Data Compression Approach for Plagiarism Detection

PHD THESIS

2016 HUSSEIN KHALED HUSSEIN SOORI
Declaration:

“I hereby declare that this Ph.D. thesis was written by myself. I have quoted all the references I have drawn upon.”

HUSSEIN KHALED HUSSEIN SOORI

Signature:

Date: June 29th, 2016.
Acknowledgments:
First, I would like to express my deepest gratitude to my supervisor, prof. RNDr. Václav Snášel, CSc. for his guidance, support and encouragement, and for giving me his valuable advice and time during my PhD programme.

I also would like to thank my friends in the Administration Department and my colleagues in the Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VŠB – Technical University of Ostrava for their support.

Last but not least, I would like to extend my heartfelt thanks to my girlfriend, Hana Halásová for her patience and endless encouragement throughout the time of writing this thesis.
Abstrakt:
V naší digitální éře, je potřeba nástrojů pro detekci plagiátorství z důvodů obrovského počtu denně rostoucích dokumentů ať již v akademické sféře či mimo ní. Patří zde zprávy, úkoly studentů, bakalářské, magisterské či disertační práce. Zatímco někteří studenti používají metodu vyjmou a vložit, další skupina studentů se uchyluje k různým způsobům plagiátorství, včetně změn struktur vět, parafrázování i nahrazení slov jejich synonymem.Tato práce je zaměřena na vytvoření nástroje pro detekci textového plagiátorství při odhalování plagiátů v arabských a českých textech, dále na provádění počátečních částí kompresního algoritmu s jejimi modifikacemi, kde podobnost textu může být měřena na základě podobnosti kompresními-metrik. Dále se tato práce zaměřuje na to, že začleňuje tuto techniku v lexikonu českých synonym a v českém stemmer, kde odhaluje sémantické plagiátorství včetně parafrázování a restrukturalizace českých textů. Na druhé straně hledání kořenů slov a schopnost rozdělování slov na slabiky je velmi důležité v oblastech vyhledávání informací, dolování dat a zpracování jazyka. Vytvoření kvalitních pravidel pro rozklad na slabiky a hledání kořenů slov je stéžejní. Ještě vyšší poptávka je u jazyků, jimiž hovoří širší populace, jako je například arabština. Tato práce představuje novou metodu pro rozklad arabských slov na slabiky, založenou na arabských samohláskách. Práce také představuje snadnou metodu pro hledání kořenů slov pro arabský jazyk. Pro dokončení výsledku této metody, je nutné před použitím hledání kořenů slov, využít on-line syntetického analyzátoru. Ten se využívá pro lepší kategorizaci různých slovních druhů. Po té, tyto výstupní slova je nutné porovnat pomocí elektronického slovníku.

Klíčová slova: rozklad na slabiky, hledání kořenů slov, komprese dat, podobnost, detekce plagiátorů, textové plagiátorství

Abstract:
In our digital era, the need for plagiarism detection tools is growing with the tremendous number of documents produced on daily basis in and outside academia in all fields of science. This includes, reports, students’ assignments, undergraduate and graduate theses and dissertations. While some students use cut and paste methods, some other students resort to different ways of plagiarism including, changing the sentence structure, paraphrasing and replacing words with their synonyms.

This thesis focuses on creating textual plagiarism detection tools for detecting plagiarism of Arabic and Czech texts by implementing initial parts of a compression algorithm with its modifications where text similarity can be measured by compression-based similarity metrics. Next, it expands on this work by integrating this technique with a Czech synonyms thesaurus and a Czech stemmer to detect semantic plagiarism, including, paraphrasing and restructuring of Czech texts. On the other hand,
stemming and syllabification are very important in information retrieval, data mining and language processing. Creating good stemming and syllabification rules is crucial. The demand goes even higher with languages spoken by wider population, such as Arabic language. This thesis presents a novel method for syllabification of Arabic text based on Arabic vowel letters. The thesis also presents a light stemming method for Arabic language. To fine-tune the results of this method, an online parser is used, before stemming, to better categorize the different parts of speech and, later, the output words are matched with an electronic dictionary.

**Keywords:** syllabification, stemming, data compression, similarity, plagiarism detection, text plagiarism
## Contents

1 INTRODUCTION .................................................................................................................. 15

1.1 OBJECTIVE OF THE THESIS ....................................................................................... 17

1.2 ORGANIZATION OF THE THESIS ............................................................................... 17

2 SYLLABIFICATION AND STEMMING FOR DATA COMPRESSION IN ARABIC .......... 19

2.1 TEXT COMPRESSION AND ARABIC LANGUAGE ......................................................... 19

2.1.1 Text Compression ....................................................................................................... 19

2.1.2 Arabic Language ........................................................................................................ 21

2.1.3 Challenges to be Considered When Working with Arabic Text ................................. 21

2.2 SIMPLE RULES FOR SYLLABIFICATION OF ARABIC TEXTS ................................. 24

2.2.1 Initial Experiment ....................................................................................................... 24

2.2.2 Rule Set ....................................................................................................................... 25

2.2.3 Algorithm .................................................................................................................... 26

2.2.4 Processing of Text Files ............................................................................................ 28

2.2.5 Compression Experiments ......................................................................................... 28

2.3 SIMPLE STEMMING RULES FOR ARABIC LANGUAGE ............................................ 29

2.3.1 Stemming Rules ........................................................................................................ 30

2.3.2 Experiments ............................................................................................................... 34

2.4 STEMMING ENHANCED BY PARSING ...................................................................... 37

2.4.1 Stanford Online Parser ............................................................................................. 38

2.4.2 Stemming Rules ........................................................................................................ 40

2.4.3 Experiments and Implementation of Rules ................................................................. 42

2.4.4 Processing .................................................................................................................. 45

2.4.5 Results and Discussion .............................................................................................. 47

2.5 SUMMARY OF CHAPTER TWO .................................................................................... 48

3 DATA COMPRESSION APPROACH FOR PLAGIARISM DETECTION .......................... 51

3.1 PLAGIARISM DETECTION BY COMPRESSION ............................................................ 51

3.1.1 Methods of Plagiarism Detection .............................................................................. 52

3.1.2 Arabic Plagiarism Detection PAN Shared Task 2015 ............................................... 54

3.1.3 Creating Dictionaries out of Document .................................................................. 55

3.1.4 Similarity of Text ....................................................................................................... 55
3.1.5 Comparison of the Documents ................................................................. 55
3.2 TEXT SIMILARITY BASED ON DATA COMPRESSION IN ARABIC .................. 56
  3.2.1 Experimental Setup ................................................................................. 56
  3.2.2 False Suspicious Documents Creator Tool ........................................... 57
  3.2.3 The Experiment ...................................................................................... 57
  3.2.4 Stop Words Removal .............................................................................. 58
  3.2.5 Results ..................................................................................................... 58
  3.2.6 Visualization of Documents Similarity in PlagTool ar ............................ 60
  3.2.7 Conclusion .............................................................................................. 61
3.3 UTILIZING TEXT SIMILARITY MEASUREMENT FOR DATA COMPRESSION TO DETECT PLAGIARISM IN CZECH ................................................................. 61
  3.3.1 Experimental Setup ................................................................................. 62
  3.3.2 Document Creator Tool ......................................................................... 62
  3.3.3 The Experiment ...................................................................................... 63
  3.3.4 Stop Words Removal .............................................................................. 63
  3.3.6 Visualization of Document Similarity in PlagTool cz ............................ 64
  3.3.7 Conclusion .............................................................................................. 67
3.4 SEMANTIC AND SIMILARITY MEASURE METHODS FOR PLAGIARISM DETECTION OF STUDENTS’ ASSIGNMENTS ............................................................ 67
  3.4.1 Rationale, Training Data and Thesaurus ................................................. 67
  3.4.2 Short Texts Experiment .......................................................................... 68
  3.4.3 Pitfall of Using Data Compression Similarity Measure for Plagiarism Detection ................................................................. 72
  3.4.4 Long Texts Experiment of Students’ Assignments ................................. 73
  3.4.5 Conclusion .............................................................................................. 74
3.5 SUMMARY OF CHAPTER THREE ................................................................... 75
4 CONCLUSIONS ................................................................................................. 77
5 REFERENCES ................................................................................................... 81
**List of Tables**

Table 1 Arabic alphabets .................................................................................................................. 21
Table 2 Shape of the letter Ayn in different positions ................................................................. 21
Table 3 Example of Arabic LZ78 coding ....................................................................................... 22
Table 4 Effect of adding diacritic marks ......................................................................................... 23
Table 5 Statistics of testing files ................................................................................................... 28
Table 6 Experimental results (all sizes are in bytes) ..................................................................... 29
Table 7 Arabic alphabets transliterations ....................................................................................... 30
Table 8 List of words for each word category ............................................................................... 34
Table 9 Results for nouns ............................................................................................................. 35
Table 10 Results for verbs ............................................................................................................ 36
Table 11 Results for adverbs ......................................................................................................... 36
Table 12 Results for adjectives ..................................................................................................... 37
Table 13 Summary of experiment results ..................................................................................... 37
Table 14 Parts of speech in Stanford Online Parser .................................................................... 40
Table 15 Processed nouns - DTNN: determiner + singular common noun .................................. 42
Table 16 Processed nouns - DTNNP: determiner + singular proper noun .................................... 42
Table 17 Processed nouns - DTNNS determiner + plural common noun .................................... 43
Table 18 Processed nouns - NNPS: common noun, plural or dual .............................................. 43
Table 19 Processed verbs - VBD: perfect verb ............................................................................... 44
Table 20 Processed verbs – VBN: passive verb ............................................................................. 44
Table 21 Processed verbs – VBP: imperfect verb ........................................................................... 45
Table 22 Processed adjectives – DTJJ ................................................................. 45
Table 23 Processed nouns - DTNN: determiner + singular common noun .................... 46
Table 24 Processed nouns - DTNP: determiner + singular proper noun ....................... 46
Table 25 Processed nouns - DTNNS determiner + plural common noun ...................... 46
Table 26 Processed nouns -NNPS: common noun, plural or dual ................................ 46
Table 27 Processed verbs - VBD: perfect verb .......................................................... 46
Table 28 Processed verbs – VBN: passive verb ......................................................... 46
Table 29 Processed verbs – VBP: imperfect verb ....................................................... 47
Table 30 Processed adjectives – DTJJ ........................................................................ 47
Table 31 Summary of parser enhanced stemming results ............................................ 48
Table 32 Results of success rate PlagTool ar ............................................................... 59
Table 33 Results of success rate PlagTool cz ............................................................... 64
Table 34 Results of document similarities for short texts ........................................... 72
Table 35 Results of document similarities for long texts ............................................. 73
List of Figures

Figure 1 Arabic short vowels and diacritics ................................................................. 22
Figure 2 Context information in text files of various sizes ........................................... 25
Figure 3 Vowels, long vowels and diphthongs ............................................................. 25
Figure 4 Syllabification scheme .................................................................................... 27
Figure 5 Syllabification sample ..................................................................................... 27
Figure 6 Stanford Online Parser .................................................................................. 39
Figure 7 Source text ..................................................................................................... 59
Figure 8 Text with dispositioning of sentences and clauses .......................................... 59
Figure 9 Paraphrased text ............................................................................................. 60
Figure 10 PlagTool ar: Similarity for each suspicious paragraph in the document ......... 60
Figure 11 PlagTool ar: Histogram of document similarity ............................................. 61
Figure 12 PlagTool cz: Line chart of document similarity ............................................ 65
Figure 13 PlagTool cz: Histogram of document similarity ............................................ 65
Figure 14 PlagTool cz: Coloured visualization of similarity ........................................ 66
Figure 15 Training data (ORIGINAL TEXT) v1 ............................................................. 69
Figure 16 Testing data (SEMANTIC MATCH ONE) v2 ............................................... 69
Figure 17 Testing data (SEMANTIC MATCH TWO) v3 .............................................. 70
Figure 18 Testing data (SEMANTIC MATCH THREE) v4 .......................................... 70
Figure 19 Testing data (EXACT MATCH (with rearranged paragraphs and sentences)) v5 ........ 71
1 Introduction

Similarity detection is considered a crucial part of document processing. It covers a wide area including spam detection, plagiarism detection, etc. The need for plagiarism detection tools is growing with the growing number of natural language documents that are written day by day in schools and universities all over the world. The growing number of these documents include, students’ assignments, Bachelor’s, Masters’ and PhD theses and dissertations. While some students use to cut and paste methods, some other students resort to different ways of plagiarism including changing the sentence structure, paraphrasing and replacing the lexical meaning of words with synonyms. These require more sophisticated tools to detect plagiarism. The present thesis is devoted to create plagiarism detection tools by implementing a similarity measurement based on the initialization of Lempel Ziv comparison algorithms and its modifications and show its efficiency for detecting plagiarism of Arabic and Czech texts.

When using this technique for Arabic language, many language specific features has to be born in mind. Unlike languages that use Roman characters, Arabic is written from right to left and has twenty eight alphabets (three vowels and twenty five consonants). In case of Arabic, additional steps have to be performed before the computational processing such as, text normalization and some other steps that are specific to Arabic text processing. Hence, Arabic plagiarism detection tools require considering language-specific features in detecting text similarity.

The notion of plagiarism has expanded and went beyond just cutting and pasting a text into paraphrasing. According to Merriam Webster’s Dictionary, plagiarism is “the act of using another person’s words or ideas without giving credit to that person”. Since ideas may be reworded, this makes rewording without mentioning the source text -where the idea was taken from originally - an act of plagiarism. Semantic plagiarism may take many forms including changing the structure of sentences (restructuring) and replacing words with their synonyms, rewriting (paraphrasing) and translating ideas from other languages. The present thesis integrates a similarity measure technique previously used for text compression, along with a Czech synonyms thesaurus and a Czech stemmer to detect rewording and restructuring of Czech language texts.

The rationale behind the method proposed in the present thesis is that for someone to rewrite (paraphrase) a text, one needs to replace the original words either with different words that carry - more or less- the same meaning, i.e., synonym, or a variant of a synonym. For example, one may replace the word, important with synonyms, such as, significant, consequential, essential, relevant, etc. On the other hand, paraphrasing may also involve replacing the word at hand with a word that carries a negated opposite meaning, i.e., negated-antonym. For example, one may replace the word, important with negated-antonyms such as, far from insignificant, not inconsequential, not unessential,
not irrelevant, etc. Paraphrasing also involves restructuring of the sentence. The present work involves synonyms detection. Antonyms detection is not included in the present thesis.

On the other hand, stemming and syllabification are very important in information retrieval, data mining and language processing. Creating good stemming and syllabification rules is crucial. The demand goes even higher with languages spoken by wider population, such as Arabic language. The importance of Arabic language comes from the fact that it is the sixth most used language in the world. Arabic is considered as one of the highly inflectional languages with complex morphology where prefixes and suffixes are added to the stem to form words.

The importance of conducting research in the area of stemming and syllabification of Arabic is called for, for two main reasons: the rapidly growing number of computer and internet users in the Arab world, and the fact that the Arabic language is the sixth most used language in the world today. Another important factor is, after the Latin alphabet, Arabic alphabet is the second-most widely used alphabet around the world. Arabic script has been used and adapted to such diverse languages as the Slavic tongues (also known as Slavic languages), Spanish, Persian, Urdu, Turkish, Hebrew, Amazigh (Berber), Swahili, Malay (in Malaysia), Jawi (in Indonesia), Hausa and Mandinka (in West Africa) and Swahili (in East Africa). Therefore, computational processing of the Arabic language and Arabic alphabet is crucial and in demand.

Several approaches to text compression were developed in the field including, character-based compression suitable for small files, word-based compression suitable for very long files and syllable-based approach, which uses the syllable as a basic element. Algorithms for syllabification of English, German and other European language are well known. However, syllabification algorithms for Arabic and their usage in text compression has not been fully investigated. The present thesis describes a novel method for syllabification of Arabic text based on Arabic vowel letters.

In addition to that, the problem of stemming is very important in information retrieval, data mining and language processing. As mentioned above, processing of Arabic language is eminent for the facts that that it is widely spoken and currently the number of computer and Internet users in the Arab world is growing tremendously. However, the sophisticated and complex morphology of Arabic could be challenging for researchers where stemming rules must deal with many language-specific properties of Arabic. Many light stemming rules have been introduced in the past to fulfil this important issue. The present thesis attempts to expand on these efforts and introduce a light stemming method. To fine-tune the results of the light stemming method introduced, this thesis uses an online parser, before stemming, to better categorize the different parts of speech and, later, to match the output words with an electronic dictionary.
1.1 Objective of the Thesis

This thesis aims at finding applicable methods for plagiarism detection tools for Arabic and Czech texts by implementing a similarity measurement based on an initialization of Lempel Ziv compression algorithms and its modifications and show its efficiency for detecting textual plagiarism. It also aims at extending this work to include a semantic plagiarism detection method of texts including rewording, restructuring and using synonyms. In addition to that, the present work aims at presenting rules for syllabification of Arabic text using vowel letters, and a light stemming method, for data compression, and showing the viability of these rules for syllabification and stemming.

1.2 Organization of the Thesis

This thesis is divided into four chapters. Chapter one is an introduction to the thesis. This chapter outlines the framework of the thesis. It also includes the objectives and the organization of the thesis. Chapter two is mainly divided into two parts. The first part covers a syllabification method for Arabic language based on vowel letters. The second part includes a light stemming method for data compression enhanced by an online parser for the categorization of Arabic parts of speech. Chapter three is divided into three parts. The first and second parts are dedicated to plagiarism detection tools for Arabic and Czech texts by implementing a similarity measurement based on a comparison algorithms. The third part includes a semantic plagiarism detection tool for reworded and restructured Czech texts by using a synonyms thesaurus and a stemmer. The last chapter includes the conclusions drawn from this thesis.
2 Syllabification and Stemming for Data Compression in Arabic

This chapter includes a preliminary review of data compression and Arabic language. It also includes two methods for Arabic text compression. The first is a novel syllabification method and the second is a light stemming method enhanced by a parser. The results of these experiments were published in Soori et al. [33], [35] and [77].

2.1 Text Compression and Arabic Language

The importance of conducting research in the area of syllabification of Arabic is called for, for two main reasons: the rapidly growing number of computer and internet users in the Arab world and the fact that the Arabic language is the sixth most used language in the world today. Another important factor is, after the Latin alphabet, Arabic alphabet is the second-most widely used alphabet around the world. Arabic script has been used and adapted to such diverse languages as the Slavic tongues (also known as Slavic languages), Spanish, Persian, Urdu, Turkish, Hebrew, Amazigh (Berber), Swahili, Malay (in Malaysia), Jawi (in Indonesia), Hausa and Mandinka (in West Africa) and Swahili (in East Africa) [10]. Therefore, the computer processing of the Arabic language and Arabic alphabet is increasingly becoming a crucial task.

2.1.1 Text Compression

The key to data compression is the ability to model data to improve the efficiency of prediction [25]. The modeling of the data consists of modeling the context [1]. Modeling textual data means the modeling of natural languages. The most widely analyzed language is English. The first analysis of English was made by Shannon, who computed its entropy [26]. The latter analysis of English was closely related to the PPM compression method. Bell et al., in an excellent survey in [1], describe the state of the modeling of data at the end of the year 1989. They describe the principles of data modeling for context models (such as the PPM compression method). Moreover, the paper describes the principles of other possible models - finite state models like Dynamic Markov Compression [6, 13] and grammar or syntactic models [4, 15]. PPM-based compression usually uses contexts with a limited length. For English, a context with a maximal length of 6 symbols is standard [1]. Cleary et al. [5] and Bloom [2] inspect the possibility of unlimited context length. Skibinski and Grabowski [28] summarize both the previous methods and describe their own improvements based on variable-length context. The problem of the memory requirements of the PPM model is discussed by Drinic et al. in [7]. Shkarin [27] describes the practical implementation of the PPM method using instruction sets of modern processors, Lelewer and Hirschberg [19] suggest a variation of PPM optimized for speed.
Compression based on a combination of several models using neural networks was presented by Mahoney [21]. This concept is called context mixing. Algorithms based on this principle are the most efficient in several independent compression benchmarks.

Another research study in the field of text compression focuses on analyzing context information in files of various sizes. The main advantage of this approach is that the compression algorithms may be used without any modification. The efficiency (scalability) of the compression methods used for text compression may be related with the selection of the basic elements.

The first approaches focused on the compression of large text files. For large files, words were used as the basic elements. This approach is called word-based compression [14, 8, 31]. All the basic compression methods were adapted to this type of compression. The first modification was a modification of Huffman encoding called Huffword [31]. Later, other methods followed: WLZW as a modification of the LZW method [14, 8], and WBW as a modification of the Burrows-Wheeler transformation-based compression [9, 22]. PPM has also its modification for word-based compression [31]. The last modification for word-based compression was made for LZ77 compression [23]. As written above, all these modifications left the main compression unchanged. The only change is made in the first part of the algorithms, where the input symbols are read. The input file is read as a character sequence. From this sequence several types of elements are decomposed. Usually, two basic element types are recognized - words, which are represented as sequences of letters, and non-words, which are represented as sequences of white spaces, numbers, and other symbols. Algorithms must solve problems with capital letters, special types of words, hapax-legomenons (words which are present only once in the file), etc. A third type of element is sometimes also defined. The most commonly used one is a tag, which is used for structured documents such as XML, HTML, SGML, and others. A very interesting survey of dictionary-based text compression and transformation was presented by Skibinski et al. [29].

The second approach evolved by Lansky [18, 17] is syllable-based compression. Lansky made experiments with LZW, Huffman encoding, and Burrows-Wheeler transformation-based compression methods adapted for using syllables as basic elements [18, 17]. The main problem of this approach is the process of separation of syllables, also called hyphenation algorithms. This algorithm is specific to any language. The rules for word hyphenation may be heuristic, linguistically based, or other. The first two approaches were described for the Czech, English, and German languages in [18, 17]. A soft computing approach to hyphenation is described in [16], where a genetic algorithm is used for preparing the optimal set of syllables for achieving optimum compression using Huffman encoding.
2.1.2 Arabic Language

 Creating good stemming rules for the Arabic language comes from the importance of Arabic language as the sixth most used language in the world. Stemming and syllabification are very important in information retrieval, data mining and language processing. Arabic is considered as one of the highly inflectional languages with complex morphology. Unlike most other languages, it is written horizontally from right to left. It consists of 28 main letters as shown in Table 1.

<table>
<thead>
<tr>
<th>د</th>
<th>ن</th>
<th>م</th>
<th>م</th>
<th>ن</th>
<th>ه</th>
<th>و</th>
<th>ي</th>
</tr>
</thead>
<tbody>
<tr>
<td>ghun, ghain</td>
<td>ayn</td>
<td>wa</td>
<td>lam</td>
<td>mimm</td>
<td>laam</td>
<td>kenaf</td>
<td></td>
</tr>
</tbody>
</table>

Table 1 Arabic alphabets

The shape of each letter depends on its position in a word—initial, medial, and final. There is a fourth form of the letter when written alone/ separated. One example of this can be given for the letter ayn (ع) as in Table 2:

<table>
<thead>
<tr>
<th>Initial</th>
<th>Medial</th>
<th>Final</th>
<th>Separate</th>
</tr>
</thead>
<tbody>
<tr>
<td>ع</td>
<td>ع</td>
<td>ع</td>
<td>ع</td>
</tr>
</tbody>
</table>

Table 2 Shape of the letter Ayn in different positions

With Arabic having highly complex morphology and grammatical properties, this poses a challenge for researchers in this field.

2.1.3 Challenges to be Considered When Working with Arabic Text

A few challenges may face researchers as for as the special nature of Arabic script is concerned. Moreover, the letters alif, waaw, and yaa (standing for glottal stop, w, and y, respectively) are used to represent the long vowels a, u, and i. This is very much different from Roman alphabet which is naturally not linked. An example of Arabic LZ78 coding is shown in table 3 for the sentence, (فلما كلت كلمتي كلمتي كلت كلمتي كلمتي كلمتي كلمتي الكلت كلمتي).
Other orthographic challenges can be the persistent and widespread variation in the spelling of letters such as hamza (ء) and Ta’ marbuTa (ۡ). In addition to the increasing lack of differentiation between word-final ya (ي) and alif maqSura (ى). Typists often neglect to insert a space after words that end with a non-conector letter such as، و، ر and ز [3].

In addition to that, Arabic has eight short vowels and diacritics as shown in Figure 1. Typists normally ignore putting them in a text, but in case of texts where typists put them, they are pre-normalized—in value- to avoid any mismatching with the dictionary or corpus. As a result, the letters in the decompressed text, appear without these special diacritics.

Diacritization has always been a problem for researches. According to Habash [12], since diacritical problems in Arabic occur so infrequently, they are removed from the text by most researchers. Other text recognition studies in Arabic include, Andrew Gillies et al. [11], John Trenkle et al. [30] and Maamouri et al. [20].

Short vowels and diacritics also determine the word identity and, in many instances, can change the meaning and part of speech of the word as shown in Table 4, where we may see a total change in word meaning and part of speech as a result of adding the diacritic marks which resulted in producing three different words in meaning and four different parts of speech for the same word رجل (man).
<table>
<thead>
<tr>
<th>Word</th>
<th>Meaning</th>
<th>Part of Speech</th>
</tr>
</thead>
<tbody>
<tr>
<td>رجل</td>
<td>man</td>
<td>noun (subject)</td>
</tr>
<tr>
<td>رجل</td>
<td>man</td>
<td>noun (object)</td>
</tr>
<tr>
<td>رجل</td>
<td>foot</td>
<td>Noun</td>
</tr>
<tr>
<td>رجل</td>
<td>to go on foot (rather than, <em>e.g.</em>, ride a bike)</td>
<td>Verb</td>
</tr>
</tbody>
</table>

Table 4 Effect of adding diacritic marks

Never the less, it is always advised that these vowels and diacritics are often normalized before textual computational processing in most light stemming or morphological approaches [12]. Mainly the reasons for not including them in word processing is the claim that they do occur so infrequently, and that in Modern Standard Arabic (MSA), people tend not to use them and, as a result of that, the meaning is left for the native speaker’s intuition, or, in some cases, can be determined from the context.

Another morphological feature in Arabic is that unlike Roman letters, which are separated naturally, Arabic has an agglutinated nature where letters are linked to each other in some cases, while unlinked in some other cases, depending on position of the letter in the root, stem and word. For example, in English the personal pronoun *he* in *he plays* is separated from the following verb *plays*. In Arabic the personal pronoun is represented by the letter (ي) which is linked prefix to the root verb (لعب) to form (يلعب) = *he plays*. The same is true when it comes to different Affixes.

Arabic has four types of affixes: prefixes, these are letters (normally one) that change the tense of the verb from past to present, such as the letter (ي) in case of the verb (لعب) and (يلعب) above; suffixes, these represent the inflectional terminations (endings) of verbs, as well as, the female and dual/plural markers for the nouns; postfixes, these are the pronouns attached at the end of the word; antefixes, these are prepositions agglutinated to the beginning of words. Mainly the problem is not only that these are linked to the beginning or the end, but they often require changes in the root and this can be really a challenging task for Arabic text processing.
2.2 Simple Rules for Syllabification of Arabic Texts

In the past, several works about analyzing of the Arabic language were published, as discussed in sections 2.1.2 and 2.1.3 above. However, the language analysis is only one-step in language processing. Several approaches to the text compression were developed in the field of text compression. The first is the character-based compression that is suitable for small files. Another approach is called word-based compression which is very suitable for very long files. The third approach is called syllable-based, which uses the syllable as a basic element. Algorithms for the syllabification of the English, German or other language with Roman alphabets are well known, but syllabification algorithms for Arabic and their usage in text compression has not been fully investigated. The following sub-sections describe a novel method for syllabification of Arabic text based on Arabic vowel letters as described in [33].

2.2.1 Initial Experiment

For the purpose of this experiment, three different approaches to text compression, based on the selection of input symbols are defined: character-based, syllable-based, and word-based. The question is which type is the most suitable for defined texts. In [17], authors compare the single file parsing methods used on input text files of a size 1KB-5MB by means of the Burrows-Wheeler Transform for different languages (English, Czech, and German). They consider these input symbol types: letters, syllables, words, 3-grams, and 5-grams. Comparing letter-based, syllable-based, and word-based compression, they came to the conclusion that character-based compression is the most suitable for small files (up to 200KB) and that syllable-based compression is the best for files of a size 200KB-5MB. The compression type that uses natural text units such as words or syllables is 10-30% better than the compression with 5-grams and 3-grams. For larger files, word-based compression methods are the most efficient.

These findings correspond to the amount of contextual information stored in files. Contextual information in large text files (larger than 5 MB) is very big and therefore it is sufficient to process them using word-based compression as shown in Figure 2a. Medium-sized text files (200KB-5MB) have less contextual information than large text files, and, therefore, it is insufficient to take the word as a base unit. However, we can take a smaller grammatical part and use syllable-based compression as shown in Figure 2b. Contextual information in small files (10KB - 200KB) is difficult to obtain. In small files, the contextual information is sufficient only when they were processed by characters as shown in Figure 2c.

The efficiency of the text compression may be improved with simple modification when multiple documents are compressed. In [24], cluster analysis is used for improving the efficiency of
the compression methods. All the main compression methods adapted for word-based compression are tested and the compression ratio is improved by up to 4 percent.

2.2.2 Rule Set

The rules for syllable separation are well defined for English, German, Czech and many other European languages. However, as far as Arabic is concerned, research in this area is still young. Arabic script is different from languages using Roman characters in many aspects, for example: text direction (right to left) and the agglutination of letters in a word. This study defines several rules for the decomposition of Arabic words into syllables. These rules are based on splitting the text according to some linguistic criteria called here, split markers. The basic criterion for splitting the text is tracing the vowel letter in a text. In this proposed method, the term Arabic vowels includes vowels, long vowels and diphthongs as shown in Figure 3.

Figure 3 Vowels, long vowels and diphthongs

It is to be mentioned here that the letters (٘, ٕ) in Figure 3 are dual letters, i.e., they both contain two letters of which the first is a consonant letter and the second is a vowel. It is also worth
mentioning here that, interestingly enough, in the Arabic keyboard, the Unicode value of dual letters is not always equal to the total value of two agglutinated letters if taken separately. For example, while the total Unicode value of the dual letters in (ي, ی) are the same if taken separately, the dual letter characters such as (ژ, ژ) in Figure 3, carry different Unicode values if taken separately.

Split markers include also some other kinds of vowels which are indicated by diacritical marks called, short vowels. These are placed above or below the letter, such as, (ََ, ََ, ََ) (see Figure 1).

Another diacritical marks that are included among the split markers in this study are nunations such as (ََ, ََ, ََ) (see Figure 1). In written Arabic, a nunation is indicated by doubling the vowel diacritic at the end of the word. These diacritical marks mentioned above in Figure 1 are used in fully vocalized texts in Arabic. However, they are also used occasionally in Modern Standard Arabic (MSA) to clarify meaning and part of speech overlap (see Table 4).

2.2.3 Algorithm

The rule set defined in the previous sub-section leads to an algorithm which may be defined as follows:

Read the input file and process all words by the following steps.

[1] When the length of the word is less than 3 characters, ignore it, even if it contains a vowel.
[2] When the second or third letters are vowels, make a split between the second and third letter.
[3] Otherwise, make a split after the third letter.

Input: A text file F.
Output: A set of words with separated syllables.

ForEach (Word W in file F)
  If (|W|>=3)
    If (IsWovel(W[1]) && IsWovel(W[2]))
      MakeSplit(W, 1, 2)
    Else
      MakeSplit(W, 2, 3)

Note: |W| is a length of the string,
      IsWovel is a function that returns true when a letter in parameter is a wovel, and return false otherwise,
      MakeSplit is a function that splits a word into W into two parts between the indices passed as parameters.

The syllabification schema is depicted in Figure 4.
As may be noticed, the algorithm is quite simple. A sample syllabification using this algorithm may be seen in the Figure 5. The red hyphens are used as splitting marks.

Figure 4 Syllabification scheme

Figure 5 Syllabification sample
2.2.4  Processing of Text Files

For testing purposes, several text files written in Arabic language were tested. The first group comprises short texts from Arabic newspapers. The second group contains several larger files. The statistics of the used files is shown in Table 5. For each file, decomposition was tested into: 2-grams, 3-grams, characters, words and the syllables method as described in this algorithm. The whitespaces are handled as a one syllable/word in syllable and word decomposition. The size of the alphabet extracted from the testing files using the vowel method is also referred to as depicted in Table 5.

The testing files are selected from two sources. The first source is four short texts taken from the web pages of Arabic international news channels, and the long text is taken from shareware Arabic books from the web. All texts are encoded using codepage 1256 - Arabic Script.

<table>
<thead>
<tr>
<th>File:</th>
<th>Short1</th>
<th>Short2</th>
<th>Short3</th>
<th>Short4</th>
<th>Long1</th>
<th>Long2</th>
</tr>
</thead>
<tbody>
<tr>
<td>File size (B)</td>
<td>1922</td>
<td>7926</td>
<td>2910</td>
<td>12278</td>
<td>1547552</td>
<td>845431</td>
</tr>
<tr>
<td>Unique Chars</td>
<td>45</td>
<td>64</td>
<td>51</td>
<td>71</td>
<td>95</td>
<td>60</td>
</tr>
<tr>
<td>Unique Syllables</td>
<td>320</td>
<td>898</td>
<td>490</td>
<td>1117</td>
<td>9342</td>
<td>6521</td>
</tr>
<tr>
<td>Unique Words</td>
<td>232</td>
<td>837</td>
<td>391</td>
<td>1187</td>
<td>55006</td>
<td>28336</td>
</tr>
<tr>
<td>Unique 2-grams</td>
<td>301</td>
<td>623</td>
<td>425</td>
<td>712</td>
<td>2208</td>
<td>1251</td>
</tr>
<tr>
<td>Unique 3-grams</td>
<td>429</td>
<td>1397</td>
<td>680</td>
<td>1843</td>
<td>19377</td>
<td>11113</td>
</tr>
</tbody>
</table>

Table 5 Statistics of testing files

As may be seen from Table 5, the number of the extracted unique syllables is rather high, but it is always less than the number of 3-grams. As for the files bigger than 10 kb, it is also lower than the number of words.

2.2.5  Compression Experiments

In this section, two experiments are performed. The first algorithm is based on Burrows-Wheeler transformation with combination Huffman encoding and Move-To-Front and RLE transformation. This algorithm is chosen because its efficiency is based on the relationships between sibling symbols. The second algorithm is LZW compression algorithm which gains its efficiency from repetitions of symbols.

During experiments, a large enough dictionary was used for LZW algorithm and large enough block size for BWT based algorithm to avoid clearing of dictionary and splitting the input into several blocks.

The results of the experiment are shown in Table 6. For comparison purposes, it is necessary to compare sizes of the compressed texts together with the size of the dictionary used.
As may be seen from the results, the most efficient compression algorithm for small files (Short1-Short4) is character-based compression. All the other approaches did not achieve good results. This corresponds with the information mentioned in Table 5. The best results for larger files (Long1 and Long2) were achieved by 2-grams approach, but the character and syllable approach have almost the same efficiency. These experiments show that the present syllabification algorithm needs to be improved and more experiments will have to be performed in this regard in the future.

2.3 Simple Stemming Rules for Arabic Language

Processing of Arabic language is eminent for the fact that currently the number of computer and Internet users in the Arab word is growing tremendously. The problem of stemming is very important in information retrieval, knowledge mining and language processing. Arabic has very sophisticated and complex morphology, in addition to stemming rules that must deal with many specific properties of Arabic, as discussed in sub-section 2.1.3 above. Some studies have worked in the past to fulfil this important issue. In this section, the present thesis tries to expand on these efforts and introduce a light stemming method.

This section, describes very simple light rules for stemming of Arabic words as described in [35, 77]. Two of these rules are universal, i.e. they are applicable to any word category. The other rules are set for each of the four categories of part of speech: nouns, verbs, adverbs and adjectives.
After that, the method attempts to enhance the results by using an online morphological parser to distinguish between parts of speech, set some extracting rules to produce stems, and finally, match these stems with an electronic dictionary. In this later stage nouns, verbs and adjectives are dealt with.

### 2.3.1 Stemming Rules

The rules are divided into five sets. The alphabets in the rules are transliterated as per the Arabic alphabets transliterations in Table 7. These sets of words are designed to target verbs, nouns, adjectives and adverbs.

<table>
<thead>
<tr>
<th>Arabic Alphabet</th>
<th>Transliteration</th>
<th>Arabic Alphabet</th>
<th>Transliteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>ا</td>
<td>alif</td>
<td>ع</td>
<td>ayn</td>
</tr>
<tr>
<td>ب</td>
<td>baa</td>
<td>غ</td>
<td>ghayn</td>
</tr>
<tr>
<td>ت</td>
<td>ta</td>
<td>ف</td>
<td>faa</td>
</tr>
<tr>
<td>ث</td>
<td>tha</td>
<td>ق</td>
<td>qaaf</td>
</tr>
<tr>
<td>ج</td>
<td>jiim</td>
<td>ك</td>
<td>kaaf</td>
</tr>
<tr>
<td>ح</td>
<td>haa</td>
<td>ل</td>
<td>laam</td>
</tr>
<tr>
<td>خ</td>
<td>kha</td>
<td>م</td>
<td>miim</td>
</tr>
<tr>
<td>د</td>
<td>daal</td>
<td>ن</td>
<td>nuun</td>
</tr>
<tr>
<td>ذ</td>
<td>thal</td>
<td>هـ</td>
<td>haa</td>
</tr>
<tr>
<td>ر</td>
<td>raa</td>
<td>ة</td>
<td>taMarboota</td>
</tr>
<tr>
<td>ز</td>
<td>zay</td>
<td>و</td>
<td>waaw</td>
</tr>
<tr>
<td>س</td>
<td>siin</td>
<td>لا</td>
<td>laamAlif</td>
</tr>
<tr>
<td>ش</td>
<td>shiin</td>
<td>ء</td>
<td>hamza</td>
</tr>
<tr>
<td>ص</td>
<td>saad</td>
<td>ئ</td>
<td>hamzaOnYaa</td>
</tr>
<tr>
<td>ض</td>
<td>daad</td>
<td>ؤ</td>
<td>hamzaOnWaaw</td>
</tr>
<tr>
<td>ط</td>
<td>taa</td>
<td>ي</td>
<td>yaa</td>
</tr>
<tr>
<td>ظ</td>
<td>dhaa</td>
<td>ئ</td>
<td>alifMaqsoora</td>
</tr>
</tbody>
</table>

Table 7 Arabic alphabets transliterations
The first set is focused on any word type. The first necessary step is to normalize the words using the following rules: convert all hamzated alif (أ) and (إ) to bare alif (ا) and change alifmaqSura (ء) to yaa (ي).

\[
\text{[hamzatedAlif]}W \rightarrow \text{[alif]}W \\
\text{[AlifmaqSura]}W \rightarrow \text{[yaa]}W \\
\]

After normalization, remove all possible prefixes (preposition, definite articles and conjunctions) using following process. Remove baa from all beginning of every word.

\[
\text{[baa]}W \rightarrow []W \\
\]

Remove waaw from all beginning of every word, if the remaining is three or more characters.

\[
\text{[waw]}W \&\& |W| > 3 \rightarrow []W \\
\]

Then remove the following linked characters from the beginning of a word, if the remaining number of characters is three or more characters:

\[
|W| > 3 \\
\text{[AlifLam]}W \rightarrow []W \\
\text{[Alif waw lam]}W \rightarrow []W \\
\text{[Alif baa lam]}W \rightarrow []W \\
\text{[Alif kaf lam]}W \rightarrow []W \\
\text{[Alif faa lam]}W \rightarrow []W \\
\]

The second set of rules are designed for suffixes. The rule removes linked suffixes/characters if the remaining is two or more characters:

\[
|W| > 2 \\
\text{[W][Ha alif]} \rightarrow [W] \\
\text{[W][Alif nuun]} \rightarrow [W] \\
\text{[W][Alif taa]} \rightarrow [W] \\
\text{[W][Waaw nuun]} \rightarrow [W] \\
\text{[W][Yaa nuun]} \rightarrow [W] \\
\text{[W][Yaa ha]} \rightarrow [W] \\
\text{[W][Ya taaMarboota]} \rightarrow [W] \\
\]

The third set of rules are designed for Adjectives and Adverbs, more precisely for changing adjectives and adverbs from feminine into muscular form, if these characters are as the last in a word.

\[
\text{[W][TaaMarboota]} \rightarrow [W] \\
\text{[W][Taa]} \rightarrow [W] \\
\]
The fourth sets of rules are designed for nouns; it is to change nouns from plural into singular form. The rule applies if the final remaining is longer than two characters. In this rule, if the word has only three characters, remove the following characters if the additional conditions below are fulfilled.

Remove alif from the beginning of every word;
Remove alif, if it is found as the character before the last one;
Remove waaw, if it is found as the character before the last one;
Remove alif from the beginning of every word, only if the character before the last is also alif.

If the word has four characters, remove the following characters if the additional conditions are fulfilled. Remove alif, if it is found as the character before the last character.

If the word has five characters, remove the following characters if the additional conditions are fulfilled. Remove alif from the beginning of the word, replace it with waaw, and delete the character before the last character.

The fifth set of rules are designed for verbs. It converts inflectional forms of the verb into stems. The character/s are removed from the end of the word, if the remaining is at least 2 characters.
W[nuun yaa alif] -> W

Remove: alif if found as the second character in a word, only if alif is also found as the last character in a word.

w[alif]W[alif] -> wW[alif]

If alif is found as a second character in a word and taa as the last character, remove alif and remove taa from the word.

w[alif]W[taa] -> wW

Notation:

w – single character.
W – one or more letters.

These sets of rules are applied on the list of words according to four categories: nouns, verbs, adjectives and adverbs as shown in Table 8.
a. Nouns

تمساح حصان سمك أرنب حوت خد حدوت دجاج
إصابع قلب أسنان مكتبة مكتب دجاجة دجاج لحم قميص زوجة
الصين بلد مكتبة خليج صحراء ضباب تلج عاصفة سكرتيرة سكرتير
الإجهاض المحامي الخريف الكرة

b. Verbs

سمح إعترفت سأل عض كسر اتصل اعترف ركل تعامل رسم
تزوج تغلبت على زرع كتب رن استمتع توس

c. Adverb

 قادر نشيط غاضب مستيقظ سي أحسن أفضل يغلي مكسور

d. Adjectives

 كبيرة كبيرة صغيرة صغيرة جميلة جميلة قميص قميص أزرق أوكم
أزرق أزرق أزرق أوكم

Table 8 List of words for each word category

2.3.2 Experiments

The suggested rules must be tested against real data. For this purpose, some news articles were used, from the BBC Arabic and Al Jazeera Arabic news portal articles, of which some of the most frequent nouns, verbs, adverbs and adjectives in those articles were selected. These words were divided into groups of files according to four parts of speech. Each group consists of several words, selected precisely to cover all type of words. The lists of words used in the experiments are shown in Table 8. All source data were encoded using code page 1256 Arabic.

In this experiments, the rules described in the previous sub-section 2.3.1 were applied. There are three common rules used for each word type and one specific rule per group. Accordingly, each word was processed by four step rules and therefore, there are five versions of a word during its processing.
The first version is the original word. The second version normalizes alif and alif maqṣura. The third version shows the word after it had already processed prepositions, definite articles and conjunctions. The fourth version contains the processed suffixes. The fifth version uses the rule specific for each word type. The results for the nouns list are depicted in Table 9.

<table>
<thead>
<tr>
<th>Original</th>
<th>Step 1</th>
<th>Step 2</th>
<th>Step 3</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>تمساح</td>
<td>تمساح</td>
<td>حوت</td>
<td>حت</td>
<td>حوت</td>
</tr>
<tr>
<td>حوت</td>
<td>حوت</td>
<td>خد</td>
<td>خد</td>
<td>خد</td>
</tr>
<tr>
<td>خد</td>
<td>خد</td>
<td>خود</td>
<td>خود</td>
<td>خود</td>
</tr>
<tr>
<td>اصبع</td>
<td>اصبع</td>
<td>اصبع</td>
<td>اصبع</td>
<td>اصبع</td>
</tr>
<tr>
<td>اصبع</td>
<td>اصبع</td>
<td>اصبع</td>
<td>اصبع</td>
<td>اصبع</td>
</tr>
<tr>
<td>السن</td>
<td>السن</td>
<td>دجاج</td>
<td>دجاج</td>
<td>دجاج</td>
</tr>
<tr>
<td>دجاج</td>
<td>دجاج</td>
<td>قميص</td>
<td>قميص</td>
<td>قميص</td>
</tr>
<tr>
<td>زوجة</td>
<td>زوجة</td>
<td>لغات</td>
<td>لغات</td>
<td>لغات</td>
</tr>
<tr>
<td>لغات</td>
<td>لغات</td>
<td>الصين</td>
<td>الصين</td>
<td>الصين</td>
</tr>
<tr>
<td>بلد</td>
<td>بلد</td>
<td>دحراء</td>
<td>دحراء</td>
<td>دحراء</td>
</tr>
<tr>
<td>دحراء</td>
<td>دحراء</td>
<td>ضباب</td>
<td>ضباب</td>
<td>ضباب</td>
</tr>
<tr>
<td>ضباب</td>
<td>ضباب</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
</tr>
<tr>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
</tr>
<tr>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
<td>سكرتيرة</td>
</tr>
<tr>
<td>اجهاض</td>
<td>اجهاض</td>
<td>اجهاض</td>
<td>اجهاض</td>
<td>اجهاض</td>
</tr>
<tr>
<td>اجهاض</td>
<td>اجهاض</td>
<td>اجهاض</td>
<td>اجهاض</td>
<td>اجهاض</td>
</tr>
<tr>
<td>كرة</td>
<td>كرة</td>
<td>الكرة</td>
<td>الكرة</td>
<td>الكرة</td>
</tr>
</tbody>
</table>

Table 9 Results for nouns

The results for the noun category produced some undesirable output especially in case of plural. Some other letters were deleted by the processor as they were taken as suffix prepositions. The errors are highlighted using bold font.

The verbs list results are depicted in Table 10. The verb category result produced good results in case of two, three, four and five-letter stems. However, some letters were removed by the processor from trilateral verb stems from the middle and the end. Modification of rules is to be considered in the future.
The results for adverbs are depicted in Table 11. The results for the adverb category produced very good results in both muscular and feminine cases.

The results for adjectives are depicted in Table 12. Three undesirable results were produced in the adjective category of which two are in case of singular feminine and one in case of dual feminine. Additional rules have to be enhanced in the future to overcome this problem.
The summarization of the experiments is depicted in Table 13. As may be seen, the best results were achieved for adverbs. Two errors were produced for verbs, three errors for adjectives and six errors showed up in case of nouns.

<table>
<thead>
<tr>
<th>Word type</th>
<th>Errors</th>
<th>Word count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nouns</td>
<td>6</td>
<td>21</td>
</tr>
<tr>
<td>Verbs</td>
<td>2</td>
<td>13</td>
</tr>
<tr>
<td>Adverbs</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Adjectives</td>
<td>3</td>
<td>12</td>
</tr>
</tbody>
</table>

In this section simple rules for stemming Arabic words are described. Rule sets were defined for nouns, verbs, adverbs and adjectives. As may be seen from the experiments results, the rules were more successful in case of adverbs. As for nouns, verbs and adjectives, errors were produced. Most errors were occurred in case of suffix processing.

As can be seen from Table 10, some verbs such as (سأل) and (على) were over processed and the result was not successful. These are common problems in light stemming. This occurs when the word is not recognized in the stemming process as to which part of speech it belongs. To overcome this common problem, in the next section, a parser is used to distinguish the difference parts of speech before stemming.

2.4 Stemming Enhanced by Parsing

In the previous sections, some methods were used, for example, the vowel letter method [33]. This method is mainly dependent on syllabification of words and focused on splitting words according to vowel letters. Another approach [35] was a simple approach into stemming, where four categories of words were selected: nouns, verbs, adjectives and adverbs, from short news item texts. These two...
approaches produced some good results. However, in the light stemming method, two problems showed up.

The first problem had to do with parts of speech recognition. For example, the verb لعب (plays) starts with the letter yaa (ي). In Arabic, adding the suffix yaa (ي) used to change the word from its past/root form into its present form. When some rules are set to remove the letter yaa (ي) so to produce the root form of لعب, these rules always removed the letter yaa (ي) from other words with different parts of speech such as the noun اليمن (Yemen) where the letter yaa (ي) is part of the root form of the noun.

The second problem occurs within the sub-parts of speech. For example, when trying to remove the determiner aliflam (ال) (the definite article 'the') from common nouns as in الطالب (the student). The rules set remove the aliflam (ال) from all nouns including some proper nouns such as, المانيا (Germany) where aliflam (ال) is part of the original noun and not a determiner or a definite article.

For these reasons, in this section, the Stanford Online Parser [36] is used before stemming to better categorize the different parts of speech and later to be match the output words with an electronic dictionary.

This parser enhanced stemming method is executed using the following steps. First, the text is normalized and the input sentences are parsed using Stanford Online Parser. Next, the words in the text are grouped according to their parts of speech. And finally, the stemming is performed according to the rules set, and the results are matched with an electronic dictionary.

2.4.1 Stanford Online Parser

The Stanford Online Parser is a powerful parser that parses texts in three languages: Arabic, Chinese and English. This parser is using dependency grammar. The Arabic parts of the parser [36] is depending on the Penn Treebank project that was launches in 2001 in the University of Pennsylvania. According to their corpus documentation [20], the corpus is designed for those who study or use languages professionally or academically, as well as, for those who need to analyze text corpora in their work. The Penn Arabic Treebank is particularly suitable for language developers, computational linguists and computer scientists who are interested in various aspects of natural language processing. An example of parsing an Arabic sentence is shown in Figure 6, where we can see the different levels of parsing using dependency grammar.
Stanford Parser

Please enter a sentence to be parsed:

دخلت العالِميا الحرب العالمية في سنة 1939.

Your query

دخلت العالِميا الحرب العالمية في سنة 1939.

Tagging

VBD/NFP العالِميا/DINN الحرب/VBD/NFP في/IN سنة/NFP 1939./CD

Parse

(ROOT
  (S
    (VBD دخلت)
    (NNP العالِميا))
    (NNP الحرب))
    (IN في)
    (NFP سنة)
    (CD 1939.)))

Figure 6 Stanford Online Parser

Stanford Online Parser for Arabic language includes 27 different parts of speech. These parts of speech are shown in Table 14.
<table>
<thead>
<tr>
<th>ADJ_NUM</th>
<th>adjectival number</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>coordinating conjunction</td>
</tr>
<tr>
<td>CD</td>
<td>cardinal number</td>
</tr>
<tr>
<td>DT</td>
<td>determiner/demonstrative pronoun</td>
</tr>
<tr>
<td>DTJJ</td>
<td>determiner + adjective</td>
</tr>
<tr>
<td>DTJJR</td>
<td>determiner + Adjective, comparative</td>
</tr>
<tr>
<td>DTNN</td>
<td>determiner + singular common noun</td>
</tr>
<tr>
<td>DTNNP</td>
<td>determiner + singular proper noun</td>
</tr>
<tr>
<td>DTNNS</td>
<td>determiner + plural common noun</td>
</tr>
<tr>
<td>FW</td>
<td>Foreign word</td>
</tr>
<tr>
<td>IN</td>
<td>subordinating conjunction (FUNC_WORD) or preposition (PREP)</td>
</tr>
<tr>
<td>JJ</td>
<td>adjective</td>
</tr>
<tr>
<td>JJR</td>
<td>adjective, comparative</td>
</tr>
<tr>
<td>NN</td>
<td>common noun, singular</td>
</tr>
<tr>
<td>NNP</td>
<td>proper noun, singular</td>
</tr>
<tr>
<td>NNPS</td>
<td>common noun, plural or dual</td>
</tr>
<tr>
<td>NOUN_QUANT</td>
<td>noun quantifier</td>
</tr>
<tr>
<td>PRP</td>
<td>personal pronoun</td>
</tr>
<tr>
<td>PRPS</td>
<td>possessive personal pronoun</td>
</tr>
<tr>
<td>PUNC</td>
<td>punctuation</td>
</tr>
<tr>
<td>RB</td>
<td>adverb</td>
</tr>
<tr>
<td>RP</td>
<td>particle</td>
</tr>
<tr>
<td>UH</td>
<td>Interjection</td>
</tr>
<tr>
<td>VBD</td>
<td>Interjection perfect verb (**nb: perfect rather than past tense)</td>
</tr>
<tr>
<td>VBG</td>
<td>verb, gerund or present participle (perfect verb)</td>
</tr>
<tr>
<td>VBN</td>
<td>passive verb (**nb: passive rather than past participle)</td>
</tr>
<tr>
<td>VBP</td>
<td>imperfect verb (**nb: imperfect rather than present tense)</td>
</tr>
<tr>
<td>VN</td>
<td>-</td>
</tr>
<tr>
<td>WP</td>
<td>relative pronoun</td>
</tr>
<tr>
<td>WRB</td>
<td>wh-adverb</td>
</tr>
</tbody>
</table>

Table 14 Parts of speech in Stanford Online Parser

2.4.2 Stemming Rules

According to Stanford Online Parser for Arabic language, there are 27 different parts of speech as shown in Table 14. For the experiment in the present thesis, a number of rules are set for three main
parts of speech: nouns, verbs and adjectives. The transliterations of Arabic alphabets are used in the rules as per the transliterations shown in Table 7 (see section 2.3.1). The sets of rule for every part of speech or sub-part of speech are divided into a step or steps. Every step is to be implemented in the order of numbering. The full sets of rules start with specifications as follows:

**Specifications**

\( W \) – any word or its part (word refers to any part of speech in the rules: noun, verb, adjective, etc.)

- \([\text{arabic}]\) – letter
- \(\text{Ins}(x, y)\) – return true when \(x\) is anywhere in \(y\)
- \(|x|\) – length of word \(x\)
- \([x]\) – letter \(x\) is at the beginning of the word

**Nouns Rules:**

a) **DTNN**: determiner + singular common noun

   **Step 1**: If the noun starts with (لاا)، delete it and replace it with (ال)

   **Step 2**: If the noun starts with (ال) but not (لاا)، delete the (ال).

d) **NNPS**: common noun, plural or dual

   **Step 1**: delete (ات) or (ين) from the end of every noun if found.

   **Step 2**: If the word has less than five letters, add (ة) to the end.

   **Step 3**: delete any (و) found before the last added (ة).

**Verbs Rules:**

a) **VBD**: perfect verb (**nb: perfect rather than past tense**)

   **Step 1**: delete the (و) from the beginning of every verb if the remaining is not 2 letters.

   **Step 2**: delete (!) or (ت) or (هن) from any verb if it comes at the end.

   **Step 3**: if (!) or (س) are found at the end, replace them with (ى).

b) **VBN**: passive verb (**nb: passive rather than past participle**)

   **Step 1**: delete (ي) from the beginning if found.

   **Step 2**: delete (ت) from the beginning if found and replace it with (!) only if the number of letters after deleting the (ت) is 4.

c) **VBP**: imperfect verb (**nb: imperfect rather than present tense**)

   **Step 1**: delete (ت) or (تن) or (تي) from the beginning of every verb if found.

   **Step 2**: delete (ي) from the end of every verb if found.

   **Step 3**: delete (ن) or (يون) or (س) or (ه) from the beginning of every verb if found.

   **Step 4**: if the remaining letters of any verb, so far, is 2 letters, add (ى) to the end.

   **Step 5**: if any verb ends with (ى)، delete the (ى) and replace it with (ى)، and then add (!) to the beginning.

   **Step 6**: if any word starts with (ن)، add (!) to its beginning, only if there is (ت) anywhere in that word.

   **Step 7**: if any word ends with (ن) or (نن) or (نون)، delete them and replace them with (لاً).

   **Step 8**: if any word has more than 3 letters and starts with (تن)، delete the (تن) only and link the (ن) with the next letter.
**Adjectives Rules:**
a) **DTJJ**: determiner + adjective  
**Step 1:** delete (ال) from the beginning of every adjective if found.  
**Step 2:** delete (ة) from the end of every adjective if found.  
Step 3: if a word ends with (ِ), delete the (ِ) and add (ِ) to the beginning of the word.

### 2.4.3 Experiments and Implementation of Rules

The rules must be tested against real data. For this purpose, some news articles, from the BBC Arabic and Al Jazeera Arabic news portals were used. These articles are parsed by Stanford Online Parser. After the extraction of words according to their parts of speech, the rules are implemented.

The following Tables 15-22 show the rules for every category: nouns, verbs and adjectives respectively, and the extracted words preceded by the rule for every category. In the following Tables 15-22, repeated words are deleted and sample words of every part of speech or sub-part of speech are shown.

**Nouns Rules:**
a) **DTNN**: determiner + singular common noun  
**Step 1:**  
\[
|\text{alif}\text{Laam}\text{alif}{|W} \rightarrow |\text{alif}\text{Laam}|W
\]
**Step 2:**  
\[
|\text{alif}\text{Laam}\text{alif}{|W} \rightarrow |W
\]

| البحر | البنزين | البيئة | الجرف | الحد | الاتصالات | الشرق | السيناتورين | المنافذ | الشؤون | البر | المصدر | المصادر | الحكومة | البترول | المحيط | عالم | الاتصالات | الامن | الامور | الامراض | البترول | المعالجة | العناصر | المحيط | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | الطلب | السلامة | العناصر | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف | الأخطار | المحيط | المشاغل | الموارد | المياه | العواصف |
|--------|----------|--------|-------|-----|--------|-------|------------|--------|---------|-----|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Table 15 Processed nouns - DTNN: determiner + singular common noun |

b) **DTNPP**: determiner + singular proper noun  
**Step 1:**  
\[
|\text{alif}\text{Laam}|W \rightarrow W
\]
c) DTNNS: determiner + plural common noun
   Step 1:
   [alif laam]W -> W
   W[yaa nuun] -> W
   [W|alf taa|] -> [W|taamarbota|]
   W[waw nuun] -> W
   W[nuun] -> W

   الأمريكيين الأولويات الجمهوريون الديمقراطيون الشركات العشرات
   لتحقيق للحماية المحافظون المشتركون العمليات الملوثات المنتجات
   المندوبيون المنصات الولايات

   Table 17 Processed nouns - DTNNS determiner + plural common noun

d) NNPS: common noun, plural or dual
   Step 1:
   W[Alif taa] -> W
   W[yaa nuun] -> W
   Step 2:
   |W| < 5 -> W[taMarboota]
   Step 3:
   W[waaw][taMarboota] -> W[taMarboota]

   تغييرات تقديرات جماعات سنوات طبقات طنين عشرات عمليات
   كميات مجتمعات مقترحات منصات

   Table 18 Processed nouns - NNPS: common noun, plural or dual

Verbs Rules:
a) VBD: perfect verb (**nb: perfect rather than past tense)
   Step 1:
   |[waaw]W|>2 -> W
   Step 2:
   W[alif] -> W
   W[taa] -> W
   W[waaw nuun] -> W
   Step 3:
   W[alif haa] -> W[alifMaqsoora]
   W[ta haa] -> W[alifMaqsoora]
b) VBN: passive verb (***nb: passive rather than past participle)
Step 1:
[yaa]W -> W
[ta]W -> W
Step 2:
|W| = 5 -> [alif]W

Table 20 Processed verbs – VBN: passive verb

(c) VBP: imperfect verb (***nb: imperfect rather than present tense)
Step 1:
[ta]W -> W
[ta ta]W -> W
[yaa]W<3>W
[yaa]W<4>W
[yaa]W -> W
Step 2:
W[waaw] -> W
Step 3:
[nuun]W -> W
[waaw nuun]W -> W
[haa]W -> W
[haa alif]W -> W
Step 4:
|W| = 2 -> W[alifMaqsoora]
Step 5:
W[yaa] -> [alif]W[alifMaqsoora]
Step 6:
Step 7:
W[waaw laam] -> W[alif laam]
W[waaw laam waaw nuun] -> W[alif laam]
W[waaw nuun] -> W[alif laam]
Step 8:

<table>
<thead>
<tr>
<th>الامريكية</th>
<th>الدينية</th>
<th>التشريعي</th>
<th>التشريعي</th>
<th>التشريعي</th>
<th>التشريعي</th>
<th>التشريعي</th>
<th>التشريعي</th>
</tr>
</thead>
<tbody>
<tr>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
</tr>
<tr>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
</tr>
<tr>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
</tr>
<tr>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
</tr>
<tr>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
</tr>
<tr>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
</tr>
<tr>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
<td>ما</td>
</tr>
</tbody>
</table>

The noun rules results are depicted in Tables 23-26. The stemming results for the noun rules produced very good results in case of DTNNP and NNPS as shown in Tables 24, 25 and 26 respectively. As for DTNN and DTPNNS, some undesirable results occurred, mainly in case of...
(بالكلاب) in Table 23, where the preposition was linked by the Stanford Online Parser with the plural common noun.

Table 23 Processed nouns - DTNN: determiner + singular common noun

<table>
<thead>
<tr>
<th>بحر بنزين</th>
<th>بيئة جرف</th>
<th>دفاع سيناتورين</th>
<th>شرق شؤون</th>
<th>بر مصادر سابق</th>
</tr>
</thead>
<tbody>
<tr>
<td>حكومة</td>
<td>محيط</td>
<td>نفط</td>
<td>وصول</td>
<td>عالم احتياطي</td>
</tr>
<tr>
<td>تنقيب</td>
<td>عام</td>
<td>دولة</td>
<td>بار</td>
<td>مور ميال نسان</td>
</tr>
<tr>
<td>بالكلاب</td>
<td>طاقة</td>
<td>طلب</td>
<td>عواصف</td>
<td>مثبت</td>
</tr>
<tr>
<td>محيط</td>
<td>مشاغل</td>
<td>موارد</td>
<td>مياه</td>
<td>نفط</td>
</tr>
<tr>
<td>وحل</td>
<td>نواع</td>
<td>بيئة</td>
<td>عاصمة</td>
<td>خطط</td>
</tr>
<tr>
<td>فصال</td>
<td>درشة</td>
<td>سهل</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 24 Processed nouns - DTNNP: determiner + singular proper noun

| انترنت | برازيل | دوحة | عاج | مكسيك |

Table 25 Processed nouns - DTNNS determiner + plural common noun

<table>
<thead>
<tr>
<th>امريكي</th>
<th>اولوية</th>
<th>جمهوري</th>
<th>ديموقراطي</th>
<th>شركة</th>
</tr>
</thead>
<tbody>
<tr>
<td>عشيطة</td>
<td>شركة</td>
<td>عشرة</td>
<td>لتجهيز</td>
<td>للحماية</td>
</tr>
<tr>
<td>لمحافظ</td>
<td>مشتر</td>
<td>عاملة</td>
<td>ملوثة</td>
<td>منتجة</td>
</tr>
<tr>
<td>مندوب</td>
<td>منصة</td>
<td>ولاية</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ونائي</td>
<td>لنظام</td>
<td>نساعة</td>
<td>مجتمع</td>
<td>مقترح</td>
</tr>
<tr>
<td>منصة</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 26 Processed nouns - NNPS: common noun, plural or dual

The verb rules results are depicted in Tables 27-29. In spite of the few errors in VBD, the verbs rules produced very good results in case of VBD and VBN in Tables 27 and 28. However, in case of imperfect verb VBP in Table 29, we may notice higher error rate.

Table 27 Processed verbs - VBD: perfect verb

<table>
<thead>
<tr>
<th>اجري</th>
<th>اجرى</th>
<th>ادى</th>
<th>استخرج</th>
<th>أصبح</th>
<th>اقترح ايد</th>
</tr>
</thead>
<tbody>
<tr>
<td>أتى</td>
<td>بامكان</td>
<td>تسبب</td>
<td>تسرب</td>
<td>جعل</td>
<td>فقد كان</td>
</tr>
<tr>
<td>مضى</td>
<td>تزيد</td>
<td>قال</td>
<td>قدم</td>
<td>يضيف</td>
<td>يقول</td>
</tr>
</tbody>
</table>

Table 28 Processed verbs – VBN: passive verb

| وجد | سحب | رفع | استخدم | درج |

Table 29 Processed verbs - VBP: imperfect verb
The results for the adjectives’ rules are depicted in Table 30. All the rules made for adjectives category DTJJ produced successful results.

### 2.4.5 Results and Discussion

In this experiment, rules were set for parts of speech. The words from the corpora where sorted out and categorized using Stanford Online Parser for Arabic language. The Stanford Online Parser identifies 27 different parts of speech. However, in this experiment, the rules set were for three main parts of speech: nouns, verbs and adjectives. Every rule for every part of speech was divided into one or more steps.

The stemming results for the noun rules produced very good results in case of DTNNP and NNPS as shown in Tables 24, 25 and 26 respectively. As for DTNN, some undesirable results occurred, mainly in case of (بالكلاب) where the preposition was linked by Stanford Online Parser with the plural common noun. Another example in the same category can be seen in case of dual common noun (سيناتورين). We may see in this category the highest error rate mainly because in Arabic names can take three forms, single, dual and plural. In this regard, more rules needed to focus on the plural and dual suffixes. Linked prepositions to nouns also occur in case of DTNNS as in (لتحقيق) and (لحماية) where the preposition (ل) is linked to the plural common nouns. Interestingly enough, none of the nouns “investigation” or “protection” are plural nouns, but were parsed wrongly by the parser, most probably because they were linked to prepositions.

All the rules made for adjectives category DTJJ produced successful results and they are depicted in Table 30.
The verb rules results are depicted in Tables 27-29. In spite of the few errors in VBD, the verbs rules produced very good results in case of VBD and VBN. However, in case of imperfect verb VBP, we may notice higher error rate. As an example, if we compare the stemming results in case of (تيوجه) and (شير) in Table 29 with the original words before processing in Table 21, we may notice that all the above imperfect verbs are linked to personal pronouns by the parser. This is always one of the main reasons for stemming errors. The challenge here is to minimize the errors pertaining to linked personal pronouns with verbs or affixes in general. The rules has to be optimized in the future.

Although the results obtained were not all very good, the overall evaluation of these rules proved that the rules are applicable and produced good results. This study was conducted for three parts of speech and the results were very encouraging. Thus, future work will include setting rules for all parts of speech in order to be used for data compression of Arabic text files. These rules will also be applied on a larger scale corpora such as Al Watan 2004 (20291 documents) and Al Khaleej (5690 documents – nearly 3 million words). The summary of parser enhanced stemming result are depicted in Table 31.

<table>
<thead>
<tr>
<th>Part of Speech</th>
<th>Errors</th>
<th>Word count</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTNN: determiner + singular common noun</td>
<td>15</td>
<td>44</td>
</tr>
<tr>
<td>DTNNP: determiner + singular proper noun</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>DTTNS: determiner + plural common noun</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>NNPS: common noun, plural or dual</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>VBD: perfect verb (***nb: perfect rather than past tense)</td>
<td>3</td>
<td>21</td>
</tr>
<tr>
<td>VBN: passive verb (***nb: passive rather than past participle)</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>VBP: imperfect verb (***nb: imperfect rather than present tense)</td>
<td>9</td>
<td>48</td>
</tr>
<tr>
<td>DTIJ: determiner + adjective</td>
<td>0</td>
<td>33</td>
</tr>
</tbody>
</table>

Table 31 Summary of parser enhanced stemming results

2.5 Summary of Chapter Two

Section 2.2 describes a novel approach by setting simple set of rules for syllabification of Arabic language based on splitting vowel letters. The efficiency of the compression was compared using the syllable approach with characters, words, 2-grams and 3-grams approaches on two groups: short and long files. The best result for short files were achieved by character based approach. For long files, the best results were achieved by 2-grams approach, but the character approach and syllable approach were also efficient.

In section 2.3 simple rules for stemming Arabic words were described. Rule sets were defined for nouns, verbs, adverbs and adjectives. The rules were more successful in case of adverbs. As for nouns, verbs and adjectives, errors were produced. Most errors were occurred in case of suffix processing.
In section 2-4, the experiment for light stemming was enhanced by a parser. Experiment rules were set for three parts of speech: nouns, verbs and adjectives. The words from the corpora were sorted out and categorized using Stanford Online Parser. Every rule for every part of speech was divided into one or more steps. The stemming results for the noun rules produced very good results in case of DTNNP and NNPS. As for DTNN, some undesirable results occurred, mainly where the preposition was linked by Stanford Online Parser with the plural common nouns. More rules needed to focus on the plural and dual suffixes. Linked prepositions to nouns also occur in case of plural common nouns DTNNS where prepositions are linked to these nouns which leads the parser to wrongly categories the nouns as plural while they are in fact singular. All the rules made for adjectives category DTJJ produced successful results. The verb rules produced very good results in case of VBD and VBN. However, in case of imperfect verb VBP, we may notice higher error rate mostly when verbs are linked to personal pronouns or affixes in general.
3 Data Compression Approach for Plagiarism Detection

Similarity detection is considered an essential part of document processing. It covers a wide area including spam detection and plagiarism detection. This section is devoted to creating a plagiarism detection tools for Arabic and Czech.

The need for plagiarism detection tools is growing with the growing number of natural language documents that are written day by day in schools and universities all over the world. The growing number of these documents include, students’ assignments, Masters’ and PhD theses and dissertations. While some students resort to cut and paste methods, some other students use different ways of plagiarism including changing the sentence structure, paraphrasing and replacing the lexical meaning of words with synonyms. These constantly require new and more sophisticated tools to detect plagiarism.

Formerly plagiarism can be described as a document \( d \) and a potential plagiarism source \( D \) where detection in the external approach is to identify pairs of texts \((s, s')\), from \( D \) and \( D' (d \in D') \) respectively, where \( s \) and \( s' \) can be similar. However, this similarity may differ as it could be an exact match, paraphrased, summarized, restructured or even semantic similarity, which includes using different words or different languages.

This chapter tries to implement an initialization of a similarity measurement based on Lempel Ziv comparison algorithms and its modifications to show its efficiency for detecting plagiarism of Arabic and Czech texts. However, in the present thesis, only some part of Lempel Ziv algorithm is used. In addition to that, this chapter also extends into implementing this technique to semantic textual detection of plagiarized texts. The results of these experiments were published in Soori et al. [58], [67] and [78].

3.1 Plagiarism Detection by Compression

The present method uses Lempel-Ziv compression method. The main principle of this method is the fact that for the same sequence of data the compression becomes more efficient. Lempel-Ziv compression method is one of the most currently used methods in data compression in various kinds of data like texts, images, audio [47], [48]. The present method is inspired by Chuda et al. [49]. In their method, they tried to detect similarity for Slovak text. In [49] they decided to use and adapt the principle of creating a dictionary from which they obtain the parameters for evaluating similarity. They created a dictionary since it is part of the encoding process of Lempel-Ziv method as a table where the row contains two components: index or code-word, as well as, the corresponding character string. As far as the prefix is concerned, this dictionary should fulfil the condition, if some string of characters is saved in the dictionary, then all its prefixes are saved in the dictionary as well. However, the main divergence in their approach has to do with insertion of whole words rather than just
characters in Lepel-Ziv. Another key point is that, as the input text comes, the first initialization of the dictionary is not needed since the dictionary is initialized continuously by new words. In addition to that, the most important divergence in their method [49] is that they monitor the count of phrases in the dictionary and they do not deal with counting of the output string, but with the input text, where the text is divided into words. For example, if the input word is not in the dictionary, then it is added there, but if it is found in the dictionary, then the next word is added, and so on. This creates a phrase. If this phrase is in the dictionary, then the next word from the input is added to it. However, if the phrase is not in the dictionary, then it is added to the dictionary, and the parameter count of the compression is incremented. The process is repeated over and over till the end of the input. Mainly their approach deals with words and phrases rather than characters.

The technique used in this chapter is inspired by Chuda et al. [49]. This technique is described in sections 3.2 and 3.3 respectively, for textual plagiarism detection in Arabic and Czech. However, the idea is extended in section 3.4 with a synonyms thesaurus and a stemmer to detect textual semantic plagiarism.

3.1.1 Methods of Plagiarism Detection

With the huge amount of online and offline written data, plagiarism detection has become an eminent need for various fields of science and knowledge. Various context based plagiarism detection methods have been published in the literature.

Traditional methods of Plagiarism detection use manual observation and comparison of documents but these methods are no longer viable due to the tremendous number of documents available online in various fields of science and knowledge on daily basis.

Primarily, plagiarism detection methods were developed to detect plagiarism of programming codes in languages such as Java, C++, python, etc. Later, the idea was extended to include text plagiarism in English and other natural languages.

Mainly, research in this area uses different computational and linguistic techniques within wide range of areas such as data mining [63], information retrieval [64], similarity measures [65], cross lingual semantic detection techniques [66], etc.

Each of these techniques used approaches the problem from a different perspective. One of the most widely used methods is context based method that depends on similarity measurement between two documents. This method compares the fingerprints of documents where fingerprints are made by hashing subsets of documents. In this technique, the Winnowing algorithm [38] is widely used. It depends on the selection of finger prints of hashes of k-grams. It is based on the idea of finding the similarity of certain lengths of small partial matches where t is the guarantee threshold and k is the
noise threshold. Basically the idea is based on two conditions: the substring found is at least as long as the threshold, and if there is any match that is shorter than the noise threshold $k$, then it is not detected.

Another method is Stanford Copy Analysis Mechanism (SCAM) [39]. This method is based on a copy detection server which is made of a repository and a chunker. Documents are broken up into small chunks (sentences, words...etc.) and, after that, registered in the repository. Each chunk is sorted out and labeled. After that, every new unregistered document is broken up into chunks and compared with the registered documents already in the repository. The method is based on the idea that smaller units of chunks increase the probability of finding similar texts. This method uses Relative Frequency Mode (RFM).

Text similarity measurement can also be used for data compression. Platos et al. [42] utilized similarity measurement in their compression algorithm for small text files. Some other attempts to use the similarity measurement for plagiarism detection include Prilepok et al. [43] who used this measurement to detect plagiarism of English text.

On the other hand, some machine techniques have demonstrated high accuracy as in the case of k-nearest neighbors (KNN), artificial neural networks (ANN) and support-vector machine learning (SVM).

KNN depends on the idea of $X$ member and its relation to its nearest neighbor $n$. Winnowing algorithm [38] is one of the widely used algorithms in this regard. It depends on the selection of fingerprints of hashes of k-grams. The idea is based on the optimization of results by trying the variations of the $n$ value and how distant or remote is $x$ to its neighbors. In this case, plagiarism is decided on whether or not $x$ falls into the neighboring members, and the number of neighbors. One of the drawbacks of this technique is its slow computational processing because it requires calculation of all the remote and distant members.

The ANN learning classifier functions by modeling the way the brain works with mathematical models. These two are commonly implemented to serve the purpose by plagiarized and non-plagiarized texts. It resembles the mechanism of human neuron with other neurons that work in another brain layer. In addition to the efficiency of this technique to detect plagiarism, it has also proven very effective in other areas such as speech synthesis [47] and image compression [48].

Another machine techniques in this regard is SVM. SVM is a statistical classifier that deals mainly with text. This technique tries to find the boundaries between the plagiarized and non-plagiarized text by finding the threshold for these two classes. The boundary set is called support vectors.

Other methods use a hybrid of computational and linguistic techniques. For example, Khan et al. [68] who use a method based on phrases taken from first sentence of paragraphs based on the co-occurrences of 5 tokens and eventually evaluated by precision, recall and $f$-measure.
Some other approaches tend to deal with semantic plagiarism detection where the system is able to detect rewording, restructuring and using synonyms and cross lingual plagiarism [69, 70, 71, 72].

Plagiarism detection methods extend the exact match and semantic detection into a third dimension where the text is examined intrinsically in terms of the consistency of style, vocabulary, usage of short or long sentences/ phrases, and the complexity of structure as in [73, 74, 75, 76].

The most widely used writer’s style method is a stylometry statistical method [40], which is based on the idea that every writer has her/ his own style which can be detected by dividing the documents into smaller parts and comparing the linguistic features such as the length of text (sentences, paragraphs and chapter), frequency of use of punctuations, parts of speech, use of function words, richness of the vocabulary used, etc. In these intrinsic methods [41], the detection is performed within the same document and does not take into account outside references. The drawback of stylometry approach comes when the writer has more than one style then this approach can detect false-positive plagiarism.

3.1.2 Arabic Plagiarism Detection PAN Shared Task 2015

According to the PAN2015 report [87], recently very few studies have been conducted on Arabic external text detection [58, 81,83,84,85,86,] and even fewer on intrinsic methods [82], but these studies have been evaluated using different corpora and strategies, which makes the comparison between them very difficult. This makes it not easy to draw a clear conclusion on the performance [87]. The AraPlagDet PAN2015 was held for the first time in 2015 [87]. In PAN 2015, participants had two shared tasks: external plagiarism detection and intrinsic plagiarism detection. Eight runs have been submitted and tested on a corpora developed for the track. In this shared task, they used a corpus made of a combination of two corpora: the Corpus of Contemporary Arabic CCA, and another corpus manually collected from hundreds of random documents from Arabic Wikipedia. According to this shared task report, the evaluation corpus they used is considered medium-sized in comparison with the PAN competition corpora [89, 90, 88]. In this shared task, four methods by four responding participants were reported where all the methods had granularity of more than 1.05 which, according to their report, is not a good result in comparison with PAN 2014 competition results [91]. In PAN 2015, only three participant’s methods were able to detect reshuffling. The participants who could not detect reshuffling mainly used finger printing approach. They also found out that semantic similarity was the most difficult task to perform where manual paraphrasing was involved. Many false positive cases were detected in the AraPlagDet PAN 2015 report, especially in cases where repetition of phrases occur even within the same document. The report also found false positive cases between the manually randomly selected Arabic Wikipedia corpus and the CCA corpus.
3.1.3 Creating Dictionaries out of Document

Creating a dictionary is one task in the encoding process for Lempel-Ziv 78 method [50]. The dictionary is created from the input text, which is split into separate words. If a current word from the input is not found in the dictionary, this word is added. In case the current word is found, then the next word from the input is added to it. This eventually creates a sequence of words. If this sequence of words is found in the dictionary, then the sequence is extended with the next word from the input in a similar way. However, if the sequence is not found in the dictionary, then, it is added to the dictionary with the increased number of sequences property. The process is repeated until the end of the input text is reached. In the present experiment, every paragraph has its own dictionary.

3.1.4 Similarity of Text

The similarity measure approach is one of the most used approaches to detect plagiarism. To a certain extent, this method resembles the methods used in information retrieval in that it decides the retrieval rank by measuring the similarity to a certain query.

Generally speaking, we may rank the similarity-based plagiarism detection techniques as: text based techniques that depend on cosine and fingerprints as in [53, 54], graph similarity that depend on ontology as in [55, 56] and line matching as in bioinformatics [57].

The main idea of text similarity lies in measuring the distance between two texts. In this case, the ideal scenario would be when the distance between the texts is metric. Formally, the distance is defined as a function over Cartesian product over a set $S$ with nonnegative real value where the metric $d$ is a distance that satisfies four conditions as follows:

\[
\begin{align*}
  d(x, y) &\geq 0 \\
  d(x, y) &= 0, \text{ if and only if } x = y \\
  d(x, y) &= d(y, x) \\
  d(x, z) &\leq d(x, y) + d(y, z)
\end{align*}
\]

The conditions mentioned above as: (1), (2), (3) and (4) are called: axiom of non-negativity, axiom of identity, axiom of symmetry and the triangle inequality respectively. This kind of a definition is valid for any metric, e.g. Euclidean Distance. However, applying it on document or data similarity is a herculean task.

3.1.5 Comparison of the Documents

Mainly, there are two properties in a dictionary where the comparison between two documents takes place: a list of word sequences and a word-sequences-count in the list. A dictionary is created for each
of the compared files. The next phase involve comparison of these dictionaries where the main property for comparison lies in the number of common sequences in these dictionaries. The number, in this case, is represented by the parameter in the following formula which represents a similarity metric where the comparison is taking place between two documents.

$$SM = \frac{sc}{\min(c_1, c_2)}$$

(5)

Where:

- $sc$ – Count of common word sequences in both dictionaries.
- $c_1, c_2$ – Count of word sequences in dictionary of the first or the second document.

Here the interval is the $SM$ value which satisfies the condition, If $SM = 1$, then the documents are equal, but if the value is $SM = 0$, then the documents are totally different.

3.2 Text Similarity Based on Data Compression in Arabic

Data compression can be used for textual similarity measurement. There are some data compression algorithms for similarity of text files. Some of these use compression methods as discussed in section 3.1.1 above. In this section, this technique is used to detect plagiarism of Arabic texts. The results of the experiments in this section were published in Soori et al. [58].

When using this technique for Arabic language, many language specific features has to be born in mind. Unlike languages that use Roman characters, Arabic is written from right to left and has twenty eight alphabet letters (three vowels and twenty five consonants). Hence, Arabic plagiarism detection tools require considering language specific features in detecting text similarity. Arabic alphabets are much different from Roman alphabets, which are naturally not agglutinated. In addition to that, Arabic has eight short vowels and diacritics (see Figure 1 above). Accordingly, Arabic text requires normalization of diacritics. According to Habash [12], since diacritical problems in Arabic occur so infrequently, they are removed from the text by most researchers. Typists normally ignore putting them in a text, but in case of texts where they exist, they are pre-normalized - in value - to avoid any mismatching with the dictionary or corpus during text processing.

3.2.1 Experimental Setup

In this experiment, Al-Khaleej-2004 corpus of Arabic texts is used. Al-Khaleej corpus contains 5690 documents. It is divided into four topics: local news, economy, sports and international news.
For this experiment, the local news category was chosen. This dataset contains only documents in Arabic language. In this experiment, suspicious document collection was needed to test the suggested approach.

These document were created as follows: 150 false suspicious and 100 source documents from Khaleej-2004 corpus by using a small tool that was designed to create false suspicious documents.

### 3.2.2 False Suspicious Documents Creator Tool

The purpose of this tool is to create false suspicious documents. The tool is designed following these steps. All the documents from the corpus were split into paragraphs, and each paragraph is labeled with new line mark for a quick reference of its position in the corpus. From this paragraphs list, two separate collections of documents were created. The first is a source document collection, and the other is the suspicious collection. For the source documents collection, one - five paragraphs were selected from the list of paragraphs. These paragraphs were added to a newly created document and marked as source document one. This step was repeated for all 100 documents. The collection contains 252 distinct paragraphs.

The process of creating the suspicious documents is very similar to process of creation the source documents. From each suspicious document one - five paragraphs were randomly selected. The tool randomly selects the paragraphs. Each document contains some paragraphs from the source document and some unused paragraphs. This step is repeated for all 150 documents. For creating a collection of suspicious documents, 330 paragraphs were used: 159 paragraphs from source documents and 171 unused paragraphs from Al-Khaleej corpus.

For each created suspicious document, an XML description file was created. This file contained information about the source of each paragraph in the used corpus, i.e. starting and ending points, byte and file name. This step is repeated for all 150 documents. The created dataset contained 150 suspicious parts (24 plagiarized parts and 126 unplagiarized parts), and 100 source documents, which were considered as the testing data for the algorithm.

### 3.2.3 The Experiment

The comparison of the whole documents where only a small part of the document may be plagiarized is useless, because other characteristics and the whole text of the new document may hide the characteristic of the plagiarized part. Therefore, the documents were split into paragraphs. Paragraphs were chosen because it is believed here that they are better than sentences for the reason they contain more words and should not be affected by stop words, such as, preposition, conjunctives, etc. The paragraphs were separated by an empty line between them. A dictionary for each paragraph was created from the source document, according to the method described in section 3.1.3. As a result of
the fragmentation of the source documents, 252 paragraphs and their corresponding dictionaries were obtained. These dictionary paragraphs serve as reference dictionaries that were used to compare the dictionary paragraphs with the suspicious documents created earlier.

The set of suspicious documents was processed in a similar way. Each suspicious document was fragmented into paragraphs. After fragmentation of the suspicious documents, 330 paragraphs were obtained. Then, a corresponding dictionary was created using the same algorithm without removing diacritics and stop words. This dictionary was compared with the dictionaries from the source documents. To improve the speed of the comparison, only subset of dictionaries were chosen for comparison because comparing one suspicious dictionary to all source dictionaries consumes too much time. The subset is chosen according the size of a particular dictionary with tolerance rate of ±20%. For example, if the dictionary of the suspected paragraphs contains 122 phrases, all dictionaries with the number of phrases between 98 and 146 were chosen. This 20% tolerance significantly improves the speed of the comparison. Moreover, it is believed here that this tolerance percentage does not affect the overall efficiency of the algorithm. This way, the paragraph with the highest similarity to each paragraph of the tested paragraph is picked up.

3.2.4 Stop Words Removal

Stop words removal has been proven to increase the accuracy level of text similarity detection. For this reason, in this experiment, stop words were removed from the used texts.

As a source of stop words, and in order to acquire a long list of stop words, two lists of Arabic stop words were used in this experiment: The classic Shereen Khoja’s list of stop words from Khoja Stemmer 2004 [51] and the Basic Arabic Stemmer [52].

The list in Khoja Stemmer 2004 [51] contains 168 stop words. The April 2013 release of The Basic Arabic Stemmer stop words list [52] is also used in this experiment. This release contains 1300 stop words. After comparing the two lists, 42 common words between the two stop-word lists were found. The removal of stop words and diacritics was used to increase the similarity detection and the speed of the algorithm. The present algorithm is modified in a way so that after the fragmentation of the text, all stop words are removed from the list of paragraphs and the text is normalized by removing diacritics, then the rest of them are processed by the same algorithm.

3.2.5 Results

For the purpose of this experiment, a document is considered, plagiarized if all its parts are found in the attached annotation XML file, partially plagiarized if not all its parts are detected successfully in the annotation XML file, i.e., 3 out of 5 parts in the annotation XML file are found, and non-plagiarized, if none of its parts are detected in the annotation XML file.
<table>
<thead>
<tr>
<th></th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plagiarized documents</td>
<td>90/ 126 71.42%</td>
</tr>
<tr>
<td>Partially plagiarized documents</td>
<td>36/ 126 28.58%</td>
</tr>
<tr>
<td>Non-plagiarized documents</td>
<td>24/ 24 100.00%</td>
</tr>
</tbody>
</table>

Table 32 Results of success rate PlagTool ar

In this experiment 71.42% of the documents were found to be plagiarized documents in 90 out of 126 documents, 28.58% of the documents were found to be partially plagiarized in 36 out of 126 documents, and 100% of the documents were found to be non-plagiarized in 24 out of 24 documents. This results are depicted in Table 32.

In case of partially plagiarized documents, suspicious paragraphs could be found in another document, or paragraphs with higher similarity as a paragraph with the same content. This case can occurs if one of the paragraphs is shorter than the other paragraph. To illustrate this case, a brief example is mentioned in the form of extracts from the processed text as follows.

This first paragraph comes from one of the suspicious documents collections. The paragraph consists of two sentences. After removing stop words and diacritics we get 28 word sequences as shown in Figure 7.

This paragraph’s similarity with the first paragraph is $SM = 1.0$.

The second paragraph, shown in Figure 8, contains the same words and sentences. This paragraph has different position of sentences and clauses in comparison with the first paragraph in Figure 7. This is illustrated with the dispositioned parts in red and blue colours. This paragraph’s similarity with the first paragraph is $SM = 1.0$. 
The third paragraph, as shown in Figure 9, is more or less a similar paragraphs. It contains the same meaning, but with different words and sentence construction. This paragraph’s similarity with the first paragraph shown earlier in Figure 7 is $SM = 0.4$.

3.2.6 Visualization of Documents Similarity in PlagTool ar

In PlagTool ar, three visual representations were used for the visualization of paragraph similarities. These visual representations should give the user a simple quick overview of the results of the suspicious document.

The first visual representation is represented by a line chart. This chart shows the similarity for each suspicious paragraph in the document. The user may easily see which part of the document is plagiarized and the number of the plagiarized parts. Higher similarities represent paragraphs with more plagiarized content. This is depicted in Figure 10.

The second visual representation is a histogram of document similarity as shown in Figure 11. The histogram shows brief overview of how many paragraphs have the same similarity and how many parts of the suspicious document are plagiarized.
The last visual representation is used to easily visualize the similarity in the form of colored text highlights. Three colours were used for visualization. The red color means that the paragraph has a similarity rate greater than 0.2. The orange color shows the paragraphs with lower similarity ranging between 0 and less than 0.2. This paragraph shares only few similar words with the source paragraphs. The green text means that the paragraph was not found in the source text and, accordingly, it is not plagiarized.

3.2.7 Conclusion

In this section, the similarity detection algorithm at hand was applied on a real dataset with Arabic texts. This confirmed the ability to detect plagiarized parts of the documents with the removal of stop words and diacritics, which improved the processing speech. The algorithm for similarity measurement based on the Lempel Ziv compression algorithm and its dictionaries was very efficient in detection of the plagiarized parts of the Arabic text documents with 150 suspicious parts (24 plagiarized parts and 126 unplagiarized parts), and 100 source documents, which were considered as the testing data for the algorithm. All plagiarized documents in a dataset were marked as plagiarized and in most cases all plagiarized parts were identified, as well as, their original source. The findings of this experiment were published in Soori et al. [58].

3.3 Utilizing Text Similarity Measurement for Data Compression to Detect Plagiarism in Czech

Research to find plagiarized texts came as an answer to the massive increase in written data in various fields of science and knowledge and in the urge to protect intellectual property of authors. Many text plagiarism methods have been utilized for this purpose as discussed earlier in section 3.1.1.

Similarity detection covers a wide area of research including spam detection, data compression and plagiarism detection. In addition to that, text similarity is a powerful tool for documents processing. On the other hand, plagiarism detection includes many texts documents such as, students’ assignments, postgraduate theses and dissertations, reports, academic papers and articles. This section
investigates the viability and applies an initialization of similarity measurement based on Lempel-Ziv comparison algorithms and its modifications for detecting plagiarism of Czech texts.

As mentioned above, text similarity measurement can also be used for data compression. Platos et al. [42] utilized similarity measurement in their compression algorithm for small text files. Some other attempts to use the similarity measurement for plagiarism detection include Prilepok et al. [43] who used this measurement to detect plagiarism of English texts and Soori et al. [58] who used the technique to detect plagiarism of Arabic texts. However, the idea of [43] and [58] was originally inspired by Chuda et al. [49]. In this section, the method is utilized to detect plagiarism of Czech texts. The results of the experiments in this section were published in Soori et al. [67].

3.3.1 Experimental Setup

In this experiments, a local small corpus of Czech texts from the online newspaper iDNES.cz [60] was used. This corpus contains 4850 words. The topics were collected from various news items including sports, politics, social daily news and science. For this experiment, suspicious document collection were needed to test the suggested approach. The collection of 100 false suspicious and 50 source documents were created from a locally compiled corpus for this experiment from iDNES.cz by using a small tool that was designed to create source and suspicious documents as described in the next subsection.

3.3.2 Document Creator Tool

The purpose of the document creator tool is to create some documents to be used as testing data. The tool is designed as follows. All the documents were split from the corpus into paragraphs. Next, each paragraph is labeled with a new tagged line for quick reference that indicates the location of the paragraph in the corpus. After that, two separate collections of documents were created from this list of paragraphs: a source document collection and a suspicious document collection. Finally, one to five paragraphs were selected randomly from the list of paragraphs. The first group is added to a newly created document and marked as source documents. This step is repeated for all the 50 documents. The collection obtained contained 120 different sets of paragraphs.

The same steps were repeated for the group of created suspicious documents where from each suspicious document one to five paragraphs were randomly selected. Each document contained some paragraphs from the source document created earlier, and some unused paragraphs. These steps were repeated for all the 100 documents. To create the list of suspicious documents, out of 227 paragraphs, 109 paragraphs from the source documents and 118 unused paragraphs from our local corpus were used. For each of the created suspicious document, an XML description file is made. This file included
information about the source of each paragraph in our corpus, its starting and ending point, byte and file name. This step is repeated for all the 100 documents.

3.3.3 The Experiment

The idea of comparing a whole document where only a small part of it is plagiarized is futile for the reason that the plagiarized part may only be a small chunk hidden among the total volume of text in the new document. Hence, splitting the documents into paragraphs is essential. Paragraphs were chosen because, it is believed here, that they retain some better characteristics than sentences in terms of carrying more sense words, and they are not affected by stop words, *i.e.* preposition, conjunctions, articles, etc. The paragraphs were separated by a line separator and a dictionary was created for each paragraph from the source document, according to the steps mentioned earlier. From the fragmentation of the source documents, 120 paragraphs and their corresponding dictionaries were made. These dictionary paragraphs were used as reference dictionaries to be compared against the created suspicious documents.

Similarly, the suspicious document sets were processed in the same manner. The suspicious document were fragmented into 227 paragraphs. Next, a corresponding dictionary was created using the same algorithm, without removing stop words in this step. After that, this dictionary is compared against the dictionaries from the source documents. To accelerate the comparison speed, only a subset of the dictionaries were chosen for comparison for the reason that comparing one suspicious dictionary to all source dictionaries is time consuming. The subset of dictionaries were chosen as per the size of a particular dictionary with a tolerance rate of 20%. For instance, if the dictionary of the suspected paragraphs contains 122 phrases, the dictionaries chosen were all the dictionaries with number of phrases between 98 and 146. The 20% tolerance rate improves the comparison speed significantly. It is believed here that this tolerance rate does not affect the overall efficiency of the success rate of the algorithm. This way, the paragraph with the highest similarity to each paragraph of the tested paragraph is spotted.

3.3.4 Stop Words Removal

Removal of stop words increases the accuracy level of text similarity detection. In this experiment, stop words were removed from the text. For this purpose, a list of Czech stop words from Semantikoz [60] was used. The list of the stop word used contained 138 stop words. The algorithm used here is modified so that after the fragmentation of the text, all stop words are removed from the list of paragraphs and the remaining words are processed by the same algorithm.
### 3.3.5 Results

For the purpose of this experiment, a plagiarized document refers to a document of which the PlagTool cz managed to find all its plagiarized parts from the attached annotation XML file, a partially plagiarized document refers to a document of which the PlagTool cz managed to find parts of it to be plagiarized in the attached annotation XML file (for instance, three out of five parts of the original document), and a non-plagiarized document refers to a document of which the PlagTool cz did not manage to find any plagiarized parts of it in the annotation XML file.

<table>
<thead>
<tr>
<th>Document Type</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plagiarized documents</td>
<td>76/92 82.60%</td>
</tr>
<tr>
<td>Partially plagiarized documents</td>
<td>16/92 17.40%</td>
</tr>
<tr>
<td>Non-plagiarized documents</td>
<td>8/8 100.00%</td>
</tr>
</tbody>
</table>

Table 33 Results of success rate PlagTool cz

In this experiment 82.60% of the document were found to be plagiarized documents in 76 out of 92 documents, 17.40% of the documents were found to be partially plagiarized in 16 out of 92 documents, and 100.0% of the documents were found to be non-plagiarized in 8 out of 8 documents. This results are depicted in Table 33.

### 3.3.6 Visualization of Document Similarity in PlagTool cz

The PlagTool cz application is divided into four parts. Three parts show document visualization and one part shows the result. In this tool, three visual representations were used for showing paragraph similarity. These visual representations help the user to identify the suspicious documents and identify their locations.

The first representation is a line chart. The line chart shows the similarity for each suspicious paragraph in the document. The user may easily locate the plagiarized parts of the document and the number of the plagiarized parts where higher similarities represent paragraphs with more plagiarized content, as shown in Figure 12 where paragraphs two and three are plagiarized.
The second representation is a histogram of document similarity. The histogram depicts how many paragraphs are similar and accordingly how many paragraphs are plagiarized. This is shown in Figure 13.
The last visual representation is used to show the similarity in the form of colored highlighted texts. Five colors have been used. The black color is used to show the source document. The red color indicates that the paragraph has a similarity rate greater than 0.2. The orange color shows the paragraphs with lower similarity ranging between less than 0.2 and 0, where the paragraph has only few similar words with the source document. The green text indicates that the paragraph is not found in the source document and accordingly it is not plagiarized. The blue color indicates the location of the plagiarized paragraph in the source document. A snapshot of PlagTool, with the color functions, are depicted in Figure 14.

![Figure 14: PlagTool cz: Coloured visualization of similarity](image)

As shown in Figures 12, 13 and 14, the left table shows all the paragraphs from the selected document, starting and ending byte (file location) in the document, the detected highest similarity rates and the name of the document with the highest similarity rate.

The middle part opens suspicious document with color highlighting. The color highlighting helps the user to easily identify the plagiarized paragraphs.

The part on the right side shows the content of paragraphs with the highest detected similarity to source documents. This helps the user to compare and see difference between the paragraph from the suspicious document and the paragraph from the source document.

The bottom part displays information about the results using three tabs. The first two tabs show various charts of the selected suspicious document. These charts give the user information about the overall rate of plagiarism for the selected document. The last tab shows information about the suspicious document from the testing data set. It also shows which paragraphs are plagiarized, and
from which document they were copied. This information is useful for validating the success rate of the detected plagiarism.

3.3.7 Conclusion

In this section, the similarity detection algorithm at hand was applied on Czech text with a locally complied multi-domain corpus taken from the online version of the daily newspaper, iDNES.cz. The algorithm for similarity measurement based on Lempel Ziv compression algorithm and its modifications, and its dictionaries with the removal of stop words, was very efficient in detecting the plagiarized parts of the documents in the testing data, with 50 original documents of non-plagiarized parts, and 100 suspicious documents. The documents were generated so that every document is made of 1-5 paragraphs. The suspicion rate in the documents was randomly chosen from 0.2% to 0.8%. All plagiarized documents in the dataset were marked as plagiarized and, in most cases, all the plagiarized parts were identified, as well as, their original versions, with a success rate of 82.6%. This confirmed the ability of this algorithm to detect the plagiarized parts of the documents, as well as the viability of this approach for Czech language. Future studies may enhance the efficiency of the algorithms by including combined and more sophisticated methods. The findings of this experiment were published in Soori et al. [67].

3.4 Semantic and Similarity Measure Methods for Plagiarism Detection of Students’ Assignments

For more than a decade, research on plagiarism detection has been focusing on developing methods to detect plagiarism of exact matches in texts. However, the notion of plagiarism has expanded where paraphrasing may extend the notion of cutting and pasting a text into paraphrasing texts. According to Merriam Webster’s Dictionary [62], plagiarism is ’the act of using another person’s words or ideas without giving credit to that person’ [62]. Since ideas may be reworded, this makes rewording without mentioning the source text -where the idea was taken from- an act of plagiarism. As mentioned earlier in section 3.1.1 above, semantic plagiarism may take many forms including changing the structure of sentences (restructuring) and replacing words with their synonyms or rewriting (paraphrasing) [69, 70, 71, 72]. This section integrate a similarity measure technique previously used for text compression in [58, 67] along with a Czech synonyms thesaurus and stemmer by Pala et al. [79, 80] to detect rewording and restructuring of Czech language texts. The results of the experiments in this section were published in Soori et al. [78].

3.4.1 Rationale, Training Data and Thesaurus

The rationale behind the method proposed in this section is that for someone to rewrite (paraphrase) a text, one needs to replace the original words either with different words that carry -more or less- the
same meaning, *i.e.*: synonyms, or a variant of a synonyms. For example, one may replace the word, *important* with synonyms, such as, *significant, consequential, essential, relevant*, etc. On the other hand, paraphrasing may also involve replacing the word at hand with a word that carries a negated opponent meaning, *i.e.*, negated-antonym. For example, one may replace the word, *important* with negated-antonyms such as, far from *insignificant, not inconsequential, not unessential, not irrelevant*, etc. Paraphrasing also involves restructuring of the sentence. This work involves synonyms detection. Antonyms detection is not included in this work.

For our experiment, a corpus of 100GB is used. The corpus contains BA, MA, and PhD students’ assignments, semester works and theses, from different majors. After removal of metadata and sorting out the corpus into majors, semesters and academic years, the corpus size dropped to 77GB with 13.671 files of plain text, out of which the computer science major students’ works were used to apply our experiment on. The total size of the extracted testing data used for our initial experiment was 1.98 GB of plain text.

For our experiment, we integrated the similarity measurement used earlier in sections 3.2 and 3.3 with a Czech structured thesaurus and a stemmer called, Slovník českých synonym (Vocabulary of Czech synonyms) by Pala & Vsiansky [79, 80]. These thesaurus and stemmer are used currently by Apache OpenOffice for their Czech language pack [22]. The thesaurus contains 386,891 tokens, 166,331 words and it is in UTF-8 format.

### 3.4.2 Short Texts Experiment

The experiment starts by giving a random short text (175 words), taken from an online version of the daily newspaper iDNES.cz [60] to three Czech native speakers. This short text is called (v1) and it is depicted in Figure 15.
Černý rok pro Erste. Majitel České spořitelny čeká rekordní ztrátu
3. července 2014 19:42
Rakouská finanční skupina Erste Group Bank, do které patří Česká spořitelna, letos očekává čistou ztrátu až 1,6 miliardy eur (44 miliard Kč). Můžou za to zejména problémy v Maďarsku a Rumunsku. Podle agentury Reuters to vyplývá z prognózy, kterou rakouská firma ve čtvrtkve zveřejnila.
Billboard společnosti Erste
Finanční skupina Erste Group Bank očekává letos rekordní ztrátu (ilustrační snímek). | foto: Jiří Benák, iDNES.cz

Autoři: ČTK, jj

The three candidates were asked to re-write this short text in their own words and were not given any specific hints on how to rewrite the text. Accordingly, each of them rewrote the given short text in her/ his own words. After receiving the three different versions from the three Czech native speakers, their three different versions were called, (v2), (v3) and (v4) respectively, as depicted in Figures 16, 17 and 18.

Špatná zpráva pro Erste. Vlastníka České spořitelny čeká rekordní propad
Rakouský bankovní dům Erste Group Bank, kterému patří také Česká spořitelna, letos očekává ztrátu až 1,6 miliardy eur (44 miliard Kč). Důvodem jsou hlavně potíže na Maďarském a Rumunském trhu. Agentura Reuters to vyvozuje z prognózy, kterou rakouské společnost publikovala ve čtvrtkve.
Plakát společnosti Erste.
Bankovní dům Erste Group Bank očekává letos negativní výsledek hospodaření (ilustrační snímek). | foto: Jiří Benák, iDNES.cz.


Figure 15 Training data (ORIGINAL TEXT) v1

Figure 16 Testing data (SEMANTIC MATCH ONE) v2
Černý rok pro Erste. Majitel České spořitelny čeká rekordní ztrátu

3. července 2014 19:42

Rakouská finanční skupina Erste Group Bank, jejíž součástí je Česká spořitelna, v tomto roce očekává čistou ztrátu až ve výši 1,6 miliardy eur. Hlavním důvodem jsou problémy v Maďarsku a Rumunsku. Tento výsledek byl zveřejněn agenturou Reuters na základě ve čtvrtek zveřejněné prognózy.

V loňském roce čistý zisk Erste Group Bank klesl na 61 miliónů EUR z 483,5 miliónů, kterých dosáhl v předchozím roce.

Prognóza vychází z předpokladu, že se Maďarská vláda bude snažit přenést na banky část ztrát vzniklých u devizových úvěrů kvůli oslabení domácí měny.

Výsledky Bulharské divize budou podle Erste negativně ovlivněny snahou Bulharské centrální banky o zvýšení rezerv kvůli omezení špatných úvěrů v bankovním systému.

Erste je jedním z klíčových hráčů na bankovním trhu střední a východní Evropy. Skupina působí, kromě České Republiky a Rakouska, také na Slovensku, v Rumunsku, Maďarsku, Chorvatsku a Srbsku.
Černý rok pro Erste. Majitel České spořitelny čeká rekordní ztrátu 3. července 2014 19:42
Společnost předpokládá, že její aktivity v Maďarsku negativně zasáhne snaha tamní vlády přenést na banky část ztrát, které Maďarům u devizových úvěrů přineslo oslabení domácí měny.
Billboard společnosti Erste
Finanční skupina Erste Group Bank očekává letos rekordní ztrátu (ilustrační snímek). | foto: Jiří Benák, iDNES.cz
Loni se čistý zisk Erste Group Bank propadl na 61 milionů eur z 483,5 milionu eur v předchozím roce.
Výsledky bulharské divize podle Erste negativně ovlivní zvýšení rezerv související se snahou tamní centrální banky o omezení špatných úvěrů v bankovním systému.
Erste je klíčovým hráčem na bankovním trhu střední a východní Evropy. Vedle České republiky a Rakouska působí v Rumunsku, na Slovensku, v Maďarsku, v Chorvatsku a v Srbsku.
Rakouská finanční skupina Erste Group Bank, do které patří Česká spořitelna, letos očekává čistou ztrátu až 1,6 miliardy eur (44 miliard Kč). Můžou za to zejména problémy v Maďarsku a Rumunsku. Podle agentury Reuters to vyplývá z prognózy, kterou rakouská firma ve čtvrtek zveřejnila.
Autoři: ČTK, jj

After that, another short text was added for validation reason called (v6). (v6) had the same word count as (v1) but the text contained totally different words from (v1) and from the other versions, (v2), (v3), (v4) and (v5).

For matching the words in the suspicious documents (v2, v3, v4, v5) with the source document (v1), only a first level -main entry- from the synonyms thesaurus was used. In other words, words in the source document were compared with their synonyms in the thesaurus, but not with the synonyms of their synonyms, i.e., sub-entries in the electronic thesaurus.

Before searching the synonyms in the dictionary, every word was connected to its basic form of the word using the stemmer. This helped to find the synonyms in the dictionary where all words are in their basic forms.

Eventually, the results were obtained as depicted in Table 34. As may be noticed from the results, the similarities of rewritten documents, (v2), (v3) and (v4) with (v1) are above 0.5%. However, the similarity of two different documents with the same word count (v1) and (v6) are close to 0.1%.
### Table 34 Results of document similarities for short texts

<table>
<thead>
<tr>
<th></th>
<th>v1</th>
<th>v2</th>
<th>v3</th>
<th>v4</th>
<th>v5</th>
<th>v6</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>1</td>
<td>0.556</td>
<td>0.632</td>
<td>0.673</td>
<td>0.827</td>
<td>0.086</td>
</tr>
<tr>
<td>v2</td>
<td>0.556</td>
<td>1</td>
<td>0.505</td>
<td>0.528</td>
<td>0.565</td>
<td>0.086</td>
</tr>
<tr>
<td>v3</td>
<td>0.632</td>
<td>0.505</td>
<td>1</td>
<td>0.621</td>
<td>0.621</td>
<td>0.054</td>
</tr>
<tr>
<td>v4</td>
<td>0.673</td>
<td>0.528</td>
<td>0.621</td>
<td>1</td>
<td>0.658</td>
<td>0.097</td>
</tr>
<tr>
<td>v5</td>
<td>0.827</td>
<td>0.565</td>
<td>0.621</td>
<td>0.658</td>
<td>1</td>
<td>0.108</td>
</tr>
<tr>
<td>v6</td>
<td>0.086</td>
<td>0.086</td>
<td>0.054</td>
<td>0.097</td>
<td>0.108</td>
<td>1</td>
</tr>
</tbody>
</table>

#### 3.4.3 Pitfall of Using Data Compression Similarity Measure for Plagiarism Detection

As we may notice in Table 34, in the experiment on short texts, the similarity rate between (v1) and (v5) is always more than 80%, but never 100%, in spite of the fact that both documents contain exactly the same words and word count. This theoretically illogical percentage is seen by us as a slight drawback to our combined technique. This drawback may be explained as follows.

The notion of data compression following Lempel-Ziv 78 method [50] mainly involves creating a dictionary as a major part of the encoding process. The input text is split into separate words. In case a word is not found in the dictionary, then it is added, but if the word is found, then the next word from the input is added. This creates a sequence of words. If this sequence is not found in the dictionary, then it is added, otherwise, it is extended with the next word from the input.

However, as a result of changing the arrangement of sentences and paragraphs in (v5), the sequence of words created yields totally different chains/sets of sequences. This affected the overall percentage of matches in the present experiment because the sequences generated from (v1) were not totally the same as the sequences generated from (v5). To overcome this drawback, in the future work, this method will be fine-tuned by modifying the algorithm so to be able to detect rearranged sequences of sentences and paragraphs.
3.4.4 Long Texts Experiment of Students’ Assignments

After obtaining the results for the short texts, this combined method was applied on the compiled corpus of students’ assignments. The following steps were applied for each document.

In the first step, all punctuation marks such as commas, full stops, exclamation marks, etc. were removed. After removing the punctuation marks, the input text was converted into chains of words.

In the next step, stop words were removed from this series and all words were converted to their basic forms. Every word in the series of words was looked up for the synonyms in the synonym dictionary for its several meanings. The synonyms were put in an entry of our internal dictionary of words. This entry was built from the dictionary every time, whenever the current processed word is not found in the dictionary. In case, the current word occurs in more than one entry in the dictionary, the entry with the higher count meanings was chosen.

After finding the synonyms to all words from the series of input words, every text was converted into a series of IDs in the dictionary entries. Each dictionary had its own ID. In the phase of measuring the similarity, the present approach deals with these IDs. The similarity is measured by applying Chuda et al.’s method [49], as described in section 3.1 above. However, instead of words, the present method works with the IDs of the dictionary entries.

The threshold between the plagiarized and non-plagiarized documents is set in the present experiment to 0.3. It is understood here that this value is higher than other current plagiarism detection tools where the value is set to only 0.2. However, it is set with a higher percentage because, in the present experiment, bibliography lists were not removed. More importantly, in addition to that, in students’ assignments, one may always find a lot of commonly used words such as technical terms, scientific definitions, etc.

For the purpose of this experiment, if the similarity of two documents is higher than the threshold set as 0.3, the two documents are considered as similar and, accordingly, suspicious. If the similarity is below or equal to 0.3 the two documents are not the same. Table 35 presents the results for long texts in a summarized way, despite the big number of files processed and the huge similarity matrix.

<table>
<thead>
<tr>
<th>No. of Plagiarized Documents</th>
<th>Total Document Count</th>
<th>Plagiarized in %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results</td>
<td>211</td>
<td>883</td>
</tr>
</tbody>
</table>

Table 35 Results of document similarities for long texts

In this experiment, the settings and preprocessing of text was the same as in the short text experiment. In the dataset, 211 similar documents were found with plagiarism rate of 23.9% out of the total number of 883 documents as depicted in Table 35. This rate appears to be very high. However, when looking closer to the results, one may notice that this number is doubled. This can be explained
as follows. In the present experiments, all documents in the training data were compared to all documents in the corpus. Hence, the tuples of documents were counted twice. For example, if documents A and B are compared and their similarity is above 0.3, then according to the metric definition, mentioned under Section 3.1 above, the two documents in the opposite order have the same similarity. This is the reason behind counting the documents twice.

3.4.5 Conclusion

This experiment presents a semantic plagiarism detection method for Czech texts. The method combines a data compression technique used earlier for similarity detection, a synonymy thesaurus and a stemmer. Contrary to the conventional 0.2% threshold used in plagiarism detection tools, the detection threshold for the present experiment was set as 0.3 because the present work was conducted on raw text. Two experiments were ran on two corpora. The first was a small corpus and contains one original text, three rewritten (paraphrased) texts by three Czech native speakers, one more version with sentences and paragraphs reordered, and one last version that contained totally different words but with the same word count as the original text. In the experiment on the short corpus, the proposed method was able to distinguish well between plagiarized and non-plagiarized texts in six versions. The second corpus was a long text corpus made of 1.98 GB of plain text and contained students’ assignments, dissertations and theses. Out of 883 document, 211 suspicious documents were detected with 23.9% suspicious tuples of documents. These documents may be considered as plagiarized. However, after scanning the documents manually, it appeared that these documents contained a number of similar words -usually- technical terms, definitions, references in bibliography lists, etc. The results obtained were good in spite of the pitfall in case of version five (v5) where a text was made with the exact words but reordered sentences and paragraphs. Still, in (v5) the method managed to detect similarity and the detection rate exceeded the set 0.3 threshold with a detection rate of 0.82. The proposed approach proved to be useful to detect semantically similar (reworded/ paraphrased) documents. Additional modifications are needed to fine-tune the obtained results by building a file-specific stop word list. In addition to that, a method is needed to detect exact match documents with rearranged sentences and paragraphs, as well as, removal of bibliography lists and other standard text templates often used in parts of students’ assignments, dissertations and theses. Future semantic plagiarism detection work is planned to involve antonyms thesaurus so to detect negated antonyms, cross lingual plagiarism, back-translated texts, and implementing a writer’s style method to detect consistency within a same document.
3.5 Summary of Chapter Three

In section 3.2, a similarity detection algorithm using text similarity for Arabic language is applied on Al-Khaleej corpus with 150 suspicious parts (24 plagiarized parts and 126 un-plagiarized parts), and 100 source documents, which were considered as the testing data for the algorithm. The findings of the study show that the similarity measurement based on Lempel Ziv comparison algorithms is efficient for the plagiarized part of the Arabic text documents with a successful rate of 71.42%. All plagiarized documents in a dataset were marked as plagiarized and in most cases all plagiarized parts were identified, as well as, their original source. This confirmed the ability to detect plagiarized parts of the documents. Stop words and diacritics were removed to increase the similarity detection and the speed of the algorithm. The algorithm for similarity measurement based on the Lempel Ziv compression algorithm modifications and its dictionaries proved to be very efficient in detection of the plagiarized parts of documents. The findings of this experiment were published in Soori et al. [58]. Future studies will try to improve the efficiency of the algorithms by combining more sophisticated computation, statistical and linguistics hybrid detection methods.

Section 3.3 attempts to apply data compression based similarity method for plagiarism detection. The method has been used earlier for plagiarism detection for Arabic language. In this section this method is utilized for Czech language text by using a locally compiled multi-domain Czech corpus from iDNES.cz, with 50 original documents with non-plagiarized parts, and 100 suspicious documents. The documents were generated so that every document could have from 1 to 5 paragraphs. The suspicion rate in the documents was randomly chosen from 0.2% to 0.8%. The findings of the study show that the similarity measurement based on Lempel-Ziv comparison algorithms with its modifications is efficient for the plagiarized part of the Czech text documents with a success rate of 82.60%. Future studies may enhance the efficiency of the algorithms by including combined and more sophisticated methods. The findings of this experiment were published in Soori et al. [67].

Section 3.4 presents a semantic plagiarism detection method for Czech texts. The method combines a data compression technique used earlier for similarity detection, a synonymy thesaurus and a stemmer with a detection threshold set as 0.3, and using two corpora: one short text corpus taken from the an online daily newspaper and a second corpus of 1.98 GB of plain text that contained students’ assignments, dissertations and theses. The method was tested first on short texts. Then, the method was applied on the bigger corpus of students’ assignments. The results on short texts showed accurate results to detect paraphrased texts of semantic similarity, but lower accuracy was detected in case of texts with identical words but rearranged sentences and paragraphs. The results of the experiment conducted on the long texts corpus of students’ assignment showed a semantic plagiarism rate of 23.9%. However, after manual scanning of documents, some noise results occur as a result of
frequent encounter of the same technical terms and scientific definitions and references in bibliography lists in different documents. These results will be fine-tuned and optimized in the future by building a file-specific stop word list. The findings of this experiment were published in Soori et al. [78].
4 Conclusions

This chapter includes the main conclusions drawn from this work. It also mentions some ideas for future work. The present work aimed at presenting rules for syllabification of Arabic text using vowel letters, and a light stemming method for data compression, and showing the viability of these rules for syllabification and stemming.

Section 2.2 describes a novel simple set of rules for syllabification of Arabic language based on splitting vowel letters. The algorithm uses only three basic rules, but it achieves very good results in the identifying of syllables. This algorithm was used in text compression for two types of input files. The short files contain up to 12,000 characters. The long files are more than 800,000 characters. The efficiency of the compression was compared using the syllable approach with characters, words, 2-grams and 3-grams approaches on these two groups of files. The best result for short files were achieved by character based approach, as was expected. For long files, the best results were achieved by 2-grams approach, but the character approach and syllable approach were also efficient. In the future, a correction must be done especially in regard to the non-word and non-syllable handling and more sophisticated syllabification algorithm must be tested. The findings of this experiment were published in Soori et al. [33]

Section 2.3 describes very simple light rules for stemming of Arabic words. Two of these rules are universal, i.e. they are applicable to any word category, and one rule for each of the four categories of part of speech. In this section simple rules for stemming Arabic words are described. Rule sets were defined for nouns, verbs, adverbs and adjectives. In the experiment results, the rules were more successful in case of adverbs. As for nouns, verbs and adjectives, errors were produced. Most errors were occurred in case of suffix processing. Some verbs such as (سأل) and (على) were over processed and the result was not successful. These are common problems in light stemming. This occurs when the word is not recognized in the stemming process as to which part of speech it belongs. To overcome this common problem, in the next section, a parser is used to distinguish the difference parts of speech before stemming. The findings of this experiment were published in Soori et al. [35]

In section 2.4, the Stanford Online Parser is used before stemming to better categorize the different parts of speech and later to be match the output words with an electronic dictionary. This parser enhanced stemming method is executed using the following steps. First, the text is normalized and the input sentences are parsed using Stanford Online Parser. Next, the words in the text are grouped according to their parts of speech. And finally, the stemming is performed according to the set rules, and the results are matched with an electronic dictionary. The Stanford Online Parser identifies 27 different parts of speech. However, in this experiment, the rules set were for three main parts of speech: nouns, verbs and adjectives. Every rule for every part of speech is divided into one or more steps. The stemming results for the noun rules produced very good results in case of DTNNP and
NNPS. As for DTNN, some undesirable results occurred, mainly in case of (بالكلاب) where the preposition was linked by Stanford Online Parser with the plural common noun. Another example in the same category is in case of dual common noun (سنياتورين). This category has the highest error rate. In this regard, more rules needed to focus on the plural and dual suffixes. Linked prepositions to nouns also occur in case of DTNNS as in (لـ) where the preposition (تحقيق) is linked to the plural common nouns. Interestingly enough, none of the nouns (تحقيق) “investigation” or (حماية) “protection” are plural nouns, but were parsed wrongly by the parser, in this case because of the linked preposition.

All the rules made for adjectives category DTJJ produced successful results. In spite of the few errors in VBD, the verbs rules produced very good results in case of VBD and VBN. However, in case of imperfect verb VBP, higher error rate may be noticed. As an example, if we compare the stemming results in case of (تيحها), (شير), (ختبئى) and (تشر) with the original words before processing, we may notice that all the above imperfect verbs are linked to personal pronouns. This is always one of the main reasons for stemming errors. The challenge here is to minimize the errors pertaining to linked affixes and in this case to personal pronouns with verbs. The rules has to be optimized in the future.

Although the results obtained were not all very good, the overall evaluation of these rules proved that the rules are applicable and produced good results. This study was conducted for three parts of speech and the results were very encouraging. The findings of this experiment were published in Soori et al. [77] Future work will include setting rules for all parts of speech in order to be used for data compression of Arabic text files. These rules will also be applied on a larger scale, for instance, Al Watan 2004 and Al Khaleej corpora.

This thesis also aimed at finding applicable methods for plagiarism detection tools for Arabic and Czech texts by implementing a similarity measurement based on a compression algorithms and showing its efficiency for detecting textual plagiarism. It also aimed at extending this work to include a semantic plagiarism detection method of texts including rewording, restructuring and using synonyms.

In section 3.2, a similarity detection algorithm based on Lempel Ziv compression and its modifications is presented for Arabic textual plagiarism detection. The technique has been used earlier. However, it is used here with removing stop words and Arabic diacritics to increase similarity detection and the speed of the algorithm for Arabic text. The technique is applied on Al-Khaleej corpus with 150 suspicious parts (24 plagiarized parts and 126 unplagiarized parts), and 100 source documents, which were considered as the testing data for the algorithm. The findings of the study show that the initial part of the similarity measurement based on Lempel Ziv comparison algorithms and its modifications is efficient for the plagiarized part of the Arabic text documents with a successful rate of 71.42%. All plagiarized documents in the dataset were marked as plagiarized and, in most cases, all plagiarized parts were identified, as well as, their original source. The algorithm for
similarity measurement based on the Lempel Ziv compression algorithm modifications and its dictionaries proved to be very efficient in detection of the plagiarized parts of documents of the Arabic text. The findings of this experiment were published in Soori et al. [58]. Future studies can improve the efficiency of the algorithms by combining more sophisticated, computational and linguistic, hybrid detection methods.

Section 3.3 attempts to apply an initial part of data compression based similarity method for plagiarism detection. The technique has been used earlier in section 3.2 for Arabic language. However, in this section, the similarity detection algorithm at hand was applied on Czech language with a locally compiled multi-domain corpus taken from the online version of the daily newspaper, iDNES.cz. The algorithm for similarity measurement based on Lempel Ziv compression algorithm and its dictionaries and the removal of stop words, was very efficient in detecting the plagiarized parts of the documents in the testing data, with 50 original documents of non-plagiarized parts, and 100 suspicious documents. The documents were generated so that every document is made of 1 - 5 paragraphs. The suspicion rate in the documents was randomly chosen from 0.2% to 0.8%. All plagiarized documents in the dataset were marked as plagiarized and, in most cases, all the plagiarized parts were identified, as well as, their original versions, with a success rate of 82.6%. This confirmed the ability of this algorithm to detect the plagiarized parts of the documents, as well as the viability of this approach to be applied on Czech text. Future studies may enhance the efficiency of the algorithms by including combined and more sophisticated methods. The findings of this experiment were published in Soori et al. [67].

Section 3.4 presents a semantic plagiarism detection method for Czech texts. The method applies a data compression technique used earlier for textual similarity detection in sections 3.2 and 3.3. However, in this section, the technique is combined with a synonymy thesaurus and a stemmer. Contrary to the conventional 0.2% threshold used in plagiarism detection tools, the detection threshold for the present experiment was set as 0.3 because the present work was conducted on raw text.

Two experiments were ran on two corpora. The first was a small corpus and contains one original text, three rewritten (paraphrased) texts by three Czech native speakers, one more version with sentences and paragraphs reordered, and one last version that contained totally different words but with the same word count as the original text. In the experiment on the short corpus, the proposed method was able to distinguish well between plagiarized and non-plagiarized texts in six versions.

The second corpus was a long text corpus made of 1.98 GB of plain text and contained students’ assignments, dissertations and theses. Out of 883 document, 211 suspicious documents were detected with 23.9% suspicious tuples of documents. These documents may be considered as plagiarized. However, after scanning the documents manually, it appeared that these documents contained a number of similar words -usually- technical terms, definitions, references in bibliography lists, etc.
The results obtained were good in spite of the pitfall in case of version five (v5) where a text was made with the exact words but reordered sentences and paragraphs. Still, in (v5) the method managed to detect similarity and the detection rate exceeded the set 0.3 threshold with a detection rate of 0.82.

The present approach proved to be useful to detect semantically similar (reworded/paraphrased) documents. Additional modifications are needed to fine-tune the obtained results by building a file-specific stop word list.

In addition to that, a method is needed to detect exact match documents with rearranged sentences and paragraphs, as well as, removal of bibliography lists and other standard text templates often used in parts of students’ assignments, dissertations and theses. Future semantic plagiarism detection work is planned to involve antonyms thesaurus so to detect negated antonyms, cross lingual plagiarism, back-translated texts, and implementing writer’s style method to detect consistency within a same document. The findings of this experiment were published in Soori et al. [78].

Hussein Khaled Hussein Soori
5 References


Author’s Publications

Publications Related to the Thesis

Conference Papers


Soori, Hussein, Michal Prilepok, Jan Platoš, Eshetie Berhan, and Vaclav Snasel. "Text Similarity Based on Data Compression in Arabic." In AETA 2013: Recent Advances in Electrical Engineering and Related Sciences. Springer Berlin Heidelberg, 2014. ISSN 1876-1100, pages 211-220. SJR = 0.122.


Publications Not Related to the Thesis

Journal Article

Conference Paper