used. In this corpus, paraphrasing generation are divided into two categories as *artificial and simulated*. Artificial plagiarism cases have been automatically generated by a computer program using random text operations, semantic word variations and POS-preserving word shuffling. On the other hand, simulated plagiarism cases have been generated manually using *Amazon Mechanical Turk* platform [17]. The artificial plagiarism cases contain three different obfuscation levels: *none (or very few), low and high*. The corpus also consists of translation plagiarism cases between English-German and English-Spanish. This thesis study focuses on the mono-lingual perspective using English texts and aims to compare the performance of the proposed approach with the detectors participating in the *3rd International Plagiarism Detection Competition (PAN11)* in [51].

1.5. Contributions of the Thesis

The main contributions of this study are:

1. An automatic plagiarism detection system is proposed to determine plagiarized sections of documents. The proposed system is a novel contribution as it uses POS tag n-grams that helps to show syntactic similarity over different text fragments. Especially, in manual or automatic obfuscation strategies performed on suspicious texts to hide plagiarism, mostly words are replaced with synonyms, and extensive modifications are not made regarding the syntactic structure of sentences. Therefore, similarity can be detected based on POS tag n-grams, as it reveals syntactic similarities between two sentences that have the same meaning and one of which is plagiarized, although all the words are different from each other.

2. In this study, a two-step replacement strategy of POS tags is proposed in order to make the search process faster. First, POS tags are replaced with one-character length symbols to reduce size of the POS tags and speed up the candidate selection process. Second, the generated n-grams are sorted in the whole corpus from most used to least used and these n-grams are replaced again with symbols or one, two, or three length

characters, starting with the most used one. Finally, the source documents containing POS tag n-grams are indexed by the search engine to make faster searching operations and to quickly find plagiarism candidate sentences.

3. The representation of candidate sentences which is carried out after the indexing and candidate retrieval steps is one of the main contributions of this study. All of the steps performed up to this representation can be considered data preparation. This preparation process includes text preprocessing, indexing and searching tasks. The text preprocessing task allows the sentences to be converted into the POS tag n-grams structure. Then, the indexing of the source sentences and the searching for the candidates are performed with the capabilities of the search engine. On the other hand, the presentation of the candidates ensures that possible plagiarism is determined in the fastest way without requiring a detailed analysis. Therefore, if the data is prepared up to the list of candidates, the plagiarized sections can be detected with a simple distance-based calculation between two consecutive candidates by means of representation methodology, which is one of the main contributions of this study.

4. The proposed method improves the detection performance of the low and high obfuscation paraphrasing cases even though only POS tag n-grams are used to measure syntactic similarities. The experiments are performed with four types of paraphrasing in the PAN-PC-11 dataset containing different levels of modifications and obfuscation strategies such as none or very few, low, high and simulated. A different number and scope of thresholds and parameters are defined to evaluate the variety of the test results. The performance of the proposed approach is compared with plagiarism systems participating in the PAN11 competition according to various measures. The experimental results demonstrate that the proposed approach in this thesis achieved the best performance in the overall metric called *plagdet* in low and high types of obfuscation.

However, the proposed method obtained competitive results in none and simulated obfuscation types for all four evaluated metrics.

1.6. Thesis Structure

This study consists of seven chapters. The rest of this thesis study is structured as follows:

Chapter 2 Plagiarism Background

This chapter provides general information to the plagiarism approach. The definition, taxonomy and detection techniques of plagiarism are introduced. The types of plagiarism are explained in detail with examples. The advantages and disadvantages of the various plagiarism methods are analyzed. Also, similarity metrics used in many plagiarism algorithms are summarized.

Chapter 3 Literature Review

This chapter gives a summary of the development process of plagiarism detection over time. This chapter also presents an overview of the related studies involved in external plagiarism detection. The methods applied to these studies are given by classifying them. In addition, NLP techniques used in related works are discussed in this section. Following that the approaches of PAN11 detectors are introduced. The methods and contexts of the detectors are summarized.

Chapter 4 Plagiarism Detection Based on POS Tag N-Grams

This chapter gives detailed information about the proposed method. The architecture and steps of the plagiarism detection process and the algorithms are introduced. The step-by-step outputs of the tasks and examples of the operations are shown in this chapter.

Chapter 5 Data Analysis

This chapter introduces the PAN-PC-11 corpus which is used in this study. An overview about the statistics of documents and plagiarism cases in the corpus

are given. Following that obfuscation strategies and evaluation metrics of PAN-PC-11 are explained.

Chapter 6 Experimental Evaluation

In this chapter, the experimental environment and settings are introduced, the results are reported, the performance of the proposed system is evaluated and the execution times of the processes performed in this study are analyzed. In addition, the effects of different threshold values and parameters on the experimental results are discussed.

Chapter 7 Conclusion

This chapter presents a summary and final conclusion about this study with the contributions. Besides, suggestions for the future improvements are discussed.

2. PLAGIARISM BACKGROUND

This chapter provides a detailed overview of the definition of plagiarism. It also presents a survey about the plagiarism types. It introduces the concept of plagiarism detection and similarity metrics.

This chapter is organized as follows: in Section 2.1, the description of plagiarism is explained. Section 2.2 reviews the types of the plagiarism and two main types of plagiarism, textual and source code, are introduced in this section. Section 2.3 presents the plagiarism detection problem and discusses its three tasks: intrinsic, external and cross-lingual plagiarism. Finally, in Section 2.4, similarity metrics used in plagiarism detection studies are introduced.

2.1. Definition of Plagiarism

As stated in [7], plagiarism is described as the use of other person's thoughts, expressions or works without proper citation and presenting them as one's own. Especially in the recent years, plagiarism has become an important ethical problem both in scientific studies and in the business world [65, 116]. With grown of the Internet, fast access to the resources on the web has enhanced the problem of plagiarism. Copying various works from the Internet without proper citation may be considered as plagiarism. In this case, some legal issues such as copyright infringement may arise.

However, from a legal perspective, proving plagiarism may not be an easy process. Plagiarism can only occur when it is proven that words are copied directly or expressed using other words. In some cases, even one-to-one match between two different texts does not prove a plagiarism since both texts may have been written about the same subject. In the circumstances, some information such as the names of person, place or technical terms may have been used in common. In [18], it is reported that independent texts have 50% or more common vocabulary. Therefore, it is necessary to prove that the suspicious text has the same meaning of original text in detecting plagiarism. A long string of matching characters, the same sequence or similar distribution of the words, a similar writing style or the same typos between two texts can be considered as

plagiarism indicator. This problem is not limited to written texts, but is also found in software code that is copied and reused without citation to the original author [19].

Deciding whether to consider a work as plagiarism and automating plagiarism detection is a difficult task. It sometimes can be clearly seen that a certain part of text is a copy from another. However, plagiarists often use some techniques that try to distinguish the plagiarized text from the original. To hide an act of plagiarism, the suspicious text is often rewritten: the order of words in a sentence is changed, words are replaced with synonyms, some words are added/removed or the text can be summarized. These changes make difficult to detect plagiarized text for automated systems. Many automated detection programs use structural and lexical similarities of the documents [20].

It is harder to detect semantic similarity in a text. Especially performing this process with a computer algorithm requires great effort. In recent years, many commercial and academic products have been developed to facilitate the detection of plagiarism. Most of these products can detect verbatim plagiarism, but fail when the works are paraphrased. In paraphrased works, detecting plagiarism requires to identify similarities that go beyond keywords and verbatim overlaps [21].

2.2. Types of Plagiarism

In this section, two main types of plagiarisms, textual and source code plagiarism, are introduced.

2.2.1. Textual Plagiarism

Textual plagiarism is common in education such as student assignments and research publications. Textual plagiarism occurs in various types and in documents written in natural language. These documents may be written in the same or different languages. An example of plagiarism created by copy-paste method from PAN-PC-11 corpus is shown below:

Original sentence:

Plagiarism can **only occur** when it is **proven** that words are **copied directly** or expressed **using** other words. In some **cases**, even one-to-one match between two **different** texts **does not prove a** plagiarism **since** both **texts** may have been written about the same **subject**.

Plagiarized sentence:

Plagiarism can **happen only** when it is **demonstrated** that words are **directly copied** or expressed **utilizing** other words. In some **circumstances**, even a one-to-one match between two **distinctive** texts **doesn't constitute** plagiarism, **as** both **documents** may have been written about the same **topic**.

In the example, it is seen that the plagiarized sentence was created by replacing some of the words in the original sentence. The original words were directly copied into the plagiarized sentence and no proper citation was given. The plagiarized sentence was not written in such a way that what is intended to be conveyed is expressed in the author's own words. In the example, the changed words are highlighted as underlined and bold. It is seen that all the remaining words are the same in both sentences. The syntactic structure of the original sentence was preserved and words were changed with their synonyms. On the other hand, by changing the orders of some words in the original sentence, it was tried to reduce the consecutive matching of words. It is clear that both sentences have similar syntactic structure and there are no creative or intelligent obfuscation strategies.

However, plagiarists sometimes resort to plagiarism by deliberately misspelling words. As in the other example below, bold and underlined words are intentionally misspelled in order to avoid matching the same words in both sentences.

Original sentence:

Neither of them were **their** **friends**.

Plagiarized sentence:

Niether of them were **thier** **frinds**.

Two sentences are compared structurally or semantically to detect plagiarism. As a result of this process, the usage of the same words in both sentences at a certain rate increases the possibility of plagiarism. For this reason, misspelling techniques can be used intentionally as in the example to prevent the same words from being matched with each other.

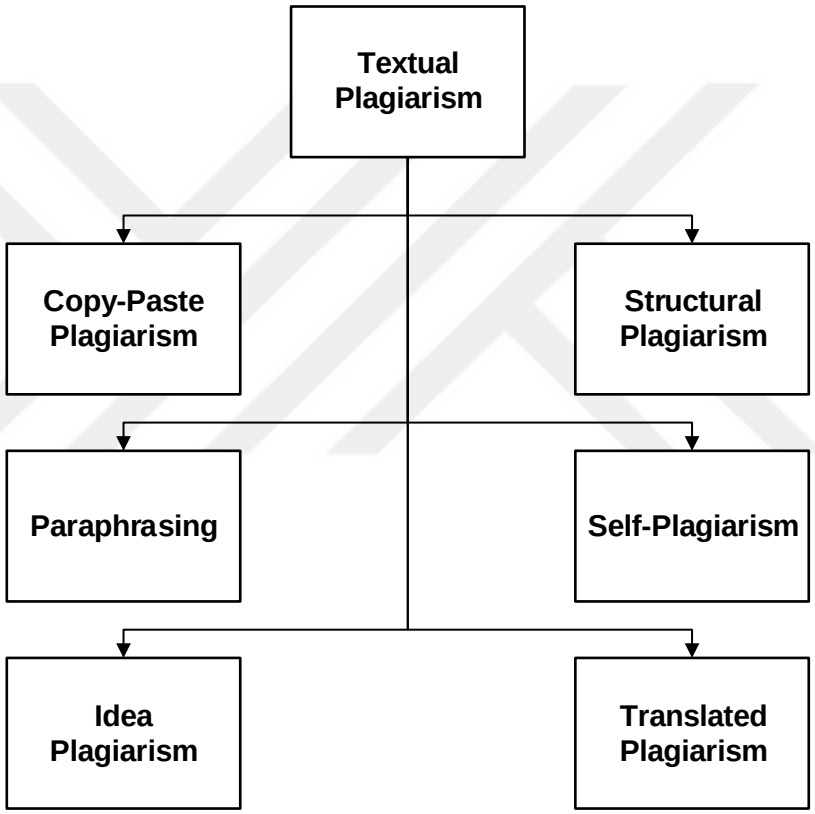

Figure 2.1. Forms of Textual Plagiarism

As can be seen from Figure 2.1, some different forms of textual plagiarism are listed as follows:

1. **Copy-paste Plagiarism:** In this type of plagiarism, sentences or paragraphs from another work are directly copied as your own without citation.

2. **Paraphrasing:** It refers to the presentation of the idea described in another work in different ways by changing the structure of the sentences, changing the words or their order, replacing words with synonyms and making changes in the grammar [22]. Changes made as a result of the paraphrasing action are made without citing a source directly. Paraphrasing includes some serious and intelligent modifications that are hardest to detect by plagiarism detection systems.

3. **Idea plagiarism:** In this type of plagiarism, ideas are taken from other sources and used as your own. Examples include taking someone else's opinion, translating various concepts or expressions, using someone's findings or results without citation or permission [23]. An idea taken from the original text is reused regardless of its original words or form. Given the levels of difficulty in detecting plagiarism, the simplest plagiarism cases to identify are text fragments that copied directly. However, detecting the plagiarism of ideas usually requires semantic analysis as it is a more complex task.

4. **Structural plagiarism:** It is the act of expressing someone else's text with different words by changing the structure of the sentence. It can also be described as the type of plagiarism in which the structure of an original text is copied to another. It also includes changing the grammatical structure of the sentences by translating them [24].

5. **Self-plagiarism:** It means reusing your own previously published work entirely or its some portions without citing it for a new research paper. In this type of plagiarism, the author tries to reconstruct his/her previous work by changing the writing pattern of the old research. Self-plagiarism violates the necessity that the content of an article be original and not previously published [25]. Since authors are the owners of their own works, they think that they can reuse their works in various ways. However, especially because of the possibility of violating a publisher's copyright, self-plagiarism remains ethically debated.

6. **Translated plagiarism:** It is the act of using a work written and published in a different language by being translated into another language without citation. Some cases that have emerged in the recent years have shown that researchers translate a published research and submit it to various journals for publication as their own, without attribution to the original author [25]. As it is more difficult to detect, translated plagiarism has become one of the major problems for automated plagiarism detection tools [26].

Figure 2.2 shows an example of translated plagiarism in [150]. In Figure 2.2, it is seen that a text written in English was plagiarized in Spanish as verbatim. The English text has been translated directly into Spanish without any structural or semantic changes by an automatic translation tool [151]. The original English text was not expressed in the author's own words and an appropriate citation was not made. When the plagiarized Spanish text is translated back into English with the same automatic translation tool, it is easily seen that the two English texts are copy-paste. The same phrases between the two English texts are highlighted in different colors in Figure 2.2. The original English text has been converted with the same sentence structure and words.

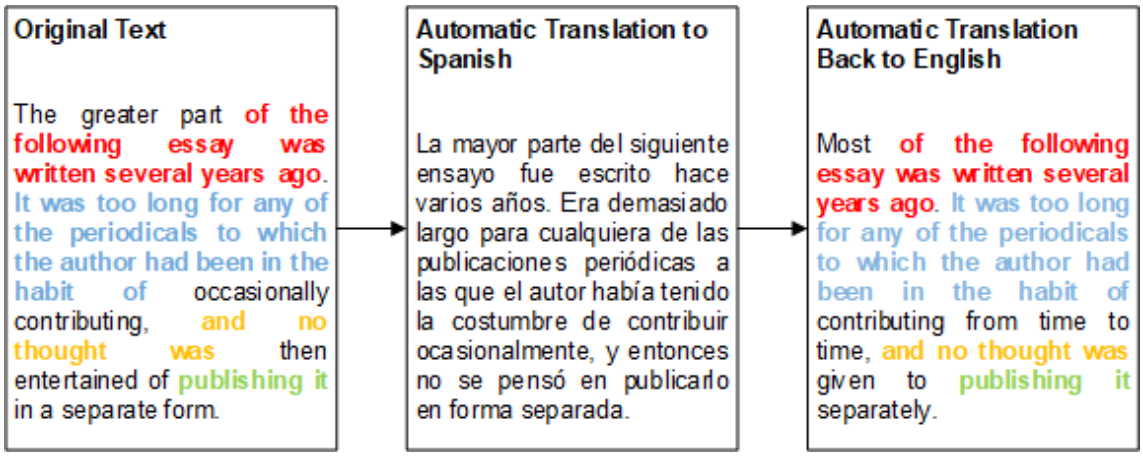

Figure 2.2. An Example of Translated Plagiarism

However, even in the fragments of the texts that are not exactly the same, e.g. (1) *the greater part* and *most*, (2) *occasionally* and *from time to time*, (3) *in a separate form* and *separately*, it is clearly seen that there are words or phrases,

which has same meaning. Although words with the same meaning are expressed differently, the original English sentence and the Spanish translation are exactly the same.

The main problem in textual plagiarism detection is determining whether plagiarism exists and how to measure the degree of similarity. Many researchers have proposed various detection methods for the text similarity problem. These methods are basically divided into two similarity calculation methods. The first is based on statistics and the other one is based on semantic. The method, which is based on statistics, requires a large data set and long training process. The semantic based method, on the other hand, has high precision. However, the scope of sentences or words is limited in this method [27].

2.2.2. Source Code Plagiarism

Plagiarism in coding is a type of plagiarism that is studied and researched in the literature. At the programming level, plagiarism does not only mean copying the source code, but also includes comments within the code, program input data and interface designs [28]. Compared to textual plagiarism, detecting plagiarism in source codes is very difficult but easy to do. For example, when students are given an assignment to write a program, it will be inevitable that similar codes will emerge. While some students write source code of the assignment on their own, other students simply take code from them and make some changes, such as changing variable, method or class names, changing the order of the statements or changing the functions and variables of a class [29]. It is almost impossible to manually compare the program pairs, especially if the number of lines of code is too much. It takes a lot of effort to examine all codes in two different programs and detect plagiarized pairs.

There are also some methods to detect plagiarism in the source codes, just like textual plagiarism. A source-code plagiarism detection system should show which parts of the two programs are similar. Today, most plagiarism detection systems today detect plagiarism largely at the syntactic level. Because a plagiarism detection system does not have a human-like perspective, it may not detect plagiarism at the semantic level, such as data structures and algorithms

[30]. There is not yet a standard reference model for determining whether one of two similar programs is plagiarized from the other. The similarity score of a program group depends on the program set itself. If a fixed similarity threshold value is used to detect suspicious source-code pairs, too many plagiarism or false identification may arise. For this reason, an adaptive threshold that takes into account the similarity distribution of the program set is required to handle plagiarism cases [31].

Original code:

```
1: public Paraphrase removeWord() {
2:    Paraphrase p1;
3:    int ran;
4:    if(result == 1)
5:       return null;
6:    ran = (int) Math.random() * 100);
7:    p1 = phrases.get(ran % total);
8:    result = result + 1;
9:    return ran;
10: }
```

Modified code:

```
1: public Paraphrase deleteWord() {
2:    Paraphrase pp;
3:    int randomNum;
4:    randomNum = (int) Math.random() * 10);
5:    pp = phrases.get(randomNum + w);
6:    w = w - 1;
7:    sum = sum++;
8:    return pp;
9: }
```

Figure 2.3. An Example of Source Code Plagiarism

In the plagiarism process of the source codes, two types of modifications basically are applied: lexical changes and structural changes [32].

1. **Lexical Changes:** These are easy changes that can be made using a text editor without the need for any programming language knowledge.
    - Comments are changed, added or rewritten.
    - The format of the code is changed.
    - The names of variables, functions and identifiers are changed.
2. **Structural Changes:** Structural changes are programming language dependent and require a programming knowledge to make changes to the source code.
    - Loops can be changed.

- Nested if statements and case statements are replaced with each other.
- Procedure and function calls are replaced with each other.
- The order of the operands can be changed.
- New statements can be added that will not affect the output of the program [29].

As can be seen from the source code plagiarism example in Figure 2.3 [149], both lexical and structural changes were made in the modified code. As lexical changes, the function name (*removeWord* and *deleteWord*) and variable names (*p1*, *ran*, *result* and *pp, randomNum sum*) have been changed. Structurally, as can be seen in the 6th line of the modified code, new expression has been added and the writing style of the supplementation operand has been changed in the 7th line so as not to affect the result of the operation. Also, the 4th and 5th lines in the original code were removed in the modified code.

2.3. The Concept of Plagiarism Detection

The main goal of plagiarism detection is to reveal similar information between two documents. Plagiarism detection can be done manually or automatically. Manually detecting plagiarism on text is a difficult process, as text can be interpreted in different ways from person to person. In addition, as the amount of information in the text increases, it becomes more difficult to identify similar sections and is time consuming. In order for teachers or instructors to detect plagiarism manually, they need not only to read every assignment, but also to know all possible sources of plagiarism [33]. For this reason, the use of automatic tools in plagiarism detection has become a necessity and with the developed systems, it has been possible to detect plagiarism cases more quickly and effectively [34]. The aim of an automated plagiarism detection system is to reduce the time spent comparing texts, making it possible to compare multiple documents. This makes it as easy as possible to find possible similar texts by searching a large number of electronic sources. Automatic plagiarism detection

systems should minimize the number of cases that are erroneously classified as plagiarized and cases that are incorrectly classified as non-plagiarized [8].

Plagiarism detection can be classified in two ways [35] as monolingual and cross-lingual. Monolingual perspective deals with finding plagiarized cases between two same languages. Most plagiarism detection systems fall into the monolingual plagiarism category. It can be divided into two according to whether external references are used during plagiarism detection process.

2.3.1. Intrinsic Plagiarism Detection

This detection method focuses on cases where there is no reference data set or external sources. In intrinsic plagiarism detection, the author's writing style, known as stylometry, is analyzed.

Table 2.1. The Research Areas of Intrinsic Plagiarism Detection

Research Area	Description
Intrinsic plagiarism detection	It is the task of identifying the plagiarized sections of a potentially suspicious text without comparing them with the original sources.
Authorship anomaly detection	The task of determining anomalous passages in a textual document.
Multi-author document segmentation	It is the task of detecting which authors wrote each passage in a textual document with unknown authorship when N number of authors is given.
Authorship verification	Given a textual document d and a collection of textual documents D written by an author, it focuses to determine if d is written by this author.
Plagiarism direction identification	It is the task of finding out both original and suspicious passages when two textual documents which share one or more passages are given.
Linear text segmentation	It is the task of detecting the locations in a textual document that cause a change in topic.
Speaker diarization	When audio or video records which consist of unknown number of speakers are given, it aims to determine who spoke when?

Identifying plagiarized parts in a suspicious document relies on detecting irregularities and anomalies in the document [36]. In terms of writing style,

passages that are not compatible with each other in the document can be considered as an indicator of plagiarism. A brief explanation of some research areas [39] related to intrinsic plagiarism detection can be seen in Table 2.1.

There are many research works in the literature on writing style. According to [37], stylometric features generally consist of the following five categories:

- Character-level text statistics,
- Sentence-level syntactic features,
- Part-of-speech features of the word classes,
- Closed-class word sets used for special words,
- Structural features that show the text structure.

On the other hand, [38] proposed the following stylometric properties:

- Character-based types (character levels and character n-grams),
- Lexical-based types (word counts, word n-grams, word relationships etc.),
- Syntactic-based types (POS counts, chunks and phrases, sentence structure etc.),
- Semantic-based types (synonyms replacement, semantic analysis, semantic dependencies, word embeddings etc.),
- Application-specific types (content-specific, structure-specific and language-specific).

2.3.2. External Plagiarism Detection

This type of plagiarism detection aims to find similarities in the suspicious documents by comparing them with a collection of documents. This collection of documents can be consisted of online sources on Internet or offline sources containing the original set of documents [40]. In external plagiarism detection, a

suspicious document is compared to a series of documents. A threshold value is set for calculating the similarity of this suspicious document with other documents. Any document exceeding this similarity threshold is determined to have been plagiarized.

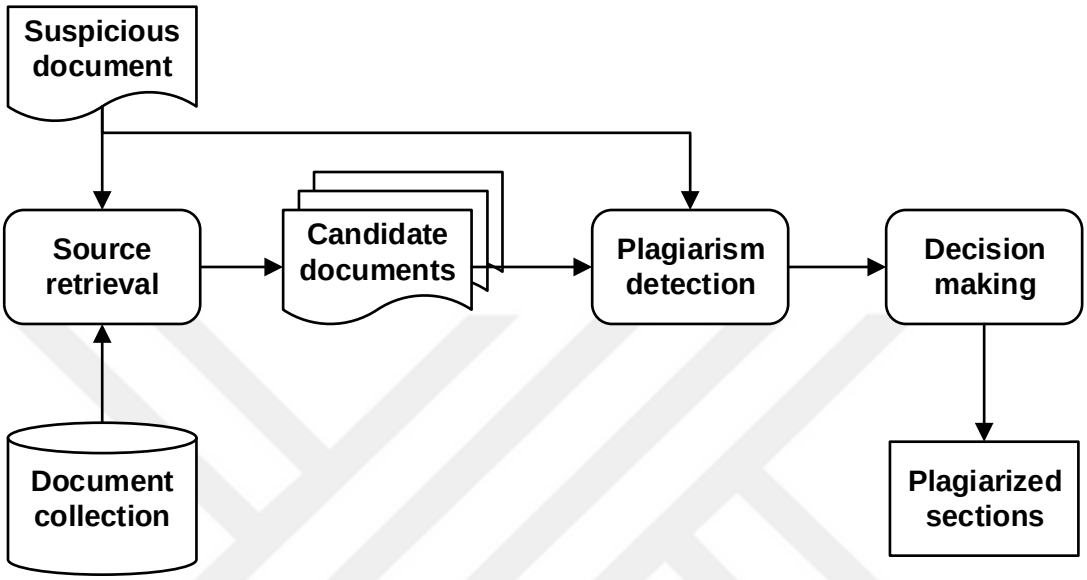

Figure 2.4. General Process of External Plagiarism Detection

The basic flow of external plagiarism detection [41] is shown in Figure 2.4. As can be seen from Figure 2.4, there is a collection of documents consisting of source documents. These source documents are retrieved sequentially from this collection of documents and each source document is pre-processed with the suspicious documents. Many NLP techniques such as removing stops words, stemming, lemmatization, chunking etc. can be used in this step. As a result of this process, candidate documents are produced. Then, a detailed plagiarism detection algorithm is performed between the suspicious document and source candidate documents. Finally, in the decision-making step, also called post-processing, it is decided whether the suspicious document has been plagiarized from the source document, and if plagiarized, the plagiarized sections of both the source and suspicious document are shown.

Algorithm 2.1. The General Mechanism of External Plagiarism Detection

Input:
C_{susp} is suspicious corpus
C_{src} is source corpus
doc_{susp} is suspicious document where $doc_{susp} \in C_{susp}$
doc_{src} is source document where $doc_{src} \in C_{src}$

Variables:
$countSusp$ is the number of documents in C_{susp}
$countSrc$ is the number of documents in C_{src}
$ranking$ is the score of the syntactic similarity
CL is the candidate list of doc_{susp}
$countCL$ is the number of candidate documents in CL
$candidate$ is the candidate document in CL
sim is the similarity score between doc_{susp} and $candidate$
$threshold$ is the threshold value

1: $countSusp \leftarrow$ number of documents in C_{susp}
2: **for** $i = 1$ **to** $countSusp$ **do**
3: $doc_{susp} \leftarrow$ get $(C_{susp(i)})$
4: $countSrc \leftarrow$ number of documents in $\mathrm{C_{src}}$
5: **for** $j = 1$ **to** $countSrc$ **do**
6: $doc_{src} \leftarrow$ get $(C_{src(j)})$
7: **if** candidate retrieval algorithm of doc_{susp} & $doc_{src} > ranking$
8: $CL \leftarrow$ add (doc_{src})
9: **end if**
10: **end**
11: **end**
12: **for** $i = 1$ **to** $countSusp$ **do**
13: $doc_{susp} \leftarrow$ get $(C_{susp(i)})$
14: $countCL \leftarrow$ number of documents in CL
15: **for** $j = 1$ **to** $countCL$ **do**
16: $candidate \leftarrow$ get $(CL_{(j)})$
17: $sim \leftarrow$ get similarity between $(doc_{susp}, candidate)$
18: **if** $sim\ (doc_{susp}, candidate) > threshold$
19: doc_{susp} is plagiarized of $candidate$
20: write plagiarized sections of doc_{susp}
21: **end if**
22: **end**
23: **end**

The basic mechanism and steps of the external plagiarism detection system are given in Algorithm 2.1. As can be seen from the algorithm, plagiarism detection process is carried out using source and suspicious document sets. Source documents represent original texts. One of the main goals in the process is to

find candidate documents to be associated with each suspicious document. Therefore, candidate source documents are identified according to a candidate retrieval algorithm and added to the candidate list for each suspicious document. Thus, it is aimed to reduce the time for post-processing step. Finally, a detailed analysis is performed between the suspicious document and the source candidate document. In this step, source and suspicious documents are compared according to various features of the plagiarism detection algorithm and a similarity calculation is made. If the similarity score between the two documents is greater than a threshold, it is decided that the suspicious document has been plagiarized and plagiarized sections are displayed with the corresponding sections in the original document.

If the size of the reference dataset is large, the number of source texts can be huge. In this case, the total comparison duration between the suspicious text and the source texts can take considerable time. In order to overcome this situation, candidate documents are retrieved as a subset of source texts [42]. Most external plagiarism detection methods involve the location of plagiarized passages within candidate texts. In Chapter 3, some of the external plagiarism detection methods used in detecting suspicious plagiarized texts are introduced in detail.

The use of n-grams provides flexibility to external plagiarism detection task in terms of detecting rewritten fragments of text [43]. Other approaches have focused on the external plagiarism detection problem like a traditional classification problem used in machine learning. For example, the external plagiarism detection system introduced in [44] consists of two stages: documents are indexed in the first stage and similarities are detected in the second stage. This method uses n-grams of words where the value of n varies between 4 and 6 and calculates the number of matches of these n-grams between suspicious documents and source documents to detect plagiarism.

2.3.3. Cross-Lingual Plagiarism Detection

Documents with similar content are also available across different languages: for example, articles written in multiple languages, news about similar events or other translated documents. Determining the similarities of such documents among

different languages is a research area that has developed within the scope of plagiarism in recent years [45].

Cross-lingual plagiarism is focused on detecting similarities between source and suspicious documents in different languages. The correct translation of a document depends on the quality of the translation tool. Therefore, translating documents with poor-quality and limited capability translation tools can result to produce incorrect documents. If there is no an advanced translation tool, a dictionary or other translation sources containing some datasets can also be used to translate the texts [46]. Figure 2.5 shows the general detection process of cross-language plagiarism. First, an algorithm is used to determine the language of the documents in the dataset that written in their original language. Then, these documents are translated into English with a translation algorithm. After this step, a dataset includes of all original and suspicious documents in English is obtained, and all subsequent operations are applied as in mono-lingual plagiarism approach.

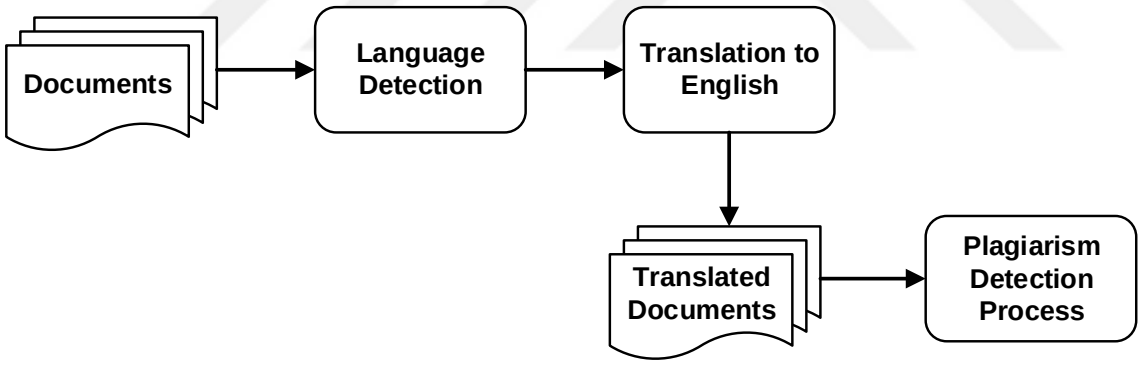

Figure 2.5. Cross-Lingual Plagiarism Detection Process

Existing studies of cross-lingual plagiarism detection have taken advantage of the syntactic and lexical features of writing, statistical dictionaries, or similarities with a multilingual collection of documents. Many of these approaches are designed for verbatim copying, and performance decreases when dealing with high obfuscated plagiarism cases that contain paraphrasing [47]. Two methods can be used to detect cross-lingual plagiarism: first, the cross-lingual similarity method, and second, the monolingual similarity by translating the document into

the other language. Four main types of cross-language similarity assessment models [13, 48] have been proposed in the literature: (a) syntactic-based model, (b) semantic-based model, (c) dictionary-based model and finally, (d) corpus-based model.

2.4. Similarity Metrics

The basic process to detect plagiarism is to measure similarity. In this context, it is necessary to reveal and calculate the similarity between two documents or text fragments. It is seen that two basic similarity metrics, string-based and vector-based, have been used in the studies carried out so far on plagiarism detection.

2.4.1. String Similarity Metrics

These methods are also called edit-based measures and mostly used by external plagiarism detection algorithms to calculate approximate string matching. For example, Hamming distance is a similarity measure that estimates the number of distinct smallest units between two data of equal length [22]. Let n and m be two strings, Levenshtein distance defines the minimum edit distance that converts the n to m [69]. Another commonly used method, Longest Common Sequence (LCS), measures the length of the longest character pair between n and m, according to the order of the characters [70].

2.4.2. Vector Similarity Metrics

Vector-based similarity metrics are also called as token-based measures. Many vector-based similarity metrics have been introduced in recent studies. One of them, matching coefficient was used to calculate the similarity between two vectors of equal length [112]. On the other hand, the Jaccard coefficient defines the number of common elements versus the total number of elements between two exactly the same sets. [113].

The Dice coefficient is almost identical with the Jaccard coefficient, but the number of terms common in the dice measure is reduced [57]. The Overlap coefficient focuses on the match between subsets and calculates the similarity

between these subsets [114], while the Cosine similarity is used to compute the cosine angle between two or more vectors [115].

Table 2.2. Comparison of Similarity Metrics

Metrics	**Type**	**Time Complexity**	**Space Complexity**
Exact matching	Character comparison	$O(min(m,n))$	$O(1)$
Hamming	Edit-based	$O(max(m,n))$	$O(1)$
Levenshtein	Edit-based	$O(mn)$	$O(n^m)$
Longest common subsequence	Edit-based	$O(mn)$	$O(mn)$
N-gram	Token-based	$O(m+n)$	$O(m+n)$
Jaccard	Token-based	$O(m+n)$	$O(1)$
Dice	Token-based	$O(m+n)$	$O(1)$
Cosine	Token-based	$O(m^n)$	$O(m+n)$

Table 2.2 shows the comparison of time and space complexities of string similarity metrics and vector-based similarity metrics [117, 118]. As can be seen from Table 2.2, m denotes "number of terms/characters" and n denotes "size of the dataset/document".

In this thesis study, syntactic and semantic features were used to detect plagiarism cases. The determination of syntactic features was implemented by creating POS tag n-grams. Candidate sentence retrieval was performed with the exact string matching of POS tag n-grams. The semantic similarity between the source and suspicious sentences was calculated with the Longest Common Subsequence (LCS) algorithm.

3. LITERATURE REVIEW

Many different methods have been proposed in the literature on plagiarism detection so far. In this regard, plagiarism detection continues to develop as an active research area of computer science today. Section 3.1 provides a historical summary of the development process for plagiarism detection. Section 3.2 presents a summary of the related works performed with different methodologies for external plagiarism detection. In Section 3.3, a brief overview is provided about the detectors participating in PAN11.

3.1. Development Route of Plagiarism Detection

Many techniques for plagiarism detection have been developed over time. In this section, a brief summary of the major techniques and approaches developed for plagiarism detection from the past to the now is presented.

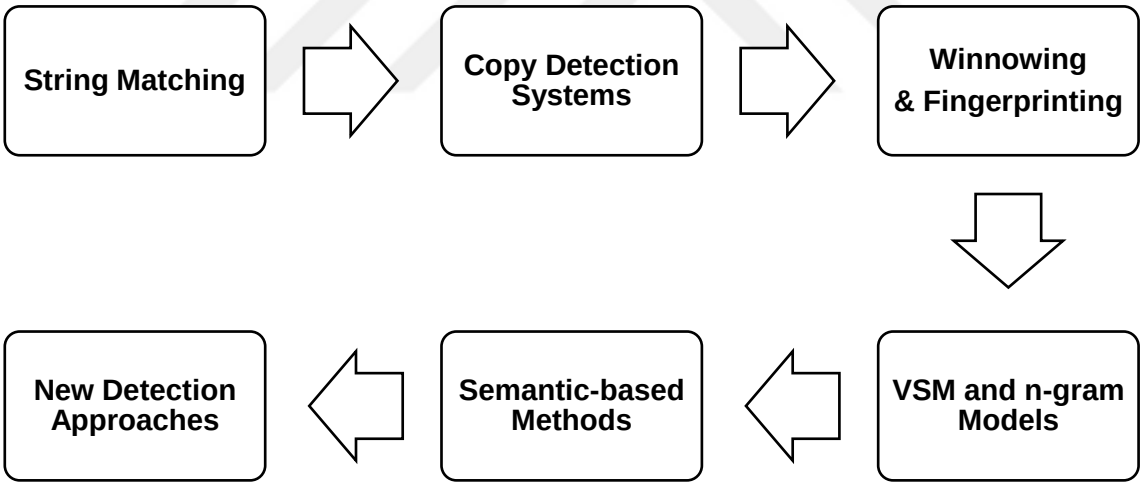

Figure 3.1. Development Route of Plagiarism Detection Techniques

As can be seen in Figure 3.1, one of the first methods used in the past years to detect plagiarism is string matching. In [138], two methods such as overlapping n-gram and window sliding-based was used for the string matching approach. String matching algorithms use the grammatical features documents as they obtain better results in detecting word-to-word plagiarism.

In the following years, copy detection systems based on natural language processing techniques have been developed. The general features of the copy detection systems developed between 1993 and 2001 are given in Table 3.1 [148]. As can be seen from Table 3.1, copy detection systems mostly used the string matching technique. However, character or word length was commonly preferred as the text chunking. Since the choice of text blocks is very big, copy detection systems don't choose the chunk or passages in many cases. This situation makes it hard to define some part of plagiarism comparatively. Selecting a very small chunk of text can cause some problems too. Because this can lead to incorrect evaluation and enhance the number of calculation. Word frequency-based methods focus to semantic attributes more, although they don't deal with deep semantic [148].

Table 3.1. Copy Detection Systems

System	Year	Technique	Similarity Algorithm	Text Chunk
Sif	1993	String matching	Number of common fingerprints	50 bytes after anchor
COPS	1995	String matching	Matching ratio of fingerprints	Sentence
SCAM	1995	Word frequency	RFM (Recency, Frequency, Monetary)	Word
YAP3	1996	RKR-GST, longest matching string	Matching ratio	
KOALA	1996	String matching	Matching ratio of fingerprints	20 characters
CHECK	1997	Key words	Cosine function and matching ratio of section	Word and variable granularity
Shingling	1997	String matching	Matching ratio of fingerprints	10 words
MDR	2000	Suffix tree, longest matching string	Matching ratio	60 characters
CDSDG	2001	Key words	Overlap of semantic and overlap structure	Word and variable granularity

Fingerprinting is one of the other methods developed to detect plagiarism. It defines two different approaches which are the fingerprint algorithm and the winnowing algorithm. The fingerprint algorithm is performed using a hash function to n-grams of a document. The essence of the Winnowing approach is to

determine the smallest fingerprint value and compute the sample fingerprint matching ratio to get the similarity of two passages [148]. Detailed explanations of fingerprint and winnowing algorithms are provided in Section 0.

VSM (Vector Space Model) and n-gram models are one of the most used methods for plagiarism detection. As can be seen in Figure 3.2, the VSM method is often used to represent every text in a textual corpus as a vector. The numbers are in the matrix are frequency of words in the document. VSM consists of two different frequencies such as word (TF) and inverse text (IDF). The VSM method uses word frequencies to attain the properties vector and thus calculates the similarity between two or more documents with the cosine degree [148].

	doc1	doc2	doc3	doc4	doc5
Keyword1	4	0	2	2	1
Keyword2	1	0	3	2	0
Keyword3	1	5	0	1	0
……..……..					
……..……..					

Figure 3.2. The Representation of Word and Document Matrix

In the earlier days, Cosine, Jaccard, Dot Product, Dice and other coefficients were used as the basis for calculating similarity. Then, semantic-based detection tools began to be developed to calculate semantic similarity. These systems intended to used VSM based algorithms. However, VSM is basically not very sufficient in analyzing the semantic on the texts in depth. Types of plagiarism, containing paraphrasing and rewording, are difficult to identify. Many researchers first tried to compute semantic similarity between words or passages through dictionaries such as WordNet to overcome these issues. Afterwards, machine learning algorithms such as Support Vector Machine (SVM) and Naïve Bayes and other similarity calculation methods such as fingerprinting, word similarity, latent semantic analysis started to be used [148, 165]. In 2007, a method was

proposed based on artificial neural networks in [143]. Although this method focuses on analyzing deep learning algorithms of neural networks, it also uses string matching technique.

In the following years, studies on the improvement of candidate retrieval algorithms became popular. In [144], the K-means algorithm was used to carry out the operations related to finding candidates. The K-means approach produces encouraging results, although it struggles with heavily modified data. Though, K-means clustering generates non-overlapping K sets of document properties and therefore isn't able to distinguish text boundaries effortlessly. Fuzzy C Means Clustering method has been proposed in [145] to solve this non-recognition problem. Another topic that has been studied a lot in recent years is word embedding. In [146], a new method with dispensed notation is proposed to calculate word similarity using Word2Vec and LCS. In [147], a method that combines word embeddings with Jaccard index is proposed to compute the lexical similarity. In this method, the proposed system runs speedy because it doesn't deal with the techniques such as stemming and POS tagging applied in the preprocessing step.

3.2. Related Works on External Plagiarism Detection

Automatic plagiarism detection is considered as a series of actions that involve associating a suspicious text with plagiarized passages of original text. One of the biggest challenges in this task is to discover relevant portions of the original text that have been plagiarized by various obfuscation methods [53]. Different techniques and approaches regarding the external plagiarism detection system have been presented by researchers to date. In this context, there are many studies using lexical, syntactic and semantic similarity methods.

As can be seen in Table 3.2 [22, 49, 50], plagiarism detection approaches are analyzed and classified according to various criteria. These methods are categorized by the type of plagiarism, whether it is mono-lingual or cross-lingual and by plagiarism class. The different methods proposed for plagiarism detection are classified according to the basic function on which the plagiarism detection algorithm is based. These functions include different techniques such as string

matching, syntactic, semantic, grammatical or structural features. As mentioned in Section 2.3, plagiarism detection is categorized into two different types as external and intrinsic. As can be seen from the Table 3.2 , the stylometric-based method is only of the type of intrinsic plagiarism detection. All remaining methods fall under the category of external plagiarism. In addition, other methods except cross-lingual fulfill the task of detecting mono-lingual plagiarism. On the other hand, plagiarism cases consist of easily detectable verbatim plagiarism and intelligent cases with more complex modifications on texts. The semantic-based, citation-based and fuzzy-based methods are focused on detecting both literal and intelligent plagiarism cases.

Table 3.2. Plagiarism Methods in Different Criteria

Method	**Type**	**Language**	**Plagiarism Classification**
Character-Based	External	Mono-Lingual	Literal
Syntactic-Based	External	Mono-Lingual	Literal
Semantic-Based	External	Mono-Lingual	Literal/Intelligent
Grammar-Based	External	Mono-Lingual	Literal
Citation-Based	External	Mono-Lingual	Literal/Intelligent
Vector-Based	External	Mono-Lingual	Literal
Structure-Based	External	Mono-Lingual	Literal
Cluster-Based	External	Mono-Lingual	Literal
Fuzzy-Based	External	Mono-Lingual	Literal/Intelligent
Stylometric-Based	Intrinsic	Mono-Lingual	Literal
Cross-Lingual	External	Cross-Lingual	Literal

In [51], three main steps are defined for the external plagiarism detection task: (1) candidate documents retrieval, (2) detailed plagiarism analysis, (3) post-processing. Let d_{susp} be a suspicious document, d_{src} be a source document and D be a collection of source documents where $d_{src} \in D$. First, in the candidate retrieval step, a set of candidate documents denoted as d_{src}^{*} is retrieved from D for each d_{susp}. d_{src}^{*} includes the most relevant candidate documents for d_{susp}. The purpose of this step is to minimize the number of candidate source documents to be compared with d_{susp}. In the detailed analysis step, d_{susp} is compared to each document in d_{src}^{*} to find out if plagiarism has occurred. If there is a certain similarity higher than predefined threshold value between a section of

d_{susp} and a section of d_{src}, then these sections are considered as a potential plagiarism case. In the final post-processing step, all candidate pairs are re-analyzed and the ones that are not sufficiently similar are ignored.

Plagiarism detection systems in the literature have used lexical, syntactic and semantic features to measure similarities between source and suspicious documents and to identify plagiarized sections. In this study, POS tag n-grams have been used to determine syntactic similarities between sentences. They have also been used to index source document sentences so that candidate sentences can be rapidly accessed by a search engine. Different techniques and methods for external plagiarism detection have been proposed in the literature. In the rest of this section, an overview of some existing methods are introduced.

3.2.1. Character-Based Methods

Many plagiarism detection methods fall into this category [22]. These methods work as a string matching technique, based on different characters such as n-gram or word n-gram methods using syntactical features. It is the most widely used and well-known method by many researchers to reveal the degree of matching between different strings [54, 55]. These methods are classified based on the length of different features such as character, word or n-grams. In this method, comparisons are made at character n-gram or word n-gram level. Similarity between two or more documents can be predicted using both an exact match and an approximate match. In an exact match, each letter or word must match in the same order with the corresponding letter or word. Current researches show that most detection systems are developed using exact string matching based on word n-grams [22].

Two sorted word-based n-gram approaches such as 3-grams and 1-skip-3-grams with different length of n are proposed in [56]. As the complexity of paraphrasing increased, the performance of accuracy decreased. In [57], a three-step character-based method is proposed. First, non-overlapping 250-character chunks are extracted, second, the word-based similarity score is calculated using the dice measure. Finally, a threshold value is applied to determine plagiarized segments. The performance of this method is weak on account of the very low

recall value. Different n-gram methods including most commonly used n-grams, named entity-based n-grams and several lengths stop word-based n-grams were applied in [58]. A graph clustering algorithm is used to identify shared fingerprints or clusters of n-grams. The authors proposed a new n-gram method implemented with the Rabin-Karp algorithm based on string matching in [59]. On the other hand, 16-gram character matching was used in [60] and 8-gram word matching was used in [61]. In addition, the researchers also used the string matching technique. The string matching technique is based on the degree of similarity between two strings. For this pulrpose, various proximity measures such as string similarity metric or vector similarity metric are used.

Table 3.3. An Example of the Generation of Stop Words n-grams

Representation	Output
(1) Text passage	These savage birds **are** very common **in** Maine, where **they** make great havoc among **the** flocks **of** wild-ducks **and** Canada grouse, **and will** even, where driven **by** hunger, venture **an** attack **on the** fowls **of the** farm-yard.
(2) The text after removing all tokens except stop words.	are in they the of and and will by an on the of the
(3) Stop words 8-grams of the text.	[are, in, they, the, of, and, and, will] [in, they, the, of, and, and, will, by] [they, the, of, and, and, will, by, an] [the, of, and, and, will, by, an, on] [of, and, and, will, by, an, on, the] [and, and, will, by, an, on, the, of] [and, will, by, an, on, the, of, the]

A novel method [52] based on stop word n-grams (SWNG) has been proposed to detect plagiarized passages in the document collections. The author reported that stop word sequences reveal syntactic clues in document structure. As can be seen in Table 3.3 , the system removes all but stop words from sentences and generates n-grams of remaining stop words. Then, a portion of the source

documents is determined to find the common stop word n-grams of the source and suspicious documents in the candidate retrieval process. In the next step, a detailed analysis is done to predict the plagiarized passages. Finally, in the post-processing step, the similarity score between all source and suspicious passage pairs is sequentially calculated to verify the plagiarism detected. The approach proposed in that study has some limitations in the plagiarism detection process.

Plagiarism becomes more difficult to detect when there are more than one plagiarized passages in a suspicious document, as the proposed work is mainly based on syntactic structure. However, short passages in source and suspicious documents less than a certain number of words were not included in the plagiarism detection process and not reported as plagiarism cases. To overcome the limitations of SWNG approach, the plagiarism detection process in this thesis focuses on both syntactic and semantic features. In addition, short sentences are taken into account and included in the plagiarism detection algorithm. The proposed method in this thesis calculates the semantic similarity to solve the problem of multiple plagiarized passages in the suspicious documents. Thus, sentences with the same syntactic structure but different contexts are eliminated and excluded from the plagiarism detection process.

Table 3.4. Detection Abilities of Character-Based Methods

Character-Based Methods	**Copy-Paste**	**Low Paraphrasing**	**High Paraphrasing**
Exact String Matching	Good	Poor	Unfit
Approximate String Matching	Good	Poor	Unfit
Fingerprinting	Good	Poor	Unfit
Vector Space Models	Good	Poor	Unfit
Semantic Enhancements	Good	Poor	Unfit

Fingerprinting is another character-based method used to detect plagiarism. Fingerprinting algorithms check some portions of a document for plagiarism. The checked portions proceed according to a certain technique, such as character matching, to determine plagiarism. It has two different approaches, which are (1) the fingerprint algorithm and (2) the winnowing algorithm [36, 63, 64]. An illustration [163] of fingerprint algorithm and winnowing algorithm can be seen in

Figure 3.3. The fingerprint algorithm is performed using a hash function to n-grams of a document. N-gram is a substring of 'n' length of the tokens in the document. On the other hand, the Winnowing algorithm is created by adding the *window concept* to the fingerprint algorithm. The hash value selected from this window is the minimum fingerprint value.

Fingerprinting:	**Winnowing:**
A do run run run, a do run run (a. Some text)	A do run run run, a do run run (a. Some text)
adorunrunrunadorunrun (b. The text with irrelevant features removed)	adorunrunrunadorunrun (b. The text with irrelevant features removed)
adoru dorun orunr runru unrun nrunr runru unrun nruna runad unado nador adoru dorun orunr runru unrun (c. The sequence of 5-grams derived from the text)	adoru dorun orunr runru unrun nrunr runru unrun nruna runad unado nador adoru dorun orunr runru unrun (c. The sequence of 5-grams derived from the text)
77 72 42 17 98 50 17 98 8 88 67 39 77 72 42 17 98 (d. A hypothetical sequence of hashes of the 5-grams)	77 74 42 17 98 50 17 98 8 88 67 39 77 74 42 17 98 (d. A hypothetical sequence of hashes of the 5-grams)
72 8 88 72 (e. The sequence of hashes selected using 0 mod 4)	(77, 74, 42, 17) (74, 42, 17, 98) (42, 17, 98, 50) (17, 98, 50, 17) (98, 50, 17, 98) (50, 17, 98, 8) (17, 98, 8, 88) (98, 8, 88, 67) (8, 88, 67, 39) (88, 67, 39, 77) (67, 39, 77, 74) (39, 77, 74, 42) (77, 74, 42, 17) (74, 42, 17, 98) (e. Windows of hashes of length 4)
	17 17 8 39 17 (f. Fingerprints selected by winnowing)
	[17,3] [17,6] [8,8] [39,11] [17,15] g. Fingerprints paired with 0-base positional information)

Figure 3.3. Illustration of Fingerprinting and Winnowing Algorithms

As can be seen in Table 3.4 [62], all these proposed plagiarism detection approaches have been able to effectively detect cases of simple copy-paste and paraphrasing with minor modifications. However, the performance of these systems in detecting high-complexity plagiarism cases has decreased. More efficient results can be obtained when a hybrid system is created by supporting these methods with other approaches. All character-based methods can detect copy-paste plagiarism easily. However, it is equally inconvenient to detect changes made as the paraphrasing complexity level increases.

3.2.2. Syntactic-Based Methods

In these approaches, syntactic-level document parts are extracted as different kind of process. The output of these extractions can be sentences, phrases, chunks or part-of-speech tags (POS). These methods take advantage of syntactic properties such as POS of sentences and implement POS tags like verbs, nouns, adjectives in a document to detect plagiarism [22, 68]. Chunking and POS tagging provide syntactic information and make it easier to find deeper modifications within a document. Parsing trees of the document are created and the related expressions are extracted by the chunking technique. In POS tagging process, any token that can be word, punctuation or symbol is labelled with its tag identifier, which facilitates in more meaningful comparisons [40].

A plagiarism detection system with TF-ISF (term frequency-inverse sentence frequency) weighted and POS tagging was proposed in [67]. It has been stated that the POS tagging approach outperforms better precision in their system. This is due to the proposed system only compares words with the same tag. Therefore, it uses syntactic knowledge to eliminate false detections. The authors in [69, 70] used POS tag features and similarity metrics to analyze and calculate similarity between two documents. The authors [54] used POS tags to represent a text structure for the operations of comparison and analysis. In their study, documents including the same POS tags specifications are processed for more identification of plagiarism.

An external plagiarism detection system including a combined set of syntactic and semantic features have been proposed in [65]. In this system, syntactic

similarities between sentences are based on word-order similarities. On the other hand, semantic similarities between sentences are based on similarities of their semantic vectors. The authors implemented sentence-based approach in the preprocessing step too. In this thesis, the original forms of the words were used without transforming them into root forms. Stop words have also been remained in the documents to keep the syntactic structure of the sentences. Similarities between sentences are determined using an integrated approach that included syntactic and semantic features. Detailed comparison is made using words different from the sentence pair. Later, the authors [66] also proposed the usage of semantic role labeling to measure similarities.

In [165], the authors proposed a plagiarism detection method based on syntactic parsing to match the source and suspicious paragraphs. The aim of the proposed method is to parse the suspicious documents and obtain a word list that has the same meaning with them while considering the POS tags of the words. The proposed system has two main steps such as pattern analysis and similarity measurement. In pattern analysis step, each sentence is parsed to their words and then, each word is tagged with its POS tag. POS tags are converted to metadata format, which represents a paragraph. In similarity measurement step, words of each suspicious paragraph are compared with each paragraph in the original database. The similarity between the paragraphs is calculated with Jaccard Coefficient. The comparison process is performed depending on whether the same word or similar word is matched with the same POS.

3.2.3. Semantic-Based Methods

Semantic-based methods focus on the semantic representation of a document and to identify paraphrasing that has the same meaning with the original text. Various approaches have been proposed over time, using techniques such as semantic role labelling (SRL) and machine learning, which are included in this category [40]. In this method, similarity is calculated by semantically comparing two different words in a text. Semantic-based methods mostly use some libraries and technologies such as WordNet, semantic webs and other thesaurus [44-46]. For example, Resnik [76] used WordNet to calculate semantic similarity. A sentence commonly consists of a set of ordered words. Two sentences can have

the same meaning, although the order of their words is used differently. For example, even if a sentence can be changed from active voice to passive voice and vice versa, its meaning may not change. In such cases, WordNet is used to measure semantic relatedness or similarity degree between two textual information [22]. The use of such methods is limited because it is difficult to find a measure of semantic similarity for sentences [24].

As examples of other methods, the authors [71] have been proposed the SRL-based method that performs detailed analysis of a document semantically using role labelling. On the other hand, Kalleberg [20] used seed classification with various similarity scores such as cosine and dice to detect plagiarized sections of the text in a thesis study. Ceska [72] has proposed a semantic-based plagiarism detection system that uses singular value decomposition (SVD), while Sahu [73] presented a plagiarism detection system based on the k-Nearest Neighbor (k-NN) algorithm to cluster strings and detect matches with neighbor words. The grade of semantic similarity between a pair of words used in knowledge-based measurements was computed using various attributes and information from a dictionary in the study of [75]. In another approach, the authors [77] proposed a method that counts the number of nodes of the shortest path between two texts and specifies semantic similarity based on this technique. A semantic and syntactic based method using POS tags and Latent Dirichlet Allocation (LDA) has been proposed in [74]. In this study, it is focused on determining topics in the sentences and semantic similarities between the sentences using the LDA model. The sentences are syntactically modified and each sentence is converted into a POS tag array. Then, this array is assigned to a topic using LDA. In the next step, topic similarities between the sentences are used to identify plagiarism cases. Plagiarism is inferred based on whether the two sentences are syntactically related to each other. Finally, the plagiarized sentences are retrieved depending on their topics and degrees of syntactic similarity.

Syntactic and semantic based methods are computationally expensive. However, they have made significant improvement in the performance of detecting complicated modifications in the documents [40].

3.2.4. Grammar-Based Methods

These methods use natural language processing techniques to detect plagiarism. Therefore, they can effectively detect verbatim plagiarism and paraphrasing actions. A semantic-based approach generally cannot identify the position of the plagiarized parts in a document. Grammar-based methods eliminate this limitation of the semantic-based method and can solve this problem efficiently [35, 78].

In grammar-based methods, a similarity calculation is made for plagiarism detection by using string-matching approach between original and suspicious documents. These methods can easily detect copy-paste plagiarism, but fails to detect idea or intelligent plagiarism such as paraphrasing [79].

3.2.5. Citation-Based Methods

These methods involve in deep analysis of a document depending on the references cited. Citations are mostly used within scientific publications. In these methods, as can be seen from Figure 3.4 [12], citation patterns are analyzed to identify plagiarism. These methods are semantically related to plagiarism detection methods, as they utilize the semantics of the citation in a document. Similarity is calculated after analyzing similar patterns in citation sequences [87]. Several approaches such as citation order analysis (COA) and bibliographic coupling are used to detect plagiarism [80, 81].

A citation-based plagiarism detection system called CitePlag has been proposed in [82]. In this system, detection process that examine citation series of scientific documents are used to identify similar patterns. The authors [83] used citation evidences along with structural detection to identify plagiarism in their study. In another approach, the authors [84] used text-based citation analysis to discover verbatim plagiarism in academic papers from different websites in the field of NLP. In this study, different types of plagiarism such as reusing, paraphrasing, self-plagiarism and self-reuse is detected.

The overall detection performances of textual-based and citation-based methods according to various plagiarism types are compared in Table 3.5 [88]. Textual-

based methods can detect copy-paste plagiarism even for short fragments. However, the detection performance of textual-based methods decreases for more complex cases that are intelligently hidden in the text. It can detect some of the translation plagiarism cases. Citation-based methods, on the other hand, are capable of detecting various cases of paraphrasing and some of the structural changes made to the text.

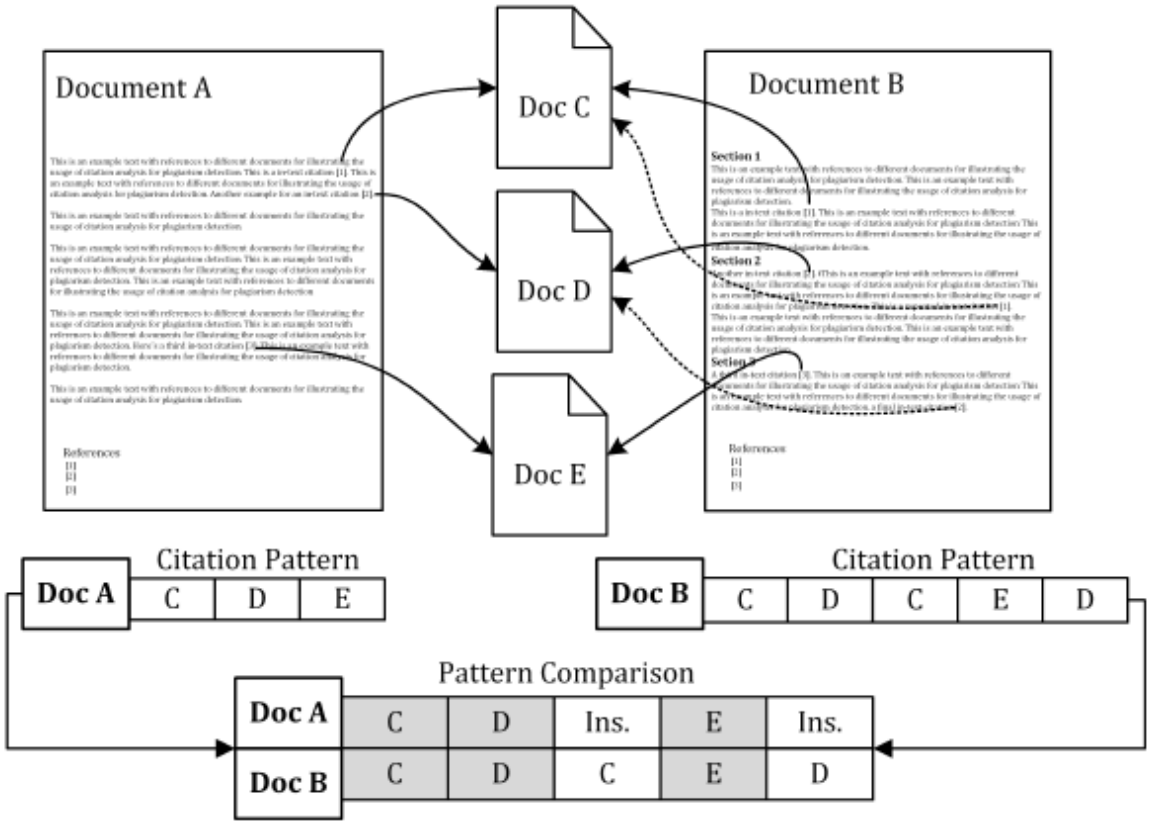

Figure 3.4. General Concept of Citation-Based Plagiarism Detection

In [85, 86], semantic and citation-based plagiarism detection methods are proposed for the usage of semantics in the cited documents. Because these methods utilize the semantic concept located in the citations, they search and analyze the same pair of documents based on the citation. The authors [81] proposed a new approach for detecting plagiarism on a citation basis. It is aimed to detect plagiarism in scientific documents that are read but not cited in this approach.

Table 3.5. Comparison of Detecting Abilities

Plagiarism Type	Textual-Based	Citation-Based
Copy-paste	Good results even for short fragments.	Unsuitable, short fragments cannot be detected.
Disguised Plagiarism	Some cases can be identified.	It depends on the length of the fragments.
Idea/Structural Plagiarism	Unsuitable, fails in intelligent plagiarism.	Some cases can be identified.
Translated Plagiarism	Some cases can be identified.	Some cases can be identified.

The authors [89] propose a new approach about the plagiarism detection. A more detailed version of this approach [80] was later published. This study is a citation-based plagiarism detection and analyzes a document in citation order rather than the text itself. Different algorithms such as the longest common citation sequence of two documents or citation chunking are compared based on a sequential pattern analysis. This method is language independent and significantly reduces the complexity of comparing the entire document by focusing on the reference list. The authors evaluated the performance of their proposed method on the PubMed Central Open Access Subset dataset. This dataset only consists of medical publications and 185,170 documents. The authors conducted a user study which people in different positions (undergraduate, graduate, specialists) ranked their documents with plagiarism scores. Then, these rankings are compared to a plagiarism ranking resulting from different algorithms. This study also includes a comparison with character-based approaches. The authors reported that their proposed method outperformed better results than character-based plagiarism detection in cases of structural and idea plagiarism as well as paraphrasing. However, their approach did not produce successful results in the copy-paste plagiarism.

3.2.6. Vector-Based Models

These methods implement lexical and syntactic properties as tokens rather than strings [76]. In these methods, similarity is calculated through vector similarity coefficients. For example, the word n-gram is represented as a vector of n expressions or tokens and similarity can then be computed using the

corresponding, Cosine, Dice, Jaccard, Euclidean or Manhattan coefficients [35]. It has been stated by the authors [98] that the Cosine coefficient is used in cases where partial plagiarism detection will be performed without sharing the content of a document. The Cosine coefficient is therefore useful for detecting plagiarized passages in texts whose submission is took into account concealed.

Vector-based methods use syntactic and lexical properties and indicate the documents in the vector space. For the document comparison and representation process, various weighting schemes are used. According to the frequency of terms within a document or sentence, two commonly used weighting schemes are TF-IDF and TF-ISF. While TF-IDF is used for both retrieving candidates and detailed analysis steps, TF-ISF is mostly performed for detailed analysis [40].

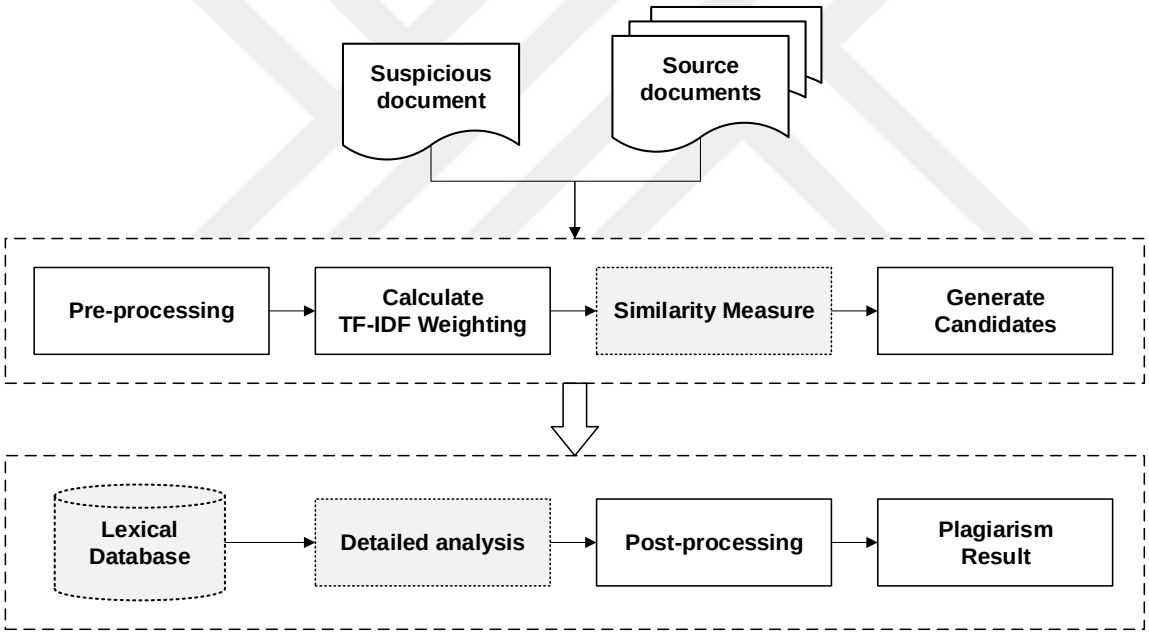

Figure 3.5. General Plagiarism Process of TF-IDF with Similarity Measure

Figure 3.*5* shows the general working principle of vector-based plagiarism detection using TF-IDF weighting scheme [162]. As can be seen from Figure 3.*5*, first, suspicious document and source documents are pre-processed. Then, the representation of documents or sentences is done with the TF-IDF weighting scheme and the candidate retrieval step is performed using a similarity metric. A lexical database such as WordNet is used for detailed analysis. After the detailed

analysis and post-processing step, whether there is plagiarism in the suspicious document or not is shown as a result.

A TF-IDF weighted VSM model was proposed by the authors [94] for both candidate retrieval and detailed analysis, and cosine similarity was used for document comparisons. A TF-ISF weighting scheme is proposed for the detailed analysis step using dice and cosine similarity measures in [95]. In another approach, Vani and Gupta [67] use both TF-ISF weights and POS tags to determine plagiarism in the sentences. The authors investigated the impact of several similarity measures to decide detection efficiency as well. The authors proposed an approach based on query formulation for retrieval of plagiarism in the original texts using the TF-IDF weighting scheme in [96]. The first five keywords are used to formulate of the queries. Then, sorting operation is performed according to the TF-IDF weighting values of the words in the plagiarized passages. A method combining TF-IDF with TF-IDF weights and extracts key terms from the suspicious document as query, is proposed in [97] to retrieve the plagiarized original document.

The authors [90] proposed an automatic plagiarism detection system for obfuscated texts. The proposed method is based on the SVM classifier, which uses various syntactic, lexical and semantic features to detect plagiarism. The system consists of four main steps: In the paragraph-level comparison step, first, the text is preprocessed and stop words and punctuations are removed. Then tokenization task is performed. Finally, suspicious and source documents are compared at paragraph level. In the second step, sentence-level comparison, the sentences are compared based on the number of common unigrams between them. Then, the detected plagiarism cases are further examined by the SVM classifier in the third step. Finally, in the post-processing step, the consecutive detected text fragments are combined and the shorter ones are removed.

In [91], a sentence-based comparison approach has been proposed to detect plagiarism between source and suspicious documents using text embedding vectors. Word vectors are merged with an aggregation procedure to indicate a textual document. This notation contains syntactic and semantical information to provide efficient text alignment between source and suspicious documents. This

approach includes two major stages: In the first stage, two sentences are nominated as candidate seeds. Each vector including the sentence representation in the plagiarized text is compared with the vectors of the original text using the cosine similarity measure. Then, a set of variables containing Jaccard coefficient and combining the thresholds are set to filter and combine these candidate seeds. In the second stage, each candidate sentences are filtered by the maximum number of spaces and characters between two sentences to detect the correct plagiarism cases. The authors evaluated the proposed method separately for English, Persian and Arabic languages.

The authors [92] proposed a document-level plagiarism detection system based on VSM. The aim of this approach is to analyze the strength of syntactic features extracted using shallow NLP techniques such as POS tags and chunks. The authors applied the techniques of lemmatization and chunking for the extraction of features. The proposed system includes three steps: feature extraction, feature selection and classification. In first step, all documents are represented in the VSM model after POS tags and chunks of suspicious and source documents are extracted. Then, two-stage feature selection algorithms are applied to increase the efficiency of the classification step. Finally, classification is made using different classification algorithms such as Naive Bayes (NB), Support Vector Machine (SVM) and Decision Tree to identify plagiarized sentences presented to the user.

The authors proposed a matching approach for the cross-lingual text alignment task that takes into account both semantic and syntactic knowledge in [93]. In this study, a VSM based method using a multilingual word embedding dictionary and a weighting practice is utilized to retrieve the possible candidate fragments. The documents are modeled handling word graphs to process words and their relations in pairs analysis. The proposed approach consists of four main steps: First, in the preprocessing step, basic techniques in linguistic are performed on source and suspicious texts. In the candidate identification step, a set of potential source fragments is retrieved for each suspicious document and then, a pair of fragments with a similarity degree is bigger than a predefined threshold is considered a possible plagiarism candidate. In the fragment analysis step, all

pairs of candidate fragments are compared on a sentence basis. As a result of this analysis, the locations of the beginning and end of the plagiarism case are determined in both the suspicious and source documents. Finally, in the post-processing step, the detected plagiarism sections are analyzed whether they conform to the predefined lowest length and the lowest spacing between them.

3.2.7. Structure-Based Models

These methods investigate how words are written in a particular block of text in a document [13, 38]. It focuses on contextual similarity such as header, sections, paragraphs or how words are used across in whole document, to find similarity between two documents. Information in the context is usually processed using the structure of tree notations, which can be provided in ML-SOM [99]. In the other study [100], the authors proposed a method that detects plagiarism in two stages: In the first stage, document clustering and retrieval of candidates are performed handling a notation of tree structure. In the final stage, plagiarism is detected using ML-SOM.

In [101], the authors represented a textual document as graph that shows semantic relatedness. In this work, every sentence is symbolized by a node of the graph and sentence relations are represented by edges. Graphical structures ensure a further detailed representation of the documents and make easier the deep analysis. This method has high potential compared to other plain document representations. However, it is more effective to use a combined approach with text and structured information. There are also methods proposed in [83, 102] to detect plagiarism especially in scientific publications, based on different approaches such as logical structure extraction (LSE) and general classes, using the structural information of documents.

3.2.8. Cluster-Based Models

These methods are generally useful for retrieving information during the search process of any plagiarized document. Besides, compared to other methods, the comparison time is less during the detection process [49].

As seen in Figure 3.6 [119], clustered fragments are created from feature matrix or vectors of a suspicious document. Then, the clustered fragments are divided into n clusters. These n clusters are analyzed according to a heuristic algorithm. Let class $\in \{plagiarized, not\ plagiarized\}$, finally, these n clusters are classified according to whether they are plagiarized or not.

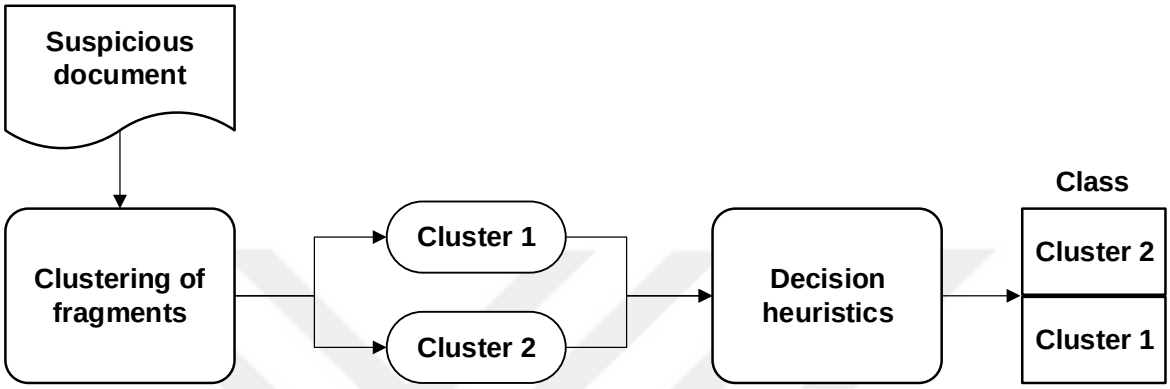

Figure 3.6. Steps of Cluster-Based Plagiarism Detection

In the use of these methods, the grouping of documents as unsupervised or supervised in the process of information retrieval emerges as an important factor [22]. Various NLP problems studied by researchers such as plagiarism detection, text classification and summarization etc. are used to reduce the search space in the information retrieval task [103-105]. This provides a significant reduction of document comparison time during plagiarism detection. There are also several studies proposed in [106, 107] that use certain words or keywords to group similar parts of the text fragments.

3.2.9. Fuzzy-Based Models

In fuzzy-based methods, corresponding remains of text are predicted or become ambiguous. Machine learning techniques are also applied to these methods for similarity analysis. Sentences in the documents are presented as numbers or characters to identify plagiarism. It performs various similarity calculations that sequence from one to zero. One is exactly the same that means a plagiarism has been detected, while zero is completely different that means that there is no plagiarism. The fuzzy concept indicates that every word in a text is associated

with an indefinite set of words with similar meanings. It can be modeled by considering the similarity between the expression in a text and the downy set [35]. Fuzzy-based methods provide an easy way to conclude a precise similarity based on ambiguous, noisy or incomplete input information.

A correlation matrix consisting of words and correlation factors measuring the grade of similarity between two or more words was constructed in [108]. Then, the grade of similarity between sentences is obtained by calculating the correlation factor of the word pairs in distinct text passages of the relevant documents. The authors introduced a tool that identifies the grade of similarity between documents based on the fuzzy information retrieval (IR) approach in the other study [109].

Alzahrani and Salim [110] proposed an approach in which fuzzy-based semantic similarity metric calculations are used in the detailed analysis stage. Then, the authors included POS tags knowledge and fuzzy-based rules focused on extremely obfuscated plagiarism types into this similarity metric in [34]. Based on the tests performed, the authors discussed that their approach is statistically significant and demonstrates the potential of semantics methods to determine the plagiarism. Later, an advanced fuzzy-based semantic similarity measure using POS tags is proposed in [111].

3.3. Brief Overview of PAN11 Systems

Systems developed according to various methods for plagiarism detection have been investigated in PAN competitions. The results regarding both internal and external plagiarism detection were discussed in these competitions. In this thesis study, PAN-PC-11 dataset is used to evaluate the results produced by the proposed plagiarism detection system. The performance of the proposed system has been compared with the results of nine detectors participating in the PAN11 competition. In addition, the performances of the detectors participating in the PAN11 competition summarized in [51] are discussed. Table 3.6 gives an overview of the properties and methods of participating detectors in PAN11. Each study in PAN11 is denoted by Dn, which stands for detector in Table 3.6, and n is the number of the detector.

Table 3.6. Properties of the Detectors that Participated in PAN11

#	Study	Method	Description
D1	[120]	Based on number of matching words & WordNet	The system consists of three main parts: preprocessing of the text, detection of plagiarism candidates and post-processing, which includes the steps of removing overlapping passages, merging of passages and excluding ambiguous pairs of passages.
D2	[121]	Character n-grams	The system includes the following four steps: preprocessing of the text, calculating a similarity matrix for each pair of source and suspicious documents, sorting the pairs by their similarity and finally, analyzing the top-ranked pairs in detail.
D3	[122]	Word n-grams	The system consists of two phases: first, it executes a plagiarism search space reduction method and then performs an exhaustive search to find possible plagiarized passages.
D4	[123]	To avoid the usage of hash types	The system is built in Amazon Web Services. The source files are indexed in the system and then the detector discovers the plagiarism cases using the index. The similarity is measured by cosine distance to confirm passages.
D5	[124]	Contextual n-grams (CTNG)	The proposed system is based on CTNG. CTNGs are the basic information for indexing and modeling documents for plagiarism detection task. These n-grams are obtained through a series of techniques involving capitalization, stop words removal, stemming and internal sorting.
D6	[125]	Vector Space Model (VSM) & word n-grams	The system consists of two steps: candidate selection and detailed analysis. A VSM-based model is performed to retrieve candidate documents, and then, a window-based similarity degree is calculated to determine plagiarism.
D7	[126]	N-gram fingerprinting & WordNet	The main steps of the system are tokenization, stemming, POS tagging, word sense disambiguation and generating a relative semantic similarity matrix for each word pair.
D8	[127]	Running Karp-Rabin (RKR) Greedy String Tiling (GST) algorithm	The system consists of three steps: first is preprocessing and indexing, second is candidate document selection using an Information Retrieval based approach and finally, detailed analysis. The source candidate document and suspicious document are compared using the RKR GST algorithm.
D9	[128]	Character n-gram distribution & WordNet	The system consists of three stages: knowledge preparation, candidate retrieval and plagiarism detection. The source sentences are stored in an index, and then, a sentence-to-sentence mapping is made between a query created from suspicious sentences and source files in the index.

4. PLAGIARISM DETECTION BASED ON POS TAG N-GRAMS

This chapter describes a part-of-speech (POS) tag n-grams based approach for the problem of plagiarism detection. In this study, the type of plagiarism that is focused on is textual plagiarism instead of plagiarism in programming code.

The remainder of this chapter is organized as follows: Section 4.1 explains the general process of the proposed approach based on POS tags n-grams. In Section 4.2, the plagiarism detection process is described in detail with all its phases. Each phase of the system is covered in subsections of Section 4.2.

4.1. General Approach

The architecture and general process of proposed plagiarism detection system consists of four main phases as shown in Figure 4.1.

First, in the preprocessing phase, source and suspicious documents are made ready for processing. For this, a series of operations are applied to remove line breaks and solve encoding problems in the documents. Then, all source and suspicious documents are separated into their sentences via a sentence segmentation step, each sentence is split into tokens, and finally each token is tagged with its POS tags. In POS tagging step, POS tags are replaced with one-character length symbols in order to reduce the file size. Thus, search process is able to be shortening in the next candidate retrieval phase. The details of the preprocessing phase are described in Section 4.2.1.

In n-gram generation phase, POS tag n-grams of source and suspicious sentences are created. Source sentences converted to POS tag n-gram structure are indexed through a search engine library. Thus, it is aimed to provide a rapid and direct access to the source sentences to be used as candidates. Details of the n-gram generation and indexing process are in Section 4.2.2.

In candidate retrieval phase, every sentence of suspicious documents is searched in the source index in turn. Search engine queries are created from the POS tag n-grams of each sentence in the suspicious documents to obtain a set

of candidate sentences. It is aimed to find similarities and possible plagiarism passages between the suspicious sentence that is searched in the index and the source sentences. Thus, in the final phase to decide whether there is plagiarism, a syntactic or semantic similarity calculation will be made only on candidate source sentences of a suspicious sentence, not on all source documents. Details on the representation of candidate sentences are in Section 4.2.3.

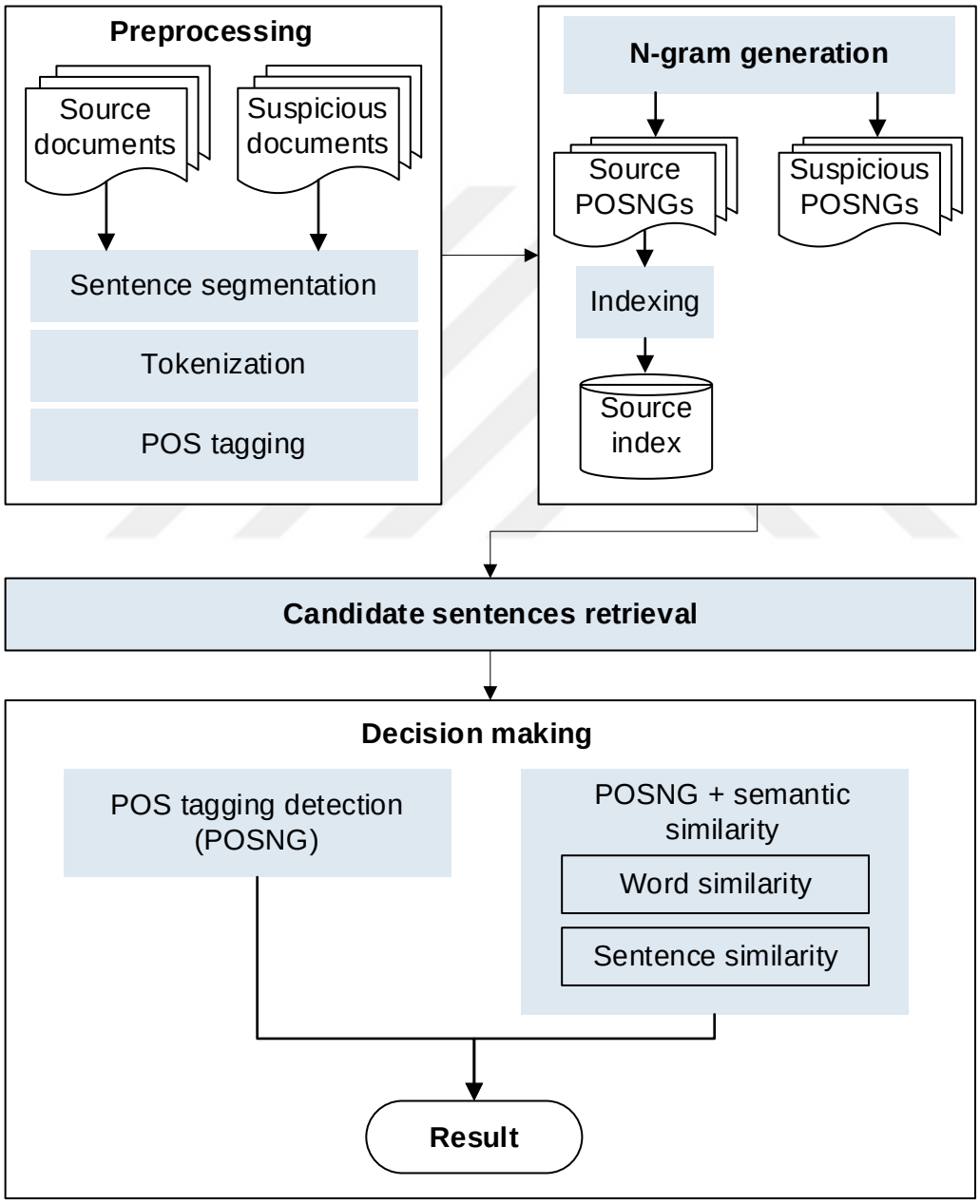

Figure 4.1. The Architecture of Plagiarism Detection Process

In decision making phase, suspicious sentences and their candidate sentences are compared according to two plagiarism detection methods. In the first method, it is only decided whether the suspicious sentences are plagiarized according to their POS tag n-grams (POSNG). In the other method, semantic similarity is applied along with POSNG method. Semantic similarity is calculated at both the word and sentence level. The plagiarized sections are identified based on the POS tag n-grams and the semantic similarity measure of the vector semantic model. Each method is described in detail in Section 4.2.4.

4.2. Plagiarism Detection Process

In this section, all the phases and steps performed in the plagiarism detection process of the proposed approach are described in detail.

4.2.1. Preprocessing

In this phase, basic NLP techniques are applied on the source and suspicious documents in a total of six steps. As can be seen in Figure 4.2, at the end of each step, an output is produced to be used in the next step.

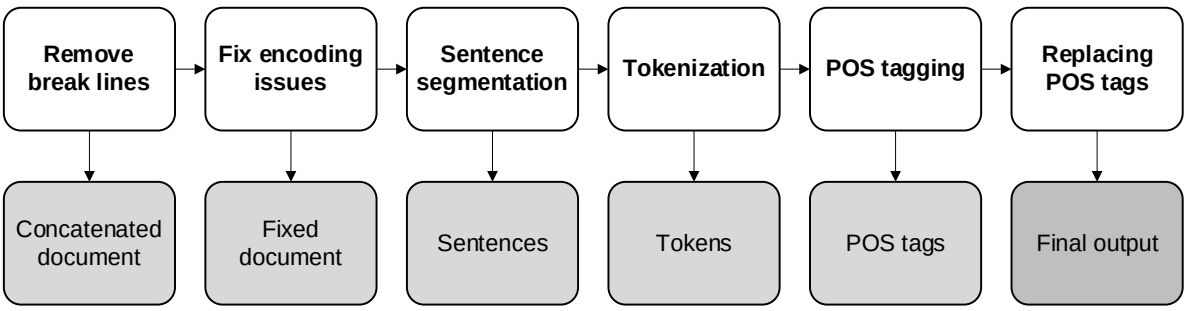

Figure 4.2. The Steps of Preprocessing Phase

In the preprocessing phase, first of all, break lines of the documents are removed to make the document ready to be split into sentences. As can be seen in Figure 4.3, the contents of the documents [152] in the dataset are written into the text files line by line. However, since the content of the document is written randomly to the files, each line does not exactly constitute a sentence. There may be one or more sentences in a line and in some cases a sentence may appear to cover

more than one line. Since the proposed method in this study allows sentences to be generated using a regular expression pattern, it requires the collection of consecutive lines into a single line.

	Original document
1	This Isabel, mother of Philippa, was a very important acquaintance indeed
2	for Columbus. It must be noted that he left the shop and poor Bartholomew
3	to take care of themselves or each other, and went to live in the house of
4	his mother-in-law. This was a great social step for the wool-weaver of
5	Genoa; and it was probably the result of a kind of compromise with his
6	horrified relatives at the time of her marriage. It was doubtless thought
7	impossible for her to go and live over the chart-maker's shop; and as you
8	can make charts in one house as well as another, it was decided that
9	Columbus should live with his mother-in-law, and follow his trade under her roof.
10	Columbus, in fact, seems to have been fortunate in securing the favour of his
11	female relatives-in-law, and it was probably owing to the championship of
12	Philippa's mother that a marriage so much to his advantage ever took place at all.
13	His wife had many distinguished relatives
14	in the neighbourhood of Lisbon; her cousin was archbishop at this very
15	time; but I can neither find that their marriage was celebrated with the
16	archiepiscopal blessing or that he ever got much help or countenance from
17	the male members of the Moniz family.
	New document after removing break lines
1	This Isabel, mother of Philippa, was a very important acquaintance indeed for Columbus. It must be noted that he left the shop and poor Bartholomew to take care of themselves or each other, and went to live in the house of his mother-in-law. This was a great social step for the wool-weaver of Genoa; and it was probably the result of a kind of compromise with his horrified relatives at the time of her marriage. It was doubtless thought impossible for her to go and live over the chart maker's shop; and as you can make charts in one house as well as another, it was decided that Columbus should live with his mother-in-law, and follow his trade under her roof. Columbus, in fact, seems to have been fortunate in securing the favour of his female relatives-in-law, and it was probably owing to the championship of Philippa's mother that a marriage so much to his advantage ever took place at all. His wife had many distinguished relatives in the neighbourhood of Lisbon; her cousin was archbishop at this very time; but I can neither find that their marriage was celebrated with the archiepiscopal blessing or that he ever got much help or countenance from the male members of the Moniz family.

Figure 4.3. The Representation of Removing Break Lines

Each of the text fragments highlighted in different colors in Figure 4.3 represents distinct sentences. In the text fragments of the original document, it is clear that each colored sentence is contained in three, four or five lines. For successful sentence segmentation task, line breaks are removed and each source and suspicious document is rewritten as a single line. In Figure 4.3, the sentences

highlighted with colors consist of the first thirteen lines in the original document, while the new document after the lines are removed consists of a single line.

In the preprocessing phase, all source and suspicious documents are updated at the end of each NLP step. All documents in the dataset are encoded with UTF-8 format. As can be seen from the example [153] in Table 4.1, some special symbols or non-English letters are used in this corpus. Therefore, a conversion process is made between ISO-8859-1 and UTF-8 character sets when reading from or writing to a file.

Table 4.1. Examples of non-English Letters in the Corpus

Example	Text
Example-1	A FREE TRANSLATION OF MAURUS JÓKAI'S ROMANCE "A SZÉP MIKHÁL"
Example-2	A lady gardener who understood her business had to know what species of flowers could be planted and sown under the zodiacal signs ♈, ♉, ♊, or ♋, ♌, ♍; to which the signs ♎, ♏, and ♐ are baleful; and how seldom those flourish which are planted under the signs ♑, ♒, and ♓; in fact, she had to have her almanac at her fingers' ends.

While updating documents in each step, there may be encoding problems due to these special characters. These encoding problems need to be fixed so that documents can be correctly split into sentences and then, tokens can be created properly. Otherwise, the starting location and lengths of the sentences in the document may be obtained incorrectly. This situation causes the location and length of plagiarized passages to be incorrectly determined and poor system performance. Therefore, after removing the line breaks, various bugs such as encoding problems, extra spaces, and punctuation errors in the documents are fixed in order to make the source and suspicious documents ready for splitting into sentences.

4.2.1.1. Sentence Segmentation

Let C_{src} and C_{susp} be two corpuses that containing the set of source and suspicious documents respectively. Since the proposed method is sentence-

based, in this step, the sentence segmentation task is performed and the documents in both C_{src} and C_{susp} are split into their sentences.

Algorithm 4.1. The General Mechanism of Sentence Segmentation

Input:
$text$ is any document in the corpus

Variables:
$sentenceList$ is the list of sentences
$shortSentenceList$ is the list of short sentences
$shortSentenceIndex$ is the list of starting point of sentence in the document
$previousIndex$ is the starting location of the previous sentence in the document
$minWordNumber$ is the minimum word number in a sentence
$sentence$ is the sentence getting from document
$index$ is the starting point of $sentence$ in the document
$length$ is the total character length of the $sentence$
$previousIndex$ is the previous $index$ of former $sentence$
$wordNumber$ is the number of words in the $sentence$
$wordCountFormerSentence$ is the number of words in the former $sentence$
$concanatedSentence$ is the combination of the sentences whose total number of words are > $minWordNumber$

```
List sentenceList, shortSentenceList, shortSentenceIndex
previousIndex ← -1
minWordNumber ← 12
while not (end of file)
    sentence ← get sentence of text
    index ← text.indexOf(sentence, previousIndex + 1)
    length ← get length of sentence
    previousIndex ← index
    Array wordNumber ← get word number of sentence
    if wordNumber < 1024
        if wordNumber < minWordNumber
            add sentence to shortSentenceList
            add index to shortSentenceIndex
            wordCountFormerSentence ← wordNumber
            if wordCountFormerSentence ≥ minWordNumber
                for i = 1 to shortSentenceList do
                    concanatedSentence ← shortSentenceList_i
                end
            end if
        end if
    end if
end
```

The reason for using the sentence-based approach in this study is that the sentence is one of the meaningful block of a text and consists of a group of words that express a complete thought [112].

In this thesis study, stemming or lemmatization techniques are not applied because the Word2Vec model is used to measure the semantic similarity between words. Word2Vec represents each word with a vector consisting of a specific list of numbers. Therefore, since the degree of semantic similarity between different words is measured by the cosine similarity between these vectors, it is not necessary to convert the words to their base forms by performing the stemming or lemmatization steps.

Some of the functional words such as prepositions, determiners and articles, called *stop words,* are content-independent and appear frequently in the documents. Since stop words do not contain any semantic information [52], they can be removed from the text to make calculations faster and retrieve the relevant data quickly [129]. Let D_{src} be a source document in C_{src} and D_{susp} be a suspicious document in C_{susp}. In the proposed method, the sentences of D_{src} and D_{susp} are formed without removing the stop words so that their syntactic structures do not change. Even if the contents of D_{src} and D_{susp} are completely different, the same short sentences can be encountered in D_{src} and D_{susp}, either on or off-topic.

In the task of sentence segmentation, short sentences have been ignored in many studies. The authors [74] excluded sentences with less than 5 words from the plagiarism detection process. On the other hand, the authors [130] considered sentences containing less than 7 non-stop words as short sentences and ignored them in the plagiarism algorithm. LaRocque [131] classifies sentences containing less than 12 words, including stop words, as short.

Since the proposed method in this study is POS tag-based, short sentences are used to preserve the syntactic structure of the text as a whole. Sentences were generated by combining the short ones with the others during the sentence segmentation process. Let *S* be a sentence, *S'* be the sentence before *S*, and *N* be the number of words, including stop words, in *S*. If $N > 12$, then *S* is considered

as a separate sentence, otherwise *S* is added to *S'*. Therefore, *S'* and *S* are combined to be a new sentence. This operation proceeds until the total number of the words of the combined sentences is greater than 12. The general mechanism of the sentence segmentation task is given in Algorithm 4.1.

Table 4.2. The Representation of Sentences in the Text Files

#	Sentences
1	But of how little real importance is it to establish the bare fact, that Shakespeare was an attorney's clerk before he was an actor!**<|0,133**
2	Suppose it proved, beyond a doubt,--what have we learned? Nothing peculiar to Shakespeare; but merely what was equally true of thousands of other young men, his contemporaries, and hundreds of thousands, if not millions, of those of antecedent and succeeding generations.**<|134,270**
3	It has a naked material relation to the other fact, that he uses legal phrases oftener than any other dramatist or poet; but with his plastic power over those grotesque and rugged modes of speech it has nought to do whatever.**<|406,225**
4	That was his inborn mastery. Legal phrases did nothing for him; but he much for them.**<|632,84**
5	Chance cast their uncouth forms around him, and the golden overflow from the furnace of his glowing thought fell upon them, glorifying and enshielding them forever.**<|718,164**
6	It would have been the same with the lumber of any other craft; it was the same with that of many others,--the difference being only of quantity, and not of kind.**<|883,162**
7	How, then, would the certainty that he had been bred to the law help us to the knowledge of Shakespeare's life, of what he did for himself, thought for himself, how he joyed, how he suffered, what he was?**<|1046,204**

As can be seen from Table 4.2, sentences [154] are stored with their starting index and length in the document. For example, as seen at the end of the second sentence in Table 4.2, the starting location of the sentence after the symbol "**<|**" is 134; the second value, 270, indicates how many characters the sentence consists of.

4.2.1.2. Tokenization

The next operation after sentence segmentation is *tokenization*. Tokenization is the task of dividing sentences into the smallest possible units and is a necessary

step for POS tagging. As shown in Table 4.3, sentences are divided into tokens, each represented as *T*. Tokens are defined as non-whitespace sequences and a Whitespace Tokenizer model [155] is used to convert sentences into their tokens.

4.2.1.3. POS Tagging

Finally, in the preprocessing step, the *POS tagging* task is performed. POS tagging is one of the most important tasks in NLP and is defined as labelling each token in a sentence with its most suitable syntactic class [74].

Table 4.3. Step-by-step Output of a Sentence in the Preprocessing Phase

#	Technique	Result of *S* after step
		S: He thinks he had of countenance how he had merely understood travel yesteryear, and he state to himself: "I must salvage how her contradicted from myself!"
Step 1	Tokenization	He/**T** thinks/**T** he/**T** had/**T** of/**T** countenance/**T** how/**T** he/**T** had/**T** merely/**T** understood/**T** travel/**T** yesteryear/**T** ,/**T** and/**T** he/**T** state/**T** to/**T** himself/**T** :/**T** "/**T** I/**T** must/**T** salvage/**T** how/**T** her/**T** contradicted/**T** from/**T** myself/**T** !/**T** "/**T**
Step 2	POS tagging classification	He/**PRP** think/**VBP** he/**PRP** had/**VBD** of/**IN** countenance/**NN** how/**WRB** he/**PRP** had/**VBD** merely/**RB** understand/**VB** travel/**NN** yesteryear/**NN** ,/**,** and/**CC** he/**PRP** state/**NN** to/**TO** himself/**PRP** :/**:** “/`` I/**PRP** must/**MD** salvage/**VB** how/**WRB** her/**PRP** contradicted/**VBD** from/**IN** myself/**PRP** !/**.** “/**''**
Step 3	Replacement POS tags with one length letter or digit.	He/**p** think/**S** he/**p** had/**b** of/**i** countenance/**n** how/**W** he/**p** had/**b** merely/**r** understand/**v** travel/**n** yesteryear/**n** ,/**1** and/**c** he/**p** state/**n** to/**t** himself/**p** :/**2** “/**5** I/**p** must/**m** salvage/**v** how/**W** her/**p** contradicted/**b** from/**i** myself/**p** !/**0** “/**6**
Step 4	Result	p S p b i n W p b r v n n 1 c p n t p 2 5 p m v W p b i p 0 6

Syntactic analysis is performed to examine how different writers might paraphrase the relevant sections of the same content in different documents. POS tag series are applied to train the style of particular a writer, and a set of rules is created to detect paraphrased passages in other documents [132]. Let S_{src} and S_{susp} be two sentences, where $S_{src} \in D_{src}$ and $S_{susp} \in D_{susp}$. Each token

of S_{src} and S_{susp} is tagged with its part-of-speech using Stanford Log-linear Part-Of-Speech Tagger [156]. At the end of this tagging process, each token in the sentence is shown with its part-of-speech such as verb, noun, adjective etc.

Table 4.4. The Penn Treebank POS Tagset

#	Tag	Description	Example	#	Tag	Description	Example
1	CC	coordin. conjuction	and, but, or	24	SYM	symbol	+, %, &
2	CD	cardinal number	one, two	25	TO	“to”	to
3	DT	determiner	a, the	26	UH	interjection	ah, oops
4	EX	existential ‘there’	there	27	VB	verb base form	eat
5	FW	foreign word	mea culpa	28	VBD	verb past tense	ate
6	IN	preposition/sub-conj	of, in, by	29	VBG	verb gerund	eating
7	JJ	adjective	yellow	30	VBN	verb past participle	eaten
8	JJR	adj., comparative	bigger	31	VBP	verb non-3sg pres	eat
9	JJS	adj., superlative	wildest	32	VBZ	verb 3sg pres	eats
10	LS	list item marker	1, 2, One	33	WDT	wh-determiner	which, that
11	MD	modal	can, should	34	WP	wh-pronoun	what, who
12	NN	noun, sing. Or mass	llama	35	WP$	possessive wh-	whose
13	NNS	noun, plural	llamas	36	WRB	wh-adverb	how, where
14	NNP	proper noun, sing.	IBM	37	$	dollar sign	$
15	NNPS	proper noun, plural	Carolinas	38	#	pound sign	#
16	PDT	predeterminer	all, both	39	“	left quote	‘ or “
17	POS	possessive ending	‘s	40	”	right quote	’ or ”
18	PRP	personal pronoun	I, you, he	41	(	left parenthesis	[, (, {, <
19	PRP$	possessive pronoun	your, one’s	42	)	right parenthesis	],), }, >
20	RB	adverb	quickly, never	43	,	comma	,
21	RBR	adverb, comparative	faster	44	.	sentence-final punc	. ! ?
22	RBS	adverb, superlative	fastest	45	;	mid-sentence punc	: ; … _ -
23	RP	particle	up, off				

In this study, *The Penn Treebank POS Tag Set* [164], which has a total of 45 POS tags, including punctuation marks and symbols, is used, as can be seen from Table 4.4.

Algorithm 4.2. The General Mechanism of POS Tagging Production

Input:
$corpus$ is the dataset containing source and suspicious documents

Variables:
$documentNumber$ is the number of documents in the source or suspicious corpus
$tags$ is the array of tokens in the any sentence
$count$ is the number of n-gram
tag is the POS tag n-gram
$finalPOSTag$ is the final output containing all the POS tag n-grams

```
1: producePOSTags()
2: for each corpus {suspicious, source}
3:     documentNumber ← get number of document in corpus
4:     for i = 1 to documentNumber do
5:         read documents {contains sentence list} line by line
6:         for each sentence
7:             tags array ← get tokens of the sentence
8:             for j = 1 to tags do
9:                 tags[j] ← generalize(tags[j]) {e.g. JJR -> k}
10:                append tags[j] to STRING_BUILDER
11:                count ← count + 1
12:                if count = ngram {e.g. ngram = 4}
13:                    tag ← STRING_BUILDER
14:                    finalPOSTag ← finalPOSTag + tag
15:                else if count > ngram
16:                    delete first tag of STRING_BUILDER
17:                    finalPOSTag ← finalPOSTag + tag
18:                end if
19:            end for
16:            write finalPOSTag set to file
17:        end for
18:    end for
19: end for
```

Let *S* be a sentence as seen in Table 4.3. The result of preprocessing steps of *S* is shown in Table 4.3. An NLP technique is applied to *S* at each step and a string of certain characters is obtained as a result of this process. In the proposed system, the POS tag n-grams of the source sentences are indexed to provide

quick access to the candidate sentences. Thus, it is ensured that the search process of suspicious sentences in this index is carried out faster. Indexing performance is dependent on system disk space and memory usage and may vary based on these values. In terms of disk usage, indexing is directly proportionate to the total size of all documents in the corpus. Therefore, as can be seen from Step 3 in Table 4.3, each POS tag is replaced with one-character length to reduce size of the sentences to be indexed and speed up the candidate search process. A unique one-length character is determined for each POS tag in the Penn Treebank POS Tagset and stored in the system for use in this replacement step.

The mechanism of generation of POS tags in the proposed system is seen in Algorithm 4.2. First, the tokens of each sentence are found, then the POS tags are determined by the POS Tagger software and finally, each POS tag is replaced with symbols of one-character length.

4.2.2. N-gram Generation

In this step, n-gram sets of each sentence are created. N-gram is one of the most popular techniques in NLP and can be defined as the contiguous sequences of the characters or words [64]. A better estimate can be made in its applications depending on the length of *n*. N-grams show the consecutive common elements in a text data and maximize identifying similar sentences [133].

4.2.2.1. Methodology

In the former studies, the authors [134, 135] and Stamatatos [52] proposed word n-grams (WNG) and stop word n-grams (SWNG) for the string matching respectively. WNG methods are based on matching of strings to each other. In these methods, stop words are filtered out from the text and the remaining words are converted to their roots. On the other hand, SWNG removes all but stop words from the text unlike traditional methods for finding common n-grams.

In most cases of plagiarism, the text is modified by changing the order of the stop words or removing them from the text. For this reason, SWNG fails to detect such

plagiarized passages. WNG, on the other hand, is particularly successful in detecting cases of copy-paste plagiarism. But if the words are replaced with synonyms, WNG is not be able to accurately identify the plagiarized passages. The method proposed in this study, unlike the other two methods, is based on the part-of-speech tag n-grams (POSNG). POS tags are effective in knowing the syntactic properties of documents and give clues if the two passages have a common syntactic structure.

Table 4.5. Generation of POS Tag n-grams

n-gram	**POSNG of Sentence**
Original	p S p b i n W p b r v n n 1 c p n t p 2 5 p m v W p b i p 0 6
3-gram	pSp Spb pbi bin inW nWp Wpb pbr brv rvn vnn nn1 n1c 1cp cpn pnt ntp tp2 p25 25p 5pm pmv mvW vWp Wpb pbi bip ip0 p06
4-gram	pSpb Spbi pbin binW inWp nWpb Wpbr pbrv brvn rvnn vnn1 nn1c n1cp 1cpn cpnt pntp ntp2 tp25 p25p 25pm 5pmv pmvW mvWp vWpb Wpbi pbip bip0 ip06
5-gram	pSpbi Spbin pbinW binWp inWpb nWpbr Wpbrv pbrvn brvnn rvnn1 vnn1c nn1cp n1cpn 1cpnt cpntp pntp2 ntp25 tp25p p25pm 25pmv 5pmvW pmvWp mvWpb vWpbi Wpbip pbip0 bip06

In n-gram generation step, POSNG is created from the documents in C_{src} and C_{susp}. An example of the generation of 3, 4 and 5-grams POS tags of the sentence *S* is shown in Table 4.5. After this step, a second replacement process starts in order to reduce the size of the POSNGs again and speed up the search time. For this, all POSNGs produced in the original and suspicious documents are searched and then sorted in the entire corpus from the most recent to the least.

Table 4.6. Number of Occurrences of n-grams

#	**n-gram**	**Number of Occurrences**	**Replaced Symbol**
1	idni	5,881,735	.
2	nidn	5,394,456	,
3	dnid	5,175,588	;
4	idjn	4,901,428	‘
5	djni	3,854,853	#
6	idn1	2,998,981	<

For example, according to the test results using 4-grams, 1,050,203 unique 4-grams were produced in the whole data set. As an example, Table 4.6 shows how many times the first six most commonly used 4-grams occur in the whole dataset. These unique n-grams are sequentially replaced with symbols starting from one-character length up to their own n-gram lengths.

1	idni .	486692	vneR 99P	1050175	sVrb
2	nidn ,	486693	sjNc 99Q	1050176	HdrP
3	dnid ;	486694	v6un 99R	1050177	px0H
4	idjn '	486695	snmV 99S	1050178	nmVH
5	djni #	486696	krgz 99T	1050179	mVHc
6	idn1 <	486697	d1Ht 99U	1050180	pjxr
7	nidj >	486698	mair 99V	1050181	VzjJ
8	jnid $	486699	wpro 99W	1050182	J0eg
9	djn1 %	486700	tqsi 99X	1050183	v71p
10	dnin /	486701	qsii 99Y	1050184	ozp7
11	idn0 &	486702	2o0z 99Z	1050185	rkw1
12	aidn =	486703	cnd3 990	1050186	gWRp
13	Vidn @	486704	J3cr 991	1050187	1WBb
14	dn1c _	486705	P6bd 992	1050188	sVqb
15	dnig |	486706	zid7 993	1050189	aaeh
16	niPn a	486707	W0nb 994	1050190	1jkf
17	1idn b	486708	n5rH 995	1050191	VPm1
18	djn0 c	486709	0qSr 996	1050192	cogE
19	idai d	486710	rzee 997	1050193	hvWj
20	dniP e	486711	zee1 998	1050194	vnfP
21	bdni f	486712	2ssr 999	1050195	oVwS
22	daid g	486713	ieEp	1050196	nthH
23	tvdn h	486714	P2sS	1050197	35p3
24	ridn i	486715	cvve	1050198	oWrc
25	idja j	486716	aStx	1050199	o66W
26	nidg k	486717	cvcR	1050200	czem
27	vdni l	486718	Rawm	1050201	0wz7
28	jn1c m	486719	u25r	1050202	wz7v
29	iPn1 n	486720	06u3	1050203	Hxjp

Figure 4.4. List of Changed n-grams with the Symbols

Symbols used for replacement of n-grams, which can also be thought of as new or updated n-grams, consist of punctuation marks, symbols, uppercase and

lowercase letters and numbers in order to achieve the maximum single character length. For this, a set containing all these characters is created and then new n-grams with lengths such as one, two, and three are produced sequentially according to the current n-gram length. For example, if 4-grams are using in the plagiarism detection process, these existing 4-grams are replaced with their smaller n-grams up to a maximum length of 3.

As can be seen from Figure 4.4, a file is created in which each unique n-gram and its replacement symbol are written on the same row with a space between them. It is clear in Figure 4.4, a replacement operation is not done starting from the row 486712. Because up to that row every 4-gram has been replaced with its corresponding symbol of smaller length. Since it will not be possible to create a string of 3 or less length from this row, there is no need to change 4-gram. After this replacement process, the size of the total source and suspicious data set decreased by 34.4%.

Table 4.7. Replacement of POSNGs with Symbols

POSNGs	**After Replacement**
idj1 dj1p j1pb 1pbd pbdn bdni dnid nido idon donn onnt nntg ntg1 tg1c g1cx 1cxb cxbr xbri bria riai iaid aidn idni dniP niPn iPni Pnig nig0	,J K5 e9 #Y x h ' ao BM 'g_ $5s ,I6 nS lH ,/p v9 .Lo ,89 .o8 Mg #q @ . g c .b &q ,q
ipbV pbVi bVig Vig1 ig1p g1pS 1pSj pSji Sjid jida idaS daSn aSnc Sncj ncja cjai jaid aidn idnb dnbi nbig bigc igcg gcg1 cg1W g1Wo 1Woc Wocd ocdb cdbp dbpi bpid pidJ idJn dJn1 Jn1c n1cp 1cpS cpSb pSbj Sbjt bjtv jtvp tvp0	.w .j $n &7 cZ a$ p8 mH I_ $r @J ,a$ <eZ >sv gO /, C @ F ,P xy .i5 #b >X 'EB %bQ c>b q5% _ir #6e #sH #/ .S5 =Z C' .q. .q #r ,1D #Du jhP $o /K <Q

Table 4.7 shows the final view after the replacement with symbols of the two sample sentences transformed to POS tag n-grams format. As can be seen from the replacement operation, the final version of the sentences is unreadable and creates an impression as if it is encrypted. Each symbol in the final version such as $.w$ or $\$5s$ actually represents a string. Strings created during the replacement operation are unique and have case-sensitive property.

As a result, the POS tag n-grams created after the replacement process are stored with the document and sentence number to be used in the next indexing operation.

4.2.2.2. Indexing

The POSNG of source documents in C_{src} is indexed at the sentence level using a text-based search engine called Lucene [157]. Lucene provides high performance indexing of documents via an API and has a powerful search capability. It is a platform independent solution that uses powerful and efficient search algorithms. It is therefore a popular library widely used in both academic and commercial settings. Lucene basically provides search operation on documents.

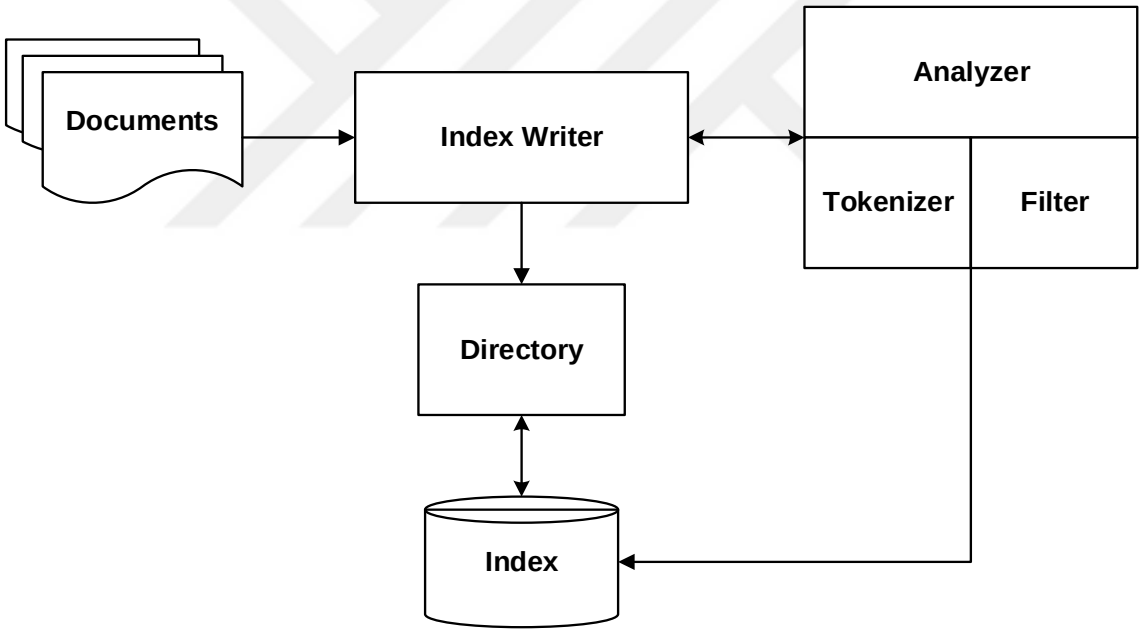

Figure 4.5. Components of Indexing Operation

A document can actually be defined as a set of specific fields. On the other hand, the field contains a name, which is a string, and one or more values. In Lucene, the document structure is not restrained in no way. Fields are only limited to keep one type of data such as text, numeric or binary. Text information is stored in the index in two ways: (1) strings store the whole data as single; (2) text data is stored

as a set of tokens. It is possible to split a text fragment into the tokens and write custom tokens in Lucene [139].

Algorithm 4.3. The General Mechanism of Indexing and Searching

Input:
$corpus$ is the dataset containing source or suspicious documents

Variables:
$documentNumber$ is the number of documents in the corpus
doc is document to be indexed
$candidateLimit$ is the number of candidates
$candidateSentence$ is the source sentence returned from the searching operation

```
index()
for each corpus {source}
    documentNumber ← get number of document in source corpus
    for i = 1 to documentNumber do
        read doc[i] {contains POS tags n-grams} line by line
        while not end of doc[i]
            add doc[i] to INDEX
        end for
    end for
end for
search()
initialize index
for each corpus {suspicious}
    documentNumber ← get number of document in corpus
    candidateLimit ← threshold
    for i = 1 to documentNumber do
        read documents line by line {contains POSTags list}
        search line {POSTags} in INDEX
        for j = 1 to candidateLimit do
            add candidateSentence[j] to set
        end for
    end for
end for
```

Lucene has a searching component that seeks a query and returns a series of documents ordered by relevance of the documents most similar with the highest scoring query. Lucene administrates a directory through a set of documents. While the documents are added to or removed from the collection, it ensures quick access to the index. In Lucene, an index can store a group of documents,

which have different number of fields. Lucene indexes terms which consist of a domain name with a token. Terms in the document are described as the pairs of field value and name. The Lucene index supplies an association from terms to the documents, called an inverted index. This inverted index provides a mechanism to score searching results. If a set of terms match with the same document, that document is considered to be related to the search criteria [139].

Figure 4.5 shows the basic internal structure of an index [140]. Documents contain a collection of fields and their contents. These documents are stored in a directory by an index writer and then indexed. The directory provides a similar interface to the file system of the operating system and includes any number of sub index [139]. It allows to updating and deleting documents from the index rapidly. The contents in the documents are tokenized and filtered by an analyzer.

The Lucene index is conceptually similar to a database, but differs in some important aspects. The table in a relational database must have its schema and constraints definitions when it is created. However, there is no such constraint in the Lucene index. Lucene index can also be considered as a folder where documents are stored. Any kind of document can be put into to the folder. Lucene indexes the documents regardless of their contents [141].

The index stores a set of source documents containing POSNG terms and provides a term-to-document mapping. If a number of POSNG of a suspicious document is mapping with the same POSNG in the source document, that source document is probable considered to be related to the suspicious document.

The indexing and search algorithms of the proposed system are shown in Algorithm 4.3. In the indexing process, each source document whose sentences are converted to POS tag n-grams (POSNG) format is indexed sequentially. In the search process, all suspicious documents are taken in order and every sentence of them is searched in the source index. A limit is set for candidate sentence results returned from the index. The candidate source sentences with the highest score up to the determined limit number are listed in order starting from the highest score.

4.2.3. Candidate Sentences Retrieval

The aim of the candidate sentence retrieval step is to reduce number of the source sentences that are compared with the suspicious sentences. Let D_{src} be a source document and D_{susp} be a suspicious document. After indexing the POSNG of each D_{src} in C_{src}, the POSNG of each suspicious sentence of D_{susp} is searched in the source index. The search operation is carried out specific to the field because the search engine indexes terms that consisted of a field name and a token.

Table 4.8. The Representation of Candidate List

#	D_{src}	Source sentence	Suspicious sentence	Candidate rank (*c*)	Difference |*d*|
1	00121	27	57	1	-
2	00121	1411	91	2	1384 & 34
3	00123	1172	128	4	-
4	00124	698	148	2	-
5	00125	439	100	1	-
6	00130	909	52	4	-
7	00132	276	80	3	-
8	00138	1154	100	2	-
9	00154	196	151	3	-
10	**00178**	**6**	**119**	**1**	**-**
11	**00178**	**7**	**117**	**1**	**1 & 2**
12	**00178**	**8**	**118**	**1**	**1 & 1**
13	**00178**	**9**	**116**	**1**	**1 & 2**
14	**00178**	**10**	**120**	**1**	**1 & 4**
15	**00178**	**11**	**121**	**1**	**1 & 1**
16	00195	3273	68	4	-
17	00207	532	79	3	-
18	00213	2178	52	3	-
19	00213	3578	144	3	1400 & 92
20	00213	4035	161	3	457 & 17

A query that contains the POSNG of suspicious sentences is created and used in the source index to find candidate sentences. As a result of this query, a certain number of sentences, called *c*, starting from the first sentence with the highest score in the index are returned as candidates. A portion of the candidate sentences of D_{susp} is shown in Table 4.8. Each row specifies a candidate containing *source document, source sentence*, *suspicious sentence* and

candidate rank. The *difference* value called as |*d*| shows the difference between the numbers of the source and suspicious sentences if two consecutive candidates belong to the same source document.

Algorithm 4.4. The Process of Decision Making with $POSNG_{PD}$

Input:
C_{susp} is suspicious corpus
doc_{src} is source document where $doc_{src} \in C_{src}$
doc_{susp} is suspicious document where $doc_{susp} \in C_{susp}$

Variables:
CL is the sorted candidate list of doc_{susp}
d is the threshold
PL is the list of common plagiarized sentences of doc_{src} and doc_{susp}
S_{src} is the id of source sentence
S_{susp} is the id of suspicious sentence
current is candidate
next is the other candidate after *current* in *CL*

```
1: count ← number of documents in C_susp
2: for i = 1 to count do
3:     CL ← read candidate list of doc_susp(i)
4:     for j = 1 to CL do
5:         current ← CL_j
6:         next ← CL_j+1
7:             if current{doc_src} = next{doc_src}
8:               if |next{S_src} – current{S_src}| and |next{S_susp} – current{S_susp}| < |d|
9:                     PL ← add {S_src(j), S_src(j+1)}
10:                    PL ← add {S_susp(j), S_susp(j+1)}
11:             else
12:                    if PL ≠ Ø
13:                        write PL to plagiarized file
14:                        PL ← Ø
15:                    end if
16:                end if
17:            else
18:                if PL ≠ Ø
19:                    write PL to plagiarized file
20:                    PL ← Ø
21:                end if
22:             end if
23:    end
24: end
```

One of the contributions of the proposed method in revealing the similarities between the text fragments is the candidate representation methodology as seen in Table 4.8. The candidate list of each suspicious document is generated during the search process. This list includes the numbers of the suspicious sentences and the unique identifiers of the candidate source sentences that are mapping to each suspicious sentence. These identifiers of candidate source sentences are source document number and source sentence. The candidate list is sorted in ascending according to these numbers. As a result of sorting process, the linear flow that occurs depending on the distance between two consecutive candidates shows the probability of plagiarism.

For example, the first row in Table 4.8 shows that the 27th sentence of D_{src} is the first rank candidate (it is clear from *c* = 1) of the 57th sentence of D_{susp}. Candidate sentences are sorted by D_{src} and the number of source sentence. For each D_{src}, if the difference between two consecutive source and suspicious sentences is less than a specified value of *d*, as can be seen between 10th and 15th rows, that passage can be considered an indication of plagiarism. This approach is the main reason behind ranking source and suspicious sentences to detect possible plagiarism passages.

4.2.4. Decision Making

Let S_{src} and S_{susp} be two sentences, where $S_{src} \in D_{src}$, $S_{susp} \in D_{susp}$ and S_{src} is the candidate of S_{susp}. The aim of this step is to investigate whether S_{susp} has been plagiarized from its candidate S_{src}. The process of decision making was carried out using two methods: *POSNG plagiarism detection (POSNG$_{PD}$)* and *POSNG$_{PD}$ with semantic similarity between sentences (POSNG$_{PD+SSBS}$)*. While POSNG$_{PD}$ method is a syntactic algorithm based on POS tag n-grams, POSNG$_{PD+SSBS}$ method includes adding semantic similarity feature to POSNG$_{PD}$. Each method is described in its own subsection.

4.2.4.1. POSNG Plagiarism Detection

The main purpose of the POSNG$_{PD}$ algorithm is to detect whether there is plagiarism between the candidate source and the suspicious sentence, which is

determined to have the same POS tag n-grams in the largest number. In this method, the sentences of two consecutive candidates are analyzed depending on the number of D_{src} and the value of threshold difference |*d*|. If the difference between consecutive candidates $\{S_{src(j)}, S_{src(j+1)}\}$ and $\{S_{susp(j)}, S_{susp(j+1)}\}$ is less than |*d*|, then the set of $\{S_{susp(j)}, S_{susp(j+1)}\}$ is considered as a plagiarism case. The steps of POSNG$_{PD}$ method can be seen detailed in Algorithm 4.4.

1. Candidate list of $doc_{susp(i)}$ is read for each doc_{susp} in C_{susp}.

2. CL_j and CL_{j+1} are drawn from the candidate list as two consecutive candidates, named *current* and *next*, respectively.

3. If the numbers of doc_{src} of *current* and *next* are same, it is determined that these two candidates are retrieved from the same source document.

4. Then, the *difference* is calculated between the consecutive sentences of $\{S_{src(j)}, S_{src(j+1)}\}$ and $\{S_{susp(j)}, S_{susp(j+1)}\}$.

5. If the value of $|S_{src(j+1)} - S_{src(j)}|$ and $|S_{susp(j+1)} - S_{susp(j)}|$ is less than threshold, the passage composed of $S_{susp(j)}$ and $S_{susp(j+1)}$ is considered as plagiarized from the passage composed of $S_{src(j)}$ and $S_{src(j+1)}$. If subsequent candidates meet the same condition for the same doc_{src}, then other S_{src} and S_{susp} are added to the plagiarized passages too.

Figure 4.6 shows the matching chart of sentences in two documents consisting of original and suspicious text fragments given in Table 6.4. Plagiarists usually try to change the words and structure of the sentences when plagiarizing. They do not change much in the general flow of a text fragment and in the order of the sentences. They can combine some sentences and split others into more than one sentence. But this act does not affect the meaning and the integrity of the text in that section. Therefore, when the matching sentence numbers of the original and suspicious sentences are considered as a point, if all points constitute a linear representation as seen in Figure 4.6, this situation can be considered as a plagiarism indicator.

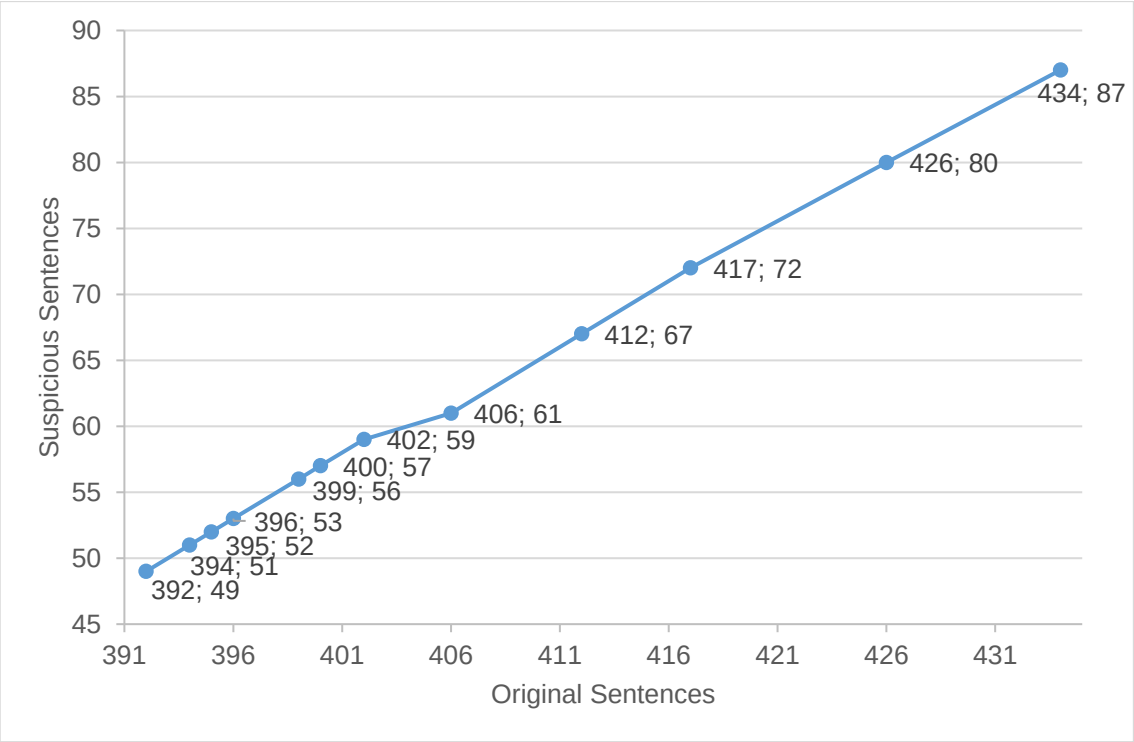

Figure 4.6. Matching Chart of Original and Suspicious Sentences

4.2.4.2. POSNGPD with Semantic Similarity

Semantic similarity is one of the most important tasks in detecting plagiarism cases. Finding semantic similarity between two different text data requires more effort than lexical and syntactic methods. POSNG$_{PD+SSBS}$ contains the operation of the POSNG$_{PD}$ method as well as a sentence-based comparison between S_{susp} and S_{src} in addition to POSNG$_{PD}$. Comparisons are made at the word level and sentence similarity is calculated with the LCS technique as a result of matching words. If the semantic similarity score calculated between S_{susp} and S_{src} is greater than a threshold value, then S_{susp} is considered as plagiarized from S_{src}.

First, the punctuation marks and stop words are removed from *S* and *P* in [161], respectively. Thus, words and punctuation seen with "-" in Table 4.9 are excluded from the semantic calculation process. Then, the semantic similarity score (between 0 and 1) of each word pair of *S* and *P* is calculated using the Word2Vec model. As can be seen in Table 4.9, the highest similarity measure is obtained for each *S* and *P* word as a result of the semantic similarity calculation.

Table 4.9. An Example of Semantic Similarity between Two Sentences

	Eventually (1)	,	a	huge (2)	cyclone (3)	hit (4)	the	entrance (5)	of	my	house (6)	.
Finally (1)	**0.9682**	-	-	0.8004	0.7076	0.8733	-	0.7124	-	-	0.7014	-
,	-	-	-	-	-	-	-	-	-	-	-	-
a	-	-	-	-	-	-	-	-	-	-	-	-
massive (2)	0.8793	-	-	**0.9560**	0.7270	0.8686	-	0.7442	-	-	0.6735	-
hurricane (3)	0.6755	-	-	0.6764	**0.9252**	0.7367	-	0.4433	-	-	0.5705	-
attacked (4)	0.7748	-	-	0.7360	0.6266	**0.8746**	-	0.6755	-	-	0.6477	-
my	-	-	-	-	-	-	-	-	-	-	-	-
home (5)	0.6834	-	-	0.8446	0.6558	0.8665	-	**0.8324**	-	-	**0.8916**	-
.	-	-	-	-	-	-	-	-	-	-	-	-

For example, let w_1 be the set of similarity measures for the word "*eventually*". As is shown in Table 4.9, w_1 = {0.9682, 0.8793, 0.6755, 0.7748, 0.6834}. Since the highest measure of w_1 is 0.9682, "*eventually*" matches with "*finally*" in *P*. Likewise, let w_2 be the set of similarity measures for the word "*hurricane*" and as is shown in Table 4.9, w_2 = {0.6755, 0.6764, 0.9252, 0.7367, 0.4433, 0.5705}. It is clear from the measures in the set of w_2 that "*hurricane*" matches with "*cyclone*" in *S* due to the highest result of 0.9252.

$$semSim(S,P) = \frac{2*LCS(S,P)}{|S|+|P|} \quad (1)$$

After the word-level comparison is completed, each word in *S* is numbered with the sequence number of the word in *P* that it matches. Likewise, each word in *P* is numbered with the ordinal number of the word it matches in *S*. Let *wo* be the set of word order. According to the enumeration method, wo_S = {1, 2, 3, 4, 5, 5} and wo_P = {1, 2, 3, 4, 6}. It is clear from the definition of LCS that *LCS* (*S*, *P*) = 4 (1, 2, 3, 4).

Finally, according to the values in the wo_S and wo_P sets, the semantic similarity, *semSim*, of *S* and *P* is calculated. According to the notation in Equation 1, the semantic similarity value between *S* and *P* is obtained as 0.7272. *LCS* is the

longest common subsequence of *S* and *P*, |*S*| is total number of words in *S*, and |*P*| is total number of words in *P*. If *semSim*(*S*, *P*) ≥ predefined threshold, then it is identified that *P* is plagiarized from *S*. According to the example, if a predefined threshold is set less than 0.7272, it is concluded that S has been plagiarized from P.

4.2.4.3. Error Analysis

In some cases, the proposed method performs poor performance because of two reasons: 1) let *S* and *P* be two sentences where *P* is plagiarized from *S*. If *S* is converted to its passive voice or the word order of the *P* is changed, the syntactic structure of the *S* is also changed. Hence, since the proposed method is a POS tag-based approach, it could fail to identify *P*. 2) in most cases of plagiarism, the words of *P* are replaced with their synonyms. In this case, the matching status between two words of *S* and *P* depends on the probability calculated by the vectors in Word2Vec. The capacity of the vector space and whether it fully covers the semantic relatedness between the words affect the similarity score. Therefore, two words that are actually similar may not match each other based on vector size. Since LCS is calculated as the maximum number of consecutive matching words of *S*, which also appears in *P*, the proposed method may not be able to identify *P*.

Let *P* be the passive conversion of *S* and is created by the words of *S* which are replacing with their synonyms or near synonyms. An example [160] of synonym or near-synonym replacement with active-to-passive conversion is shown in Table 4.10. As can be seen from the conversion, although the meaning of the P_{pas} remains the same, the syntactic structure and words of it are completely changed compared to *S*. In particular, replacing words with synonyms allows to avoid verbatim plagiarism. Therefore, it will be imperative to use methods for calculating semantic similarity between words.

As can be seen in Table 4.10, each token of *S* and *P* is tagged with their part-of-speech tags. It is clear from Table 4.10 that the number of common POS tag n-grams between *S* and P_{act} is greater than between *S* and P_{pas}. In cases where plagiarism is made by converting the source sentence to passive form, the

probability of the source sentence being retrieved as a candidate decreases. Therefore, the method based on POS tag n-grams performs better performance in active format sentences.

Table 4.10. Synonym Replacement with Active to Passive Conversion

	S	P_{act}	P_{pas}
Sentence	The award winning chef prepares each meal with loving care.	The honor triumphing cook gets ready every supper with adoring attention.	Every supper is ready with adoring attention by the honor triumphing cook.
POS tags of the sentence	The/**DT** award/**NN** winning/**VBG** chef/**NN** prepares/**VBZ** each/**DT** meal/**NN** with/**IN** loving/**VBG** care/**NN** ./.	The/**DT** honor/**NN** triumphing/**VBG** cook/**NN** gets/**VBZ** ready/**RB** every/**DT** supper/**NN** with/**IN** adoring/**VBG** attention/**NN** ./.	Every/**DT** supper/**NN** is/**VBZ** ready/**JJ** with/**IN** adoring/**VBG** attention/**NN** by/**IN** the/**DT** honor/**NN** triumphing/**VBG** cook/**NN** ./.
POS tag 3-grams of the sentence	**3na nan an4** n43 43n **3ni nia ian** 3na **nan** an4 n43 43n 3ni nia **ian**	**3na nan an4** n4r 4r3 r3n **3ni nia ian**	3n4 n4j 4ji jia **ian** ani ni3 i3n 3na **nan**

The semantic similarity between each word pair of *S* and P_{pas} is calculated, as shown in Table 4.11. Each word in *S* is numbered with the sequence number of the word in P_{pas} that it matches. Likewise, each word in P_{pas} is numbered with the sequence number of the word it matches in *S*. Let w_S and w_{pas} be the sets of word order of *S* and P_{pas} respectively.

According to the enumeration method, w_S = {6, 6, 8, 3, 1, 2, 4, 5} and w_{pas} = {5, 6, 4, 7, 8, 1, 2, 3}. It is clear from the definition of LCS that *LCS* (*S*, P_{pas}) = 4 (6, 8, 1, 2). The semantic similarity of *S* and P_{pas} is equal to 0.5 as can be calculated from the illustration of Equation 1. On the other hand, when a comparison is done between w_S and P_{act}, the set of word orders occur as follows: w_S = {1, 1, 3, 4, 6,

7, 8, 9} and w_{act} = {1, 2, 3, 4, 4, 5, 6, 7, 8}. In this case, *LCS* (*S*, P_{act}) = 6 (1, 3, 4, 6, 7, 8) and the semantic similarity of *S* and P_{act} is equal to 0.75.

Table 4.11. Semantic Similarity between the Word Matrix

	the	award (1)	winning (2)	chef (3)	prepares (4)	each (5)	meal (6)	with	loving (7)	care (8)	.
every (1)	-	0.5086	0.6706	0.5359	0.8493	**0.9784**	0.7014	-	0.5808	0.8469	-
supper (2)	-	0.5707	0.5876	0.7202	0.7558	0.6805	**0.8779**	-	0.7347	0.7024	-
is	-	-	-	-	-	-	-	-	-	-	-
ready (3)	-	0.5662	0.7308	0.6240	**0.9267**	0.8421	0.7507	-	0.7341	0.8484	-
with	-	-	-	-	-	-	-	-	-	-	-
adoring (4)	-	0.4906	0.6309	0.5391	0.6846	0.5154	0.5764	-	**0.7350**	0.5994	-
attention (5)	-	0.6246	0.7522	0.7021	0.8504	0.7774	0.6038	-	0.7235	**0.8754**	
the	-	-	-	-	-	-	-	-	-	-	-
honor (6)	-	**0.8324**	**0.7748**	0.7021	0.7465	0.6575	0.5767	-	0.6973	0.7272	
triumphing (7)	-	0.5677	**0.6689**	0.5010	0.6331	0.6214	0.5642	-	0.6240	0.5685	
cook (8)	-	0.6909	0.6569	**0.7278**	0.6994	0.6507	0.6238	-	0.7031	0.7337	
.	-	-	-	-	-	-	-	-	-	-	-

As can be seen from the semantic similarity matrix from the example, the performance of LCS with Word2Vec decreases in the sentences that are plagiarized by changing the word order. On the other hand, the proposed method shows good results in the sentences whose words are replaced with their synonyms.

As can be seen from Table 4.10, although *S* and P_{act} have six 3-grams in common, only two 3-grams (nan and ian) are in common between *S* and P_{pas}. So, the probability of P_{pas} being retrieved as the candidate of *S* is low. A sentence that normally has the same meaning and is the passive of the other will be eliminated before the detailed analysis. Therefore, active passive conversion has a negative effect on system performance. However, when *S* and P_{pas} are compared semantically, a similarity score of 0.75 is obtained. This score can be considered as a high value in terms of the similarity of the two sentences.

As can be seen from the results in this example, replacing the words of a sentence with synonyms and converting it to its passive form has no effect on semantic similarity. The main problem here is that the change in the structure of that sentence due to the active-passive conversion prevents it from being found as a candidate sentence.

5. DATA ANALYSIS

5.1. PAN-PC-11 Corpus

The experiments are conducted with PAN-PC-11 corpus where the plagiarism cases have been added to documents manually and automatically. Manual plagiarism cases, called simulated, have been created using a commercial tool, which is designed for studying crowdsourcing data by name of *Amazon Mechanical Turk* [17]. On the other hand, automatic plagiarism cases, called *artificial*, have been automatically created by a computer program with the following three obfuscation strategies: *random text operations*, *semantic word variations* and *POS-preserving word shufflings* [136].

Table 5.1. Document Statistics in the PAN-PC-11

Document Purpose and Counts					
Source documents	11093	50%	Suspicious documents	11093	50%
- English	10420	47%	- without plagiarism	5546	25%
- German	471	2%	- with plagiarism	5547	25%
- Spanish	202	1%	artificial none or very few	114	
			artificial low	2369	
			artificial high	2404	
			simulated	105	
			translation	555	

Plagiarism per Document			**Document length**		
Hardly	5%-20%	57%	Short	1-10 pp.	50%
Medium	20%-50%	15%	Medium	10-100 pp.	35%
Much	50%-80%	18%	Long	100-1000 pp.	15%
Entirely	>80%	10%			

Let P_{plg} be a plagiarized passage and created as follows from a source passage P_{src}:

Random text operations: P_{plg} is generated by the words of P_{src} are shuffling, removing, inserting or replacing randomly.

Semantic word variation: P_{plg} is generated by the words of P_{src} are replacing randomly with their synonyms, antonyms, hyponyms or hypernyms.
POS-preserving word shuffling: P_{plg} is generated by random shuffling of P_{src} while preserving the POS sequence of P_{src}.

Table 5.2. Plagiarism Case Statistics in the PAN-PC-11

Obfuscation		Case Length		
Artificial none or very few	18%	Short	<150 words	35%
Artificial low	32%	Medium	150-1150 words	38%
Artificial high	31%	Long	>1150 words	27%
Simulated	8%			
Translation	11%			

Table 5.1 and Table 5.2 provide an overview of document statistics and plagiarism cases in the corpus [51]. PAN-PC-11 consists of 11,093 source documents and 11,093 suspicious documents for the external plagiarism detection task. Since the proposed method focuses on mono-lingual plagiarism detection, non-English documents are excluded from this study and experiments are conducted with 10,420 source documents and 4,992 suspicious documents consisting of “none, low, high and simulated” plagiarism cases.

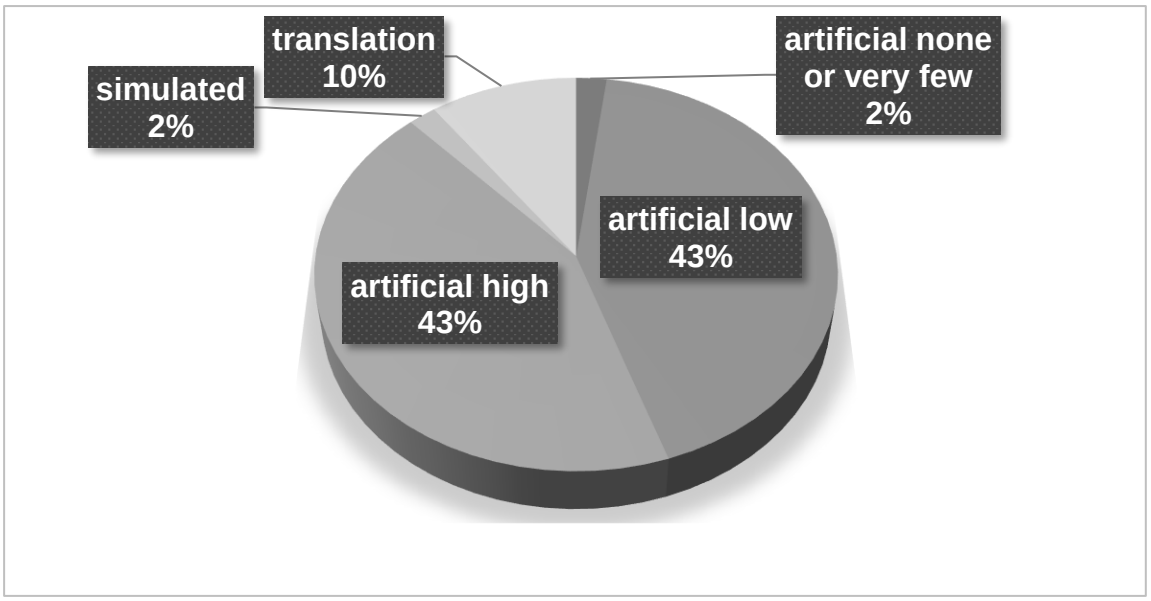

Figure 5.1. Distribution of the Plagiarism Cases by Document

Figure 5.1 shows the distribution of the cases of plagiarism in the corpus on a document basis according to paraphrasing types. Plagiarism was done in 5547 documents, which corresponds to approximately half of the total of 11093 suspicious documents in the corpus. As can be seen from the Figure 5.1, the most plagiarism types artificial high and low were performed in these 5547 suspicious plagiarized documents.

The PAN-PC-11 corpus contains *internal plagiarism detection* and *external plagiarism detection* cases. Since the proposed method is an external plagiarism detection system, the experiments are performed in this portion of the corpus. There are three different obfuscation levels of *artificial* plagiarism cases: *none (or very few)*, *low* and *high*. This study focuses on the mono-lingual perspective with four types of plagiarism cases that consists of none, low, high and simulated.

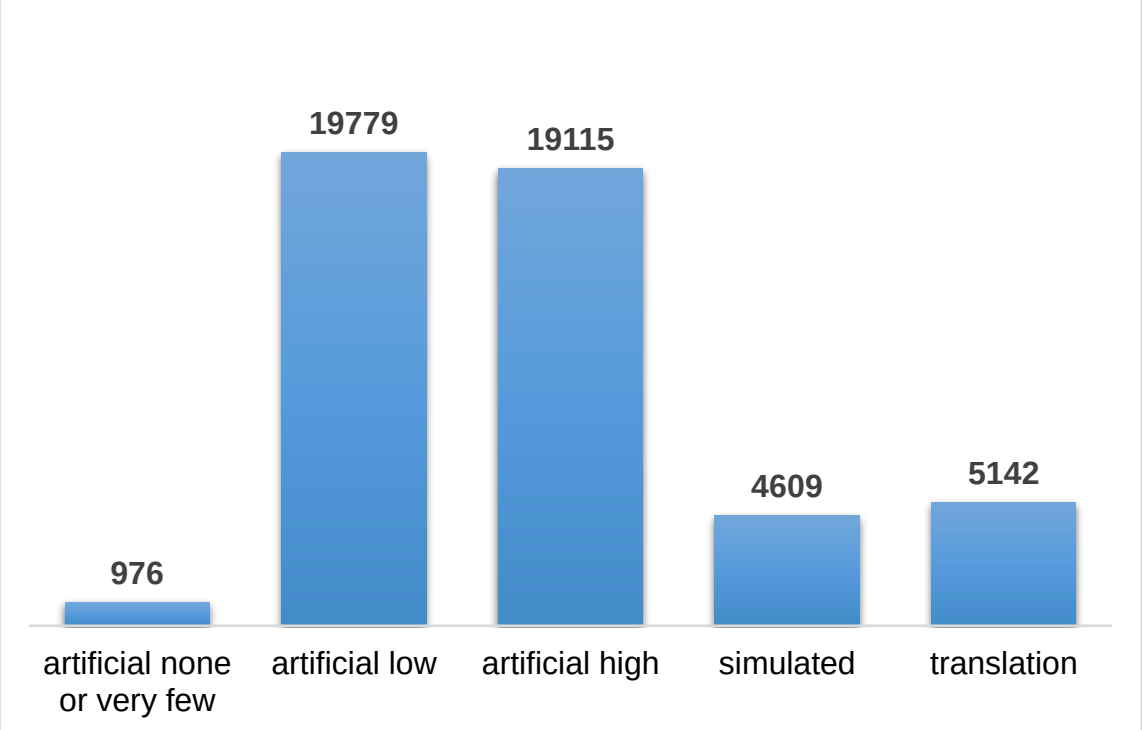

Figure 5.2. Number of Cases by Plagiarism Types

Figure 5.2 shows the total number of plagiarism cases by plagiarism types. In the corpus, all plagiarism cases in a suspicious document are written into an XML file

line by line. As seen in Table 5.3, each row represents a case of plagiarism. A plagiarism case consists of the following information:

- Plagiarism type,
- Level of obfuscation,
- Language of suspicious document,
- At what position the plagiarism begins in the suspicious document,
- How many characters in length in the suspicious document,
- From which source document the plagiarism was made,
- At what position in the source document the plagiarism begins,
- How many characters in length in the source document,
- In what language the source document is.

As can be seen from the Table 5.3, there can be more than one case of plagiarism in a suspicious document. In addition, plagiarism may have been made from more than one source document and more than one plagiarism case may have been made from a source document. However, each plagiarism case can consist of one or more sentences.

5.2. Evaluation Metrics

Performance of the proposed method is evaluated on four measures: *precision*, *recall*, *granularity* and *plagdet*. *S* denotes the plagiarism set in the dataset and *R* be the detections set found by the plagiarism detector. A plagiarism case $s = \langle s_{plg}, d_{plg}, s_{src}, d_{src} \rangle$, $s \in S$, is the set of *s* includes the characters of d_{plg} and d_{src}, constituting the passages s_{plg} and s_{src}. In the same way, a new plagiarism detector $r \in R$ is denoted as r. According to this representation, precision and recall can be measured as shown in Equation 2 and Equation 3 [51].

Table 5.3. Features of Plagiarism Cases in a Suspicious Document

Type	Obfuscation	Suspicious language	Suspicious offset	Suspicious length	Source reference	Source language	Source offset	Source length
artificial	low	en	38040	18546	source-document05720.txt	en	144290	21840
artificial	low	en	101939	17023	source-document00374.txt	en	86163	18309
artificial	low	en	158969	13398	source-document05843.txt	en	6431	16571
artificial	low	en	227939	18685	source-document05720.txt	en	97619	19530
artificial	low	en	356718	13964	source-document02977.txt	en	253550	18921
artificial	low	en	375471	2028	source-document05843.txt	en	739	1985
artificial	low	en	426670	24545	source-document00374.txt	en	166102	24832
artificial	low	en	485930	3532	source-document00573.txt	en	7399	3590
artificial	low	en	494977	757	source-document05843.txt	en	3812	767
artificial	low	en	498164	804	source-document02977.txt	en	8863	802
artificial	low	en	501737	1487	source-document03177.txt	en	72326	1483
artificial	low	en	507294	488	source-document02977.txt	en	348902	499

Precision means what portion of the detections found by system are plagiarism cases. On the other hand, recall means what portion of the plagiarism cases are identified by the system. Precision and recall are inversely proportional to each other. This means there is a relationship between precision and recall in which as the value of one increases, the value of the other decreases [137].

In some circumstances, plagiarism detectors may find multiple detections for a single case of plagiarism. As shown in Equation 4, in order to address this undesirable situation, a metric called *granularity* is measured.

The three measures mentioned above do not make it possible to do an exact ranking between them. For this reason, these three metrics are merged into one overall measure called *plagdet* defined in Equation 5.

$$precision(S,R) = \frac{1}{|R|} \sum_{r \in R} \frac{|\cup_{s \in S} (s \sqcap r)|}{|r|} \quad (2)$$

$$recall(S,R) = \frac{1}{|S|} \sum_{s \in S} \frac{|\cup_{r \in R} (s \sqcap r)|}{|s|} \quad (3)$$

where,

$$s \sqcap r = \begin{cases} s \cap r & \text{if } r \text{ detects } s, \\ \emptyset & \text{otherwise.} \end{cases}$$

$$granularity(S,R) = \frac{1}{|S_R|} \sum_{s \in S_R} |R_s| \quad (4)$$

where $S_R \subseteq S$ are the plagiarism cases found by detections in R, and $R_s \subseteq R$ are detections of s.

$$plagdet(S,R) = \frac{F_1}{log_2(1+gran(S,R))} \quad (5)$$

where F_1 is the equally-weighted harmonic mean of precision and recall.

Figure 5.3 shows a representation of the character sequence of the original and plagiarized text fragments in the corpus [142]. A document in the corpus is treated as a character sequence. The plagiarized sections of a suspicious document are denoted *S*. As can be seen from the Figure 5.3, there may be more than one plagiarized sections such as s1, s2 and s3 in a document. The location of each plagiarized section in the document and its length as characters are certain. On the other hand, plagiarized sections detected by a plagiarism detection algorithm are represented as *R*.

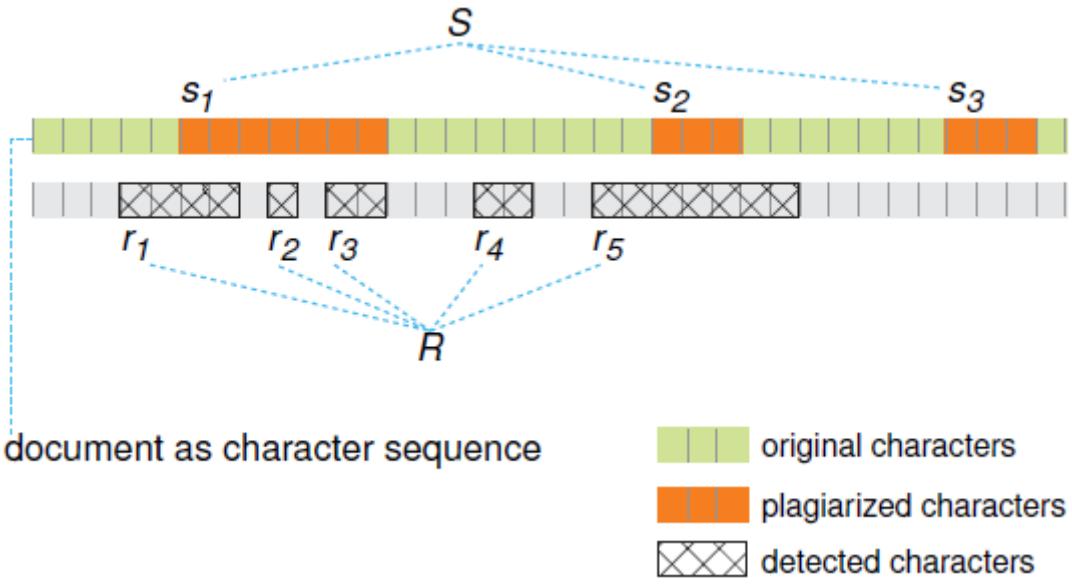

Figure 5.3. Illustration of the Character Sequence

As can be seen from the Figure 5.3, 5 different plagiarism sections from r1 to r5 are returned as results. The intersection of the plagiarized S and the sections of R found by the detection algorithm are the sections where plagiarism is correctly detected. For example, it is clear from the Figure 5.3 that the r4 section, which is identified as plagiarism, is not actually plagiarism. Some characters of the plagiarized section of s1 could be detected by the r1, r2, r3, and the s2 section by r5. However, there is no result returned that the s3 section is plagiarism.

6. EXPERIMENTAL EVALUATION

6.1. Experimental Environment

This section provides an overview of the experimental setup of the proposed study. The experiments were performed on the PAN-PC-11 corpus. Since the proposed system is a mono-lingual plagiarism detection system, the German and Spanish texts in the corpus were excluded from the experiments. All the remaining English texts were included in this study and no selected subsets were used. The performance of the system has been evaluated by experiments carried out with all original and suspicious English documents. The total size of these documents is 3.77 GB. It is mainly aimed to improve the detection performance of plagiarism types of artificial none, low, high and simulated in the PAN-PC-11 corpus.

Experiments were carried out simultaneously on two different hardware. The first hardware is running on the Windows 8.1 operating system with 6 GB RAM and 3.00 GHz processor. The other has 8 GB RAM and 3.40 GHz processor running on the Windows 10 operating system. The process of candidate sentences retrieval from the source index was run on both hardware with multitasking. The proposed method accesses OpenNLP binary files and Stanford POS tag libraries stored in the file system. In addition, in the calculation of semantic similarity between words, access to a vector file of approximately 121 MB was provided. The source codes of the proposed method were implemented in Java. The design of implementation allows to performing experiments with different parameters or threshold values. Thus, the experimental results were analyzed using different parameters. Since the proposed system is an unsupervised method, no further training process was performed. Since the system developed with this thesis uses a corpus stored in the computer memory, there is no need for an operation to be performed on the network. Therefore, any features such as bandwidth or network speed that may affect the performance of the system are beyond the scope of this study. The speed and duration of the transactions may vary depending on the capacity of the computers.

6.2. Experimental Settings

As can be seen in Table 6.1, the experiments are conducted with four parameters in order to evaluate the performance of the proposed method.

n: the number of *n-grams* used to create part-of-speech tag n-grams of the sentences (mentioned in Section 4.2.2). The set of values of $n = \{3, 4, 5\}$ is used for the proposed method.

c: the number of source sentences returned as *candidates* from the index in the candidate sentences retrieval phase (mentioned in Section 4.2.3). The experiments are carried out so that the values of $c = \{1, 5\}$.

d: the *difference* between the numbers of source and suspicious sentences of two consecutive candidates (mentioned in Section 4.2.3). The experiments are conducted with the values of $d = \{3, 5, 8, 10\}$ to evaluate the performance of the proposed method.

sim: the threshold that is used in computing the Longest Common Subsequence (LCS) between the sentences $\mathrm{S_{src}}$ and $\mathrm{S_{susp}}$ (mentioned in Section 4.2.4.2). If the LCS (S_{susp}, S_{src}) is greater than the *sim*, $\mathrm{S_{susp}}$ is identified as plagiarized from $\mathrm{S_{src}}$. The set of values of *sim* = {0.6, 0.7, 0.8, 0.9} are used in the experiments.

Table 6.1. The Set of Values of the Parameters

Parameter	Description	Set of Values
n	n-grams	{3, 4, 5}
c	candidate	{1, 5}
d	difference	{3, 5, 8, 10}
sim	similarity threshold	{0.6, 0.7, 0.8, 0.9}

In the experiments, performance of two decision-making methods, $\mathrm{POSNG_{PD}}$ (mentioned in Section 4.2.4.1) and $\mathrm{POSNG_{PD+SSBS}}$ (mentioned in Section 4.2.4.2) are compared. $\mathrm{POSNG_{PD}}$ is a syntactic based approach that aims to detect plagiarism based on the common part-of-speech tag n-grams between source and suspicious sentences. On the other hand, $\mathrm{POSNG_{PD+SSBS}}$ is the method that

contains the $POSNG_{PD}$ and also provides the semantic similarity between source and suspicious sentences.

According to the experimental results, the best performance of the proposed method is obtained in the $POSNG_{PD+SSBS}$ method with the parameters n = 3, c = 1, d = 10 and sim = 0.9. The comparison results of the proposed method, using these parameters, with PAN11 detectors are shown in Table 6.2 for each plagiarism type.

6.3. Evaluation Results and Discussion

It is aimed to improve the results of PAN11 detectors in plagiarism types of artificial none, low, high and simulated with the experiments performed within the scope of this thesis. For this reason, the experiments were carried out using different parameters and values mentioned in Section 6.2 to evaluate the performance of the proposed method and to observe what kind of results are obtained according to the changing parameters. In addition, the change that occurs with the inclusion of the semantic similarity algorithm in the proposed method, which is syntactically based, has also been investigated. The evaluation, discussion and detailed explanations regarding the experiment results performed with each different parameter are included in the subsections. The results obtained from the experiments have been demonstrated that the proposed method succeeded significantly better results than participating detectors of PAN11 in different evaluation metrics in high and low obfuscation plagiarism cases.

6.3.1. Comparison Results with PAN11 Detectors

The performance of the proposed method is compared to detectors competing in PAN11 in the overall plagdet and other measures of recall, precision and granularity. Table 6.2 shows the performance of the proposed method and of the detectors (denoted as D1 to D9) whose plagiarism detection approaches are given in Table 3.6. The results obtained from the experiments demonstrate that the proposed $POSNG_{PD+SSBS}$ method produced highly competitive results.

Table 6.2. Comparison Results of Proposed Method with PAN11 Detectors

Plagdet										
Paraphrasing	$POSNG_{PD+SSBS}$	D1	D2	D3	D4	D5	D6	D7	D8	D9
artificial none	0.89	0.97	0.85	0.91	0.81	0.81	0.60	0.66	0.40	0.01
simulated	0.45	0.49	0.50	0.47	0.17	0.32	0.05	0.29	0.02	0.00
artificial low	0.80	0.71	0.60	0.55	0.25	0.32	0.35	0.29	0.15	0.00
artificial high	0.54	0.15	0.13	0.05	0.04	0.03	0.01	0.00	0.00	0.00
Recall										
Paraphrasing	$POSNG_{PD+SSBS}$	D1	D2	D3	D4	D5	D6	D7	D8	D9
artificial none	0.95	0.97	0.90	0.88	0.87	0.79	0.70	0.81	0.72	0.01
simulated	0.31	0.33	0.36	0.31	0.09	0.20	0.03	0.18	0.02	0.00
artificial low	0.83	0.56	0.58	0.42	0.15	0.25	0.32	0.23	0.18	0.00
artificial high	0.43	0.08	0.08	0.03	0.02	0.02	0.01	0.00	0.00	0.00
Precision										
Paraphrasing	$POSNG_{PD+SSBS}$	D1	D2	D3	D4	D5	D6	D7	D8	D9
artificial none	0.85	0.97	0.84	0.94	0.75	0.82	0.53	0.58	0.32	0.08
simulated	0.90	0.99	0.96	0.98	0.86	0.93	0.38	0.80	0.43	0.01
artificial low	0.89	0.95	0.90	0.93	0.74	0.92	0.62	0.57	0.45	0.01
artificial high	0.85	0.77	0.64	0.67	0.48	0.18	0.39	0.04	0.01	0.00
Granularity										
Paraphrasing	$POSNG_{PD+SSBS}$	D1	D2	D3	D4	D5	D6	D7	D8	D9
artificial none	1.00	1.00	1.02	1.01	1.00	1.00	1.00	1.02	1.14	4.64
simulated	1.02	1.01	1.06	1.01	1.00	1.03	1.05	1.01	2.71	1.31
artificial low	1.09	1.00	1.27	1.08	1.01	1.34	1.33	1.22	2.29	1.32
artificial high	1.08	1.00	1.19	1.02	1.01	1.12	1.16	1.01	1.21	1.31

As can be seen from the results, the proposed method achieved the best performance in the overall plagdet score of low and high obfuscation paraphrasing. In particular, the plagdet result in high obfuscation cases has achieved significant success over other methods in Table 6.2. While the best plagdet performance among PAN11 systems was 0.15 in [120], the proposed method succeeded in obtaining a plagdet score of 0.54. In none and simulated cases, the proposed method has obtained competitive results with other methods. In addition, the plagdet score of the less complex none and low plagiarism types is close to the maximum value of 1.00, while the more complex simulated and high plagiarism types have lower overall performance than the none and low cases.

The results also prove that $POSNG_{PD+SSBS}$ method obtained significantly best recall performances in the low and high obfuscation paraphrasing with the scores of 0.83 and 0.43 respectively. On the other hand, the proposed method obtained competitive recall results in none and simulated plagiarism types. Because the none plagiarism type involves less complex cases, it is easier to detect than other types. Therefore, a high recall performance of 0.95 is achieved. Likewise, the recall performance of low obfuscation cases is approximately 43% more successful than the PAN11 system [121], which has the best recall performance. On the other hand, the recall performance of simulated cases appears to be the lowest in the four plagiarism categories. Since the simulated cases in the corpus were created manually, modifications were made in the texts caused changes in the sentence structure. Words that are randomly added to or removed from sentences, especially stop words, have caused the structure of n-grams of the suspicious sentences to differ from that of the source sentences. Therefore, this study based on POS tag n-grams showed poor recall performance for the simulated plagiarism cases. Cases of high complexity are mostly obtained by replacing words with synonyms, as they are generated automatically by a computer program. Therefore, high obfuscation cases, in which the POS structure is preserved even though all the words are changed, can be detected significantly with the POS tag n-grams approach. However, the recall performances of all detectors in PAN11 are seen unstable and range from very low to medium except none obfuscation paraphrasing type.

The precision results also show that POSNG$_{PD+SSBS}$ method produced best result in the high obfuscation paraphrasing and achieves 10% better performance than the top detector [120]. The precision performances of the first three methods [120-122] are higher in plagiarism types of none, low and simulated than high plagiarism cases. This shows that even the detector [120] with the best precision performance of high obfuscation type in PAN11 achieves an average precision result. Another remarkable point in precision results is that detectors with low recall performance of simulated type have the highest precision. This shows that the detectors have high performance that the cases detected as plagiarism for the simulated type are indeed plagiarism.

It is a matter of debate whether plagiarism in manually edited texts or highly obfuscated texts that are automatically modified by a computer program should be detected first. This completely depends on the level of complexity. While very simple changes can be made on the text by a human, the text can also be modified with a high degree of complexity by a computer program. Likewise, the opposite is true. When the plagdet and recall results in Table 6.2 are examined, it is seen that the simulated performances of the detectors are better than the high-obfuscated plagiarism cases. This leads to the conclusion that high obfuscated cases are more difficult to detect than simulated cases. Therefore, within the scope of this thesis, the focus is primarily on improving the performance of the detectors for the high obfuscated cases, which are more difficult to detect than the results in Table 6.2.

It is seen that the granularity performances of two detectors [120, 123] are the best and change between 1.00 and 1.01, while the others show changeable results. The best granularity performance of the proposed method, 1.00, is obtained in the type of none obfuscated plagiarism cases. The detectors in PAN11 also produced successful results for this type of plagiarism, except for one, and four detectors achieved the best performance, as in the proposed method. The granularity performance of the proposed method in type of the simulated plagiarism cases is also competitive. On the other hand, compared to the detectors in PAN11, although the proposed method achieved results close to 1.00, which is the best score, the granularity performance is average in the types

of low and high obfuscated plagiarism cases. This situation indicates that in some cases, the proposed method finds multiple detections for a single case of plagiarism.

Table 6.3. A Plagiarism Case with High Obfuscation from PAN-PC-11

	Source	Suspicious
Passages	**And when Mr. Idle and the** seven unlabouring neophytes, ranged in order, as a class, with their backs considerately placed against a screen**, had begun, in rotation, to** read the exercises **which they had not** written, even then, **each Bencher,** true to the great lazy principle of the whole proceeding**, stopped each neophyte before he had stammered through his first line, and** bowed to him, and told him **politely that he was a barrister** from that moment.	**And when Mr. Idle and the** hand of newcomer, had run in summons, as collection, with their dorsum considerately be put to blind**, had begin, in circumvolution, to** the effort **which they had not** write, yet so, but **each Bencher,** truthful authorities to great and lazy rule of fractional continue**, existed a recruit before he had bumble through their first formation, and** submit to how him do, and state it could be **courteously that he was a point**.
POS tag 3-grams of the passages	cWg Wgg ggc gcd cdo doj oja ja1 a1b 1bi bin in1 n1i 1id idn dn1 n1i 1iP iPa Par arV rVi Vid idn dn1 n1b 1bV bV1 V1i 1in in1 n1t 1tv tvd vda daw awp wpb pbr brV rV1 V1r 1rr rr1 r1d 1dg dg1 g1j 1jt jtd tdj djj jjn jni nid idj djn jn1 n1b 1bd bdn dni nip ipb pbV bVi ViP iPj Pjn jn1 n1c 1cb cbt btp tp1 p1c 1cb cbp bpr pri rip ipb pbd bdn dni nid idn dn0	cWg Wgg ggc gcd cdn dni nin in1 n1b 1bV bVi Via ia1 a1i 1in in1 n1i 1iP iPn Pnr nrv rvV vVt Vtj tj1 j1b 1bv bv1 v1i 1in in1 n1t 1td tdn dnw nwp wpb pbr brv rv1 v1r 1rr rr1 r1c 1cd cdg dg1 g1j 1ja jat atj tjc jcj cjn jni nij ijS jS1 S1b 1bd bdn dni nip ipb pbV bVi ViP iPj Pjn jn1 n1c 1cv cvt vtW tWp WpS pS1 S1c 1cn cnp npm pmv mvr vri rip ipb pbd bdn dn0

Table 6.3 shows an example [158] of high-obfuscated type of plagiarism case produced with POS tag 3-grams. The obfuscation strategies, mentioned in Section 5.1, are applied to the suspicious document to differentiate it from the source document. Despite these obfuscation strategies, the source and suspicious documents are syntactically similar and contain a sequence of common POS tag n-grams, as shown in Table 6.3.

The experimental results confirm that the proposed approach in this study accomplishes the best performance for identifying high obfuscated sections that are much more difficult to be detected than none, low and simulated plagiarism cases. The fact that the proposed method is based on syntactic analysis with POS tag n-grams ensures successful detection of high obfuscated passages. An example of this situation can be seen in Table 6.3 with an example of high-obfuscated plagiarism case from the PAN-PC-11 corpus.

6.3.2. Comparison Results of N-Grams

The performance of the proposed method is compared with different n-grams as mentioned in Section 4.2.2. The experiments are performed with 3, 4 and 5-grams. Longer n-gram sequences reveal more similar structure about the two different contents. However, since longer n-gram series occur less frequently, it is likely that the right candidates does not rank high, especially in the candidate retrieval process. On the other hand, shorter n-grams are also very repetitive, so a certain number of common bigrams or 2-grams can match even for two contents that are actually not very syntactically similar. In this case, a large number of candidates with common n-grams return at the end of the searching task. For these reasons, the experiments are performed using 3, 4, and 5 grams, which are considered the optimal length.

Figure 6.1 shows the plagdet performances in four types of plagiarism cases. As can be seen in Figure 6.1 (b), (c) and (d), 3-grams produced the best results in the types low, high and simulated plagiarism cases. The performance of the plagdet increases from 5-grams to 3-grams in these three types of plagiarism cases. Because longer n-grams occur less frequently, it is less likely to have 5-grams in common between the source and suspicious sentence pairs.

On the other hand, it is clear from Figure 6.1 (a), the plagdet performance of the 5-grams is the best for none obfuscation plagiarism cases. In both $POSNG_{PD}$ and $POSNG_{PD+SSBS}$ methods, 5-grams produced better results than 3-grams and 4-grams. The performance of 3-grams of none obfuscated plagiarism cases is the lowest in the $POSNG_{PD}$ method. In the $POSNG_{PD+SSBS}$ method, the result of 3-grams and 4-grams are very close to each other. Depending on this result, longer

n-gram sequences are thought to perform better performance in none obfuscated plagiarism type.

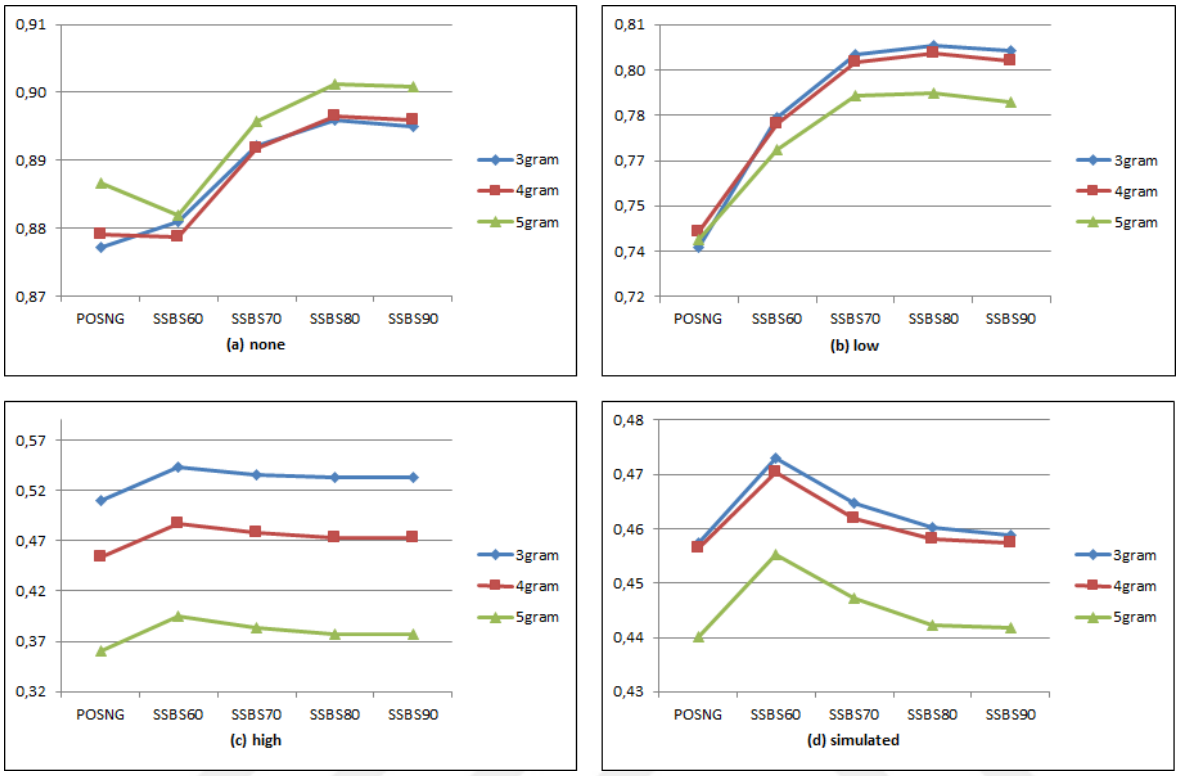

Figure 6.1. Comparison of the plagdet Using Different N-Gram Values

As can be seen in Figure 6.1 (a) and (b), the $POSNG_{PD+SSBS}$ method improves the performance compared to the $POSNG_{PD}$ method in none and low obfuscation plagiarism cases. The obfuscation complexity of none and low plagiarism cases is less than high and simulated cases. None and low obfuscation plagiarized cases mostly include the same words as the source passages without replacing them. In these cases, the words of the plagiarized sentences mostly remain in their original form without any semantic or structural changes. In the $POSNG_{PD}$ method, two consecutive candidate pairs are analyzed according to the difference of their sentence numbers and no semantic similarity is performed. On the other hand, in the $POSNG_{PD+SSBS}$ method, the pair of suspicious and the candidate source sentences is also analyzed semantically. Since, the plagiarized passages are mostly generated by adding or removing stop words of the source passages randomly, such text processing does not affect the recall performance of the $POSNG_{PD+SSBS}$ method. Because before the semantic similarity operations

are performed, the stop words are removed from the sentences. For this reason, the $POSNG_{PD+SSBS}$ method obtained better plagdet performance than $POSNG_{PD}$ method in none and low obfuscation plagiarism cases. As the similarity threshold approaches zero, sentences that are not actually plagiarized will be perceived as plagiarized, thus increasing precision and decreasing recall. Conversely, as the similarity threshold becomes closer to 1.0, the sentences that are actually plagiarized will not be detected as plagiarism, so the recall increases while the precision decreases.

Figure 6.1 (c) and (d) shows that in cases of high and simulated plagiarism when the similarity threshold is ≥ 0.8, the $POSNG_{PD}$ method is as successful as the $POSNG_{PD+SSBS}$ method without performing semantic similarity operation. This is due to the weakness of LCS using it with Word2Vec. The obfuscation strategies applied to cases of high and simulated types of plagiarism are more complex than none and low plagiarism types. In cases of high and simulated plagiarism, the words of the plagiarized sections are randomly replaced with their synonyms or the sentence structure is changed by shuffling the words. Therefore, almost all words of a plagiarized sentence may differ from the original sentence. In this case, the correct matching of the two words depends on the vector space used in Word2Vec. Therefore, two words that are actually similar may not match each other due to the similarity score calculated in Word2Vec. On the other hand, if the similarity threshold is between 0.6 and 0.8, it is shown that $POSNG_{PD+SSBS}$ outperforms the $POSNG_{PD}$ method.

6.3.3. Comparison Results of Number of Candidate Sentences

The performance of the proposed method is compared between the numbers of source sentences, denoted as *c*, that returned as candidates. The experiments are conducted with the values where *c* is 1 and 5. This means that for each suspicious sentence in the experiments, the first candidate and the top five candidates are taken respectively.

Figure 6.2 presents the plagdet performances in four types of plagiarism cases. As can be seen in Figure 6.2 (a), (b), (c) and (d), the value of candidate source sentences, $c = 1$, produced better results than $c = 5$ in all types of plagiarism

cases. When the number of candidates is five, the size of the candidate list created for all suspicious sentences also increases fivefold. This means that all five candidates for a suspicious sentence are added to the candidate list.

Figure 6.2. Comparison of the plagdet of Candidate Source Sentences

As mentioned in Section 4.2.3, candidate sentences are sorted by document number and source sentence number. In this case, two consecutive candidates, which are not actually plagiarized, can be identified as plagiarism, if the difference between the numbers of source and suspicious sentences is less than a threshold. Therefore, if the number of candidates increases, the performances of the precision and plagdet decrease. However, due to the large number of candidates, the number of real detections also increases. In this case, recall performance also improves. On the other hand, the performances of the plagdet and precision improve only when the first candidate is selected. The source sentence, which is the first candidate of the suspicious sentence, has the highest score for that suspicious sentence in the index. For this reason, it is thought that the probability of plagiarism of the first candidate is higher than the subsequent ranks.

6.3.4. Comparison of Difference of Candidate Sentences

The performance of the proposed method are compared according to the *difference* between the number of source and suspicious sentences of two consecutive candidates. The set of values of $d = \{3, 5, 8, 10\}$ was applied in the experiments.

Figure 6.3 shows the plagdet performances in four types of plagiarism cases. As can be seen in Figure 6.3 (b), (c) and (d), the value of *d* = 10 obtained the best performance in the types of low, high and simulated plagiarism cases. The plagdet performance improves when the value of *d* is from 3 to 10 in these three types of plagiarism cases.

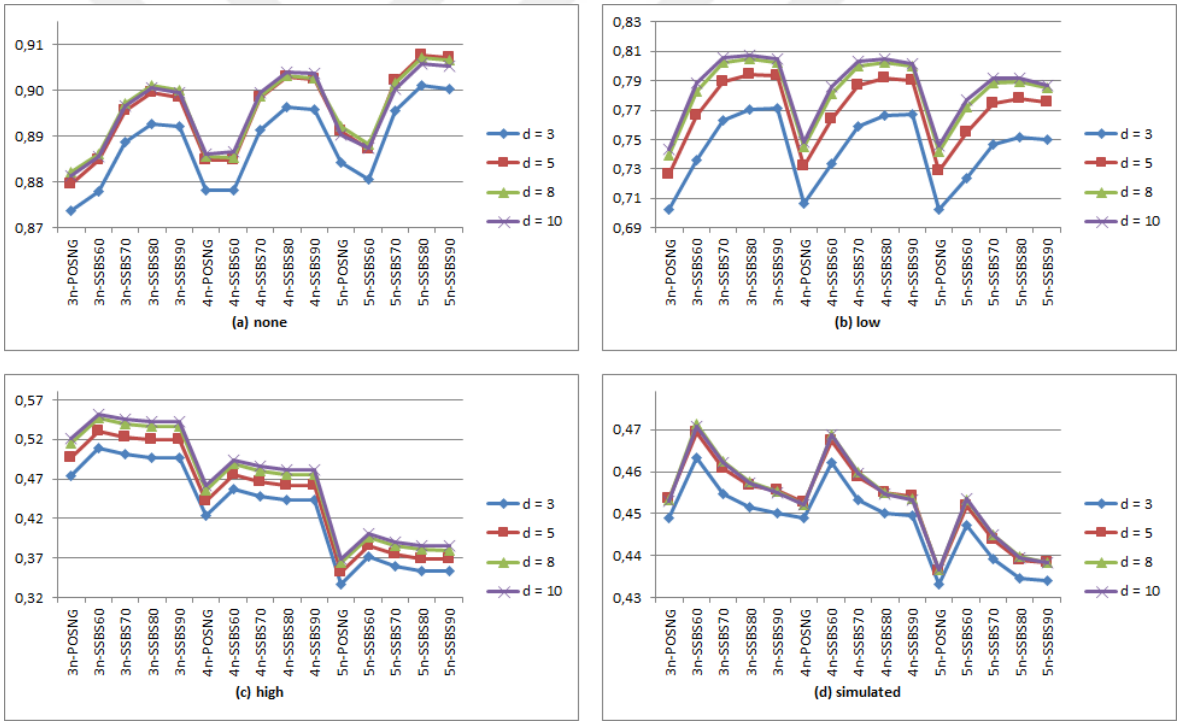

Figure 6.3. Comparison of the plagdet by Difference Value

As can be seen from Figure 6.3 (a), when the value of d = 3, the plagdet of the none obfuscated plagiarism type produced the lowest performance as in the other three types of plagiarism cases. On the other hand, the plagdet performance of the set of $d = \{5, 8, 10\}$ is very close to each other. According to the results, the plagdet performance of *d* = 8 is the best and *d* = 5 is better than *d* = 10 in POS

tag 3-grams. In POS tag 4-grams and POS tag 5-grams, the plagdet performance of *d* = 5 produced the best, *d* = 8 is the second and *d* = 10 is the third.

As mentioned in Section 4.2.3, if the difference between two consecutive source and suspicious sentences in the candidate list is less than a predefined threshold, these source and suspicious sentences may be considered as an indicator of plagiarism. Table 6.4 shows an example candidate list of $\mathrm{D}_{\mathrm{susp}}$. As can be seen in Table 6.4, the section between the 49th and 87th suspicious sentences is plagiarized from the section between the 392th and 434th source sentences. According to the experiments, the proposed method can detect the entire section performed by the parameter c = 1. If the value d = 3, then the section consisting of between the rows 1-7 in Table 6.4 is identified as plagiarism according to the $\mathrm{POSNG}_{\mathrm{PD}}$ algorithm. The rest of the candidates are investigated by $\mathrm{POSNG}_{\mathrm{PD+SSBS}}$ method whether they are plagiarized.

Table 6.4. The Representation of a Part of the Candidates List

#	D_{src}	Source sentence	Suspicious sentence	Candidate rank (*c*)	Difference |*d*|
1	07440	392	49	1	-
2	07440	394	51	1	2 – 2
3	07440	395	52	1	1 – 1
4	07440	396	53	1	1 – 1
5	07440	399	56	1	3 – 3
6	07440	400	57	1	1 – 1
7	07440	402	59	1	2 – 2
8	07440	406	61	1	4 – 2
9	07440	412	67	1	6 – 6
10	07440	417	72	1	5 – 5
11	07440	426	80	1	9 – 8
12	07440	434	87	1	8 – 7

On the other hand, if the value *d* = 10, the entire section in Table 6.4 is identified as plagiarism. This explains why the plagdet performance of *d* = 10 is the best in the types of low, high and simulated plagiarism cases. The illustration of the text sections consisting of the source and suspicious sentences in Table 6.4 is given in Figure 6.4. Suppose Figure 6.4 (a) shows similar sections between two documents. As can be seen from Figure 6.4 (a), these sections represent a long

passage consisting of consecutive sentences. If the value of d, which is the difference between the sentence numbers of two consecutive candidates, is taken as 10, it will be possible to fully identify these similar sections according to the data in Table 6.4. On the other hand, if the d value is selected as numbers less than 10, the area that is the result of the d value can be determined instead of the whole section, as seen in Figure 6.4 (b).

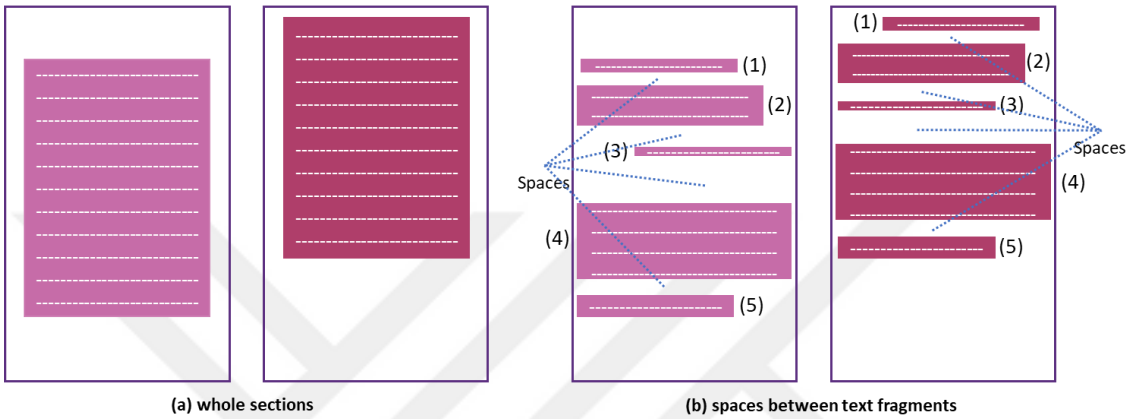

Figure 6.4. The Illustration of the Effect of Parameter d

Hence, as the value of *d* increases, the actual plagiarism cases detected by the proposed method increases. In this case, precision decreases while the performance of the recall increases. On the contrary, when the value of *d* decreases, the actual plagiarism cases detected by the proposed system decreases. In that situation, precision increases while the performance of the recall decreases.

6.3.5. Comparison of POS Tagging and Semantic Similarity

The performance of $POSNG_{PD}$ (mentioned in Section 4.2.4.1) and $POSNG_{PD+SSBS}$ (mentioned in Section 4.2.4.2) methods are also compared in the experiments.

Figure 6.5 shows the plagdet performances of the $POSNG_{PD}$ and $POSNG_{PD+SSBS}$ methods in four types of plagiarism cases. As can be seen in Figure 6.5 (a), (b), (c) and (d), the plagdet performance of $POSNG_{PD+SSBS}$ method is better than $POSNG_{PD}$ in all types of plagiarism cases. In the set of values of similarity

threshold = $\{0.6, 0.7, 0.8, 0.9\}$, the $POSNG_{PD+SSBS}$ performed better performance than $POSNG_{PD}$ method in the experiments carried out with 3, 4 and 5-grams.

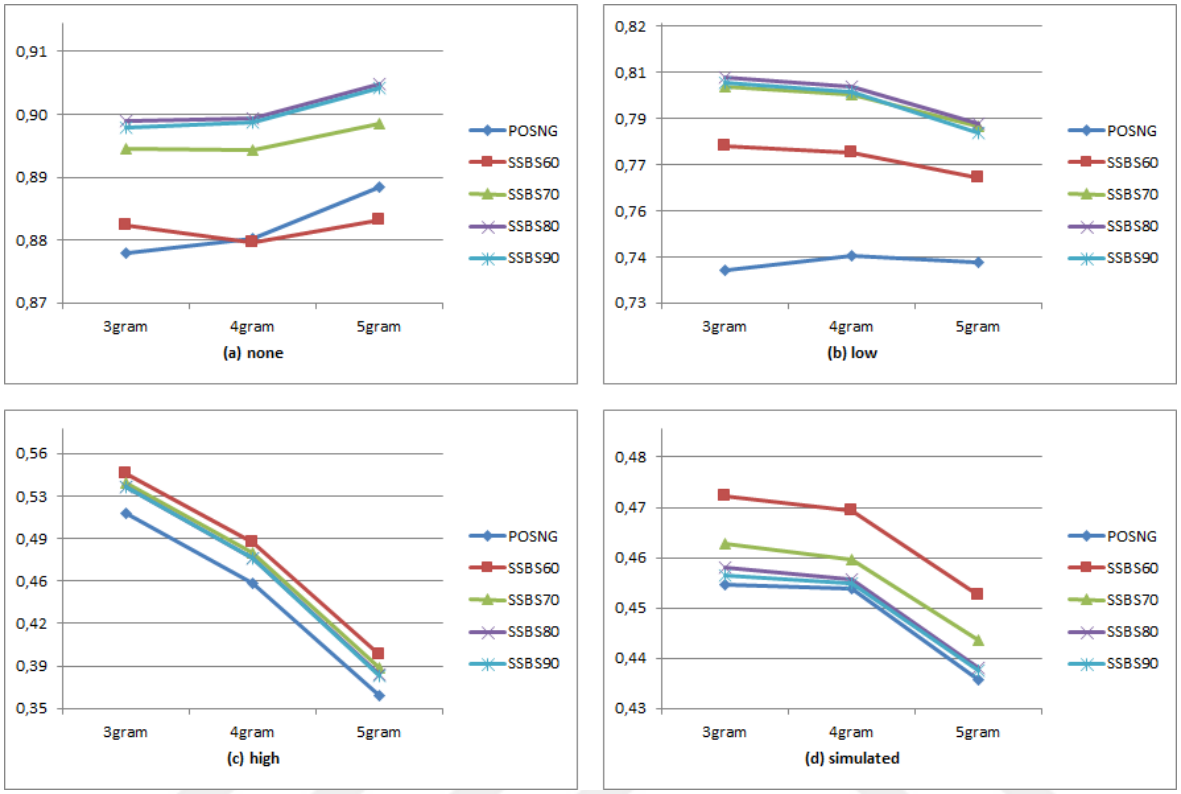

Figure 6.5. Comparison of the plagdet of $POSNG_{PD}$ and $POSNG_{PD+SSBS}$

As mentioned in Section 4.2.4.2, $POSNG_{PD+SSBS}$ method contains the $POSNG_{PD}$ and provides a sentence-to-sentence comparison between the source and suspicious documents, which are not identified as plagiarized in $POSNG_{PD}$ method. As given in Table 6.2, 35% of plagiarism cases have less than 150 words in the PAN-PC-11 corpus. A plagiarism case can be a passage consisting of more than one sentence or only one sentence. Since the $POSNG_{PD}$ method detects plagiarism according to two consecutive sentences pairs, the plagiarism case consisting of one sentence cannot be identified. As clarified in Section 6.3.4, if the difference between two consecutive source and suspicious sentences in the candidate list is greater than a predefined threshold, then the passage consisting of these source and suspicious sentences is not identified as plagiarism. Therefore, semantic similarity comparison is performed between the pair of S_{src} and S_{susp}.

The POSNG$_{PD+SSBS}$ method is able to eliminate the semantic similarity gap of POSNG$_{PD}$. Since the POSNG$_{PD+SSBS}$ performs a semantic similarity comparison between S_{susp} and its candidate S_{src}, S_{susp} that is plagiarized from S_{src} can also be detected in this method. For these reasons, the plagdet performance of POSNG$_{PD+SSBS}$ method is better than POSNG$_{PD}$.

6.3.6. Comparison of Similarity Thresholds

The performance of the POSNG$_{PD+SSBS}$ method was compared according to the values of similarity threshold that is used in computing LCS of the sentences S_{src} and S_{susp}. As mentioned in Section 4.2.4.2, if the *LCS* (S_{src}, S_{susp}) is greater than the similarity threshold, denoted as *sim*, S_{susp} is identified as plagiarized from S_{src}.

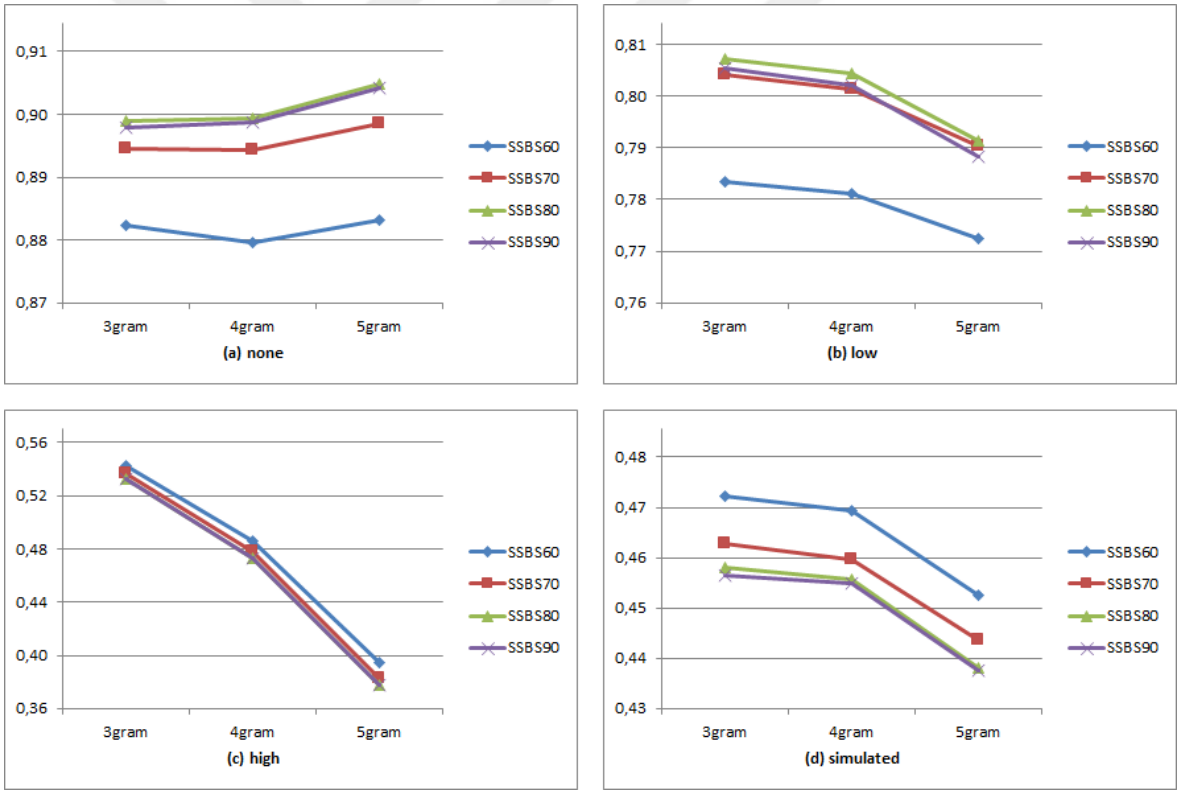

Figure 6.6. Comparison of the plagdet in Various Similarity Threshold

Figure 6.6 presents the plagdet performances in four types of plagiarism cases. As can be seen in Figure 6.6 (a) and (b), in the types of none and low, the plagdet performance of the *sim* is as follows: $0.8 > 0.9 > 0.7 > 0.6$. The best plagdet

performance is obtained when the value of *sim* = 0.8. It is clear from the Figure 6.6 (a) and (b) that when the value of similarity threshold decreases, the performance of plagdet decreases too. The reasons behind the result in Figure 6.6 (b) can be considered as follows: 1) when the *sim* is selected with lower values such as 0.6 and 0.7, the pair of S_{src} and S_{susp} that is not a true case of plagiarism can be identified as plagiarized due to the low value of *sim*; 2) when the *sim* ≥ 0.8, the pair of S_{src} and S_{susp} that is actually plagiarized, cannot be identified as plagiarism due to the high value of *sim*.

On the other hand, as can be seen in Figure 6.6 (c) and (d), the plagdet performance of *sim* in high and simulated types is as follows: $0.6 > 0.7 > 0.8 > 0.9$. The performance of plagdet improved when the value of similarity threshold decreases. This means that when *sim* = 0.9, the pair of S_{src} and S_{susp}, which is actually plagiarized, cannot be identified as plagiarism due to the high value of *sim*. The results have demonstrated an inverse correlation between none & low and high & simulated types of plagiarism.

6.4. Analysis of Execution Times

In this section, the execution times of the operations carried out regarding the plagiarism detection process in this thesis are analyzed. Experiments were performed on two desktop computers that have 6 GB RAM and 3.00 GHz processor and 8 GB RAM and 3.40 processor capacities mentioned in Section 6.1. The operations within the scope of the developed algorithm have been implemented as a single thread approach. According to the experiment results, the execution times of the operations performed in the proposed method are given in Table 6.6.

As can be seen from Table 6.6, the operations were carried out using 3, 4 and 5-grams. The operations performed are heavily based on reading existing text files and writing them to new text files. Therefore, the I/O (input and output) library is extensively used in the proposed method. First, the operations start with the arrangement of the source and suspicious documents. Accordingly, 10,420 source and 11,093 suspicious documents in the corpus are made available for use in the preprocessing step. The size of these source documents is 2.20 GB

and the size of the suspicious documents is 1.57 GB. Thus, 21,513 documents with a total of 3.77 GB are read one by one, break lines are removed and encoding problems are fixed. Since these arrangement and sentence generation operations are performed only once, the same execution times are obtained for each n-gram set. As a result, this operation was completed in approximately 11.25 minutes. The new edited texts produced as a result of this process are used to generate sentences in the next step.

Sentence segmentation task is also performed for all source and suspicious documents in the corpus. In the sentence segmentation step, each document is read sequentially and split into its sentences using a regular expression pattern. Since short sentences are not ignored in the proposed method, a combination algorithm is implemented in the generation of sentences. Accordingly, a value is set in the beginning of the process, which determines how many words a sentence will consist of at most. If a sentence contains fewer words than the value of specified number of words, then it is combined with the next sentence and the total word count of them is checked again. This process continues until the combined sentences meet the total word count criterion. Finally, all generated sentences of a document are stored with their starting points and lengths.

As can be seen from the Table 6.6, approximately 22 million sentences (12,348,670 source and 9,626,969 suspicious) were created after 65.35 minutes as a result of sentence segmentation task. Since 22 million sentences are stored with their starting point and length in their own document, the total file size including the sentences increased by 5.6% compared to the original texts and became 3.98 GB. The first two steps, which are removing break lines, fixing encoding problems and sentences segmentation are common to all n-grams and are performed once. Therefore, operations starting after this stage differ according to n-grams.

In the next steps, POS tag n-grams are produced by using source and suspicious sentences. For this, first the tokens in the sentences are determined. Then, the POS tags of these tokens are produced. At this point, in order to reduce the size of POS tags, they are replaced with symbols of one-character length and finally, POS tag n-grams are created. As can be seen from Table 6.6, the total file size

increases as n-grams increase. The execution times for the generation of POS tag n-grams were completed in approximately the same time for all three grams.

On the other hand, different results are obtained between the file size of the total POS tag n-grams and the total sentence size compared to the n-grams. The total file size of POS tag 3-grams and 4-grams decreased by 19.6% and 6.53% respectively, compared to the total sentence size, while the total file size of POS tag 5-grams increased by 8.29%. This means that the total number of characters of any sentence converted to the POS tag 5-gram format is greater than the size of the sentence itself. As can be seen from Table 6.5, the size of the sentence [159] was smaller than the size of the POS tag 5-gram derived from it. This situation caused the POS tag 5-grams to be higher than the sentence size in the entire corpus by about 8%.

Table 6.5. POS Tag 5-gram Format of a Sentence

	Sentence	**POS tag 5-grams**
	The rest of our time was spent in final training, mainly carried out at Gosfield Park and Abbot's Hall, and in preparations for going out, in which the inspection and completion of equipment of all kinds played a prominent part.	dniPn niPnb iPnbV PnbVi nbVij bVijn Vijn1 ijn1r jn1rb n1rbq 1rbqi rbqig bqigg qiggc iggcg ggcgs gcgsg cgsg1 gsg1c sg1ci g1cia 1ciai ciaie iaieq aieq1 ieq1i eq1iw q1iwd 1iwdn iwdnc wdncn dncni ncnin cnini ninid inida nidab idabd dabdj abdjn bdjn0
Size	228 bytes	245 bytes

In the next step, the most used POS tag n-grams in the whole corpus are replaced with symbols of smaller size in order to minimize the size of the index, reduce the query time and decrease the total execution time. Accordingly, all POS tag n-grams in the corpus are counted and sorted from most used to least used. For example, the first 3-gram “idn” occurs more than 17 million times in the entire corpus. This n-gram is replaced by the “.” character, thus the file size reduces drastically.

Table 6.6. Execution Times of the Operations

Operations	Data	3-gram		4-gram		5-gram	
		Output Size	Execution Time	Output Size	Execution Time	Output Size	Execution Time
Removing break lines Fix encoding issues	Source and suspicious documents	3.77 GB	11.25 min.	3.77 GB	11.25 min.	3.77 GB	11.25 min.
Generating sentences	Source and suspicious documents	3.98 GB	65.35 min.	3.98 GB	65.35 min.	3.98 GB	65.35 min.
Tokenization Determining POS tags Generating POS tag n-grams	Source and suspicious documents	3.20 GB	460.88 min.	3.72 GB	464.52 min.	4.31 GB	459.13 min.
Identifying unique n-grams Generating character space Sorting operations Replacing POS tags	Source and suspicious documents	2.23 GB	21.25 min.	2.46 GB	23.65 min.	2.71 GB	29 min.
Indexing	Source documents	2.59 GB	12.23 min.	2.92 GB	12.47 min.	3.30 GB	14.38 min.
Candidate sentences retrieval	Source index and suspicious documents	3.54 GB	~20 days	3.97 GB	~20 days	4.46 GB	~20 days
Detailed analysis Semantic similarity operations	Candidate source sentences and some suspicious sentences	Changeable	~180 min.	Changeable	~180 min.	Changeable	~180 min.

As can be seen from the Table 6.6, replacing the POS tag n-grams with characters that are smaller than their own length resulted a significant reduction in the file size for all three grams used in the experiments. Accordingly, the size of POS tag 3-grams decreased by 30%, the size of POS tag 4-grams decreased by 34% and the size of POS tag 5-grams decreased by 37%. Thus, an average of 33.6% reduction in file size was achieved in the entire corpus. On the other hand, it is seen from the Table 6.6 that the execution times of replacement of POS tag n-grams increase as n-grams increase. This is due to the fact that the POS tag n-gram file size increases in directly proportional to the n-gram number.

Then, source sentences converted to POS tag n-grams format are indexed. Approximately 22 million sentences are indexed in an average of 13 minutes. As the n-grams increase, both the index size and the execution time of the POS tag n-grams increase. However, the execution times of each n-gram are very close to each other. The increase in file sizes is seen as 12.74% from 3-gram to 4-gram and 13.01% from 4-gram to 5-gram.

The next phase, candidate sentence retrieval, is the longest running task of the proposed method. This process is to search the POS tag n-grams of approximately 9.7 million suspicious sentences within the POS tag n-grams of approximately 12.3 million source sentences. This process, which runs simultaneously on two computers, the features of which are given in Section 6.1, is completed in approximately 20 days for each n-gram. Comparison of millions of strings belonging to approximately 9.7 million suspicious sentences with millions of strings in approximately 12.3 million source sentences is the reason why the total runtime is at this duration. In this process, every source sentence closest to the suspicious sentence is scored. As a result, the top 10 source sentences with the highest scores are returned.

This total runtime has been a challenging and time-consuming factor for frequent test activities. Therefore, experiments were mostly performed on specific subsets of the corpus. However, a subset was created for each obfuscation type and the experiments were carried out in this way. According to the success of the results obtained from these subsets, the proposed method was tested on the entire corpus. On the other hand, as can be seen from Table 6.6, the index size and the

total file size used for candidate sentence retrieval increase as the number of POS tag n-grams increases.

In the last phase of the proposed method, detailed analysis is performed. In this phase, each suspicious sentence and candidate sentences are compared semantically and it is decided whether the suspicious sentence is plagiarism or not. For these operations, some suspicious sentences and their candidate source sentences are used as data. Although the total suspicious documents size is certain for each n-gram, the total source sentences size varies as each suspicious sentence has different candidate sentences. Therefore, the size of both candidate source sentences and suspicious sentences used at this phase is changeable. However, as the number of n-grams increases, so does the index size, so the total data size used at this phase will increase in directly proportional to the number of n-grams. Processes to decide whether a suspicious sentence has been plagiarized are completed in milliseconds. However, since detailed analysis was performed for all suspicious documents, it was observed that the entire process was completed in approximately 180 minutes for each n-gram. At this phase, the features of the sections in which plagiarism is detected are written to XML files intensively.

The executions times shown in Table 6.6 may deviate by about 5% to 10% due to various reasons such as the tasks on the computer running at that time or other tasks being used in the background. If computers with more powerful configurations or server platforms are used for the future work, it is foreseen that the times shown in Table 6.6 can be significantly reduced. In addition, it is considered that these times can be reduced in the same way if the multithreaded approach is applied in the implementation of the proposed method. For the future studies, it is also aimed to make remedial changes in the code structure in order to complete the transactions faster.

7. CONCLUSION

7.1. Summary

As explained in Chapter 3, a lot of research has been done on plagiarism detection. Different methods and techniques were used in these studies. As mentioned in Section 3.1, while the first studies focused on the string and character-based matching method, the usage of machine learning and fuzzy-based techniques has been widely seen in recent years to detect translation plagiarism and complicated plagiarism cases. In this thesis study, an external plagiarism detection system based on part-of-speech tag n-grams (POSNG) is introduced. The proposed method contains a set of syntactic and semantic features in the process of detecting plagiarism. It is able to detect both verbatim plagiarism and other changes that are created manually or automatically on documents with different obfuscated strategies.

The proposed method consists of four main steps: in first, also called text preprocessing, basic NLP techniques consisting of sentence segmentation, tokenization and POS tagging are applied to both source and suspicious documents respectively. In the second step, n-grams of POS tags of the sentences are created and then source sentences are indexed. In the third step, each suspicious sentence is searched in the index and candidate sentences found for that suspicious sentence are retrieved by a search engine. Finally, in the decision making step, it is investigated whether a suspicious sentence is plagiarized from its candidate. The plagiarism detection process was carried out by two methods: POSNG plagiarism detection ($POSNG_{PD}$) and $POSNG_{PD}$ with semantic similarity ($POSNG_{PD+SSBS}$). In order to detect semantic similarity, a word embedding learning technique called Word2Vec was implemented, which makes use of the semantic relatedness between words.

The experiments are conducted with PAN-PC-11 corpus, which is created to evaluate of automatic plagiarism detection algorithms. The experiments are carried out with four types of paraphrasing levels addressed in PAN-PC-11: none or very few, low, high and simulated obfuscation. Various thresholds and parameter values are used in the experiments to evaluate the diversity of the

results. The performance of the proposed method is compared with the detectors in the 3rd International Plagiarism Detection Competition. Based on the experimental results shown in Table 6.2, the proposed method appears to produce very competitive results. $POSNG_{PD+SSBS}$ method achieved the best results in the low and high obfuscated paraphrasing types relative to the overall performance metric *plagdet*. The results also demonstrate that the *recall* performance of $POSNG_{PD+SSBS}$ method is the best in the low and high obfuscated paraphrasing types. In addition, the running times of the experiments carried out with this thesis were analyzed. When the results obtained are examined, it is evaluated that improvements should be made to reduce the time of the search processes that provide candidate sentences retrieval in the entire corpus.

7.2. Future Work

For future improvements, it is intended to add the cross-lingual plagiarism detection feature to the proposed method and extend it to test the cases of translation plagiarism in the PAN-PC-11 dataset. Thus, it will be possible to obtain an overall result in the entire corpus. Besides, new levels of plagiarism such as random and summary obfuscation, have been added to the PAN12-14 datasets. In this direction, it is aimed to evaluate the proposed method with different obfuscated strategies by using these datasets.

According to the experimental results, although the best granularity performance was obtained in none obfuscation type, it was seen that the granularity performance of low and high obfuscation types showed unstable and average results. Therefore, it is aimed to improve the performance of the proposed method in terms of granularity metric, especially in low and high obfuscation types. In addition, it is aimed to implement other word embedding methods such as Doc2Vec, ELMo, BERT and fastText to evaluate the performance of the proposed method and compare the semantic similarity results with Word2Vec in future studies.

In the proposed method, the words were used as they were in the corpus in their original form and lemmatization or stemming operations were not performed. For this reason, it is aimed to apply lemmatization and stemming techniques to the

preprocessing step in order to investigate the effect on the semantic similarity and whether the POS tag structure of the words has changed. However, it is planned to improve the sentence segmentation algorithm and to analyze the usage of different numbers of stop words on the performance.

7.2.1. Improving Paraphrases

The system proposed within the scope of this thesis has achieved the most successful results compared to PAN11 detectors in detecting high obfuscation plagiarism cases. Even as can be seen from the results in Table 6.2, the proposed method has achieved better results than other systems by far in overall plagdet and recall values. However, the plagdet value of artificial high cases was 0.54 and the recall value was 0.43. This situation shows that it is very hard to identify high obfuscation cases in the corpus. When some artificial high cases are examined one by one, it is seen that there are complex obfuscated strategies and changes that even a person may have difficulty in understanding and detecting.

For the future studies, it is aimed to improve the detection performance of these high obfuscated paraphrasing cases. In addition, as mentioned in Section 4.2.4.3, when there is active-passive voice conversion, the performance of the proposed method decreases in some cases because the syntactic structure of the sentences change. For the future improvements, it is aimed to deal with the situation related to this error analysis in detail and to improve the detection performance of the sentences with active-passive conversion.

7.2.2. Plagiarism in Machine Translation

The concept of machine translation is one of the developing fields in the recent years. Especially online translation tools are frequently used today. Due to the fact that the act of plagiarism is done in research works and articles written in different languages, studies on translation have started to be carried out. The number of studies on cross-lingual plagiarism detection is increasing day by day. Similar approaches are used with the mono-lingual plagiarism detection process in the detection of cross-lingual plagiarism. Documents written in a different language are translated into English or vice versa through various translation

libraries or applications. All subsequent operations are performed with algorithms in the mono-lingual plagiarism detection process.

Unlike this approach, some word embedding technologies such as Word2Vec or Doc2Vec and vector-based methods are used without translation between languages. In this way, it is ensured that the words in the document are associated or replaced with the word in another language with the closest meaning to them.

In the corpus used in this thesis, there are 471 German and 202 Spanish texts along with English texts. Since this thesis focuses on a mono-lingual plagiarism detection approach, documents related to translation plagiarism were excluded from the experiments. It is planned to add algorithms that will detect cross-lingual plagiarism to the proposed method for future studies. Thus, cross-lingual results and overall performance of the proposed method for the entire corpus can be obtained.

7.3. Final Conclusion and Comments

To conclude, plagiarism is a serious issue and all students and academic staff should be aware of plagiarism. Although automated plagiarism detection systems are used to identify plagiarism cases, these tools report the comparison results. Hence, definitive human judgment is still needed to decide whether it is truly plagiarism.

No matter how advanced plagiarism detection programs are, it is considered that they will fail to detect some of the similarities on modified texts that are created manually or automatically by a computer program/artificial intelligence technology. In other words, it is thought that plagiarists will definitely find a way to circumvent automatic tools.

On the other hand, it is considered that a detailed human examination can detect these similarities even in the most complicated cases. The importance of automatic plagiarism detection tools here is to scan very large data that a person will never have time to spare and present possible plagiarism candidates for human evaluation to make an absolute decision. Whatever plagiarists do,

automatic detection programs will try to catch them, and plagiarists will try to find new ways to evade them.

Another issue of discussion is the necessity of explaining the working principles of plagiarism detection programs and the basic methods on which they are based. This issue can be evaluated in two ways for academic researches and commercial applications. In academic studies, it can be considered as a necessity to introduce the proposed method in detail in order to verify the reliability of the results obtained. Even the dataset and source codes may be required to fully verify the produced results. Otherwise, doubts about the reliability of the results cannot be removed.

On the other hand, open source applications also provide public information about the developed algorithm, since the source codes can be accessed. However, this is not possible in commercial applications. Even if various key functions related to the features of these applications are described, their plagiarism detection algorithm cannot be completely known. The fact that the algorithm of commercial plagiarism detection programs is not fully known can be considered as a situation that should already happen. Otherwise, anyone who knows how this detection algorithm works would have easily made the changes they needed to fool the plagiarism tool. For this reason, while the introduction of a plagiarism detection system developed within the scope of a scientific study is seen as a necessity, on the contrary, it is considered a necessity to keep the working principle of a commercial program.

However, with the development of machine learning technology, it is considered that the algorithms of plagiarism detection tools and basic software engineering mechanisms will be in the background. It is foreseen that the steps in the plagiarism detection process, such as intensive text processing, retrieving candidates or detailed analysis, will be carried out by machine learning. Hence, the detection algorithm will be a small part of the whole detection process in the form of a black box.

REFERENCES

[1] F. Sánchez-Vega, E. Villatoro-Tello, M. Montes-y-Gómez, L. Villaseñor-Pineda and P. Rosso, “Determining and characterizing the reused text for plagiarism detection,” in *Expert Systems with Applications*, vol. 40, pp. 1804-1813, https://doi.org/10.1016/j.eswa.2012.09.021, **2013.**

[2] C. Varol and S. Hari, “Detecting near-duplicate text documents with a hybrid approach,” in *Journal of Information Science*, vol. 41(4), pp. 405-414, https://doi.org/10.1177/0165551515577912, **2015.**

[3] A. Barrón-Cedeño, M. Vila, M. A. Marti and P. Rosso, “Plagiarism Meets Paraphrasing: Insights for the Next Generation in Automatic Plagiarism Detection,” in *Computational Linguistics*, vol. 39, no. 4, pp. 917-947, https://doi.org/10.1162/COLI_a_00153, **2013.**

[4] A. Barrón-Cedeño, P. Gupta and P. Rosso, “Methods for cross-language plagiarism detection,” in *Knowledge-Based Systems*, vol. 50, pp. 211-217, https://doi.org/10.1016/j.knosys.2013.06.018, **2013.**

[5] B. Stein, M. Koppel and E. Stamatatos, “Plagiarism analysis, authorship identification, and near-duplicate detection,” in *Proceedings of the ACM SIGIR International Workshop*, vol. 41, issue 2, pp. 68-71, https://doi.org/10.1145/1328964.1328976, **2007.**

[6] Webis Data PAN-PC-11, [Online]. Available: https://webis.de/data/pan-pc-11.html. [Accessed 11 April 2022].

[7] Plagiarism Definition, [Online]. Available: https://www.dictionary.com/browse/plagiarism. [Accessed 01 September 2022].

[8] A. Dhir, G. Arora and A. Arora, “Architectural Designing and Analysis of Natural Language Plagiarism Detection Mechanism,” in *Journal of Theoretical and Applied Information Technology*, vol. 4, issue 12, pp. 1150-1170, **2008.**

[9] B. Martin, “Plagiarism: a misplaced emphasis,” in *Journal of Information Ethics*, vol. 3, no. 2, pp. 36-47, **1994.**

[10] T. Gupta and L. Banda, “A hybrid model for detection and elimination of near duplicates based on Web provenance for effective Web search,” in *International Journal of Advances in Engineering & Technology*, vol. 4, pp. 192-205, **2012.**

[11] A. S. Bin-Habtoor and M. A. Zaher, “A Survey on Plagiarism Detection Systems,” in *International Journal of Computer Theory and Engineering*, vol. 4, no. 2, pp. 185-187, https://doi.org/10.7763/IJCTE.2012.V4.447, **2012.**

[12] B. Gipp and N. Meuschke, “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*, pp. 249-258, https://doi.org/10.1145/2034691.2034741, **2011.**

[13] M. Potthast, A. Barrón-Cedeño, B. Stein and P. Rosso, “Cross-Language plagiarism detection,” in *Knowledge-Based Systems*, vol. 45, pp. 45-62, https://doi.org/10.1007/s10579-009-9114-z, **2011.**

[14] S. P. Green, “Plagiarism, Norms, and the Limits of Theft Law: Some Observations on the Use of Criminal Sanctions in Enforcing Intellectual Property Rights,” in *Hastings Law Journal*, vol. 54, http://dx.doi.org/10.2139/ssrn.315562, **2002.**

[15] T. Mikolov, K. Chen, G. Corrado and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” in *ICLR (Workshop Poster)*, https://doi.org/10.48550/arXiv.1301.3781, **2013.**

[16] D. Suleiman, A. Awajan and N. Al-Madi, “Deep Learning Based Technique for Plagiarism Detection in Arabic Texts,” in *International Conference on New Trends in Computing Sciences (ICTCS)*, Amman, Jordan, https://doi.org/10.1109/ICTCS.2017.42, **2017.**

[17] J. Barr and L. F. Cabrera, “AI gets a brain: New technology allows software to tap real human intelligence,” in *ACM Queue*, vol. 4(4), pp. 24-29, https://doi.org/10.1145/1142055.1142067, **2006.**

[18] S. Finlay, “CopyCatch. Masters Dissertation,” University of Birmingham, **1999.**

[19] P. Clough, “Plagiarism in natural and programming languages: An overview of current tools and technologies,” in Department of Computer Science, University of Sheffield, **2000.**

[20] R. B. Kalleberg, “Towards Detecting Textual Plagiarism Using Machine Learning Methods. Masters Dissertation,” University of Agder, Grimstad, **2015.**

[21] Ö. Uzuner, B. Katz and T. Nahnsen, “Using syntactic information to identify plagiarism,” in *EdAppsNLP 05: Proceedings of the second workshop on Building Educational Applications Using NLP*, pp. 37-44, http://dx.doi.org/10.3115/1609829.1609836, **2005.**

[22] H. A. Chowdhury and D. K. Bhattacharyya, “Plagiarism: Taxonomy, Tools and Detection Techniques,” in *Knowledge, Library and Information Networking*, https://doi.org/10.48550/arXiv.1801.06323, **2018.**

[23] H. Maurer, F. Kappe and B. Zaka, “Plagiarism - A survey,” in *Journal of Universal Computer Science*, vol. 12, no. 8, pp. 1050-1084, **2006.**

[24] N. Khan, C. Agrawal, T. N. Ansari, “A Review on Various Plagiarism Detection Systems Based on Exterior and Interior Method,” in *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 7, issue 9, pp. 6-12, **2018.**

[25] M. S. Anderson and N. H. Steneck, “The problem of plagiarism,” in *Urologic Oncology: Seminars and Original Investigations*, vol. 29, pp. 90-94, **2011.**

[26] C. K. Kent and N. Salim, “Web Based Cross Language Plagiarism Detection,” in *Second International Conference on Computational Intelligence, Modelling and Simulation*, pp. 199-204, https://doi.org/10.1109/CIMSiM.2010.10, **2010.**

[27] K. Leilei, Q. Haoliang, W. Shuai, D. Cuixia, W. Suhong and H. Yong, “Approaches for Candidate Document Retrieval and Detailed Comparison of Plagiarism Detection,” in *CLEF 2012 Conference*, **2012.**

[28] A. G. Liaqat and A. Ahmad, “Plagiarism Detection in Java Code. Degree Project” Linnaeus University, Kalmar, **2011.**

[29] C. Kustanto and I. Liem, “Automatic Source Code Plagiarism Detection,” in *10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing*, pp. 481-482, **2009.**

[30] Y. Lee, J. Lim, J. Ji, H. Cho and G. Woo, “Plagiarism Detection among Source Codes using Adaptive Methods,” in *KSII Transactions on Internet and Information Systems*, vol. 6, no. 6, pp. 1627-1648, http://dx.doi.org/10.3837/tiis.2012.06.008, **2012.**

[31] J. Lim, J. Ji, H. Cho and G. Woo, “Plagiarism detection among source codes using adaptive local alignment of keywords,” in *ICUIMC ‘11: Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication*, pp. 24-33, **2011.**

[32] M. Joy and M. Luck, “Plagiarism in programming assignments,” in *IEEE Transactions on Education*, vol. 42, no. 2, pp. 129-133, doi: 10.1109/13.762946, **1999.**

[33] A. Knight and K. Almeroth, “Design, Implementation and Deployment of PAIRwise,” in *Journal of Interactive Learning Research*, vol. 19, no. 3, pp. 489-508, **2008.**

[34] S. M. Alzahrani, N. Salim and V. Palade, “Uncovering highly obfuscated plagiarism cases using fuzzy semantic-based similarity model,” in *Journal of King Saud University-Computer Information Sciences*, vol. 27, no. 3, pp. 248-268, **2015.**

[35] S. M. Alzahrani, N. Salim and A. Abraham, “Understanding plagiarism linguistic patterns, textual features, and detection methods,” in IEEE Transactions on Systems, Man, and Cybernetics, vol. 42, no. 2, pp. 133-149, https://doi.org/10.1109/TSMCC.2011.2134847, **2012.**

[36] E. Stamatatos, “Intrinsic Plagiarism Detection Using Character n-gram Profiles,” in *3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse*, pp. 38-46, **2009.**

[37] S.M. Eissen and B. Stein, “Intrinsic Plagiarism Detection,” in *Proceedings of the 28th European Conference on IR Research*, pp. 565-569, **2006.**

[38] E. Stamatatos, “A Survey of Modern Authorship Attribution Methods,” in *Journal of the American Society for information Science and Technology*, vol. 60, issue. 3, pp. 538-556, https://doi.org/10.1002/asi.21001, **2009.**

[39] I. Bensalem, P. Rosso and S. Chikhi, “On the use of character n-grams as the only intrinsic evidence of plagiarism,” in *Language Resources and Evaluation*, vol. 53, pp. 363-396, https://doi.org/10.1007/s10579-019-09444-w, **2019.**

[40] K. Vani and D. Gupta, “Study on extrinsic text plagiarism detection techniques and tools,” in *Journal of Engineering Science and Technology Review*, vol. 9, no.5, pp. 9-23, **2016.**

[41] B. Stein, S. Meyer, M. Potthast, “Strategies for Retrieving Plagiarized Documents,” in *30th International ACM Conference on Research and Development in Information Retrieval (SIGIR 07)*, pp. 825-826, **2007.**

[42] C. E. Karbeyaz, “A Cluster-Based External Plagiarism and Parallel Corpora Detection Method. Masters Dissertation,” Bilkent University, Ankara, **2011.**

[43] G. Oberreuter, G. L’Huillier, S. A. Ríos, J. D. Velásquez, “Fastdocode: Finding approximated segments of n-grams for document copy detection. Lab report for PAN at CLEF”, **2010.**

[44] J. Kasprzak, M. Brandejs, “Improving the reliability of the plagiarism detection system. Lab report for PAN at CLEF”, **2010.**

[45] P. Gupta, A. Barrón-Cedeño and P. Rosso, “Cross-Language High Similarity Search Using a Conceptual Thesaurus,” in *Information Access Evaluation. Multilinguality, Multimodality, and Visual Analytics*, pp. 67-75. https://doi.org/10.1007/978-3-642-33247-0_8, **2012.**

[46] N. Ehsan and A. Shakery, “Candidate document retrieval for cross-lingual plagiarism detection using two-level proximity information,” in *Information Processing and Management*, vol. 52, issue. 6, pp. 1004-1017, https://doi.org/10.1016/j.ipm.2016.04.006, **2016.**

[47] P. Clough and M. Stevenson, “Developing a corpus of plagiarised short answers,” in *Language Resources and Evaluation*, vol. 45, no.1, pp. 5-24, **2011.**

[48] M. Franco-Salvador, P. Gupta and P. Rosso, “Cross-language plagiarism detection using a multilingual semantic network,” in *Advances in information retrieval, proceedings of the 35th European conference on information retrieval (ECIR13)*, pp. 710-713, **2013.**

[49] R. K. Ahmed, “Overview of Different Plagiarism Detection Tools,” in *International Journal of Futuristic Trends in Engineering and Technology*, vol. 2, no. 10, pp. 2-4, **2015.**

[50] A. H. Osman, N. Salim and A. Abuobieda, “Survey of Text Plagiarism Detection,” in *Computer Engineering and Applications*, vol. 1, pp. 37-45, **2012.**

[51] M. Potthast, A. Eiselt, A. Barrón-Cedeño, B. Stein and P. Rosso, “Overview of the 3rd International Competition on Plagiarism Detection,” in *Working Notes Papers of the CLEF 2011 Evaluation Labs*, **2011.**

[52] E. Stamatatos, “Plagiarism Detection Using Stopword n-grams,” in *Journal of the American Society for Information Science and Technology*, vol. 62, no. 12, pp. 2512-2527, https://doi.org/10.1002/asi.21630, **2011.**

[53] D. Namdev and J Surana, “A Survey Paper on Plagiarism Detection Techniques,” in *IJCA Proceedings on International Conference on ICT for Healthcare*, vol. 1, pp. 30-34, **2016.**

[54] M. Elhadi and A. Al-Tobi, “Use of text syntactical structures in detection of document duplicates,” in *Third International Conference on Digital Information Management*, pp. 520-525, **2018.**

[55] S. Torres and A. Gelbukh, “Comparing Similarity Measures for Original WSD Lesk Algorithm,” in *Advances in Computer Science and Applications*, vol. 43, pp. 155-166, **2009.**

[56] D. A. R. Torrejón and J. M. R. Ramos, “Text Alignment Module in CoReMo 2.1 Plagiarism Detector,” in *Proceedings of 5th International Workshop PAN-13*, **2013.**

[57] R. Kuppers and S. Conrad, “A Set-Based Approach to Plagiarism Detection,” in *Proceedings of 3rd International Workshop PAN-12*, **2012.**

[58] Y. Palkovskii and A. Belov, “Developing High-Resolution Universal Multi-Type N-Gram Plagiarism Detector,” in *Proceedings of 6th International Workshop PAN-14*, **2014.**

[59] F. Alvi, M. Stevenson and P. Clough, “Hashing and Merging Heuristics for Text Reuse Detection,” in *Proceedings of 6th International Workshop PAN-14*, **2014.**

[60] C. Grozea, C. Gehl and M. Popescu, “Encoplot: Pairwise sequence matching in linear time applied to plagiarism detection,” in *3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse*, p. 10, **2009.**

[61] C. Basile, D. Benedetto, E. Caglioti, G. Cristadoro and M. Esposti, “A plagiarism detection procedure in three steps: Selection, matches and squares,” in *Proceedings of International Conference of the Spanish Society For Natural Language Processing (SEPLN)*, pp. 19-23, **2009.**

[62] N. Meuschke and B. Gipp, “State of the Art in Detecting Academic Plagiarism,” in *International Journal for Educational Integrity*, vol. 9, no. 1, pp. 50-71, https://doi.org/10.5281/zenodo.3482941, **2013.**

[63] B. Stein and S. M. Eissen, “Fingerprint-based Similarity Search and its Applications,” Universität Weimar, **2007.**

[64] S. Schleimer, D. Wilkerson and A. Aiken, “Winnowing: Local Algorithms for Document Fingerprinting,” in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, vol. 10, https://doi.org/10.1145/872757.872770, **2003.**

[65] A. Abdi, N. Idris R. M. Alguliyev and R.M. Aliguliyev, “PDLK: Plagiarism detection using linguistic knowledge,” in *Expert Systems with Applications*, vol. 42, issue. 22, pp. 8936-8946, https://doi.org/10.1016/j.eswa.2015.07.048, **2015.**

[66] A. Abdi, S. M. Shamsuddin, N. Idris, R. M. Alguliyev and R. M. Aliguliyev, “A linguistic treatment for automatic external plagiarism detection,” in *Knowledge-Based Systems*, vol. 135, pp. 135-146, https://doi.org/10.1016/j.knosys.2017.08.008, **2017.**

[67] K. Vani and D. Gupta, “Investigating the Impact of Combined Similarity Metrics and POS tagging in Extrinsic Text Plagiarism Detection System,” in *Proceedings of International Conference on Advances in Computing, Communication and Informatics*, pp. 1578-1584, **2015.**

[68] C. Leacock, G. A. Miller and M. Chodorow, “Using Corpus Statistics and WordNet Relations For Sense Identification,” in *Computational Linguistics*, vol. 24, no. 1, pp. 146-165, **1998.**

[69] Z. Su, B. Ahn, K. Eom, M. Kang, J. Kim and M. Kim, “Plagiarism Detection Using the Levenshtein Distance and Smith-Waterman Algorithm,” in *3rd International Conference on Innovative Computing Information and Control*, pp. 569-569, https://doi.org/10.1109/ICICIC.2008.422, **2008.**

[70] M. Elhadi and A. Al-Tobi, “Duplicate Detection in Documents and Web Pages Using Improved Longest Common Subsequence and Documents Syntactical Structures,” in *Fourth International Conference on Computer Sciences and Convergence Information Technology*, pp. 679-684, https://doi.org/10.1109/ICCIT.2009.235, **2009.**

[71] A. H. Osman, N. Salim, M. S. Binwahlanc, R. Alteebd and A. Abuobieda, “An improved plagiarism detection scheme based on semantic role labelling,” in *Journal of Applied Software Computing*, vol. 12, pp. 1493-1502, **2012.**

[72] Z. Ceska, “Plagiarism Detection Based on Singular Value Decomposition,” in *Advances in Natural Language Processing. Lecture Notes in Computer Science*, vol. 5221, https://doi.org/10.1007/978-3-540-85287-2_11, **2008.**

[73] M. Sahu, “Plagiarism Detection Using Artificial Intelligence Technique in Multiple Files,” in *International Journal of Scientific and Technology Research*, vol. 5, pp. 111-114, **2016.**

[74] A. Daud, J. A. Khan, J. A. Nasir, R. A. Abbasi, N. R. Aljohani and J. S. Alowibdi, “Latent Dirichlet Allocation and POS Tags Based Method for External Plagiarism Detection: LDA and POS Tags Based Plagiarism Detection,” in *International Journal on Semantic Web and Information Systems*, vol. 14, pp. 53-69. https://doi.org/10.4018/IJSWIS.2018070103, **2018.**

[75] S. Torres and A. Gelbukh, “Comparing Similarity Measures for Original WSD Lesk Algorithm,” in *Advances in Computer Science and Applications*, vol. 43, pp. 155-166, **2009.**

[76] P. Resnik, “Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language,” in *Journal of Artificial Intelligence Research*, vol. 11, pp. 95-130, **1999.**

[77] C. Leacock, G. A. Miller and M. Chodorow, “Using corpus statistics and WordNet relations for sense identification,” in *Computational Linguistics*, vol. 24, no. 1, pp. 147-165, **1998.**

[78] B. Junpeng, S. Junyi, L. Xiao-dong and S. Qin-bao, “A survey on natural language text copy detection,” in *Journal of Software*, vol. 14, no. 10, pp. 1753-1760, **2003.**

[79] P. Clough, “Old and new challenges in automatic plagiarism detection,” in *Plagiarism Advisory Service*, vol. 10, Department of Information Studies, University of Sheffield, **2003.**

[80] B. Gipp, “Citation-based Plagiarism Detection - Detecting Disguised and Cross-language Plagiarism using Citation Pattern Analysis,” Springer Vieweg Research, https://doi.org/10.1007/978-3-658-06394-8, **2014.**

[81] B. Gipp and J. Beel, “Citation Based Plagiarism Detection - A New Approach to Identify Plagiarized Work Language Independently,” in

Proceedings of the 21st ACM Conference on Hypertext and Hypermedia, **2010.**

[82] N. Meuschke, B. Gipp and C. Breitinger, “CitePlag: A citation-based plagiarism detection system prototype,” in *Proceedings of the 5th International Plagiarism Conference*, **2012.**

[83] S. M. Alzahrani, V. Palade, N. Salim and A. Abraham, “Using structural Information and Citation Evidence to Detect Significant Plagiarism cases in Scientific Publications,” in *Journal of the American Society for Information Science and Technology*, vol. 63, no. 2, pp. 286-312, **2012.**

[84] J. Mariani, G. Francopoulo and P. Paroubek. “A Study of Reuse and Plagiarism in Speech and Natural Language Processing papers,” in *Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL)*, pp. 72-83, **2016.**

[85] H. Maurer, F. Kappe and B. Zaka, “Plagiarism - A Survey,” in *Journal of Universal Computer Science*, vol. 12, no. 8, **2006.**

[86] B. Honig and A. Bedi, “The Fox in the Hen House: A Critical Examination of Plagiarism Among Members of the Academy of Management,” in *Academy of Management Learning and Education*, vol. 11, no. 1, pp. 101-123, https://doi.org/10.5465/amle.2010.0084, **2012.**

[87] B. Gipp and N. Meuschke, “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*, pp. 249-258, https://doi.org/10.1145/2034691.2034741, **2011.**

[88] B. Gipp, “Citation-based plagiarism detection - idea, implementation and evalutation,” in *Bulletin of IEEE Technical Committee on Digital Libraries*, vol. 8, **2012.**

[89] B. Gipp, N. Meuschke and J. Beel, “Comparative Evaluation of Text- and Citation-based Plagiarism Detection Approaches using GuttenPlag,” in

Proceedings of 11th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'11), pp. 255-258, **2011.**

[90] A. S. Altheneyan and M. E. Menai, “Automatic plagiarism detection in obfuscated text,” in *Pattern Analysis and Applications*, vol. 23, pp. 1627-1650, https://doi.org/10.1007/s10044-020-00882-9, **2020.**

[91] E. Gharavi, H. Veisi and P. Rosso, “Scalable and language-independent embedding-based approach for plagiarism detection considering obfuscation type: no training phase,” in *Neural Computing & Applications*, vol. 32, pp. 10593-10607, https://doi.org/10.1007/s00521-019-04594-y, **2020.**

[92] V. Kanjirangat and D. Gupta, “Text Plagiarism Classification using Syntax based Linguistic Features,” in *Expert Systems with Applications*, vol. 88, pp. 448-464, https://doi.org/10.1016/j.eswa.2017.07.006, **2017.**

[93] M. Roostaee, S. Fakhrahmad and M. Sadreddini, “Cross-Language Text Alignment: A proposed two-level matching scheme for plagiarism detection,” in *Expert Systems with Applications*, vol. 160, https://doi.org/10.1016/j.eswa.2020.113718, **2020.**

[94] M. Zechner, M. Muhr, R. Kern and M. Granitzer, “External and intrinsic plagiarism detection using vector space models,” in *Proceedings Conference of The Spanish Society for Natural Language Processing (SEPLN)*, pp. 47-55, **2009.**

[95] M. A. Sanchez-Perez, G. Sidorov and A. Gelbukh, “The Winning Approach to Text Alignment for Text Reuse Detection,” in *Proceedings of 6th International Workshop PAN-14*, **2014.**

[96] S. Suchomel, J. Kasprzak and M. Brandejs, “Diverse Queries and Feature Type Selection for Plagiarism Discovery,” in *Proceedings of 5th International Workshop PAN-13*, **2013.**

[97] L. Kong, Q. Haoliang, W. Shuai, D. Cuixia, W. Suhong and H. Yong, “Source Retrieval Based on Learning to Rank and Text Alignment Based on Plagiarism Type Recognition for Plagiarism Detection,” in *Proceedings of 6th International Workshop PAN-14*, **2014.**

[98] H. Zhang and T. W. S. Chow, “A coarse-to-fine framework to efficiently thwart plagiarism,” in *Pattern Recognition*, vol. 44, no. 2, pp. 471-487, https://doi.org/10.1016/j.patcog.2010.08.023, **2011.**

[99] M. K. M. Rahman, W. P. Yang, T. W. S. Chow and S. Wu, “A flexible multi-layer self-organizing map for generic processing of tree-structured data,” in *Pattern Recognition*, vol. 40, issue 5, pp. 1406-1424, https://doi.org/10.1016/j.patcog.2006.10.010, **2007.**

[100] T. W. Chow and M. K. Rahman, “Multilayer SOM with tree-structured data for efficient document retrieval and plagiarism detection,” in *IEEE Transactions on Neural Networks*, vol. 20, no. 9, pp. 1385-1402, https://doi.org/10.1109/TNN.2009.2023394, **2009.**

[101] A. H. Osman, N. Salim and M. S. Binwahlanc, “Plagiarism Detection Using Graph-Based Representation,” in *Journal of Computing*, vol. 2, no. 4, **2010.**

[102] S. Alzahrani, N. Salim, A. Abraham and V. Palade, “iPlag: Intelligent Plagiarism Reasoner in scientific publications,” in *World Congress on Information and Communication Technologies*, pp. 166, doi: 10.1109/WICT.2011.6141191, **2011.**

[103] M. S. Binwahlan, N. Salim and L. Suanmali, “Fuzzy swarm diversity hybrid model for text summarization,” in *Information Processing & Management*, vol. 46, no. 5, pp. 571-588, 10.1016/j.ipm.2010.03.004, **2010.**

[104] V. Mitra, C. J. Wang and S. Banerjee, “Text classification: A least square support vector machine approach,” in *Applied Software Computing*, vol. 7, no. 3, pp. 908-914, **2007.**

[105] D. Zou, W. J. Long and Z. Ling, “A cluster-based plagiarism detection method,” in *Notebook Papers of CLEF 2010 Labs and Workshops*, **2010.**

[106] M. Zini, M. Fabbri, M. Moneglia and A. Panunzi, “Plagiarism detection through multilevel text comparison,” in *Second International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, pp. 181-185, **2006.**

[107] A. Si, H. V. Leong and R. W. Lau, “Check: a document plagiarism detection system,” in *Proceedings of the ACM symposium on Applied computing*, pp. 70-77, **1997.**

[108] R. Yerra and Y. K. Ng, “A sentence-based copy detection approach for web documents,” in *International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 557-570, https://doi.org/10.1007/11539506_70, **2005.**

[109] J. Koberstein and Y. K. Ng, “Using word clusters to detect similar web documents,” in *International Conference on Knowledge Science, Engineering and Management*, pp. 215-228, https://doi.org/10.1007/11811220_19, **2006.**

[110] S. M. Alzahrani and N. Salim, “Fuzzy semantic-based string similarity for extrinsic plagiarism detection,” in *Proceedings of 2nd International Workshop PAN-10*, **2010.**

[111] D. Gupta, K. Vani and C. K. Singh, “Using Natural Language Processing techniques and fuzzy-semantic similarity for automatic external plagiarism detection,” in *International Conference on Advances in Computing, Communications and Informatics*, pp. 2694-2699, https://doi.org/10.1109/ICACCI.2014.6968314, **2014.**

[112] D. R. White and M. S. Joy, “Sentence-based natural language plagiarism detection,” in *Journal on Educational Resources in Computing*, vol. 4, issue 4, https://doi.org/10.1145/1086339.1086341, **2004.**

[113] L. Moussiades and A. Vakali, “PDetect: A Clustering Approach for Detecting Plagiarism in Source Code Datasets,” in *The Computer Journal*, vol. 48, issue 6, pp. 651-661, https://doi.org/10.1093/comjnl/bxh119, **2005.**

[114] A. Barrón-Cedeño and P. Rosso, “On Automatic Plagiarism Detection Based on n-Grams Comparison,” in *Advances in Information Retrieval, Part of the Lecture Notes in Computer Science*, vol. 5478, pp. 696-700, https://doi.org/10.1007/978-3-642-00958-7_69, **2009.**

[115] T. C. Hoad and J. Zobel, “Methods for identifying versioned and plagiarized documents,” in *Journal of the American Society for Information*

Science and Technology, vol. 54, issue 3, pp. 203-215, https://doi.org/10.1002/asi.10170, **2003.**

[116] M. Chong, L. Specia and R. Mitkov, “Using Natural Language Processing for Automatic Detection of Plagiarism,” in *Proceedings of the 4th International Plagiarism Conference*, **2010.**

[117] D. B. Bisandu, R. Prasad and M. Liman, “Data clustering using efficient similarity measures,” in *Journal of Systems Science and Complexity*, vol. 22, pp. 901-922, https://doi.org/10.1080/09720510.2019.1565443, **2019.**

[118] S. Arora and I. Pandey, “Hybrid Algorithm for Approximate String Matching to be used for Information Retrieval,” in *International Journal of Scientific & Engineering Research*, vol. 9, issue 11, **2018.**

[119] I. Bensalem, “Plagiarism Detection: A focus on the Intrinsic Approach and the Evaluation in the Arabic Language. Doctoral Dissertation” Constantine 2 University, Algeria, **2020.**

[120] J. Grman and R. Ravas, “Improved Implementation for Finding Text Similarities in Large Collections of Data,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[121] C. Grozea and M. Popescu, “The Encoplot Similarity Measure for Automatic Detection of Plagiarism,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[122] G. Oberreuter, G. L’Huillier, S. A. Ríos and J. D. Velásquez, “Approaches for Intrinsic and External Plagiarism Detection,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[123] N. Cooke, L. Gillam, H. C. P. Wrobel and F. Al-Obaidli, “A High-performance Plagiarism Detection System,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[124] D. A. R. Torrejón and J. M. R. Ramos, “Crosslingual CoReMo System,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[125] S. Rao, P. Gupta, K. Singhal and P. Majumder, “External & Intrinsic Plagiarism Detection: VSM & Discourse Markers based Approach,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[126] Y. Palkovskii, A. Belov and I. Muzyka, “Using WordNet-based Semantic Similarity Measurement in External Plagiarism Detection,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[127] P. M. A. Nawab, M. Stevenson and P. Clough, “External Plagiarism Detection using Information Retrieval and Sequence Alignment,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[128] A. Ghosh, P. Bhaskar, S. Pal and S. Bandyopadhyay, “Rule Based Plagiarism Detection using Information Retrieval,” in *Notebook Papers of CLEF 2011 LABs and Workshops*, **2011.**

[129] M. Paul and S. Jamal, “An Improved SRL based Plagiarism Detection Technique using Sentence Ranking,” in *Proceedings of the International Conference on Information and Communication Technologies (ICICT 2014)*, pp. 223-230, **2014.**

[130] N. Gustafson, M. Pera and Y. Ng, “Nowhere to Hide: Finding Plagiarized Documents Based on Sentence Similarity,” in *Proceedings IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 690-696. https://doi.org/10.1109/WIIAT.2008.16, **2008.**

[131] P. LaRocque, *The Book on Writing: the Ultimate Guide to Writing Well*. Marion Street Press, **2003.**

[132] R. V. S. Ram, E. Stamatatos and S. L. Devi, “Identification of Plagiarism Using Syntactic and Semantic Filters,” in *Computational Linguistics and Intelligent Text Processing*, pp. 495-506, https://doi.org/10.1007/978-3-642-54903-8_41, **2014.**

[133] B. Roark, M. Saraclar and M. Collins, “Discriminative n-gram language modelling,” in *Computer Speech & Language*, vol. 21, issue 2, pp. 373-392, https://doi.org/10.1016/j.csl.2006.06.006, **2007.**

[134] P. Gupta, S. Rao and P. Majumder, “External Plagiarism Detection: N-Gram Approach using Named Entity Recognizer Lab Report for PAN at CLEF 2010,” in *Notebook Papers of CLEF 2010 LABs and Workshops*, **2010.**

[135] S. Wang, H. Qi, L. Kong and C. Nu, “Combination of VSM and Jaccard coefficient for external plagiarism detection,” in *International Conference on Machine Learning and Cybernetics*, pp. 1880-1885, Tianjin, China, https://doi.org/10.1109/ICMLC.2013.6890902, **2013.**

[136] M. Potthast, B. Stein, A. Barrón-Cedeño and P. Rosso, “An Evaluation Framework for Plagiarism Detection,” in *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 997-1005, **2010.**

[137] C. D. Manning, P. Raghavan and H. Schütze, *Introduction to Information Retrieval*, vol. 1, Cambridge: Cambridge University Press, **2008.**

[138] K. Abrahamson, “Generalized String Matching,” in *SIAM Journal on Computing*, vol. 16, issue. 16, pp. 1039-1051, https://doi.org/10.1137/0216067, **1987.**

[139] Searching and Indexing with Apache Lucene, [Online]. Available: https://dzone.com/articles/apache-lucene-a-high-performance-and-full-featured. [Accessed 04 September 2022].

[140] Following a simple create Index through Apache Lucene internals, [Online]. Available: https://bhardwajgaurav.wordpress.com/2014/12/06/following-a-simple-create-index-through-apache-lucene-internals/. [Accessed 13 June 2022].

[141] Analysis of Lucene – Basic Concepts, [Online]. Available: https://alibaba-cloud.medium.com/analysis-of-lucene-basic-concepts-5ff5d8b90a53. [Accessed 13 June 2022].

[142] M. Potthast, B. Stein, A. Eiselt, A. Barrón-Cedeño and P. Rosso, “Overview of the 1st International Competition on Plagiarism Detection,” in *SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 09)*, **2009.**

[143] S. Engels, V. Lakshmanan and M. Craig, “Plagiarism detection using feature-based neural networks”, in *Proceedings of the 38th SIGCSE technical symposium on Computer science education*, pp. 34-38, https://doi.org/10.1145/1227310.1227324, **2007.**

[144] V. Kanjirangat and D. Gupta, “Using K-means cluster based techniques in external plagiarism detection,” in *Proceedings of International Conference on Contemporary Computing and Informatics*, pp. 1268-1273, https://doi.org/10.1109/IC3I.2014.7019659, **2015.**

[145] N. Ravi, V. Kanjirangat and D. Gupta, “Exploration of Fuzzy C Means Clustering Algorithm in External Plagiarism Detection System,” in *Intelligent Systems Technologies and Applications. Advances in Intelligent Systems and Computing*, vol. 384, pp. 127-138, https://doi.org/10.1007/978-3-319-23036-8_11, **2016.**

[146] K. Babaa, T. Nakatohb and T. Minamic, “Plagiarism detection using document similarity based on distributed representation,” in *Procedia Computer Science*, vol. 111, pp. 382-387, https://doi.org/10.1016/j.procs.2017.06.038, **2017.**

[147] E. Gharavi, H. Veisi, K. Bijari and K. Zahirnia, “A Fast Multi-level Plagiarism Detection Method Based on Document Embedding Representation,” in *Text Processing Lecture Notes in Computer Science*, vol. 10478, https://doi.org/10.1007/978-3-319-73606-8_7, **2018.**

[148] X. Ruoyun, “An Overview of Plagiarism Recognition Techniques,” in *International Journal of Knowledge and Language Processing*, vol. 9, no. 2, pp. 1-19, **2018.**

[149] G. Cosma, “An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis. Doctoral Dissertation” University of Warwick, **2008.**

[150] The Ethics of George Eliot's Works, by John Crombie Brown, [Online]. Available: https://www.gutenberg.org/files/17172/17172-h/17172-h.htm. [Accessed 02 September 2022].

[151] Google translate, a multilingual machine translation service to translate text, [Online]. Available: https://translate.google.com/. [Accessed 02 September 2022].

[152] Project Gutenberg's Christopher Columbus, Volume 1, by Filson Young, [Online]. Available: https://www.gutenberg.org/files/4108/4108.txt. [Accessed 02 September 2022].

[153] The Project Gutenberg EBook of Pretty Michal, by Mór Jókai, [Online]. Available: https://www.gutenberg.org/files/31886/31886-h/31886-h.htm. [Accessed 02 September 2022].

[154] William Shakespeare, Attorney at Law and Solicitor in Chancery, [Online]. Available: https://www.theatlantic.com/magazine/archive/1859/07/william-shakespeare-attorney-at-law-and-solicitor-in-chancery/627556/. [Accessed 02 September 2022].

[155] Apache OpenNLP, [Online]. Available: https://opennlp.apache.org/. [Accessed 02 September 2022].

[156] Stanford Log-linear Part-Of-Speech Tagger, [Online]. Available: http://nlp.stanford.edu/software/tagger.shtml. [Accessed 02 September 2022].

[157] Apache Lucene, [Online]. Available: https://lucene.apache.org/core/. [Accessed 02 September 2022].

[158] Full text of Works. With introd., general essay and notes by Andrew Lang, [Online]. Available: https://archive.org/stream/workswithintrodg32dickuoft/workswithintrodg32dickuoft_djvu.txt. [Accessed 02 September 2022].

[159] The Sherwood Foresters in the Great War 1914 - 1919 - History of the 1/8th Battalion by W.C.C. Weetman, [Online]. Available: https://www.hotfreebooks.com/book/The-Sherwood-Foresters-in-the-Great-War-1914-1919-History-of-the-1-8th-Battalion-W-C-C-Weetman.html. [Accessed 02 September 2022].

[160] Grammar Active and Passive Voice, [Online]. Available: https://degree.vidhyadeep.org/study/comp/mat2.pdf. [Accessed 03 September 2022].

[161] WS4J Demo, Examples Sentences, [Online]. Available: https://ws4jdemo.appspot.com/. [Accessed 03 September 2022].

[162] R. Naseem and S. Kurian. “Extrinsic Plagiarism Detection in Text Combining Vector Space Model and Fuzzy Semantic Similarity Scheme,” in *International Journal of Advanced Computing, Engineering and Application (IJACEA)* vol. 2, no: 6, ISSN: 2319–281X, **2013.**

[163] E. G. Hasan, A. W. Wicaksana and S. Hansun, “The Implementation of Winnowing Algorithm for Plagiarism Detection in Moodle-based E-learning.” in *IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, pp. 321-325, **2018.**

[164] Part-of-Speech Tagging, [Online]. Available: https://slidetodoc.com/natural-language-processing-partofspeech-tagging-parts-of-speech/. [Accessed 03 September 2022].

[165] A. R. Adam and S. Suharjito, “Plagiarism detection algorithm using natural language processing based on grammar analyzing,” in *Journal of Theoretical and Applied Information Technology*, vol. 63, no:1, pp. 168-180, **2014.**

APPENDICES

Appendix A – Publications Derived from Thesis

The paper published within the scope of this research study is as follows:
K. Yalcin, I. Cicekli and G. Ercan, “An external plagiarism detection system based on part-of-speech (POS) tag n-grams and word embedding,” in *Expert Systems with Applications*, vol. 197, https://doi.org/10.1016/j.eswa.2022.116677, **2022.**