Arab American University – Jenin

Faculty of Graduate Studies

HYBRID ARABIC TEXT PLAGIARISM DETECTION SYSTEM

By

Bilal Ghanem

Supervisor

Dr. Labib Arafeh

This thesis was submitted in partial fulfillment of the requirements for the Master`s degree in

Computer science

June / 2017

Hybrid Arabic Text Plagiarism Detection System

By

Bilal Ghanem

This thesis was defended successfully on 13/6/2017 and approved by:

Committee members	Signature
1. Dr. Labib Arafeh	.....
2. Dr. Amjad Rattrout	.....
3. Dr. Radwan Tahboub	.....

Acknowledgements

Thanks to Allah for giving me the power and help to accomplish this work. I would like to thank my parents to whom this work is dedicated. I am grateful for their unlimited love, care and continues support. I also extend my thanks to my brothers and sisters.

Special thanks go to my supervisor Dr. Labib Arafeh for his guidance and support. I would like to thank my friends who supported me while working on this thesis.

Abstract

This research presents a Hybrid Arabic Plagiarism Detection System (HYPLAG). HYPLAG is an Arabic text-based plagiarism detection approach that combines corpus-based and knowledge-based methods by utilizing Arabic semantic resource. The main aim of this research is to find out the effect of the combining process on the performance of detection process on Arabic plagiarized text cases. A preliminary study on undergraduate students was conducted to understand their behaviors or patterns in plagiarism. The results of the study show that students apply different methods to plagiarized sentences, also it shows changes in sentence's components (verbs, nouns, and adjectives). Based on these results, HYPLAG was developed taking into account other patterns of plagiarism.

Given a suspicious document and a collection of documents, HYPLAG compares the input document against the document collection in an efficient manner. It utilizes the search engine structure in the retrieving method, where the most relevant source sentence is retrieved. To ensure the validity of the input text, a set of preliminary methods are applied. One of most important methods is the stemming, where terms are replaced with their original roots. To choose an accurate stemmer from a set of Arabic current proposed stemmers, a comparative approach based on a lexicon resource is proposed.

Evaluating HYPLAG against several other approaches demonstrates its higher performance with less computational time using the same dataset "Extrinsic Arabic Plagiarism Detection Dataset, ExAraPlagDet-2015". HYPLAG achieves a precision and recall values of 92% and 87%, respectively.

Table of Contents

List of Figures

List of Tables

Abbreviations

NLP – Natural Language Processing

POS – Part-Of-Speech

NER – Named Entity Recognition

IR – Information Retrieval

VSM – Vector Space Model

SN – Semantic Networks

KB – Knowledge-based

AWN – Arabic WordNet

LCS – Least-common-subsumer

IC – Information Content-based

1. Introduction

This chapter provides an overview over text-based plagiarism problem with some statistics that affirm the seriousness of plagiarism. Furthermore, it discusses the importance of using plagiarism detection systems in researching scope. According to [1], plagiarism targets two fields, text documents and programming source codes where code is reused without a citation. In this thesis, we will focus on the Arabic text documents as they have not been addressed by researchers like other languages.

1.1 Background

Many definitions have discussed and clarified Plagiarism term. According to Wikipedia [2], it gives a general definition of plagiarism which is: “Plagiarism is the wrongful appropriation and stealing and publication of another author's language, thoughts, ideas, or expressions and the representation of them as one's own original work”. Other sources have given different cases for plagiarism as definitions, as in plagiarism.org [3], plagiarism has four main cases:

- To steal and pass off the ideas or words of another as one's own.
- To use another's production without crediting the source.
- To commit literary theft.
- To present as new and original an idea or product derived from an existing source.

Therefore, plagiarism is not associated with just copy and paste sentences from other sources, it also includes the steal of ideas of others with concealing the owner of them. Plagiarism term is located for literary theft and not others types of stealing. When an object has robbed, we say that the object is stolen but not plagiarized! Plagiarism concept are more specific with the text (researches, articles, etc.). Plagiarism problem is an old issue, even before the existence of the internet, pirates have tried to use others’ researches from libraries to enrich their own. But with the

internet the problem become more critical since there is huge access for the researchers to the electronic libraries which make the detection process also more difficult. Many researches nowadays have many researches published on a different journals, conferences and digital libraries. The availability of this content poses many challenges for authors and other counterpart researchers.

Plagiarism process differs in size, and in the amount of plagiarized text. Plagiarism is not always intentional, according to [4], there are three categories of plagiaristic behaviors. The first is the intentional behavior, where the plagiarist deliberate attempts to steal other's text with consciousness of that work. This category of plagiarism is due to lapses in ethics. The second one is the unintentional plagiarism, where it appears due to inadequate knowledge of how to integrate the source of the text into one's work, so plagiarized cases appear unintentional. The final category, is the developmental plagiarism (patchwriting), where the plagiarist in a community, tries to mimic the language and ideas of that community with exact wording from the source, which is not common knowledge. Patchwriting is a strategy that new writers use to learn the ideas and forms of expression of a community.

A survey of more than 9250 graduate students over a course within 3 successive years has shown that 24% of graduates admit to copying few sentences from Internet sources without footnoting it and 25 % admit to copying few sentences from written sources also without footnoting it [5]. This statistics and others [6,7] have emphasized that plagiarism issue is a real problem and it poses risk on literary content. Therefore, detection phase for plagiarism is needed to overcome this problem. Therefore, institutes have conducts researches in this domain to find a decisive solution.

Plagiarism has been classified into two main types in which the plagiarists try to conceal the source. Academic plagiarism where academic researchers uses others' researches within their own

by submitting plagiarized content to journals. While professional plagiarism is far away from academic scope, it's regarding to professional workplaces. Professional plagiarism is the copying of reports or contents from a competitor in a sector.

Plagiarism can be done in one of three main patterns. In many of plagiarism cases, plagiarist copies the source sentence without any changes in the sentence, this called direct copy (copy and paste). This pattern of plagiarism is very easy to detect (dummy method) since there are no changes in the plagiarized content. Others try to complicate their attempts by rewriting the source sentence in different format while conserving the original idea of sentence with original main terms, this pattern called sentence reordering. The third pattern of plagiarism is paraphrasing. Paraphrasing can be achieved by different methods, replacing terms with their synonyms, changing word order, converting numbers to different forms, using different connecting words, or changing from active voice to passive voice (or vice versa). Paraphrasing is not restricted to a specific pattern, it can include any method that can change the general form of a sentence with conserving the original idea.

With the expansion of plagiarism behavior in all fields and the availability of electronic free libraries that provide easy access to their contents, and manual checking of plagiarized content for new published researches are becoming a very hard task to be accomplished, automated plagiarism detection is very highly-needed.

Researchers have proposed different approaches and systems to detect plagiarized content (see chapter 2) for variety of languages (Arabic, English, and German etc.) and other proposes systems that are aware of cross-language plagiarism where plagiarized content are translated from one language to another. These attempts of proposing approaches are started by Ottenstein, 1976 [8] in which a solution for detecting plagiarism is proposed by partitioning papers and checking if the

partitions have common intersects. The system was proposed due to large number of students in classes.

Plagiarism detection approaches are categorized in two main methods, intrinsic and extrinsic. The intrinsic method uses techniques to extract writing style of the writer to verify the originality of the paragraphs in a document. While extrinsic method tries to investigate plagiarized cases using different techniques based on source document that has the ownership rights. HYPLAG is dedicated for extrinsic plagiarism detection cases.

Nowadays, many websites provide detection solutions such as Grammarly [9], PlagScan [10], Turnitin [11] and many others, as free or paid service. Most of these systems are dedicated for English language and a smaller number for other languages. Most of these systems use Google search engine to reveal plagiarized cases by passing exact match query which rounded by double quotation marks. This approach can reveals cases that follow copy and paste pattern, apart this, nothing can be detected. So, complex structural and rich morphological languages such as Arabic need more attentions, since these services can't infer plagiarized content that employ complex plagiarism pattern as paraphrasing, also Arabic language is one of top five languages that have the highest number of speakers according to [12, 13] which has large number of researches written in it. Even more, Arabic is the religious language of the biggest religion around the world which Islam. Therefore, providing an Arabic plagiarism detection tool is needed which will helps Arabic schools and universities to overcoming this issue easily and quickly.

In the following chapter, a literature review over proposed plagiarism detection approaches for Arabic and English languages.

1.2 Problem Statement

As plagiarists' researchers steal other researchers' ideas and information content, plagiarism is becoming a real issue. Plagiarism detection in Arabic scope is still at an early stage as compared with other detectors in other languages, mainly the Latins. So, there is an essential need for plagiarism detectors. Thus, a small number of techniques have been proposed. However, they still suffer from some limitations and drawbacks since they rely on traditional exact matching techniques to infer the plagiarized content. These techniques are not efficient with the current plagiarism techniques [14]. For instance, when a scammer copies a content from a source like: "ذهبت أنا وعائلتي لزيارة المتحف" and he/she changes it to: "زرنا المتحف أنا وأسرتي", traditional exact term matching techniques will not reveal the plagiarized statement. The new statement is different from the copied one by swapping "ذهبت" term and "عائلتي" one by choosing another synonym, and restructures the whole sentence. Therefore, finding plagiarized content is not an easy process. Depending on these facts, in this thesis we will propose a detection system for the Arabic language that adopts a hybrid approach in an attempt to enhance the primal detection techniques by employing semantic resource to better detection of plagiarism. In this context, an ontology will be exploited to encode knowledge represented in Arabic scope domain in order to carry out the matching process at a semantic level. Our approach has combined two techniques to enhance the similarity results based on a comparative study conducted in [15]. They found that corpus-based measures give in general better performance in recall, whereas; knowledge-based methods achieve better precision. So, they recommend to incorporate semantic information into text similarities measures (knowledge-based) which will yield a high accuracy approach with low execution time. Our scope of work has been defined by the following assumptions:

- This work is dedicated to plagiarism detection in Arabic language.
- It combines corpus-based and knowledge-based techniques in one approach to get a high precision and recall system with low execution time.
- It examines the plagiarized cases based on sentence length. Our experimentally sentence length used is 23 words including stop-words.
- The input suspicious document is examined against a collection of documents. So, if the retrieved sentence in the beginning of the comparison process are contain at least N number of terms then it will be passed to the next system components to begin with similarity measurement processes.
- It will not be affected by the number of input documents or the length of input document negatively.

Our approach will retrieve most of the plagiarized cases depending on the knowledge size of the used semantic resource. The more size of the knowledge is, the more accurate detection results for paraphrasing pattern will be. The proposed approach uses ranked query method to retrieve the original sources of plagiarized sentences by using an inverted index, which forms a search engine structure. The resulted sentence of the searching process will be passed to similarity measuring component with the input sentence in which their similarity will be measured using the adopted approach.

1.3 Research Methodology

The following steps present the main phases that we perform during our research work:

1.3.1 Pre-processing of input documents

At this step, suspicious input documents are pre-processed based on performing the following NLP steps (more details are provided in chapter 3):

- Document Normalization.
- Text segmentation.
- Part-of-Speech Tagging (POST)
- Named Entity Recognition (NER).
- Stemming.

By performing the above mentioned steps, we prepared the input documents for the next phases by produces a uniform level of the text. In this phase, for each sentence we extract the verbs, nouns and adjectives components as a results to be pass to the next phases.

1.3.2 Ranked Sentence retrieval process and indexing

After we applied the previous mentioned preprocesses, we use the generated components to retrieve the most relevant or similar sentence form an index. In the indexing task, we have indexed the adopted sentence components in our work with their source sentences. The indexing process will facilitate and speed up the searching task of similar sentences using the produced sentence components from the previous phase. So, for an input sentence we retrieve the most similar sentence from the index based on its components using ranked query method.

1.3.3 The detection process of plagiarized text cases and optimization processes

By retrieving the most similar sentence in the retrieval process, the similarity detection process starts by receiving two sentences with their verbs and nouns components. We exclude the adjective components in the similarity detection process to reduce the processing time. We have chosen the adjective component because it was the less important component in a preliminary study conducted to reveal the patterns of plagiarism behavior (for more details see section 4.2). In this phase, we

compare both verbs and nouns sets of both received sentences in VSM using tf-idf weighting scheme. For the sentences that need further investigation, we employing feature-based semantic measure which utilize a knowledge resource to reveal plagiarized cases. To improve the accuracy of the generated results, we built two models for adjusting and aggregating the detected plagiarized cases.

1.4 Contributions

This thesis makes the following contributions:

1) It proposes an Arabic text plagiarism detection approach that covers all of the patterns of plagiarism by combining content-based and knowledge-based similarity measuring techniques and by utilizing Arabic text components based on preliminary study.
2) The proposed approach has achieved the highest results with 92% value of precision and 87% a recall value, with the fastest execution time in around 143000 seconds according to ExAraPlagDet 2015 competition.
3) A new comparative approach for Arabic stemmers to select the most efficient stemmer based on knowledge resource has been proposed.

1.5 Thesis Organization

In Chapter 1, a background review on plagiarism topic was presented along with challenges that are faced due to its spread. Also, the motivation behind this thesis has been discussed with presenting the problem statement illustrating the contributions.

In Chapter 2, a detailed review of related plagiarism detection approaches in text-based documents is conducted and highlighted the current issues in this domain to be improved and addressed.

Chapter 3 discusses the normalization, named entity recognition, Part-of-Speech tagging and stemming methods for Arabic text which are preprocessing steps of our approach. Also a stemming comparative approach is proposed in the stemming section.

The description of HYPLAG with the adopted approaches is presented in Chapter 4. A simple study on university students to understand their behavior in plagiarism is discussed.

Chapter 5 describes the implementation steps of HYPLAG. The evaluation results of the systems is presented using traditional information retrieval metrics. In addition, we has tested the adopted corpus-based approach on the used dataset and showed how the coupling with knowledge-based approach has improved the results. A comparison between HYPLAG results and other systems that were tested on the same dataset is presented. Finally, Chapter 6 concludes the system description and provides future prospects on the approach.

2. Related literature

In this chapter, we will conduct a literature review on plagiarism detection approaches. At the first section, a detailed review on the English language plagiarism detection approaches, then another one for the Arabic language systems.

2.1 Related Works

2.1.1 Plagiarism Detection in English Language

Different approaches have been proposed in this domain to overcome such a critical problem. As we mentioned before, plagiarism detection has two forms, the Extrinsic and the Intrinsic. The Intrinsic plagiarism detection method is not well-matured as extrinsic and has one major drawback that could eliminate it. The Intrinsic method reveals plagiarized cases using the writing style of the writer. Thus, it can't reveal anything when the whole document is plagiarized. The writing style of the author can be extracted using different techniques such as the averaged word frequency [16]. While extrinsic method is more robust and achieves better results in detection ratio [102], it has a drawback that there is a need for a reference collection to compare with. Extrinsic Plagiarism detection methods can be categorized to semantic and character-based approaches. The semantic approach has two other sub-approaches as shown in Figure 2.1.

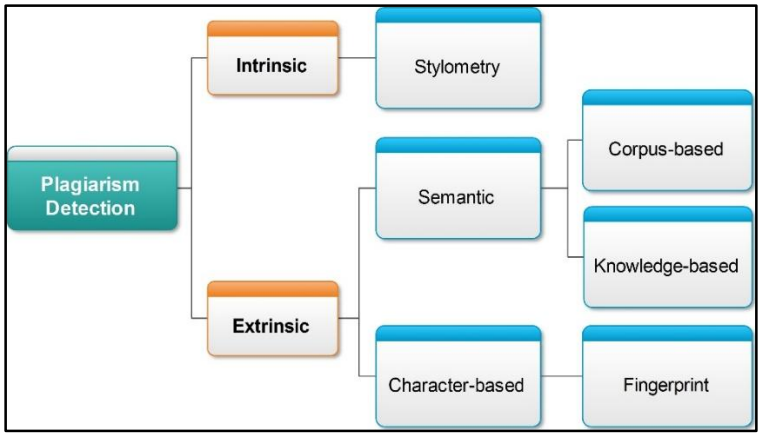


Figure 2.1: Classification of text-based plagiarism detection approaches

The first one is the corpus-based approach in which the information extracted from large corpora, like ex. corpus – based similarity process identifies the Number of times a term is repeated in a document. The knowledge-based approach utilizes the power of ontologies in representing terms to measure the similarity, by measuring the length between two terms nodes, as an example. For the second extrinsic approach, the fingerprint is used to measure the similarity of two texts by comparing their fingerprints. Fingerprints are generated by encoding n-length characters of a string using a hash function [17]. The main idea behind hash function is that it generates a unique numerical representation of the n-length input characters. This approach is widely used in this domain.

English language got the first attempts in proposing plagiarism detectors. Authors in [18] have proposed a new method for detecting the plagiarized sentences. They show that n-grams stop-words reveal important information about the plagiarized cases with exact boundaries detection. Their approach has set aside the common practice of eliminating stop-words and reports competitive results when being tested in one of the famous datasets. An extrinsic and intrinsic plagiarism detection approach has been proposed in [19]. The proposed approach exploits vector space model (VSM) to overcome plagiarism issue with its both forms. For the extrinsic part, its consists of three stages: the vectorization of sentences in each document in the corpus; the vectorization of each sentence in suspicious document and finding the nearest neighbor(s) in the reference corpus vector space by using similarity thresholding; and finally, merging subsequent plagiarized sentences into a single block. For the intrinsic detection part, the approach has applied the vectorization of each sentence in the suspicious document, after that the determination of the plagiarized sentence is obtained basing on the document's mean vector. Different Stylometry

features are adopted within VSM to support the detection process, such as: average word frequency and part of speech tags. The approach provided an acceptable result with value of precision 60% and 40% value of recall. Another extrinsic approach has been proposed in [20], which is based on the semantic relations between words and their syntactic composition. The approach targets different plagiarism types: exact copy, paraphrasing, sentence restructuring, and transformation of words by being replaced with other similar in meaning words. The approach consists of three steps, pre-processing of sentences using basic natural language processing (stemming, tokenization and stop words removal). After applying these processes, the sentences are stripped of unneeded letters or characters. Then, documents are decomposed into sentences in order to detect similar pairs. Finally, the approach starts to measure the semantic similarity by building semantic vector and word-order vector of the word sets (group of tokenized words from the suspicious and original sentences). Word-order similarity is calculated by the syntactic vector approach [21], and semantic similarity is calculated using the semantic-vector approach with cosine similarity algorithm [22]. The similarity between the two sentences is calculated using a linear equation which combines word-order and semantic similarities. Vector space model based similarity approach has been proposed in [23]. The approach depends on both path-based semantic measure and information content-based measures. In the measuring process, the approach considers two types of relationships: between verbs and between nouns to measure similarity. For each sentence, two vectors were built, one for nouns and the other for verbs, then the similarity is measured using Lin's measure algorithm [86] which is one of knowledge-based similarity measures. In [24], authors have proposed an approach for capturing similarity between sentences based on lexicon. Their approach has been achieved by the following steps: tokenizing both sentences, applying Part-of-Speech tagging to determine nouns and verbs, and applying stemming

to get a unified form. Word Sense Disambiguation (WSD), was integrated to reveal ambiguity, since many words could have more than one meaning. Thus, they need to know which sense should be used from lexicon. Finally, a matrix of senses has been built with measuring similarity between each two senses; each sentence's senses is denoted by a coordinate. The semantic similarity is measured using both Wu & palmer [83] and Leacock & Chodorow [25].

2.1.2 Plagiarism Detection in Arabic Language

Different approaches have been proposed after the first attempt in 2009 by Alzahrani [26], in which a statement-based fuzzy set information retrieval (IR) system has proposed. The paper investigated the use of statement-based plagiarism detection in Arabic documents and compared the performance of fingerprinting technique with the performance of fuzzy set technique. To increase the accuracy in the comparison process, an Arabic fuzzy thesaurus has been built as a pre-step for implementing fuzzy-based technique. Also, a dataset of around 3763 statements has been built to evaluate the comparison process. Their reported experimental results show that the fuzzy-based IR technique had outperforms fingerprints matching technique.

"RDI" is an extrinsic plagiarism detection approach [27], which was proposed for an Arabic language plagiarism detection competition [102]. It attained the first three positions with value of precision 92% and value of recall 84%. The proposed approach consists of three base models: the document retrieval, the alignment model and the filtering model. In the first model, the document was parted into sub paragraphs, Inverse document frequency weights were calculated and two queries were constructed for each paragraph in different way for optimal results, then they were received by a search engine. The alignment model consists of different techniques (skip-gram, sentence index based and common word based approaches). Finally, the filtering model applied set of rules to decide whether the sub paragraph is plagiarized or not, by checking the length of the

chunks (source and suspicious) and comparing number of common words within. “FPDA” is a Framework for Plagiarism Detection in Arabic documents [28] employs Google search engine based detector which utilizes the effectiveness of the engine to reveal the plagiarized cases from a document. The approach consists of two models, global and local. The global model task is to prepare the retrieved documents from the search engine by applying some filtrations on content to reduce computational costs, and convert the content to a unified encoding type. Then documents similarity is calculated to remove the false positive ones, which also reduces the computational costs and increases the accuracy, according to the authors. In the second model, they prepared a query by selecting at least 5 consecutive words in the suspicious document and then pushed to the search engine as a phrase query. The similarity calculations between documents are applied using precision and recall matrices. Jadalla and Elnagar [29] have proposed “Iqtebas”, which is a text-based system that applies text preprocessing techniques with fingerprints on documents and uses an inverted index to improve the comparison process. “ZPlag” is a web-based system [30] that has been built to judge students’ assignments and papers. It consists of 3 phases: preparation, processing and similarity detection phase. The system employs fingerprints techniques with Longest Common Substring (LCS) algorithm to measure similarity between fingerprints. In [31], the author has employed different methods to achieve a high accuracy detection system that reveals different types of intelligent plagiarism. The system receives the plain text to start the detection system with some of preprocessing techniques to normalize it, such as removing diacritization, tokenization and removing of stop words. Then, the phrase extraction process was started by using of N-gram algorithm to represent the document by using its phrases. A Term frequency and Inverse document frequency technique (Tf-Idf) weights matrix was built to model the document by adding the document’s unique phrases with their synonyms to reveal the hidden associations of the unique

phrases in the documents. Finally, the Singular Value Decomposition algorithm is used to detect the pairwise document by computing the similarity score. Authors have built their own dataset to evaluate their approach, and it achieves a promising capability. The system had been compared to Plagiarism-Checker-X [32] and achieved better results in three main plagiarism patterns (restructuring, synonyms replacement, and using both of them). Plagiarism-Checker-X achieved 30% as detection score while their approach achieved around 77%. Ameer and Mohd [33] have proposed a plagiarism detection approach that is based on Tf-Idf technique. Authors have inspired the approach from using Tf-Idf technique in plagiarism detection in English language. The approach is a statement based rather than document or term based. It has used some pre-processing methods, started with normalization of some Arabic letters by grouping them into one shape. For example, أ, ى, ئ, ء, ؤ, إ, آ letters have been normalized to "ا" to unify the comparison process. Finally, a Tf-Idf index of terms has been built to start the comparison process. Each term in the index has been checked if it has synonyms by using Arabic Thesaurus Project dictionary, then, Tf-Idf weights have been calculated for terms and cosine similarity is applied to measure the similarity of documents terms. The approach was evaluated on a dataset that contains different cases of plagiarism (copy and paste, restructuring and synonyms replacement). These systems with others are summarized in Table 2.1.

Table 2.1: Overview of Arabic text plagiarism detection systems

System	Approach	Investigation level	Knowledge resources	Knowledge-based measures	Dataset	Performance	
						Precision	Recall
(Alzahrani, S., 2015)	fingerprinting	Chunk-based	no	no	ExAraPlagDet 2015 [34]	83%	53%
ADP – (Alzahrani, S., 2009)	fingerprinting and fuzzy set	Statement-based	yes	no	Own	90%	85%
APlag – (Menai, M., 2012)	Fingerprinting	Chunk-based	yes	no	Own	93%	100%
Iqtebas - (Jadalla, A., 2012)	fingerprinting with windowing	Chunk-based	no	no	Own	99%	94%
ZPlag – (Farahat, F., 2015)	fingerprinting with windowing	Chunk-based	yes	no	Own	97%	94%
FPDA - (Khan, I. H., 2015)	feed windowed query to Google SE	Chunk-based	no	no	Google engine	-	-
RDI – (Magooda, 2015)	IDF weights with windowing	Chunk-based	no	no	ExAraPlagDet 2015 [34]	92%	84%
PDSA - (Hussein, 2015)	TF-IDF matrix with Singular value decomposition algorithm	Statement-based	yes	no	Own	score = 97%	
(Ameer A., 2015)	Term frequency and inverse document frequency (Tf-Idf).	Statement-based	yes	no	Own	90%	92%

2.2 Summary

Several approaches have been proposed in plagiarism detection field for many languages. These approaches can be categorized into two main types, language dependent and cross-language detectors where plagiarism can be detected through multiple languages documents or between two languages. English language has given more attention than other languages in this field especially the Arabic language. Since Arabic language has complex structure and Arabic Natural-language-processing (NLP) resources are few which make it a challenge. However, current Arabic plagiarism detection systems have not fill the gap, and there is need for more precise and accurate approaches to overcome this critical problem. Therefore, in this thesis we will propose HYPLAG as a solid plagiarism detection approach for Arabic language in which covers all plagiarism patterns.

3. Text-Preprocessing

In this chapter, we introduce some of preprocessing methods that have been used in our approach to enhance the system effectiveness. Also a brief review over Arabic language features and properties is introduced which can facilitate the understanding of the Arabic morphological nature. Then Named Entity Recognition (NER) method is introduced in section 3.4, follow it Part-Of-Speech tagging (POS) in 3.3 and finally the stemming method is reviewed. Furthermore, a comparative knowledge-based approach proposed to adopt most efficient Arabic stemmer within our approach.

3.1 Introduction

Arabic language is one of the most widely used languages around the word, it has more than 422 million of speakers across the world [35]. Arabic language has three main forms that are used in different places. The first is the Classical Arabic (CA) which is the used language in literary texts, also it's the Arabic form used in Al-Quran. The second one is the Modern Standard Arabic (MSA) which is considered as the written language of journalism and the news, it is fairly similar to Classical Arabic. The third form of Arabic is the Colloquial Arabic which refers to dialects language that predominantly spoken differently according to the Arabic countries. Arabic language consists of 29 letters with Hamza (ء). These letters can be written in different ways, depending on the position of the letter in a word. For example, the letter ي (Yaa') can be written as "ـي" if found as the last letter in a word, "يـ" the form of the middle position or "يـ" if a word starts with. Arabic letters have no upper and lower cases like English language. The Arabic writing direction from right to left (rtl), where most of the languages is left to right (ltr). Arabic words have two genders, if used for male masculine (مذكر) or feminine (مؤنث) for the female. And in case of counting things, it has three forms, the singular (مفرد), dual (مثنى) and plural (جمع) where in English there are just two, singular and plural. Arabic words can be mainly classified into three parts of speech: nouns, verbs and particles. Arabic

terms have many forms because of its morphological complexity [36]. For each term, additional characters could be added. For the word "لعب" at the beginning (Prefix: اللعب), at middle (Infix: لاعب), or at the end (Suffix: لعبوا). In many cases, special characters can lead to a new meaning. For example, for the term: "لعب" means PLAY, if we add a vowel character in the middle to be: "لعاب" the meaning has changed to SALIVA, and this case is widespread in Arabic language. In Table 3.1, an example of Arabic word forms that can be produced from a basic word. In English language, additional characters are just added at the end of term ex. Play - Plays - Played, or in some verbs infixes could be added (just in changing tenses of verbs) like: write - wrote.

Table 3.1: Different affixes than might added to the word سائق

Arabic word	English meaning	Number
سائق	Driver	Male Singular
سائقة	Driver	Female Singular
سائقان	Two Drivers	Male Dual
سائقون	Drivers	Male Plural
سائقين	Drivers	Male Plural
سائقات	Drivers	Female Plural
السائق	The Driver	Male Singular
والسائق	And the Driver	Male Singular
كالسائق	As the Driver	Male Singular
سائقه	His Driver	Female Singular
سائقها	Her Driver	Female Singular
سائقهم	Their Driver	Singular
سائقهما	Their Driver	Singular
للسائق	For the Driver	Singular

Moreover, Arabic language has a relatively free word order without using any independent particles, this feature is the challenger of the plagiarism detectors. The following sentences can describe this feature:

ذهب الطالب للمدرسة: Went the student to school;
ذهب للمدرسة الطالب: Went to school, the student;
للمدرسة ذهب الطالب: To school, went the student;
للمدرسة الطالب ذهب: To school, the student went;
الطالب ذهب للمدرسة: The student went to school;
الطالب للمدرسة ذهب: The student, to school went.

Where all these sentences describe one situation which is "The student went to school", and this feature is not available in most of the other languages with using the same words.

Another main feature of the Arabic language is the diacritics (Harakat) or in a literal meaning "forming". Diacritics are marks added to the letters in words to provide a phonetic guide, and they can change the word meaning as: عالَم with Fatha on (لـ - Lam) means world and عالِم with Kasra means scientist. Arabic diacritics include: Fatha, Dma, Kasra, Shadda, Sokun and Tanwin as illustrated in Figure 3.1).

Fatha	Kasra	Dma	Sokun	Shadda	Tanwin
ـَ	ـِ	ـُ	ـْ	ـّ	ـً ـٍ ـٌ

Figure 3.1: Arabic Diacritics

According to [37, 38], Arabic language is a complex morphology one when compared to English. Also, words synonyms are widespread in Arabic language which made it a highly inflectional language [39]. All the previous features of Arabic can form a challenge for any application that targets the Arabic language.

Text preprocessing is an initial step for most of NLP application fields and it is a primary step in text mining process. In our system "Hyplag" we have employed different preprocesses to get a unified and clear form of text to enhance the next processes on content and ensures the validity of the detection process. Figure 3.2 summarizes the preprocessing steps of HYPLAG.

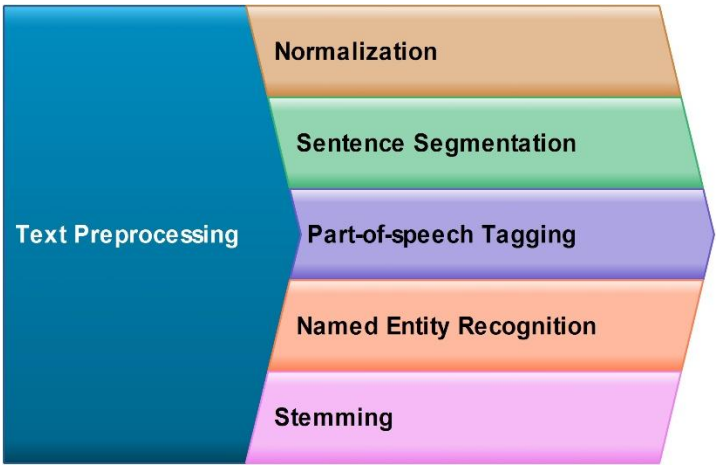


Figure 3.2: Text preprocessing steps of HYPLAG

3.2 Normalization and Sentence segmentation

Preprocessing stage starts with the normalization of step, each input sentence is processed to remove diacritics, non-Arabic letters, all words that contain numbers and words that are composed of one letter to reduce error rate. After that, the text is segmented into n-words length. The sentence's length in HYPLAG have been chosen experimentally during tuning process, where small length of sentence results in a high computations cost processes and low informative sentences, since there are stop words or particles between the main parts of speech terms. Where large length of sentences causes high informative text but it results a miss-detection of plagiarism cases since the uniqueness of the plagiarized cases will disappear (small cases are overlapped in one large case).

3.3 Part-of-Speech Tagging (POS)

POS tagging is the process of tagging an input text with its parts of speech units (verb, adverbs, adjective etc.). Researchers face difficulties in implementing such approaches due to the high ambiguity of the Arabic language. The tagging process can be achieved in three main approaches [40]. Rule-based approach in which a knowledge base of rules is constructed to define how the tags will be assigned. The knowledge base of rules is defined by the linguists. While the statistical approach is implemented by utilizing a previously tagged corpus to train

the tagger, and finally the hybrid approach in which the two previous approaches is combined in one approach. Current research works move toward this approach as it gives better tagging results [41]. A set of POS tagging systems have been proposed by [42,43,44]. In Hyplag, we have used Farasa POS tagger [45] due to its fast processing. For each input sentence, we need to extract verbs, nouns and adjectives to be used in the next system's phase.

After extracting the needed components from an input document, terms are indexed with their sentences. Each document is split to sentences of length N-words. The length of sentences is experimentally selected. For indexing, we have used Lucene Search tool [46].

3.4 Named Entity Recognition (NER)

NER is the process of extracting names of persons, organizations or places to exclude them from any alteration. Its first introduction was done by the Message Understanding Conference (MUC-6) [47]. It's an important technique in natural langue processing domain since it allows the identification of proper nouns to exclude from further processes. Different NER approaches have been proposed for the Arabic language in an attempt to enhance Arabic NLP techniques and resources [48, 49, 50, 51]. The NER task in Arabic domain is hard and complex. This is due to the fact that there is no capitalization for nouns as in English, also the proper nouns can represent several regular words as in the example: the name "راشد" means "adult", also "فارس" means "Knight". Even, in many cases, it could represent verb, such as: "حمد" as an Arabic person name means "magnify" and much more. There is a huge potential for finding named entities which will distort the detection process as they are not eligible for stemming or root extraction processes. We have coupled two techniques for the extraction process, the first is by using Farasa NER tool [52] to detect the names or the compound nouns. The other technique is by employing Term frequency-Inverse document frequency (Tf-Idf) technique as illustrated in Equation 1. It has been used in many applications in information retrieval and text mining

as a weighting scheme. Tf-Idf is a numerical statistic method that is intended to reflect how important a term is to a document in a corpus [53].

$$W_{t,d} = (1 + log\ tf_{t,d}) \times log10\ (\frac{N}{df_t}) \quad (1)$$

Where, Tf refers to number of times term (t) that occurs in the document (d), N is the corpus size and df is the number of documents that term (t) occurs in. The highest value of Tf-Idf weight is obtained when a term (t) occurs many times within a small number of documents, and the lowest when a term occurs in all documents (as stop words). The terms have the highest Tf-Idf weight value are chosen as named entity in our approach. An advantage of Tf-Idf technique over the Farasa tool is Tf-Idf technique extracts an untruthful terms, such as: "غيرمحارب", "وذلكتميز" "هذاملف", "هذهممالك" and much more, which Farasa can't detect. The stemming of these terms can lead to false positive plagiarized cases.

3.5 Stemming

Stemming is the process of converting terms into their roots or word stem forms to achieve a formal shape of the terms. Stemming has become an initial step in many research fields such as Information Retrieval (IR), Text Categorization (TC), Indexing, Text summarization (TS). It has many benefits to these fields and mostly it reduces the number of redundant terms by joining them into their original representation and therefore reduces the corpora's size or indexer length. As a result, we will have a low computation cost application which increases IR systems' matching probability. Many stemming approaches have been proposed and used as a pre-processing task in different NLP applications. Stemmers are built for specific languages, because it needs pre-understanding of language grammar roles. It has been developed for a large number of languages including Arabic, Persian, English, French, Chinese, German and more. For each language, many stemmers have been proposed. The various stemmers, for the Arabic language include: ISRI stemmer [54], Khoja stemmer [55], Berkeley

light stemmer [56], Light10 stemmer [57], Motaz stemmer [58], SP_WOAL Arabic Light Stemmer [59], Tashaphyne stemmer [60].

Arabic stemmers mainly employ one of following three main stemming approaches that are dedicated for the Arabic language:

- Root-based stemming:

  In the Root-based stemmers, a morphological analysis is conducted, and the root of the word is extracted by removing extra characters using pattern matching. Each character's position of a word is compared to its corresponding one in the pattern as shown in Figure 3.3.

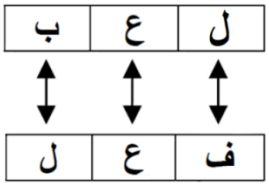


Figure 3.3: Extract the root of the word (لعب) by pattern matching

  The pattern is called "Almasdar المصدر" in the Arabic language, and there are many patterns for terms like: "فعل, افعل, افعول, استفعال, استفعلة" and others. It has been adopted by Khoja [55], ISRI [54], Motaz [58] stemmers.

- Light stemming:

  In the Light Stemming approach, the goal is to strip the words from the most common prefixes and suffixes. A list of prefix and suffix characters is being defined. A sample of the lists is: prefix (ال, لل, ل, ك, ب, و, س) & suffix (ات, ون, ين, ان, ها). This approach has been used by Light10 [57], Motaz [58] and Tashaphyne [60] stemmers.

- Statistical stemming:

  The Statistical approach is widely used in the English language but less in Arabic. In this approach, equivalence classes are formed from words that share n-gram letters as shown in Table 3.2. Then the n-gram similarity between words is calculated using the

Dice's coefficient. Finally, words are clustered based on their similarity results and represented with only one term [61].

Table 3.2: Example of shared 2-gram letters

Words	2-grams				
لعبوا	-	وا	بو	عب	لع
لعب	-	-	-	عب	لع
لاعبون	ون	بو	عب	اع	لا
لعبت	-	-	بت	عب	لع
ملعب	-	-	عب	لع	مل

However, different challenges face Arabic stemmers due to the complex nature of the Arabic language (see section 3.1) and the presence of special cases. Due to the diversity of stemming approaches, researchers are cautious to choose the appropriate stemmers that should be used in their applications.

3.5.1 Comparative Stemming Approach

In this section, we propose a knowledge resource-based approach to compare current available state-of-art Arabic stemmers based on their stemmed results. The knowledge resources that have been used is the Arabic WordNet AWN [62]. The involved stemmers include: ISRI [54], Khoja [55], Light10 [57], Motaz [58] and Tashaphyne [60]. We aim to compare these stemmers based on the availability of their results in the knowledge lexicon. Using this approach, we can abandon the manual reviewing of stemmers' results by linguistics to choose the best stemmer. A corpus for the comparison process has been built from two Arabic resources, Corpus of Contemporary Arabic (CCA) [63] and the Arabic Wikipedia [64].

Many researchers have proposed different stemmers, but we have found small number of them implemented to be involved in this study. The proposed approach has been applied on the most widely used stemmers in different research applications [65]. These stemmers include:

- Khoja Stemmer [55]:

  Khoja stemmer is a root-based which uses pattern matching to get the root. It starts by removing diacritics and punctuation of the input word, then a predefined set of lists for prefixes and suffixes are used to strip the word. At the end, the result word is validated using a root dictionary. Although, Khoja stemmer is highly complex as it involves linguistic rules in the stemming process, it is one of the early and powerful stemmer that was developed for the Arabic language [66]. The main drawback of Khoja stemmer is that it requires periodic updating to guarantee that newly discovered terms are correctly stemmed. The average score that Khoja had achieved on TREC-2001 corpus is 0.341; TREC-2001 is a corpus built to measure the performance of stemmers from the information retrieval perspective [67,68].

- Information Science Research Institute's (ISRI) stemmer [54]:

  ISRI stemmer is a root-based one. It starts by normalizing the input word (delete diacritics and non-Arabic characters) and unifying the different forms of Hamza to "ا". Then, the normalized word follows a set of rules to remove possible prefixes. Finally, the word is mapped to a group of patterns according to its length. ISRI searches for possible matches within a list of patterns, if there is no match, it removes possible suffixes. The stemming process should be stopped when the remaining length of the input word is three characters or fewer. ISRI does not validate roots against any type of dictionaries. This in turn, will lead to the fact that the extracted roots are not necessarily correct; the root could be a meaningless set of characters. ISRI stemmer had achieved 0.395 as an average precision on TREC-2001 [67].

- Light10 Stemmer [57]:

  Light10 stemmer is low complexity one since it is concerned with removing prefixes and suffixes of the words. It starts by removing diacritics, punctuation, and non-Arabic

letters, normalizing process take its place as in ISRI, and finally removing prefixes according to a set of rules. Light10 is a good example of a light stemming approach, but it has one weakness which is unable to handle irregular plural [65]. Light10 stemmer had been tested on TREC-2001 and it achieves 0.413 as an average precision.

- Motaz Stemmer [58]:

  This stemmer could be called hybrid stemmer as it is a mixture of Khoja and Light10 stemmers (Root-based and Light10 stemming). The part of root extraction is used from Khoja stemmer with additional stop words list added. Whereas for light stemming, Light10 stemmer algorithm is implemented within, and it proceeds after normalization process is conducted.

- Tashaphyne Stemmer [60]:

  Tashaphyne Stemmer provides two versions: light stemming and root extraction ones. It defines two lists of prefixes and suffixes to strip words and finding the root before it removes unrelated letters such as diacritics and non-Arabic letters. In our evaluation process, we will deal with each (light and root-based stemming versions) as different stemmers.

Arabic stemmers have been evaluated in most of the studies by listing their advantages, disadvantages or their adopted approaches [65] [69] [70] without any real proof on their achievements. So, we present an evaluation criterion that exploits the knowledge resource to judge on stemmers' results whether they are meaningful or not. We have built a corpus of Arabic verbs, nouns, and adjectives, consists of 31,000 unique tagged words of Arabic resources.

Our source of the text is from the Corpus of Contemporary Arabic (CCA) [63] and the Arabic Wikipedia [64] as we mentioned before because CCA involves hundreds of documents with a variety of topics, likewise Arabic wiki. The variety of topics is needed since we need to evaluate

the stemmers on largest number of words, so the variety of nouns and verbs is needed here. To prepare our corpus, we have conducted a pre-processing stage as shown in Figure 3.4 to extract the needed words.

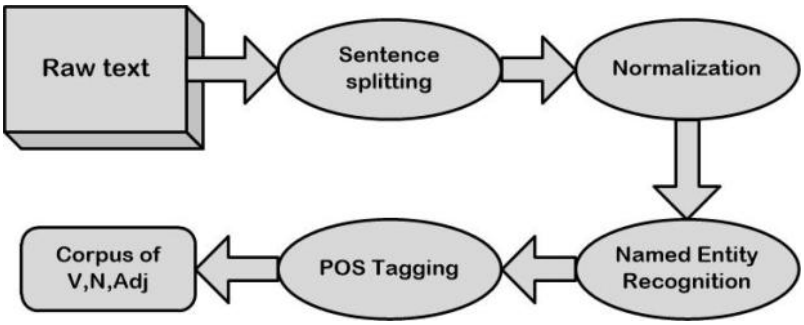


Figure 3.4: Preprocessing steps to build the stemming comparative corpus

After applying these steps, we got the corpus, a sample of the corpus is shown in the Figure 3.5.

```
<feature>
  <WORD>فسألـه</WORD>
  <POS>V</POS>
</feature>
<feature>
  <WORD>كنوع</WORD>
  <POS>NOUN</POS>
</feature>
<feature>
  <WORD>محدثـه</WORD>
  <POS>ADJ</POS>
</feature>
```

Figure 3.5: Sample of stemming comparative corpus

In this approach, the efficiency of Arabic stemmers is measured and evaluated by passing results of the stemmers to a knowledge resource to investigate the validity of the results. Each involved stemmer will start stemming the whole words of the corpus, duplicated roots or stems will not be added again. The results will be sent to the AWN to investigate whether they have a meaning or not since that AWN contains the roots of Arabic terms. So terms that have been stemmed correctly will be exist in AWN otherwise will not. The process is depicted in Figure 3.6.

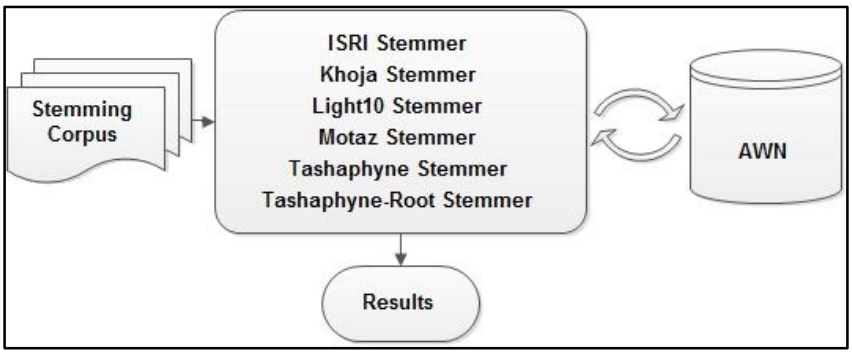


Figure 3.6: Chart of the stemming comparative approach process

By applying this approach, each stemmer will give a number of valid results that represent its performance on this corpus. The higher number of the available results in the AWN is, the better performance is achieved. The number of valid results will be represented as the score. The Pseudocode for the approach is demonstrated in Figure 3.7.

```
1  COMPARE_STEMMERS()
2  int counter=0
3  String stemmedTerm
4  boolean found = false
5  terms ← ReadTermsFromCorpus()
6  STEMMER stemmers ← SelectStemmer()
7  while terms ≠ NIL and stemmers ≠ NIL
8  do stemmedTerm ← stemmers(terms)
9      found ← CheckAvailabilityInAWN(stemmedTerm)
10     if found = true
11      then counter++
12           found = false
13 end while
14 print counter
```

Figure 3.7: Pseudocode of the stemming comparative approach

The results of this comparative approach is mentioned in section 5.3.6.1.

From our evaluation and results, we chose Motaz stemmer to be involved in our system since it has the highest stemming efficiency due to time and valid results that achieved.

According to the results, we can conclude that Arabic stemmers need more improvements since we have noticed many misstemming cases. In addition, during manual reviewing, we found that stemming could lead to ambiguity as there are few cases in which the word stemmed to another stem or root that are not related but have a meaning. The improvement of this approach can be achieved by using a robust technique in recognition of named entity, and by enriching

of AWN or using another knowledge resource along with it. The Enrichment of the AWN's content will enhance the validity checking process of stems. Anyway, in our approach we have fed all stemmers' results to the same version of AWN so this will not affect the comparative process negatively, but it will reduce the efficiency of the approach since that many words maybe stemmed correctly but not founded in the AWN.

Light stemming approach is more efficient and clearly quicker than the root-based approach that uses pattern matching technique to produce the root since it is not concerned with producing the root. Moreover, the number of unique stems is larger while in root-based stemmers is lower (as in Unique stem field in Table 5.4), and this confirms that many forms of the word will be stemmed to same root which is a fact.

3.6 Summary

Text preprocessing is an initial phase in most of NLP applications. This phase can lead for further accuracy in results and better performance of the systems. Most of employed tools is this phase are language independent tools which need deep understanding of Arabic language features to choose the most suitable one's. Some preprocessing techniques such as Stoplist removal was not used in this phase because parts-of-speech step has cover its need. In POS step, the tagger is extracts the verbs, nouns and adjectives (needed components) and discards remain words (ex. Particles), so the remain components will not affect. Also we presented in section 3.4 NER technique by combining two techniques to achieve best results.

In section 3.5, stemming technique is presented and discussed with showing its importance in NLP field. Even more, a stemming comparative approach is presented along with to employ the most efficient implemented Arabic stemmer in HYPLAG system.

4. Documents Similarity

In this chapter we introduce both techniques that we combined as a core phase in HYPLAG. Section 4.1 will introduce the adopted techniques in literature with their features. Then a preliminary study conducted on university students to study their behaviors in plagiarism is mentioned in section 4.2. In section 4.3 our sentence retrieval component is illustrated and described to give a detailed view over sentence retrieving process. Similarity detection phase in HYPLAG is presented. In addition, it is illustrated how it uses components of sentences in measuring similarity with a detailed explanation of VSM and feature-based semantic similarity measure in section 4.4. Furthermore, optimization processes are presented in section 4.5. Also, we will describe the evaluation measures that we used to measure HYPLAG performance in section 4.6.

Figure 4.1 describes the HYPLAG overall system architecture.

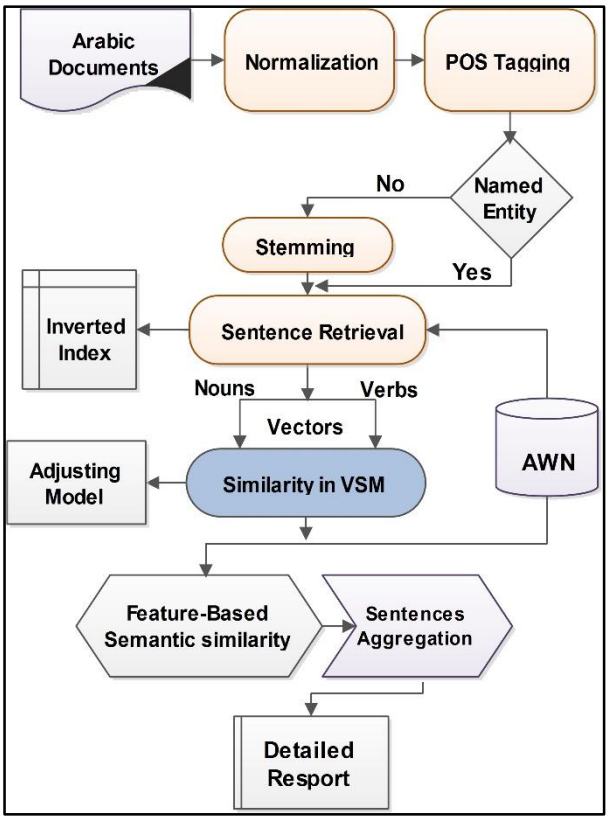


Figure 4.1: HYPLAG System Architecture

4.1 Introduction

Different similarity metrics have been proposed in the literature for the extrinsic plagiarism detection field. The similarity process is conducted on documents, sentences or text chunks level. As described in Figure 2.1, the extrinsic type of plagiarism detection has two main approaches namely, character-based and semantic. In HYPLAG case, we have adopted the semantic approach by combining its both forms to investigate the effectiveness of the coupling process in the Arabic language. Next, we review the two adopted techniques.

4.1.1 Vector Space Model (VSM) and Term Weighting Scheme

VSM is an algebraic model for representing text documents [71]. It has been proposed to enhance the representation of text in documents to be presented in more scientific form. VSM has been widely used in different domains such as information filtering [72], information retrieval [73], indexing [74] and relevancy rankings [75] since traditional bag-of-word text view become unfeasible in many applications. In VSM, documents or sentences are represented as vectors using any term weight. Different term weights can be used with VSM. Boolean model is one of the popular weighting schemes. For each term to be presented in a vector, zero "0" is added when a term is not within a document or 1 otherwise. Word count is another scheme, where the weight in the vector is presented by word count in a document. Another popular weighting scheme is the Term frequency – Inverse document frequency (Tf-Idf). This scheme is mostly used. Tf-Idf is a statistical corpus - based scheme that is used to present the importance of a term in a corpus or document. It has been used for document ranking purpose rather than similarity measure. However, VSM with Tf-Idf weighting scheme for measuring vectors similarity plays the rule of corpus-based similarity measures. Where corpus-based measures identified the degree of similarity using information exclusively derived from a large corpus [76] to give a unique numerical value for each term. Tf is the occurrences number of a term in a document, and df is the number of documents that the term t occurs in. Idf can be

calculated by dividing the corpus size over df as illustrated in Equation 1. High value of Idf means, the term t occurs in few number of documents, where low value of Idf is produced when a term t occurs in many documents. By combining both Tf and Idf, we can get three cases for terms weights in this scheme:

When term t has a highest value, it means that this term occurs many times within a small number of documents which gives it, a discrimination power over those small number of documents.

Where lower weight when the term occurs fewer times in a documents and appears in many documents.

Lowest weight value is obtained when term t occurs in most of documents. Low weight terms can be used to extract stop words list from a corpus, since stop words appears in all of documents normally.

An essence of using Tf-Idf method with large corpuses is, the larger size of corpuses used are, the more unique weights terms have. This is because of the increasing of documents size in corpus or documents length gives a lower probability of duplicating a weight value for two terms in corpus. That is, the weights in Tf-Idf scheme can present a fingerprint for weights. Where in low size corpus, Tf-Idf can't make that difference since there is huge potential of finding two terms having the same weights since they share the same source documents with the same frequency in each document. This feature can be an adversary and supporter by using Tf-Idf weighting scheme in plagiarism detection field, depending on the corpus size.

The variations of this scheme give it a high priority to be used in search engines for results ranking and scoring relevance [77]. Despite its strengths, it has limitations. Tf-Idf method does not make the jump to the relation between words, so in plagiarism detection field when a term is replaced with a synonym, Tf-Idf method can't present a weight for, to be involved in detection process [78].

4.1.1.1 Using VSM with Tf-Idf weighting scheme in plagiarism detection example

The main idea of using VSM with Tf-Idf weights in plagiarism detection is that: each term will have almost a unique weight which is represented in the corpus (depending on the corpus size, as we discussed before). The weights of sentence's terms will represent a vector in a vector space model as positional coordinates. If there are two vectors that represent two different sentences, the similarity between them is achieved by measuring the cosine of the angle θ between them as shown in Figure 4.2.

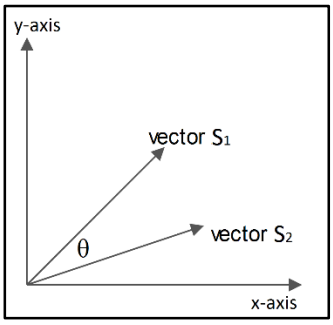


Figure 4.2: vectors representation in VSM

Where the cosine similarity equation is [79]:

$$\textbf{\textit{Cosine Similarity}}\ (\boldsymbol{S_1}, \boldsymbol{S_2}) = \frac{\boldsymbol{S_1} \cdot \boldsymbol{S_2}}{\| \boldsymbol{S_1} \| \ \| \boldsymbol{S_2} \|} \qquad (2)$$

The numerator in the equation is the dot product of two vectors S_1 and S_2 = S_1W_1*S_2W_1 + S_1W_2*S_2W_2 etc., where the denominator is the product of Euclidean length of the two vectors, where the Euclidean length of S_1 =$\sqrt{(W1)^2 + (W2)^2 + (W3)^2 + (W4)^2}$ [80] as example.

Let us assume that we have two sentences of 5-words length from two different documents, to build their vectors the following process is applied:

A	Sentence 1:		معا	القاعة	غادرنا	الانتهاء	قبل
B	Sentence 2:		الانتهاء	قبل	لوحدنا	القاعة	غادرنا

C	All Terms:	لوحدنا	غادرنا	الانتهاء	قبل	القاعة	معا
D	Tf-Idf weights:	0.552487	1.545784	0.874532	0.321548	0.654844	0.235477

E	S_1 vector:	0	1.545784	0.874532	0.321548	0.654844	0.235477
F	S_2 vector:	0.552487	1.545784	0.874532	0.321548	0.654844	0

In step A & B we extract the needed sentence components from the sentences, then in step C we added all the terms from the two sentences into one set to produce their tf-idf weights as in step D. Then in step E & F, for each term in a sentence, we replace it with its correspond tf-idf weight in which a vector of tf-idf values is formed. For the terms that are not exist in a sentence, a tf-idf value of ZERO is replaced with. In this example we built one vector for each sentence to illustrate the process, while in our approach for each sentence two vectors are built (verbs and nouns vectors).

By applying the previous cosine similarity equation on these two Tf-Idf weights vectors, we will get similarity value: cosine similarity = 0.95. Since that term (معا) is presented in the Sentence 1 and not in Sentence 2, and vice versa for term (لوحدنا) in Sentence 2.

4.1.2 Knowledge-based semantic similarity

The other type of semantic similarity measures is the knowledge-based (KB) measure. It is a method that utilizes the Semantic Networks (SN) by deriving information to measure the semantic relatedness between words. This similarity method has provided more improvements towards similarity in meaning rather than statistical representation of terms. Knowledge - based measures provide better rational similarity and accuracy since they mimic the arbitration of the human mind in measuring similarity between terms. This type of measures requires rich semantic network to perform well, also SN have to be modelled in a specific manner in case of graph representation (ex. Edge length between nodes) to allow the full utilization by the KB measures. Unfortunately, Arabic SNs are rare, uncompleted and do not have good modelled graph. These limitations impede many Arabic targeted applications from contributing and filling the gap in many fields. Also the un-good graph modelling of Arabic SN prevents some KB measures from performing well in measuring similarity.

KB measures can be grouped mainly to three main classes, each acts in a different way. Path-based measure is a similarity method that measure the similarity between two concepts in a

function of length of the path that links the concepts in the taxonomy [81]. The similarity measure achieved by using shortest path model which takes the length between nodes into consideration [82], or by using also the depth of nodes in the SN as in Wu & Palmer's measure [83] and others measures. The second class of knowledge based measures is the information content-based (IC). In this method, it assumed that each concept includes enough information in a SN. So the similarity between two concepts is based on the information content for each concept, where the more common information two concepts share, the more similar the concepts are [84]. In IC, utilizing the depth of least-common-subsumer (LCS) for two concepts is a common way [15]. For two concepts A & B, LCS is the most specific concept which is an ancestor of both A and B. Different measures have been proposed in the literature including Resnik's [85] and Lin's [86] measures.

Another KB measuring class is the feature-based method. It is based on an assumption that each concept is described by a set of words. So, the more common features two concepts have, the more similar the concepts are [15]. Tversky's measuring model [87] is the first feature-based measure in which the similarity between two concepts is increasing with more shared terms (commonality) and decreasing with the difference between them. In our approach, we adopted this concept relatedness measure due to its measuring simplicity, also due to some limitations in applying previous two measures (Path based and IC) on our used knowledge resource according to our experiments using AWN Java API [88]. Regardless of the measuring approach of previous measuring classes, Path - based and IC classes lead to inefficient similarity measuring by applying them on current available API for AWN. This inference is produced after experimental testing. For example, the noun "جندي" has a set of synsets ("محارب", "مقاتل" and "عسكري"), the similarity measuring using Wu & Palmer measure which is one of path – based measures that depends on graph representation as IC measures, produces 14% similarity score with term "محارب", and 13% for "عسكري" term, while nothing retrieved

for term "مقاتل". The similarity scores have to be larger than these scores since they are synsets to the "جندي" term. This point to some weakness in this API which is the only available AWN Java API.

4.1.2.1 Arabic WordNet (AWN)

AWN [89] as knowledge resource offers a good coverage of general Arabic words or terms. It has been adopted as a resource for different applications in different domains. The success of the Princeton WordNet [90] for the English language has motivated a similar project for the Arabic language. AWN's design contributes many advantages for its use in the context of Arabic NLP. This lexicon ontology contains 23,000 different Arabic words and 10,000 of sets of cognitive synonyms that are known as synsets (a group of words that have the same meaning). Regardless of these numbers, AWN is still small in comparison with the English WordNet. This weakness will affect our approach's results since many terms may not be found in. We chose AWN as a knowledge resource in our approach since it's the only available Arabic ontology to the best of our knowledge. AWN is used to retrieve synonyms of sentence components to enhance the detection of synonyms substitution plagiarized cases.

4.2 Preliminary Study

Before we start with the implementation of the similarity approach. We have conducted a preliminary study on university students to study the plagiarism pattern of students on an Arabic text. These kinds of studies can lead us to more effective similarity approaches by deduction of what kinds of changes students are used in plagiarism process. According to section 1.1, there are three main types of plagiarism patterns which are direct copy, sentence reordering and paraphrasing. The first type is very simple and primitive detection methods can cover. While the others are more sophisticated and need further efforts to reveal cases specially paraphrasing, since it contains synonyms extension property. So, this study is conducted to expose what kinds of changes are applied on Arabic text to obfuscate plagiarized cases.

In the preparation process, we looked for a suitable course to conduct our study on. Based on the available choices, we chose “Fundamentals of Research Method” course in the Arab American University in Jenin (AAUJ). Our choice was due to two factors:

1) The diversity of specialties is high since this course is related to university requirements courses categories, therefore it contains students from different faculties in the university.
2) The nature of this course is diversified which contains students from different studying levels (1-5 years).

Our study was in the form of questionnaire which is a paper-and-pencil instrument that the respondents requested to complete. To cover most of plagiarism behaviors, selection of suitable text is a mandatory task. The needed text has to be rich morphologically in parts of speech components to cover synonym modification cases, also to allow all choices of modifications on these components. After an extended search on Arabic articles in internet websites, we chose a suitable text from Webteb health journal [91] as illustrated in Figure 4.3.

Figure 4.3: Parts-of-Speech units of the used text using Farasa tool

To achieve accurate results in the study, the nature of this study was hidden from the students and we have encouraged them to do well by telling them that it is a competitive task to paraphrase a text with maintaining the original meaning of the sentence.

Based on components synonym replacement results, we start to develop our approach by taking into consideration the priority of components using number of changing cases for each. In the ranked query component, we used verbs, nouns and adjectives to retrieve the most relevant sentence, while in similarity component where comparisons are taking place, we used just verbs and nouns without adjectives since it has the lowest number of replacement cases (low priority) comparing to other components, also to reduce the computation cost in the comparison process.

4.3 Ranked Sentence retrieval and sentence-based Inverted Index

As we discussed in section 1.2, HYPLAG is based on search engine structure in which a query is passed to search engine. In information retrieval systems, there are two types of queries, Boolean queries and ranked queries [92]. In Boolean queries a set of operators such as AND, OR, NOT is used to link between query terms. While in the ranked queries, the target content (documents or sentences) will be scored according to a similarity function and the top N files will be retrieved. In our HYPLAG, the ranked query method is used where the input query is a set of input terms that are the used parts of speech tags of an input sentence. In this phase, the most relevant sentence to our input sentence's components is retrieved. The relevant sentence retrieval process is figured in Figure 4.4.

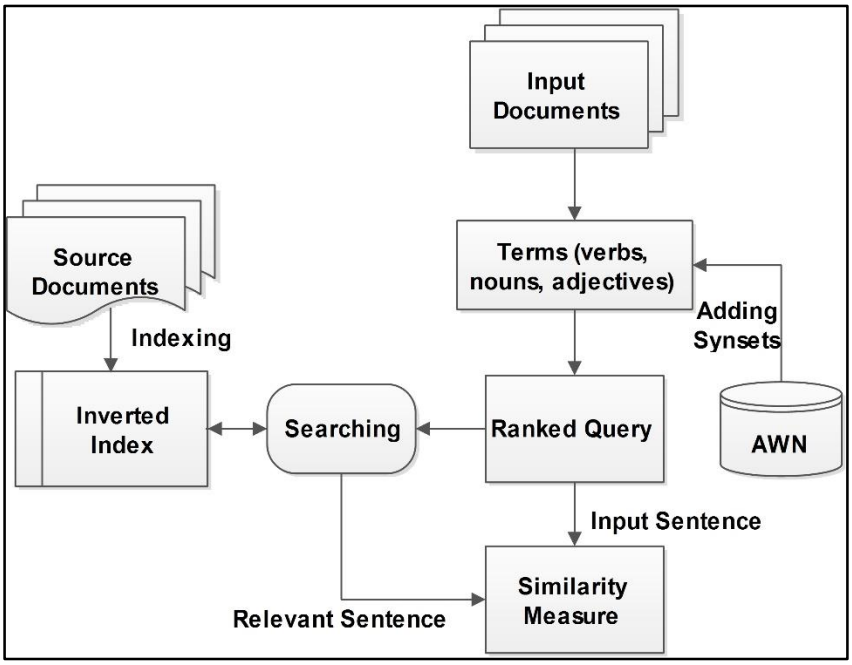


Figure 4.4: Relevant Sentence retrieval process

To ensure efficient sentence retrieval method, a sentence-based inverted index is constructed using source documents. The inverted index is one of several access methods developed to support the retrieving method in information retrieval domain. It is an index data structure that stores a mapping from content to its locations in a set of documents [93, 94]. The inverted index allows a fast retrieving process in any information retrieval environment. Typically, the inverted index structure that used in search engines are document-based where an indexed term references to a source document, since that the aim of searching scenario is to retrieve documents that contains a query (a set of terms). In the implementation stage, we indexed the terms by mapping them to their source sentences because the retrieving process are targets sentences (plagiarism case) and not a whole document. The Tf value of terms also is indexed as it's a static value according to document content to speed up the processing time as mentioned in Figure 4.5. Therefore, the input document is split to a length of n-word sentences to be indexed.

Term		Sentence_Id	Doc_No	Tf_Value
قال	→	1211	301	12
علم	→	1211	301	3
خطر	→	33	302	2
…	→	…	…	…

Figure 4.5: HYPLAG's inverted index structure

The retrieving step starts with passing the terms, for each input sentence, the set of verbs, nouns and adjectives are extracted. For each component group, the synsets are extracted from AWN to cover all synonyms replacement cases in the corpus as illustrated in Figure 4.6. Then all terms with their synsets are used to rank the sentences in the sentence ranking component. For each term, the sentences that contain these terms will have a higher rank.

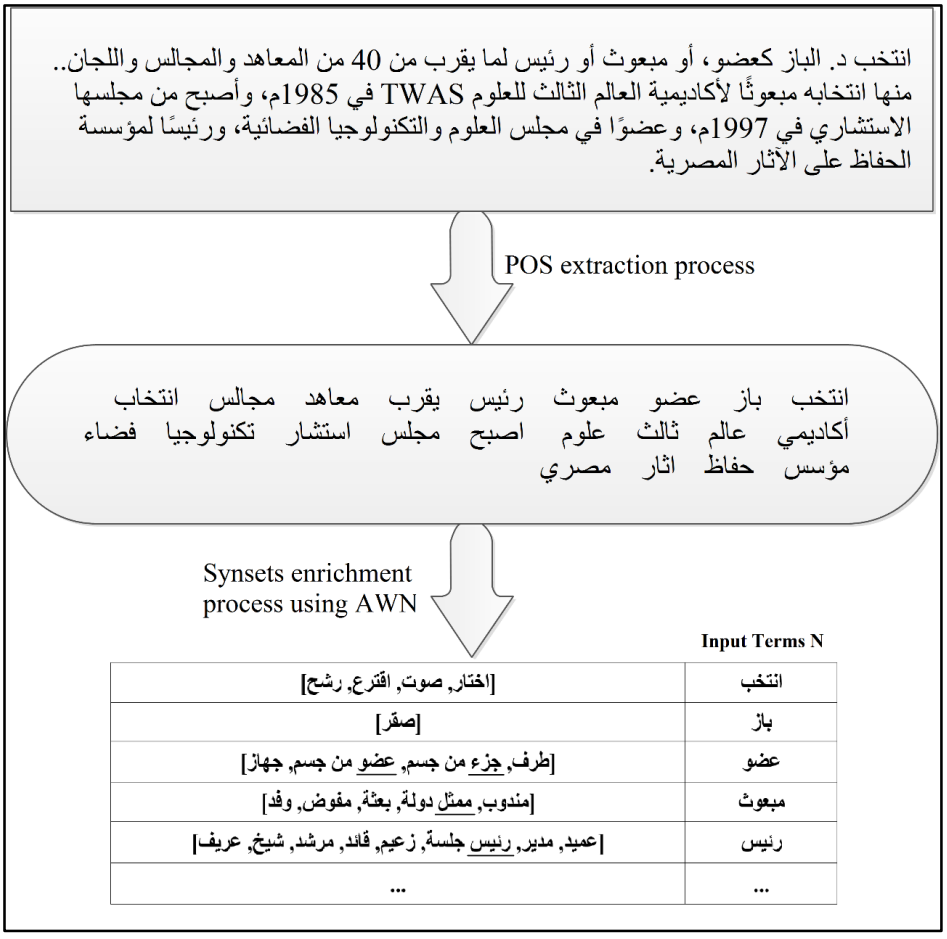


Figure 4.6: Example of input terms preparation process for ranking component using Farasa tagger

Term by term, the sentence that has the largest number of inputs terms is the sentence that most relevant to our input terms, and will have the highest relevance rank. To illustrate this process in an example, for a sentence:"غادرنا القاعة عند الانتهاء", the input terms are "غادرنا" ، "القاعه" ، "الانتهاء" without using their synonyms to clarify the example. If we have a set of sentence as in Table 4.1 in the inverted index, the most relevant sentence is the first sentence since it contains the largest number of input terms. In our example it contains all the input terms, so the rank will be 3 which is the number of existed terms.

Table 4.1: Example of sentences for the query ranking method

Rank	sentences
3	غادرنا القاعه لوحدنا قبل الانتهاء
2	غادرنا القاعه لوحدنا قبل البداية
1	ذهبنا الى القاعه التي تحتوي على المدرجات
0	ذهبنا الى المدرسه الثانوية في جنين

The selection process of relevant sentences is based on a threshold, since when an input sentence is clear and not plagiarized the most relevant target sentence will have a very low rank value. In our implementation, to accept the highest ranked sentence, it should permit the following Equation (3) in which the retrieved sentence should contains at least x value.

$$X_{number\ of\ terms} = ((Number\ of\ input\ terms\ without\ their\ synsets\ /\ 2) - Param1) \quad (3)$$

We excluded the number of all input terms with their synonyms because for each input term a number of synsets is presented for it in AWN, and synsets size is differ for each term. Also in synonyms substitution process that plagiarist uses, each terms is replaced with only one synonym term and not more, so the number of input terms without their synsets is more accurate to the selection process.

4.4 Similarity detection

The similarity detection phase consists of two methods combined together. These methods have been briefly described in section 4.1. After retrieving the most relevant sentence by sentence ranking component, similarity detection phase starts. This phase receives two sets of terms for

the input sentence, the first for verbs and the second for nouns. We excluded adjectives in this phase to reduce computation cost; we chose to exclude adjectives since they have been changes in small number of cases in our preliminary study in section 4.2.

Having two sets (verbs & nouns) for the input sentence with another two others extracted from the retrieved relevant sentence from previous phase to compare them in the vector space model. Using Td-Idf terms weights, we measured the similarity between the two verbs vectors using cosine similarity algorithm. The same process is applied for nouns vectors and one score value is resulted using weighted average equation [95]:

$$\boldsymbol{Score} = \frac{(\#\ \boldsymbol{of\ verbs}\ \times \boldsymbol{verbs\ similarity\ score}) + (\#\ \boldsymbol{of\ nouns}\ \times \boldsymbol{nouns\ similarity\ score})}{\#\ \boldsymbol{of\ verbs} + \ \#\ \boldsymbol{of\ nouns}} \quad (4)$$

Figure 4.7 illustrates the similarity detection process in an example.

Sentence A:	اعتماد النظام الغذائي الغني بالأطعمة النباتية مهم جداً للحفاظ على صحة الكلى، اعتمد في نظامك اليومي على الخضار والفواكه التي تعزز من وظائف الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، التفاح، العنب. كما يقول المختصون أن تناول نظام غذائي قليل بالسكر والجلوتين والصويا يساهم في التقليل من خطر الإصابة بالالتهابات مما يعزز من وظيفة الكلى وعملها.
Verbs vector A	اعتمد تعزز يقول تناول يساهم يعزز
Nouns vector A	اعتماد نظام اطعم نبات حفاظ صح كل خضار فواك ظائف مثل توت فلفل رومي تفاح عنب سكر جلو صويا تقليل خطر اصاب تهاب ظيف عمل

Sentence B:	اعتمد في نظامك اليومي على الخضار والفواكه التي تعزز من عمل الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، التفاح، العنب. اعتماد النظام الغذائي المليء بالأطعمة النباتية مهم جداً لحماية على صحة الكلى. أن استخدام نظام غذائي قليل بالسكر والجلوتين والصويا يساعد في التخفيف من خطر الإصابة بالالتهابات كما يقول المختصون ، مما يعزز من وظيفة الكلى وعملها.
Verbs vector B	اعتمد تعزز يساعد يقول يعزز
Nouns vector B	نظام خضار فواك عمل كل مثل توت فلفل رومي تفاح عنب اعتماد اطعم نبات جد حما صح استخدام سكر جلو صويا تخفيف خطر اصاب تهاب ظيف

Cosine similarity using Tf-Idf	Verbs vectors	0.73
	Nouns vectors	0.79
weighted average score	0.78	

Figure 4.7: Example on vectors similarity measure using verbs & nouns

The terms represented in the VSM using Tf-Idf weighting scheme in which term frequency (tf) and document frequency (df) factors are extracted using the inverted index.

After generating a similarity score in the VSM, we will have three cases in which the similarity process between the input sentence and the retrieved one will take place. The three cases are:

1) **If (final_score < min_threshould)** then discard the detection process for the input sentence.
2) **If (final_score > max_threshould)** then a plagiarized sentence is detected.
3) **If (min_threshould < final_score < max_threshould)** then verbs & nouns vectors are passed to Feature-based semantic similarity measure for more investigation.

TF-IDF technique can detects both basic methods of plagiarism (copy & paste and sentence reordering), while it can't cover synonyms replacement method. So, we have combined it with feature-based semantic measure.

In the next step when the case 3 takes place, both couples of verbs & nouns vectors are passed to feature-based measure. In this metric we adopt Tversky's model as we reference in section 4.1.2. The basic forms of Tversky's equation is suitable when the processed terms having the same number of synonyms, but in fact this is not real since some terms have small number of synsets while others have large number. So the problem with basic model is that the more unique features a concept presents the lower similarity, therefore we adopt a ratio model of Tversky's formula [96]:

$$sim_{tvr-ratio}(c_1, c_2) = \frac{F(\psi(c_1) \cap \psi(c_2))}{\beta F(\psi(c_1) \setminus \psi(c_2)) + \gamma F(\psi(c_2) \setminus \psi(c_1)) + F(\psi(c_1) \cap \psi(c_2))} \quad (5)$$

Where:

β and γ are values between 0-1 and have set experimentally.

F: is some function that reflects the salience of a set of features.

$\psi(c_1)$: represents the set of c_1 concept's features.

To measure the similarity between two concepts (C1 & C2) using the above formula, we set in the numerator the number of intersected $(\psi(c_1) \cap \psi(c_2))$ features (synsets) between these two concepts, while in the denominator the summation of the number of unique features of concept C1 $(\psi(c_1) \setminus \psi(c_2))$, the number of unique features of concept C2 $(\psi(c_2) \setminus \psi(c_1))$ and the number of intersected features between the two concepts.

After receiving the two vectors couples for both sentences. For each vector we built a matrix to represent it with its synonyms as in Figure 4.8.

Noun v^n for S^n	$Noun^1$	$Noun^2$	$Noun^3$	$Noun^4$	$Noun^n$

Mv^n

v^1	$Noun^1$	Syn^1	Syn^2	Syn^3	Syn^n
v^2	$Noun^2$	Syn^1	Syn^2	Syn^3	Syn^n
v^3	$Noun^3$	Syn^1	Syn^2	Syn^3	Syn^n
v^4	$Noun^4$	Syn^1	Syn^2	Syn^3	Syn^n
v^n	$Noun^n$	Syn^1	Syn^2	Syn^3	Syn^n

Figure 4.8: Matrix representation of a noun vector

Figure 4.8 shows an example of building a noun matrix from a noun vector, the first term in the vector is added to the first row of matrix with its synsets, and so on, until all nouns with their synsets are added. The same operation is applied for all four vectors. The matrices size depends on number of terms in vectors. The similarity measuring of same component matrices is started by reading the first row (vector) of the first matrix with the first row of the second one (matrices of the same component type) and passing them to Tversky's model as two sets. The matrices' comparison algorithm is shown in Figure 4.9. After the comparison process for both couples of matrices are completed and two scores are generated, weighted average of results is computed using Equation 4.

```
Function Matrices_Comparison(Matrix1 , Matrix2)
Matrix tmpMatrix
if Matrix2.size() smaller than Matrix1.size()
    then tempMatrix ← Matrix2;
         Matrix2 ← Matrix1;
         Matrix1 ← tempMatrix;
end if

for every row in Matrix1 do
    initialize Max = 0;
    for every row in Matrix2 do
        vectorM1 ← readRow(Matrix1);
        vectorM2 ← readRow(Matrix2);
        result ← TverskyModel(vectorM1, vectorM2);
        if result larger than Max
            then Max ← result;
        end if
    end for

    if Max equal 0
        then finalResult ← finalResult + 0;
        else finalResult ← finalResult + (Max/Matrix2.NumOfRows());
    end if
end for
return finalResult;
```

Figure 4.9: Pseudocode of the Matrices' comparison process

Finally, after the comparison process is complete, a final score produced that denotes the similarity detection value for the couple of sentences (input & retrieved sentences) whether it forms a plagiarism case or not.

After processing the input document by investigating its sentences, the document originality decision is take on. According to [97], there are different cases that represent the relation between the two compared documents in a categorized form based on their similarity score. Unrelated, when the two documents are very different and no plagiarized cases is detected. Related, documents are somehow related by sharing small number of plagiarized cases. Partly overlapped, where some sections form a full plagiarized cases. Subset, when the processed document is part of the other one. Copy, both of documents are the same. Using these categories, we will tag the suspicious input documents in the final generated report.

4.5 Optimization processes

During the similarity measure process in VSM, another extra process is used to ensure best detection value. Both input and retrieved sentences are adjusted in an adjusting model by K-overlapped terms from pervious and next sentences. Since that a plagiarism case may be divided into two sentences during sentences segmentation step, so detection results will be

reduced. An Example illustrates the issue in Figure 4.10. Therefore, we adjusted the input sentence with the previous sentence by adding K terms from it, and removing K terms from the end of the processed sentence to maintain the sentence length. The same operation is applied with the next sentence by adding K terms to the end of processed sentence and we remove K terms from the beginning. This process of adjustment is applied on the input and retrieved sentences. In this method, we ensure that plagiarized cases are detected in more accurate way. To confirm our assumption, we conduct a run on the dataset. The results are mentioned in the Experimental Results section of the5.3 next chapter.

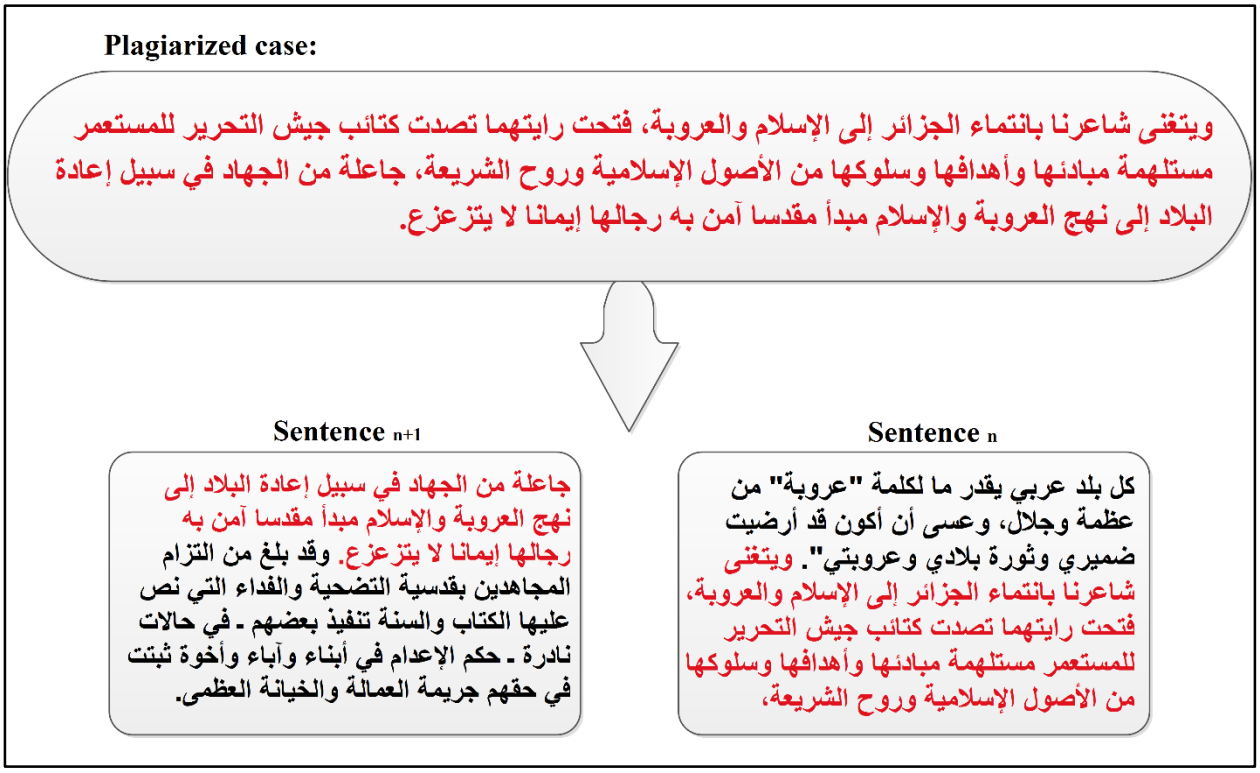


Figure 4.10: Example on segmentation of a plagiarized case

In the last step of HYPLAG, sentence aggregation model takes place. Sentences that are detected as plagiarized sentences are aggregated by checking their places, if sentence A and B are detected as plagiarized case in the same document and they are sequential then both are aggregated in one large sentence case, taking into account the situation of sentences in the other

side of plagiarized case (if they are plagiarized and sequential also, then they are also aggregated).

```
detected_Cases dc[];
order dc.inputSentences() by Id;
for every row in dc do
  if dc.current().sourceFile  is equal  dc.next().inputSentences().sourceFile
     then if dccurrent().inputSentences().Id  is equal  dc.next().inputSentences().Id
        then if AbsoluteValue(dc.current().retrievedSentences().Id - dc.next().retrievedSentences().Id) is equal 1
               then dc.current().aggregateWith(dc.next());
             end if
     else if AbsoluteValue(dc.current().inputSentences().Id - dc.next().inputSentences().Id) is equal 1
        then if dc.current().retrievedSentences().Id  is equal  dc.next().retrievedSentences().Id
               then dc.current().aggregateWith(dc.next());
             else if AbsoluteValue(dc.current().retrievedSentences().Id - dc.next().retrievedSentences().Id) is equal 1
               then dc.current().aggregateWith(dc.next());
             end if
     end if
  end for
```

Figure 4.11 Pseudocode of the sentences aggregation process

This process is applied on both input and retrieved sentences. The algorithm for the process is demonstrated in Figure 4.11. Finally, a detailed report is shown for the plagiarized cases with their original source with classifying the input document in one of mentioned categories in previous section.

4.6 Evaluation measures

To evaluate the system performance, measures have to be capable of evaluating a system based on its nature. Since that our approach is based on search engine structure, information retrieval (IR) metrics are adopted in the measuring process. Precision and recall are the most common measure in IR domain.

Precision measures the fraction of how much relevant documents are retrieved from a set, where recall is how many of relevant documents are retrieved [98].

$$\boldsymbol{Precision} = \frac{\#(\boldsymbol{relevant\ items\ retrieved})}{\#(\boldsymbol{retrieved\ items})} = \boldsymbol{P}(\boldsymbol{relevant}|\boldsymbol{retrieved}) \qquad (6)$$

$$\boldsymbol{Recall} = \frac{\#(\boldsymbol{relevant\ items\ retrieved})}{\#(\boldsymbol{relevant\ items})} = \boldsymbol{P}(\boldsymbol{retrieved}|\boldsymbol{relevant}) \qquad (7)$$

These measures are utilized in plagiarism detection domain where relevant items are the cases that are in fact plagiarized, and retrieved items are the cases that are detected by the system whether they are plagiarized or not. These measures can be described in another way by four

main terms. The terms are decided whether the cases are detected correctly or not. Figure 4.12 shows the distribution of these terms in between the factors “Plagiarized cases” and “Detected cases”.

		Detected cases	
		Yes	No
Plagiarized cases	Yes	TP	FN
	No	FP	TN

Figure 4.12: Plagiarism confusion matrix

- True positive (TP): indicates that the detected cases by the system are real plagiarized cases.
- True Negative (TN): indicates that the clear original cases in the set are not detected by the system.
- False positive (FP): indicates that a set of cases detected by the system and they are not in fact plagiarized (wrongly detected).
- False Negative (FN): indicates that a set of plagiarized cases in a set were not detected by the system.

Another adopted measure is the F-score measure [99]. F measure is an accuracy measure that is defined as a weighted harmonic mean of the precision and recall [100].

$$F - score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \quad (8)$$

These measures will be used to measure HYPLAG performance in the following chapter.

4.7 Summary

In this chapter, a detailed description of HYPLAG was presented. As a search engine based approach, we described the main steps in sentence retrieving process and how we utilize verbs, nouns and adjective as sentence components with their synsets in the retrieving process. Also,

the sentence-based indexer was presented with describing its advantages over document based in plagiarism detection process. We described the VSM in details with Tf-Idf weighting scheme, and how they have been used in plagiarism detection domain. Also, we introduced the adopted feature-based similarity metric with showing its advantages and reasons of adopting over other metrics with Arabic knowledge resource. The combined approached was described in details with giving examples on similarity measurement processes and how they have been combined. Finally, the optimization process was described in order to increase the accuracy of the detection process.

5. Evaluation and Experimental Results

After describing the core phase in our approach, the aim of this chapter is to describe the evaluation process of the approach as well as showing the experimental results. In the first section, the implementation process is described in details. Then in section 5.2, we will describe the used dataset in the case of its statistical features. In section 5.3, the evaluation process is described with proof numbers and a discussion on some factors affect the system performance. In the last section, the approach is compared to other systems that have been tested on the same dataset.

5.1 Implementation

In this section, we will reflect the theoretical ideas of our approach to practical implementation. At the beginning, selecting the most suitable programming language will give us more positive points in implementation according the availability of APIs in that language and availability of Arabic NLP libraries. According to a preparing step before implementation, we found that most of needed resources are available in the Java programming language. So, we use Netbeans IDE 8.2 as a famous environment for java language.

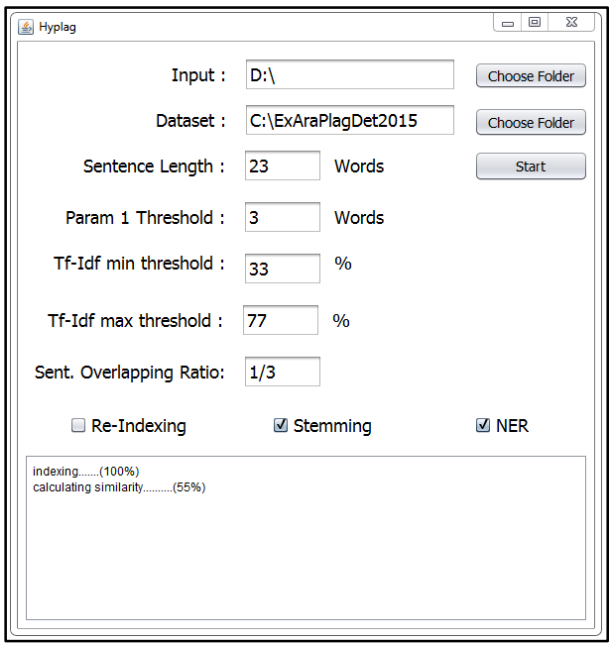


Figure 5.1: HYPLAG interface prototype

HYPLAG prototype is simple and easy to understand to the user as shown in Figure 5.1. Simply, the user is asked to choose a folder or a specific file as an input. The system accepts most of document files extensions (pdf, doc, txt, ppt, etc.). We use Apache TIKA for reading documents as it's a toolkit that detects and extracts text over thousand different file types [101]. A set of parameters are showed for tuning process during the performance measurement in Figure 5.1. After selecting the input files, the source documents or the dataset has to be chosen for indexing purpose. In the indexing process, the documents are normalized to remove the unneeded characters, numbers or other language terms. Then, documents are split into 23 words length sentences. We start with the normalization process before sentence segmentation step to ensure that the unneeded terms or character will not confuse segmentation process by inserting removed terms and characters in the word counting process. For indexing purpose, we use Apache Lucene which is a full-featured text search engine library written in Java [46]. In HYPLAG, it has been used only for indexing purpose since it affords high performance indexing solution, while sentence retrieving process was built manually. In the system prototype, the user is able to re-index the corpus.

A report of the results is generated as shown in Figure 5.2, in which the documents are categorized according to categories that in section 4.4 with an overall detection result.

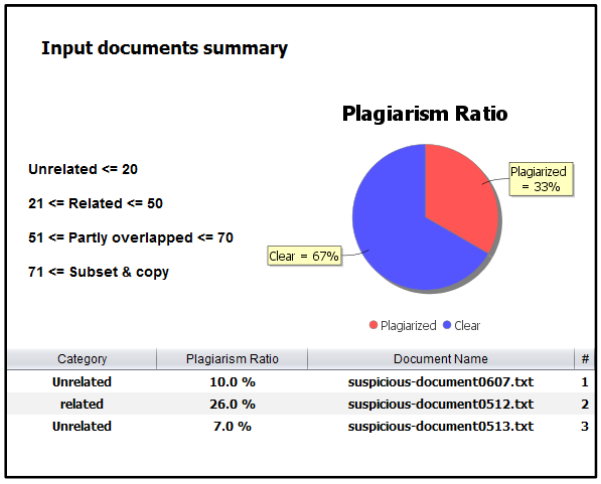


Figure 5.2: A generated report shows input documents plagiarism detection results summary

5.2 Dataset

Measuring the performance of any detector needs a reliable dataset. Most of the researchers who developed previous cited Arabic plagiarism detectors in the literature built their own datasets to measure the achievements of their own approaches. In this way we can't move towards improving detectors since each approach is tested on a different dataset. From this point of view, we stopped building a plagiarism dataset and we started looking for a shared one. To the best of our knowledge, ExAraPlagDet 2015 [34] is the only extrinsic plagiarism dataset that is available and accessible on the internet. This dataset was constructed for Arabic Plagiarism Detection competition 2015 [34].

ExAraPlagDet competition has provided two datasets, one for Training and another for the final testing. Both corpuses contain the same statistical numbers. The sources of corpuses text are the Arabic Wikipedia [64] and the Corpus of Contemporary Arabic (CCA) [63]. These sources were used due to the variety of topics and free availability. There are two types of plagiarism cases in datasets, artificial (created automatically) and simulated (created manually). For the first type, authors used phrase shuffling and word shuffling strategies. On the other hand, where manual cases were created, synonym substitution and paraphrasing are conducted. The Testing corpus consists of 1171 documents that contain 1727 plagiarism cases. Documents lengths differ from one page up to 100 pages. Also, plagiarism cases are varying in length in which it reaches 30k characters. A large statistical detail of the corpuses is shown in [102]. In the performance measurement, we will test our approach on both datasets to have enough real results about our approach performance, regardless they are similar. Figure 5.3 shows an example of plagiarism cases in a suspicious document.

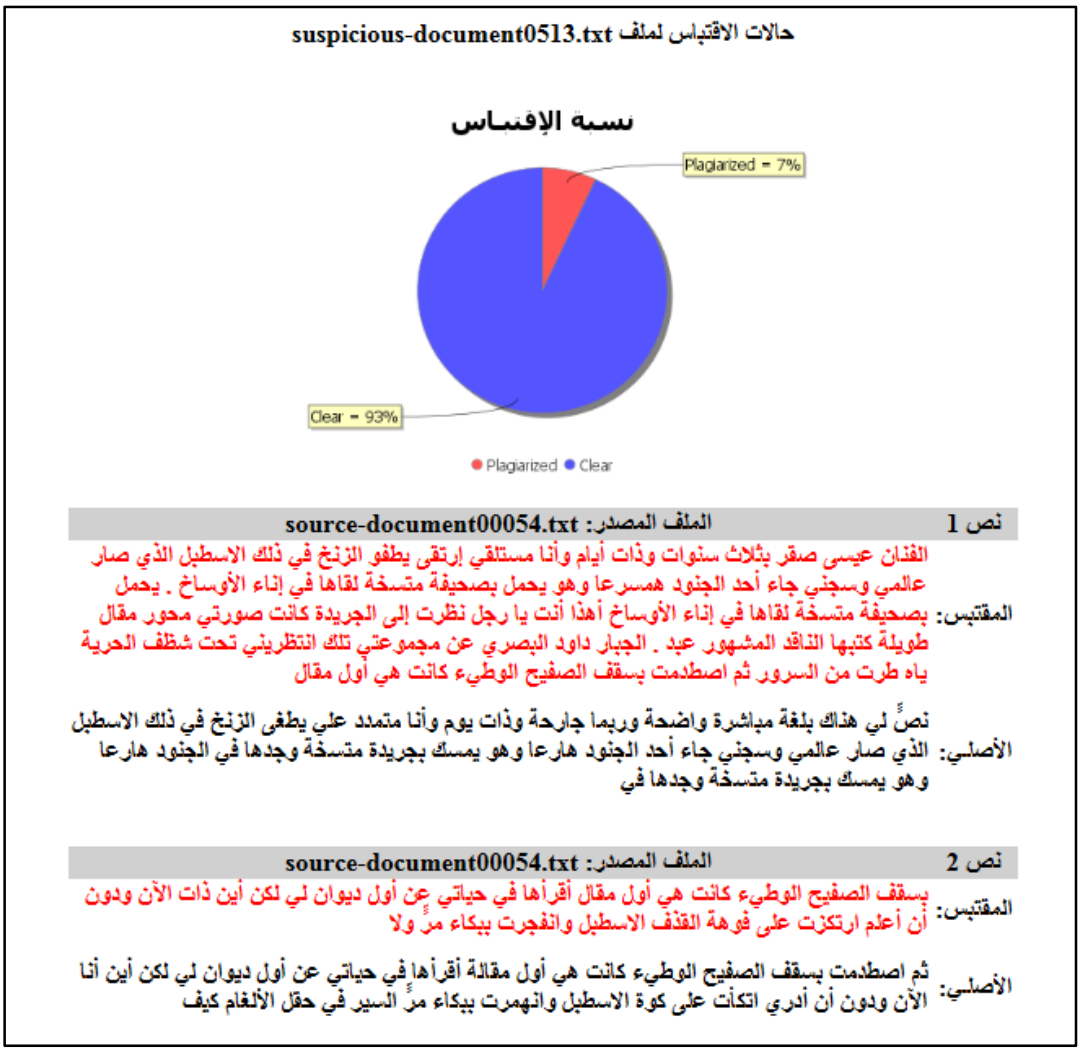


Figure 5.3: An example of a document's plagiarized cases in a report

5.3 Experimental Results

To confirm our approach performance, presenting experimental measuring is a primary step. In this section we will measure the accuracy of our approach using the datasets that were listed in section 5.2. The attained results will be provided for each dataset in terms of performance and execution time.

5.3.1 Parameters determination

To achieve high accuracy results in HYPLAG with low execution time we have to select suitable values for parameters. Therefore, in the similarity measuring phase, several runs were conducted for parameters tuning purposes to determine the most suitable values which will result the highest detection accuracy. The editing of these parameters values will change the

accuracy and the processing time of the system. After tuning process was completed, a set of parameters were used with the following values:

- Retrieval parameter (Param1) = 3 words.
- Sentence length = 23 words.
- Minimum Tf-Idf value = 0.33.
- Maximum Tf-Idf value = 0.77.
- Sentence overlapping ratio = 1/3.

In the Table 5.1 we summarized how the HYPLAG performance changes when we changed our system parameters with larger or smaller values. These results have been generated by applying several runs over one of documents' categories (simulated obfuscation category) in the test dataset.

Table 5.1: The effect of tuning process on HYPLAG performance

Parameters	Larger value cause	Smaller value cause
Retrieval parameter (Param1)	Positive Impact on F-score	Negative Impact on F-score
Sentence length	Negative Impact on F-score	Negative Impact on F-score
Minimum Tf-Idf	Negative Impact on F-score	Positive Impact on F-score
Maximum Tf-Idf	Positive Impact on F-score	Negative Impact on F-score
Sentence overlapping ratio	Negative Impact on F-score	Negative Impact on F-score

The changing process in values of some of these parameters will increase the recall value in which improve the HYPLAG accuracy. But after a specific threshold, the system starts take more execution time to accomplish the task. In the next section, we will discuss how these parameters effect the execution time. So, we settled for the above parameters' values where the system has achieve the best accuracy with lowest execution time.

5.3.2 Processing time

Processing time is an important factor in the detection process. Long processing time applications leads to disturbance of users. In HYPLAG, we quit some steps such as adding

adjectives in similarity detection process to reduce processing time. These steps will enhance the approach accuracy within a ratio, but our approach generated satisfying results without them. Also some of our used parameters in the approach are time sensitive parameters in which changing their values lead to changing in processing time. In the following, an explanation for these parameters is mentioned:

- Retrieval parameter (Param1): Larger values lead to processing more sentences that have low ranks values since that they will be accepted by Equation 3. So larger values will involve more irrelevant sentences to the similarity detection task.
- Sentence Length: Low length of sentences makes HYPLAG consumes more time to accomplish a detection task. The low length in sentence segmentation task will produces more number of sentences than large length. As a result, we will have more processes in all HYPLAG phases since that number of processed sentences will increase.
- Minimum Tf-Idf threshold: Lower values will passes more irrelevant sentences from Tf-Idf phase to feature-based measure rather than reject them, even when verbs and nouns vectors have very low similarity ratio, so more computational cost is needed and more time is consumed.
- Maximum Tf-Idf threshold: Large values will apply more similarity processes using feature-based measure even when Tf-Idf verbs and nouns vectors are very similar.

Based on tuning process, we set a values to the mentioned parameters in the previous section (see section 5.3.1) as the most eligible values in case of processing time and detection accuracy. In the last section of this chapter, we will show the measured processing time of our approach.

5.3.3 Preliminary Study's results

In this section we will mention the results of the preliminary study that discussed in section 4.2. The sample size was 59 students as shown in Table 5.2, 4 students have not changed anything (not participated) due to unclear reasons. For the rest 55 students, we have detected many synonyms replacements cases distributed as follows: 34 verbs replacement cases, 15 nouns

replacement cases and 5 cases of adjectives replacement. Also, we noticed 29 students have used sentence reordering method.

Table 5.2: Preliminary study results summary

Sample of 59 students				
55 students				**4 students**
Synonym replacement			Sentence reordering	No changes
verbs	nouns	adjectives	29	
34	15	5		

5.3.4 Similarity using sentence components

The core step of HYPLAG is to extract the sentence components. As a kind of an abstraction, the used sentence components (verbs and noun) contain an overall meaning of the sentence (informative components), in which the similarity process depends. Also, the adoption of adjectives increase the similarity results in a small ratio but it increases the execution time of the approach, so from this fact with the results of our preliminary study results in section 5.3.3 we excluded them from the similarity phase. The using of these sentence features grants more advantages in term of indexing size. They reduce the index size to the 18% of its size with the overall sentences parts of speech.

We started the measuring process using Training dataset. The Training dataset consists of 33980 sentences using sentence length of 23 words. During testing, many sentences were detected as plagiarized cases but they weren't found in the ground truth files as the mentioned example in Figure 5.4, these cases were decreased our approach performance results in which they were considered as false positive cases.

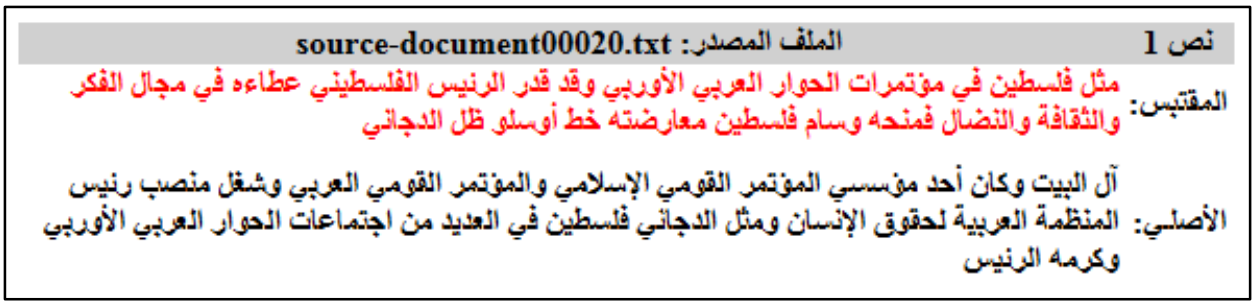
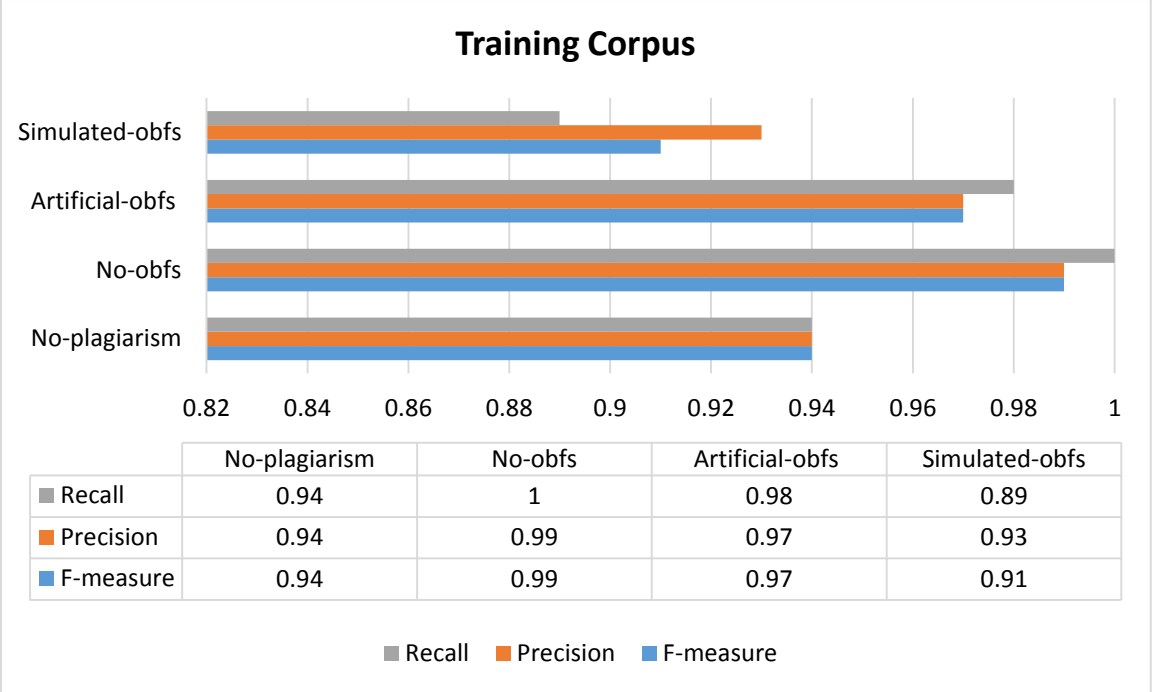


Figure 5.4: An example of a false positive (FP) case from the Training corpus

For the Training corpus, our approach has achieved 96% of precision and 96% for recall. The Training corpus consists of four main categories of documents: no-plagiarism, no-obfuscation (copy & paste), artificial-obfuscation and simulated-obfuscation. We have measured our approach on each category independently and it achieved the results as shown in Figure 5.5. To measure the effectiveness of coupling feature-based semantic measure with TF-IDF technique, we have disabled feature-based model and re-measure the performance over the simulated-obfuscation type since it represents the most complex cases which have been built by using manual synonym substitution. As we mention in Figure 5.5, the precision and recall for simulated-obfuscation type is 93% and 89%, and after disabling feature-based model the precision value is 83% and the value of recall is 87%. Combining both techniques has improved the results clearly, and we can recognize the improvements by running the approach on more complicated synonym substitution cases.



	No-plagiarism	No-obfs	Artificial-obfs	Simulated-obfs
Recall	0.94	1	0.98	0.89
Precision	0.94	0.99	0.97	0.93
F-measure	0.94	0.99	0.97	0.91

Figure 5.5: Precision and recall results on Training corpus

For the second dataset (Testing corpus), our approach has achieved 92% as precision value and 87% value of recall as shown in Figure 5.6.

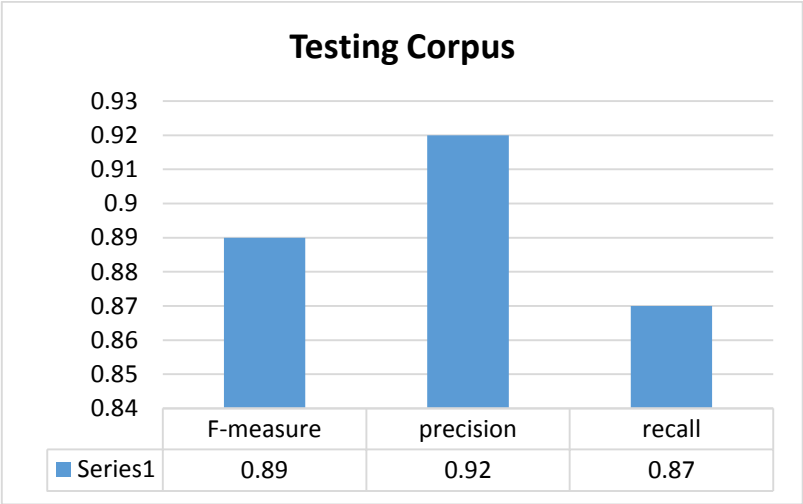


Figure 5.6: Precision and recall results on test corpus

Based on our experiments, HYPLAG can't detect sentences that have full substitution in all sentence components. Experimentally, HYPLAG will retrieve the plagiarized sentence from the index since it uses terms synsets using AWN. But in the first step of similarity detection phase where VSM is used with Tf-Idf weighting scheme, the approach couldn't detect this type

of sentences. Since synsets here are not used where nothing can relate terms to their synsets. So the similarity results will be less than Tf-Idf minimum threshold and the case will be discarded. These plagiarism cases where all components are substituted are very difficult for plagiarists to form since changing all components to their synonyms are very expensive task and not possible in most of cases. These sentences maybe contain named entities (persons, places, objects and organizations names) and can't be replaced.

5.3.5 The effect of discard Named Entities from stemming process

As we mentioned in section 3.4, we combined two methods to extract named entity from text in order to prevent them from being stemmed. We applied this technique to reduce error rate in the terms stemming process, since the process of stemming NE can lead to false positive cases. The measure of this technique effect is a hard process. It maybe couldn't be noticed if we conduct another run on the dataset without using NER because for a sentence the number of verbs and nouns are larger than NE, thus they control the detection ratio. So, the sentence length is a factor which hide the effect of the technique since that the larger number of other terms (verbs and nouns) will eliminate the effect of miss stemmed NE. Therefore, we apply the following process to conclude the effectiveness of NER process.

To measure the effect of this process we conduct an investigating process. Using an NE list (A) extracted using the two combined methods, we started stemming the names using our adopted stemmer. After that, we also produced another list (B) of stemmed terms from the dataset with exclude NE. Then we compared both lists to investigate how much stemmed named entities are lead to miss stemmed words from the corpus. The list (A) size is 5601 terms, and the used stemmer is Motaz [58]. The process reveals 95 miss stemmed unique terms, shows that the stemming process of NE can lead to miss stemmed cases. In Table 5.3, a sample of generated results by the process.

Table 5.3: Sample of generated miss stemmed NE words

NE	Stem	Meaning	Corpus term	Stem	Meaning
ويتمان	يتم	Person name	يتم	يتم	Complete
والأتراك	اتراك	Turks	أتراك	اتراك	Are you?
بالمنطقة	منطق	Region	منطق	منطق	Logic
واحات	اح	Oasis	أحيان	اح	sometime
المكي	مك	Grand Mosque	مكانهم	مك	Their place
القيامة	قيام	Resurrection	قيام	قيام	Do
وجدي	جد	Person name	جد	جد	Hard work
جدة	جد	City	جد	جد	Hard work

The effectiveness of stemmer in this investigating process plays a main role, since that less effective stemmers can stem NE to more miss stemmed cases.

5.3.6 Stemming

Stemming is a primary preprocessing process to create a uniform shape of terms to ease the comparison process. To measure the effective of stemming process, we have proposed a comparative stemming approached to compare the current available Arabic stemmers in section 3.5.1. In this section we will discuss the results of this comparative approach. Then next we will measure the effect of stemming process in the approach.

5.3.6.1 Comparative stemming approach results

After completing all preprocessing steps of the evaluation approach, as an example, we started the evaluation with Khoja stemmer. The number of words in the corpus is 31000. Khoja has started stemming process and generated 10353 stems of the corpus. The low number of stems that is generated is due to words duplications and many words have been stemmed to the same root or form, such as "مقاتلون، قتال، مقاتلات، المقاتلون، والمقاتلين" they will be "قتل" or "قتال" depending on the stemmer approach. We have proofed our point of view by printing the word and its stem(s) and ordering them alphabetically, then reviews samples of the results manually, in a sample, we found that 75 words have been stemmed to the same word, as shown in Figure 5.7.

1	أثر	16	بتأثر	31	تأثيرها	46	نؤثر	61	وتأثرهِ
2	أثرت	17	بتأثير	32	تؤثر	47	نتأثر	62	وتأثير
3	أثرهِ	18	تأثر	33	تاثيرا	48	وآثرنا	63	وتأثيرا
4	أثرها	19	تأثرا	34	تاثيراته	49	وأثر	64	وتأثيراته
5	أثرهم	20	تأثرت	35	تاثيرها	50	وأثرت	65	وتأثيرهِ
6	أثروا	21	تأثرهِ	36	تتأثر	51	وأثرهِ	66	وتأثيرها
7	إثرها	22	تأثرها	37	ستؤثر	52	وأثرها	67	وتأثيرهما
8	اثر	23	تأثرهما	38	ستؤثرهِ	53	واثرها	68	وتؤثر
9	استأثرت	24	تأثروا	39	سيؤثر	54	والأثر	69	ولاتتأثر
10	الأثر	25	تأثري	40	فتأثر	55	والاستئثار	70	ويؤثر
11	التأثر	26	تأثير	41	فتؤثر	56	والتأثر	71	يؤثر
12	التأثير	27	تأثيرا	42	لاستئثارهِ	57	والتأثير	72	يؤثران
13	التأثيرات	28	تأثيرات	43	مآثرنا	58	والتأثيرات	73	يؤثرون
14	المستأثرة	29	تأثيراتها	44	مأثرة	59	وتأثر	74	يتأثر
15	بأثر	30	تأثيرهِ	45	متأثرا	60	وتأثرت	75	يتأثرون

Figure 5.7: produced stems of word "أثـر"

After producing the 10353 words, stemmer's output was fed to the AWN in order to check their availability, AWN showed up 1313 of them and 9040 were not founded. In Table 5.4, the results of the involved stemmers. To validate this assumption, we have checked samples of the unavailable results and we found the following cases:

- CASE 1: There are some persons and places names such: "اسرائيلي، للولايات، تشرشل، بيروت" which are not available in the AWN.
- CASE 2: Many of incorrect stemmed words such: "كقسم، الآلة، لالشخص، وسن،هوجمنا".
- CASE 3: large number of words that have been stemmed in a wrong way, such as: " رين، تير، سوس، وني، خلج" and much more.
- CASE 4: Words are stemmed correctly and it wasn't being available in AWN like: " لجأ، غلب، حظي، بصق" but this case was few.

Table 5.4: Results of stemming comparison approach

Stemmer	Unique stems	AWN		Detection percentage to the corpus (%)
		Available	Unavailable	
Khoja	10353	1313	9040	4.24
ISRI	12361	1678	10683	5.44
Light10	22694	2957	19737	9.58
Motaz	25389	3187	22202	10.32
Tashaphyne	14720	2021	12699	6.55
Tashaphyne Root	07313	1020	6293	3.30

For the cases that generate words unavailable in AWN, we can conclude different reasons. For CASE 1, the appearance of such results is due to false positive results of Farasa NER tool,

which did not detect these words as entity names; these words were not numerous. CASES 1 & 2 represent a weakness in the stemmer, and this case is the target of our study. These cases existed in abundance. As we mentioned before, the richness of our knowledge resource will increase the accuracy of our study but will not negatively affect, therefore, CASE 4 appears.

About the processing time, the fastest stemmer was Motaz and the slowest was the ISRI as presented in Table 5.5, which is clearly due to the stemming approach that is adopted. Our workstation specifications include: (HP EliteBook 8560w Workstation, with processor Intel® Core™ i7-2860QM CPU @ 2.50GHZ, 16.0 GB of RAM).

Table 5.5: Processing time of stemmers

Stemmer	Processing time (sec.)
Khoja	75
ISRI	812.254
Light10	1.314
Motaz	1.002
Tashaphyne	742
Tashaphyne Root	749

Table 5.6: Overall ranking of stemmers

Stemmer	Final percentage calculation	Rank
Motaz	10.64	1
Light10	9.17	2
Khoja	1.66	3
Tashaphyne	1.11	4
ISRI	0.56	5
Tashaphyne Root	0.27	6

The stemmers' rank that is mentioned in Table 5.6 has been placed based on the following equation:

$$\boldsymbol{Rank} = (\boldsymbol{DP} \div \mathbf{100})^{\mathbf{2}} \times (\boldsymbol{T} - \boldsymbol{t}) \qquad \textbf{(9)}$$

Where DP is the detection percentage to the corpus size, T is the time trade space (we set it to 1000 since that the highest processing time is 912), t is the processing time.

From our evaluation and the results in Table 5.6, we can conclude that Motaz has the best performance and stemming efficiency due to time and valid results that achieved. Motaz stemmer has achieved 3187 valid results with 230 terms higher than Light10, and it completes the process with 0.3 second faster than Light10 so we have given rank 1 to Motaz stemmer. Therefore, Light10 comes in the second place with near number of valid results and processing time. Based on these results, we adopt Motaz stemmer in Hyplag.

5.3.6.2 The effect of stemming process

To measure the effective of stemming process, we performed a run of approach over one of documents' categories (simulated obfuscation category) in the Testing dataset. Also, we have conducted another run using one of the lower perform stemmers due to our comparative approach in section 5.3.6.1. As a knowledge, any insignificant changes in a plagiarized sentence lead to miss detection of plagiarized case without using stemmer. For example, the following two sentences clarify the situation:

1- تتكون مناطق الحكم الذاتي من مقاطعات بينما تتكون المقاطعات بدورها من البلديات.

2- ستتكون المناطق ذاتية الحكم من المقاطعات بينما ستكون البلدية بدورها المقاطعة.

Both sentences are identical in meaning and in terms that construct, but the similarity ratio is zero since all verbs and nouns are not identical in case of character sequence, so without removing the affixes and suffixes or change the terms to their roots, the comparison process will be worthless.

HYPLAG has achieved 89% for recall while 93% for precision in simulated obfuscation category documents, and after disabling stemming it achieves 85% for recall and 90% value of precision. The other run on dataset is by using other lower perform stemmer, we use Tashaphyne Root [60] stemmer since it's the lowest performed stemmer in Table 5.6. The

second run using Tashaphyne stemmer has achieved 88% value of recall and 91% value of precision.

We can deduce that the ratio of real paraphrasing in the dataset is small. Terms in plagiarized cases in dataset were paraphrased automatically and mostly nothing changed in their morphological forms (suffixes and prefixes). Therefore, the system's result is not highly affected by disabling the stemmer, and so it detects most of plagiarized cases. To ensure of this fact, we reviewed the statistical part of dataset in the source reference, and we found that the manual paraphrasing ratio of the Training corpus is 1.28% which confirms our runs result. Both runs clearly confirmed the importance of using stemming as preprocess in similarity detection.

5.3.7 Efficiency of adjusting model

In the previous chapter, we discussed the need of adjusting model in HYPLAG. To measure its effectiveness, we conduct a test run without using it in which the results will confirm it effect. The run is applied on "03-artificial-obfuscation" category. The performance of HYPLAG on this category with adjusting model is 98% for recall and 97% value of precision. After conducting the run, HYPLAG has achieved 95% for recall and 88% value of precision. Therefore, we confirm the importance of this model in HYPLAG system.

5.3.8 HYPLAG through example

In the following, we will show an Arabic sentence as an example of plagiarized text to clarify the detection performance of HYPLAG in revealing plagiarized cases. The text that we will use is the same text in our preliminary study in section 4.2. The text will have several changes applied to measure the detection ratio at each with an explanation for each case:

- The source sentence:

اعتماد النظام الغذائي الغني بالأطعمة النباتية مهم جداً للحفاظ على صحة الكلى، اعتمد في نظامك اليومي على الخضار والفواكه التي تعزز من وظائف الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، التفاح، العنب. كما يقول المختصون أن تناول نظام غذائي قليل

بالسكر والجلوتين والصويا يساهم في التقليل من خطر الإصابة بالالتهابات مما يعزز من وظيفة الكلى وعملها.

- Changing in sentence particles, Theoretical Plagiarism detection value = 100%, Real Plagiarism detection value = 100%.

اعتماد النظام الغذائي الغني **في** الأطعمة النباتية مهم جداً **في** الحفاظ على صحة الكلى، اعتمد **في** بنظامك اليومي على الخضار والفواكه **مما** تعزز من وظائف الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، التفاح، العنب. **كما** ويقول المختصون **بأن** تناول نظام غذائي قليل بالسكر والجلوتين والصويا يساهم **في** بالتقليل من خطر الإصابة بالالتهابات **مما وذلك** يعزز من وظيفة الكلى وعملها.

- Changing in some terms forms, Theoretical Plagiarism detection value = 100%, Real Plagiarism detection value = 90%. Reasoning: miss stemming of some terms.

- اعتماد **نظام غذائي غني** بالأطعمة النباتية مهم جداً للحفاظ على صحة الكلى، **إعتمد** في **النظام** اليومي على الخضار والفواكه التي **ستعزز** من وظائف الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، **تفاح**، العنب. كما يقول **مختصون** أن تناول **النظام الغذائي** قليل **السكر** والجلوتين والصويا يساهم في التقليل من خطر الإصابة **بالالتهاب** مما يعزز من وظيفة الكلى **وأعمالها**.

- Using sentence re-ordering pattern, Theoretical Plagiarism detection value = 100%, Real Plagiarism detection value = 100%.

اعتمد في نظامك اليومي على الخضار والفواكه التي تعزز من وظائف الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، التفاح، العنب. تناول نظام غذائي قليل بالسكر والجلوتين والصويا كما يقول المختصون بأنه يساهم في التقليل من خطر الإصابة بالالتهابات مما يعزز من وظيفة الكلى وعملها. للحفاظ على صحة الكلى ، اعتمد النظام الغذائي الغني بالأطعمة النباتية مهم جداً.

- Replacing some verbs and nouns with their synonyms, Theoretical Plagiarism detection value = 100%, Real Plagiarism detection value = 80%. Reasoning: unavailability of some terms in AWN.

اعتماد النظام الغذائي الغني **بالأكلات** النباتية مهم جداً للحفاظ على صحة الكلى، **استخدم** في **غذائك** اليومي الخضار والفواكه التي **تقوي** من **عمليات** الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، التفاح، العنب. كما يقول **الباحثون** أن **أكل** نظام غذائي قليل بالسكر والجلوتين والصويا **يساعد** في **التخفيف** من خطر الإصابة **بالأمراض** مما يعزز من وظيفة الكلى وعملها.

- Using paraphrasing with synonym substitution (restructuring the main idea of the text with adding and deleting some terms), Theoretical Plagiarism detection value = 100%, Real

Plagiarism detection value = 88%. Reasoning: unavailability of some terms in AWN and miss stemming of some terms.

يعد الاعتماد على الأغذية التي تحتوي على الأطعمة النباتية من اسباب الحفاظ على وظائف الكلى، حيث أن الخضار والفواكة اغناها مثل: التوت ، الفلفل الرومي، التفاح و العنب. وفي دراسة سابقة المختصون أكدوا أن الأطعمة التي تحتوي على نسب قليلة من السكر والصويا تساهم بشكل كبير من خطر الإصابة بالالتهابات.

5.4 Comparative Results

In this section, we have compared our approach with the most effective approach presented in ExAraPlag 2015 competition which is the RDI [27]. The comparison process will be based on processing time and performance measures. Also, an overview over the competition will be shown.

5.4.1 ExAraPlag 2015 participants

Different systems have been proposed in Arabic plagiarism detection domain, but we can't compare them with our system since each has been measured on different dataset. Therefore, we have compared HYPLAG with the systems that have been tested on the same dataset.

ExAraPlag is the first Arabic text plagiarism detection competition. As we mentioned before, it targets both types of detection methods, intrinsic and extrinsic. A set of approaches have been proposed to participate in the competition. The final rank was based on the performance of approaches on Testing corpus. The following Table 5.7 summarizes the competitors' results in case of precision, recall and f-measure.

Table 5.7: Performance of the extrinsic plagiarism detection participants [102] on Testing corpus

Method	Precision	Recall	F-measure
Majooda_2 - RDI	85%	83%	84%
Polkovskii_1	98%	54%	69%
Baseline	99%	54%	69%
Alzahrani	83%	53%	65%

Based on the previous results of the competition, it is clear that RDI method has achieved the highest detection accuracy with 85% value of precision and 83% of recall. Therefore, we have compared HYPLAG to RDI approach. Table 5.8 summarizes the comparative process:

Table 5.8: Comparison summary against RDI system

Training Corpus				
	Precision	Recall	F-score	processing time (sec.)
Hyplag	96%	96%	96%	148577
RDI	85%	94%	89%	161190
Testing Corpus				
	Precision	Recall	F-score	processing time (sec.)
Hyplag	92%	87%	89%	143936
RDI	85%	83%	84%	158400

Using adopted performance accuracy measures results and based on processing time that summarized in Table 5.8, we can conclude that HYPLAG clearly overcomes RDI system in detection accuracy on both corpuses and in execution time period. For the processing time, HYPLAG has processed against both datasets in less time where the difference for Training dataset is approximately 12 hours where for the Testing dataset the difference is 15 hours. For the detection accuracy in the Training corpus, HYPLAG has outperforms with F-measure value 96% whereas RDI achieved 89% detection value. For the Testing corpus, HYPLAG achieved 89% where RDI 84%. From previous results, we can clearly identify that HYPLAG has outperformed RDI in both processing time and plagiarism detection accuracy.

6. Conclusion and Future Work

6.1 Conclusion

The main goal of this thesis is to enhance the detection ratio of plagiarism in the Arabic language domain by combining two current techniques. Plagiarism is a critical problem that faces most of researchers in their scientific life. Several researching statistics showed alarming numbers on plagiarism as behavior of university students. One of our motivations in this work that Arabic language has less attention in this field due to different reasons, although of its importance in worldwide as a cultural and religion language.

Given a set of suspicious documents, the goal of our system is to extract the plagiarized cases and to compute the originality value of them using a reference corpus and a knowledge resource. The plagiarism ratio of a document is depend on size of plagiarized cases contained in it. The suspicious sentences of a document are analyzed and compared to original ones using their sentence components. Sentences maybe share the same bag of component words, but meaningfully they are not identical. Also, some modification can be applied on sentences to obfuscate on their original meaning, and these sentences are assumed to be detected as plagiarism cases. Therefore, plagiarized sentences are not required to be identical in case of terms.

HYPLAG is based on search engine structure, in which the most relevant sentence of an input sentence is retrieved using a ranking method. The search engine structure can reduce the cost of pairwise similarity by preventing m x n comparison processes. In the proposed approach, two techniques are combined for the detection process, one utilizes the semantic information from a corpus and the other uses a knowledge resource to measure the terms similarity by identifying the semantic graph relation between terms. The detection process of the approach uses two sentence components to identify the plagiarized cases which are verbs and nouns, where adjective component was used with the two previous in the sentence retrieval process to

retrieve the most relevant sentence. So, sentence components were indexed with their reference sentences. Also since that stemming phase is a core process for any text preparation process, selecting the most efficient stemmer is an important step. Therefore, a comparative approach was proposed based on a knowledge resource to select the most efficient Arabic stemmer. The experimental results indicate that stemming phase is primary step for preparation of texts.

The detection process in our approach is identified as follows: the input document is normalized then segmented in to n-words length. Named entities are extracted to be discarded from the stemming process, and parts of speech tags are identified for each input sentence. For a sentence, based on its components, the most relevant sentence from a suspicious collection is retrieved. Then the detection process starts using verbs and nouns vectors. After detection process is completed using the combined methods, adjusting and aggregation models are used to place the last steps of the approach. A detailed description of the similarity detection process is discussed in chapter 4.

The system performance was evaluated using extrinsic plagiarism detection corpus implemented for a competition (ExAraPlagDet 2015) and it's free available online. A set of approaches were proposed in the competition using different techniques, and one was evaluated as the most accurate approach which is RDI. In chapter 5, we evaluate the performance of our approach and then it has been compared with RDI approach using traditional information retrieval measures. HYPLAG shows a superior results comparing with RDI using famous information retrieval measures, with processing the same datasets in a lower processing time. Based on our approach, combining corpus-based and knowledge-based techniques clearly produces accurate results with high precision and recall values.

Extracting and utilizing sentence components from the text produces better detection accuracy, better performance and lower processing time than traditional windowing techniques where sentences are viewed as sequence of terms. The preliminary study results that was conducted

on university students is confirmed by HYPLAG performance results. Where we conclude that the components have different significance in plagiarism process.

6.2 Future work

In the future work, we shall work on extending our used knowledge resource which is Arabic WordNet (AWN) in which more cases and synsets terms can be covered. Also, we could extend the knowledge source of the HYPLAG by combining AWN with another resource such as Arabic Ontology [103] when it became available online. Rich knowledge resources can overcome plagiarism issue in Arabic domain since that Arabic is very complex wide language. Also, based on our experiments, current Arabic stemmers provide accepted performance in case of stemming terms to their origin root. Most of root-based stemmers are slow and don't provide accurate results. Therefore, implementing an accurate root-based stemmer combined with Light stemming approach can clearly provide improvements in this field with many other fields.

Bibliography

1 Bin-Habtoor, A. S., & Zaher, M. A. (2012). A survey on plagiarism detection systems. International Journal of Computer Theory and Engineering, 4(2), 185.

2 https://en.wikipedia.org/wiki/Plagiarism/ Accessed on March/2017.

3 http://www.plagiarism.org/plagiarism-101/what-is-plagiarism/ Accessed on March/2017.

4 Gerald Nelms, (2012). Plagiarism Overview: What Research on Plagiarism Tells Us.

5 http://www.plagiarism.org/resources/facts-and-stats/, Accessed on October/2016.

6 http://www.checkforplagiarism.net/cyber-plagiarism, Accessed on October/2016.

7 https://infogr.am/Plagiarism-606324, Accessed on October/2016.

8 Ottenstein, K. J. (1976). An algorithmic approach to the detection and prevention of plagiarism. ACM Sigcse Bulletin, 8(4), 30-41.

9 https://www.grammarly.com/ Accessed on March/2017.

10 https://www.plagscan.com/ Accessed on March/2017.

11 http://turnitin.com/ Accessed on March/2017.

12 https://www.accreditedlanguage.com/2016/09/09/the-10-most-common-languages/ Accessed on March/2016.

13 http://www.vistawide.com/languages/top_30_languages.htm/ Accessed on April/2016.

14 Naik, Ramesh R., Maheshkumar B. Landge, and C. Namrata Mahender. "A Review on Plagiarism Detection Tools." International Journal of Computer Applications 125.11 (2015).

15 Mihalcea, R., Corley, C., & Strapparava, C. (2006, July). Corpus-based and knowledge-based measures of text semantic similarity. In AAAI (Vol. 6, pp. 775-780).

16 Zu Eissen, S. M., Stein, B., & Kulig, M. (2007). Plagiarism detection without reference collections. In Advances in data analysis (pp. 359-366). Springer Berlin Heidelberg.

17 Hoad, T. C., & Zobel, J. (2003). Methods for identifying versioned and plagiarized documents. Journal of the Association for Information Science and Technology, 54(3), 203-215.

18 Stamatatos, E. (2011). Plagiarism detection using stopword n-grams. Journal of the American Society for Information Science and Technology, 62(12), 2512-2527.

19 Zechner, M., Muhr, M., Kern, R., & Granitzer, M. (2009, September). External and intrinsic plagiarism detection using vector space models. In Proc. SEPLN (Vol. 32, pp. 47-55).

20 Abdi, A., Idris, N., Alguliyev, R. M., & Aliguliyev, R. M. (2015). PDLK: Plagiarism detection using linguistic knowledge. Expert Systems with Applications, 42(22), 8936-8946.

21 Li, Y., McLean, D., Bandar, Z. A., O'shea, J. D., & Crockett, K. (2006). Sentence similarity based on semantic nets and corpus statistics. IEEE transactions on knowledge and data engineering, 18(8), 1138-1150.

22 Alguliev, R. M., Aliguliyev, R. M., & Mehdiyev, C. A. (2011). Sentence selection for generic document summarization using an adaptive differential evolution algorithm. Swarm and Evolutionary Computation, 1(4), 213-222.

23 Gunasinghe, U. L. D. N., De Silva, W. A. M., de Silva, N. H. N. D., Perera, A. S., Sashika, W. A. D., & Premasiri, W. D. T. P. (2014, December). Sentence similarity measuring by vector space model. In Advances in ICT for Emerging Regions (ICTer), 2014 International Conference on (pp. 185-189). IEEE.

24 Thanh Ngoc Dao, Troy Simpson, 2005: Measuring similarity between sentences.

25 Leacock, C., & Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. WordNet: An electronic lexical database, 49(2), 265-283.

26 Alzahrani, S., & Salim, N. (2009). Statement-Based Fuzzy-Set Information Retrieval versus Fingerprints Matching for Plagiarism Detection in Arabic Documents. In 5th Postgraduate Annual Research Seminar (PARS 2009), Johor Bahru, Malaysia (pp. 267-268).

27 Magooda, A., Mahgoub, A. Y., Rashwan, M., Fayek, M. B., & Raafat, H. M. (2015). RDI System for Extrinsic Plagiarism Detection (RDI_RED), Working Notes for PANAraPlagDet at FIRE 2015. In FIRE Workshops (pp. 126-128).

28 Khan, I. H., Siddiqui, M. A., & Mansoor, K. (2015). A Framework FOR PLAGIARISM DETECTION IN ARABIC DOCUMENTS.

29 Jadalla, A., & Elnagar, A. (2012, May). A plagiarism detection system for Arabic text-based documents. In Pacific-Asia Workshop on Intelligence and Security Informatics (pp. 145-153). Springer Berlin Heidelberg.

30 Farahat, F. F., Asem, A. S., Zaher, M. A., & Fahiem, A. M. (2015). Detecting Plagiarism in Arabic E-Learning Using Text Mining.

31 Hussein, A. S. (2015). A Plagiarism Detection System for Arabic Documents. In Intelligent Systems' 2014 (pp. 541-552). Springer International Publishing.

32 http://plagiarismcheckerx.com/ Accessed on April/2017.

33 Ameer A.A. Yousef and Mohd J. (2015, July). Enhanced Tf-Idf Weighting Scheme For Plagiarism Detection Model For Arabic Language

34 http://misc-umc.org/AraPlagDet/?i=1/ accessed on December/2016.

35 https://ar.wikipedia.org/wiki/لغة_عربية / Accessed on April/2017.

36 Hammo, B. H. (2009). Towards enhancing retrieval effectiveness of search engines for diacritisized Arabic documents. Information retrieval, 12(3), 300-323.

37 Sawalha, M., & Atwell, E. S. (2008). Comparative evaluation of Arabic language morphological analysers and stemmers. In Proceedings of COLING 2008 22nd International Conference on Comptational Linguistics (Poster Volume)) (pp. 107-110). Coling 2008 Organizing Committee.

38 Duwairi, R., Al-Refai, M. N., & Khasawneh, N. (2009). Feature reduction techniques for Arabic text categorization. Journal of the American society for information science and technology, 60(11), 2347-2352.

39 Saad, M. K. (2010). The impact of text preprocessing and term weighting on arabic text classification (Master dissertation, The Islamic University-Gaza).

40 El Hadj, Y., Al-Sughayeir, I., & Al-Ansari, A. (2009, April). Arabic part-of-speech tagging using the sentence structure. In Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt.

41 Tlili-Guiassa, Y. (2006). Hybrid method for tagging Arabic text. Journal of Computer science, 2(3), 245-248.

42 Habash, N., Rambow, O., & Roth, R. (2009, April). MADA+ TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR), Cairo, Egypt (Vol. 41, p. 62).

43 Diab, M. (2009). Second generation AMIRA tools for Arabic processing: Fast and robust tokenization, POS tagging, and base phrase chunking. In 2nd International Conference on Arabic Language Resources and Tools (Vol. 110).

44 Zaraket, F. A., Jaber, A., & Makhlouta, J. (2016) Sarf: Fast and Application Customizable Arabic Morphological Analyzer. Natural Language Engineering, 1(1), 1-28.

45 Zhang, Y., Li, C., Barzilay, R., & Darwish, K. (2015). Randomized Greedy Inference for Joint Segmentation, POS Tagging and Dependency Parsing. In HLT-NAACL (pp. 42-52).

46 https://lucene.apache.org/, Accessed on November/2016.

47 http://cs.nyu.edu/cs/faculty/grishman/muc6.html/, Accessed on April/2017.

48 Meselhi, M. A., Bakr, H. M. A., Ziedan, I., & Shaalan, K. (2014, December). Hybrid Named Entity Recognition-Application to Arabic Language. In Computer Engineering & Systems (ICCES), 2014 9th International Conference on (pp. 80-85). IEEE.

49 Zaghouani, W. (2012). RENAR: A rule-based Arabic named entity recognition system. ACM Transactions on Asian Language Information Processing (TALIP), 11(1), 2.

50 Benajiba, Y., Rosso, P., & Benedíruiz, J. M. (2007, February). Anersys: An arabic named entity recognition system based on maximum entropy. In International Conference on Intelligent Text Processing and Computational Linguistics (pp. 143-153). Springer Berlin Heidelberg.

51 Darwish, K. (2013, August). Named Entity Recognition using Cross-lingual Resources: Arabic as an Example. In ACL (1) (pp. 1558-1567).

52 http://qatsdemo.cloudapp.net/farasa/, Accessed on December/2016.

53 https://en.wikipedia.org/wiki/Tf–idf/, Accessed on March/2017.

54 Taghva, K., Elkhoury, R., & Coombs, J. (2005, April). Arabic stemming without a root dictionary. In Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on (Vol. 1, pp. 152-157). IEEE.

55 Khoja, S., & Garside. (1999). Stemming Arabic text. Lancaster, UK, Computing Department, Lancaster University.

56 Chen, A., & Gey, F. C. (2002, November). Building an Arabic Stemmer for Information Retrieval. In TREC (Vol. 2002, pp. 631-639).

57 Larkey, L. S., Ballesteros, L., & Connell, M. E. (2007). Light stemming for Arabic information retrieval. In Arabic computational morphology (pp. 221-243). Springer Netherlands.

58 Saad, M. K., & Ashour, W. (2010). Arabic morphological tools for text mining. Corpora, 18, 19.

59 Al Ameed, H., Al Ketbi, S., Al Kaabi, A., Al Shebli, K., Al Shamsi, N., Al Nuaimi, N., & Al Muhairi, S. (2005, September). Arabic light stemmer: A new enhanced approach. In The Second International Conference on Innovations in Information Technology (IIT'05) (pp. 1-9).

60 T. Zerrouki. Tashaphyne, (2010). Arabic light Stemmer/segment.

61 Syiam, M. M., Fayed, Z. T., & Habib, M. B. (2006). An intelligent system for Arabic text categorization. International Journal of Intelligent Computing and Information Sciences, 6(1), 1-19.

62 http://globalwordnet.org/arabic-wordnet/, Accessed on November/2016.

63 http://www.comp.leeds.ac.uk/eric/latifa/research.htm/, Accessed on November/2016.

64 https://ar.wikipedia.org/wiki/, Accessed on November/2016.

65 Dahab, M. Y., Ibrahim, A., & Al-Mutawa, R. (2015). A comparative study on Arabic stemmers. International Journal of Computer Applications, 125(8).

66 Al-Sughaiyer, I. A., & Al-Kharashi, I. A. (2004). Arabic morphological analysis techniques: A comprehensive survey. Journal of the American Society for Information Science and Technology, 55(3), 189-213.

67 Taghva, K., Elkhoury, R., & Coombs, J. (2005, April). Arabic stemming without a root dictionary. In Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on (Vol. 1, pp. 152-157). IEEE.

68 Larkey, L. S., & Connell, M. E. (2005). Structured queries, language modeling, and relevance modeling in cross-language information retrieval. Information processing & management, 41(3), 457-473.

69 Eldesouki, M. I., Arafa, W., & Darwish, K. (2009). Stemming techniques of Arabic language: Comparative study from the information retrieval perspective. The Egyptian Computer Journal, 36(1), 30-49.

70 Otair, M. A. (2013). Comparative analysis of Arabic stemming algorithms. International Journal of Managing Information Technology, 5(2), 1.

71 https://en.wikipedia.org/wiki/Vector_space_model/, Accessed on April/2017.

72 Yan, T. W., & Garcia-Molina, H. (1994, February). Index structures for information filtering under the vector space model. In Data Engineering, 1994. Proceedings. 10th International Conference (pp. 337-347). IEEE.

73 Castells, P., Fernandez, M., & Vallet, D. (2007). An adaptation of the vector-space model for ontology-based information retrieval. IEEE transactions on knowledge and data engineering, 19(2).

74 Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. Communications of the ACM, 18(11), 613-620.

75 Lee, D. L., Chuang, H., & Seamons, K. (1997). Document ranking and the vector-space model. IEEE software, 14(2), 67-75.

76 Islam, A., & Inkpen, D. (2008). Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data (TKDD), 2(2), 10.

77 Ramos, J. (2003, December). Using tf-idf to determine word relevance in document queries. In Proceedings of the first instructional conference on machine learning.

78 Divya, C. (2014). Mining Contents in Web Pages and Ranking of Web Pages Using Cosine Similarity. International Journal of Science and Research (IJSR), 3(4).

79 https://en.wikipedia.org/wiki/Cosine_similarity/, Accessed on December/2016.

80 https://en.wikipedia.org/wiki/Euclidean_distance/ , Accessed on December/2016.

81 Meng, L., Huang, R., & Gu, J. (2013). A review of semantic similarity measures in WordNet. International Journal of Hybrid Information Technology, 6(1), 1-12.

82 Lee, J. C., & Cheah, Y. N. (2016, August). Paraphrase detection using semantic relatedness based on Synset Shortest Path in WordNet. In Advanced Informatics: Concepts, Theory And Application (ICAICTA), 2016 International Conference On (pp. 1-5). IEEE.

83 Wu, Z., & Palmer, M. (1994, June). Verbs semantics and lexical selection. In Proceedings of the 32nd annual meeting on Association for Computational Linguistics (pp. 133-138). Association for Computational Linguistics.

84 Meng, L., Huang, R., & Gu, J. (2013). A review of semantic similarity measures in wordnet. International Journal of Hybrid Information Technology, 6(1), 1-12.

85 Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. arXiv preprint cmp-lg/9511007.

86 Lin, D. (1998, July). An information-theoretic definition of similarity. In ICML (Vol. 98, No. 1998, pp. 296-304).

87 Tversky, A. (1977). Features of similarity. Psychological review, 84(4), 327.

88 http://www.talp.upc.edu/index.php/technology/resources/multilingual-lexicons-and-machine-translation-resources/multilingual-lexicons/72-awn/ Accessed on October/2017.

89 http://globalwordnet.org/arabic-wordnet/, Accessed on November/2016.

90 https://wordnet.princeton.edu/, Accessed on December/2016.

91 https://www.webteb.com/articles/18079_اربع-عادات-صحية-لتحفظ-كليتيك/ Accessed on March/2017.

92 Zobel, J., & Moffat, A. (2006). Inverted files for text search engines. ACM computing surveys (CSUR), 38(2), 6.

93 Melink, S., Raghavan, S., Yang, B., & Garcia-Molina, H. (2001). Building a distributed full-text index for the web. ACM Transactions on Information Systems (TOIS), 19(3), 217-241.

94 https://en.wikipedia.org/wiki/Inverted_index/ Accessed on April/2017.

95 https://en.wikipedia.org/wiki/Weighted_arithmetic_mean/ Accessed on April/2017.

96 Pirró, G., & Euzenat, J. (2010). A feature and information theoretic framework for semantic similarity and relatedness. The Semantic Web–ISWC 2010, 615-630.

97 Bao, J. P., Shen, J. Y., Liu, X. D., & Liu, H. Y. (2003, November). Quick asymmetric text similarity measures. In Machine Learning and Cybernetics, 2003 International Conference on (Vol. 1, pp. 374-379). IEEE.

98 Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval (Vol. 1, No. 1, p. 496). Cambridge: Cambridge university press.(154-156).

99 Yang, Y. (1999). An evaluation of statistical approaches to text categorization. Information retrieval, 1(1), 69-90.

100 https://en.wikipedia.org/wiki/F1_score/ Accessed on May/2017.

101 https://tika.apache.org/ accessed on March/2017.

102 Bensalem, I., Boukhalfa, I., Rosso, P., Abouenour, L., Darwish, K., & Chikhi, S. (2015). Overview of the AraPlagDet PAN@ FIRE2015 Shared Task on Arabic Plagiarism Detection. In FIRE Workshops (pp. 111-122).

103 Jarrar, M. (2011, April). Building a formal arabic ontology (invited paper). In proceedings of the Experts Meeting on Arabic Ontologies and Semantic Networks. Alecso, Arab League. Tunis.

Appendices

Questionnaire for Plagiarism Pattern

استبيان لنمط السرقـة الأدبية

- اعتماد النظام الغذائي الغني بالأطعمة النباتية مهم جداً للحفاظ على صحة الكلى، اعتمد في نظامك اليومي على الخضار والفواكه التي تعزز من وظائف الكلى، مثل: التوت البري، الفلفل الرومي الأحمر، التفاح، العنب. كما يقول المختصون أن تناول نظام غذائي قليل بالسكر والجلوتين والصويا يساهم في التقليل من خطر الإصابة بالالتهابات مما يعزز من وظيفة الكلى وعملها[1].

.....

.....

.....

.....

.....

- هل تقترح نمطاً أو طريقة لإخفاء الجملة التي قد تنتحل "غير ما ذكر"؟

.....

.....

.....

.....

.....

.. Thank you for participating ..

[1] https://www.webteb.com/

الملخص باللغة العربية

تعد اللغة العربية لغة صرفية معقدة تختلف عن باقي اللغات اللاتينية ، وكونها من اللغات الأكثر استخداما وتحدثا في وقتنا الحالي فإنها بحاجة الى مزيد من الجهود من قبل الباحثين لزيادة أعداد الأنظمة والبرامج المصممة خصيصا لها. تكثر الأبحاث التي تكتب بالعربية حيث أنها تستهدف مواضيع ومجالات عربية وإسلامية مختلفة، ومع ذلك فإن التوجة في تحسين الأدوات التي تستهدف اللغة العربية قليل مقارنة مع اللغات الأخرى. لذلك هذه الأطروحة تعرض نظام هجين لكشف السرقة الأدبية في النصوص العربية HYPLAG وهو نهج يستهدف النصوص العربية لكشف السرقات الأدبية بدمج آلياتان احداهما تعتمد على المجموعات البيانية وأخرى تعتمد على مصادر المعرفة باستخدام مصدر دلالي عربي. يتسم النهج المعروض بقابليته على كشف النصوص المنتحلة بطرق معقدة تستخدم مرادفات ومصادر الكلمات لاخفاء المصادر الأصلية للنصوص ، حيث أن الأنظمة المعروضة سابقا تعتمد في عمليات كشف الانتحالات على الكلمات نفسها بدون التوجة للمرادفات وقياس التشابة الدلالي للمصطلحات. وهكذا فإن الهدف الرئيسي من هذا البحث هو كشف تأثير عملية دمج الآليتين على أداء عملية كشف النصوص المسروقة في اللغة العربية.

أجريت دراسة أولية على طلاب جامعيين لفهم سلوكهم في عملية السرقة الأدبية ، وأظهرت نتائج هذه الدراسة أن الطلاب يقومون بتطبيق طرق متعددة لسرقة النصوص في عمليات الإنتحال. وأيضا أوضحت الدراسة أن من هذه الطرق هي تعديلات في أنواع الكلمات مثل الأفعال ، الأسماء والصفات. بناءً على نتائج هذه الدراسة تم بناء نظام HYPLAG آخذين بعين الإعتبار كل الأنماط التي تستخدم في عمليات السرقات الأدبية.

HYPLAG يقوم بمقارنة الملفات المدخلة المشبوهة مع مجموعة كبيرة من الملفات الأخرى بطريقة فعالة حيث أنه يستغل هيكلية محركات البحث في عملية إسترجاع المصادر الأصلية للملفات المشبوهة. و للتأكد من صحة النصوص المدخلة ، يقوم النظام بتطبيق مجموعة من العمليات الأولية التي تقوم بتجهيز النص وإزالة كل مصادر التشويش على عملية الكشف. و من أهم هذه العمليات هي استخراج مصادر الكلمات (Stemming). ولإستخدام أكثر أداة فعالية في عملية استخراج مصادر الكلمات من الأدوات المطروحة حاليا للغة العربية، قمنا بإقتراح نهج جديد يستخدم مصدر لغوي لمقارنة فعالية هذه الأدوات.

تمت مقارنة HYPLAG مع عدة أنظمة أخرى وأظهر دقة عالية و سرعة في المعالجة بإستخدام نفس البيانات التي استخدمت في الأنظمة الأخرى ، حيث أن هذه البيانات طرحت سابقا في مسابقة لكشف السرقة الأدبية للنصوص العربية (ExAraPlagDet-2015)، وأظهر النظام المعروض نتائج أفضل من الأنظمة الأخرى.