جـامـعـة الـشـرق الأوسـط
MIDDLE EAST UNIVERSITY

Hybrid Technique for Arabic Text Compression
(تقنية هجينة لضغط النصوص العربية)

By

Enas Mahmoud Abu Jrai

Supervisor

Dr. Arafat Awajan

A Thesis Submitted in Partial Fulfillment of the

Requirements for the Master Degree in

Computer Science

Faculty of Information Technology

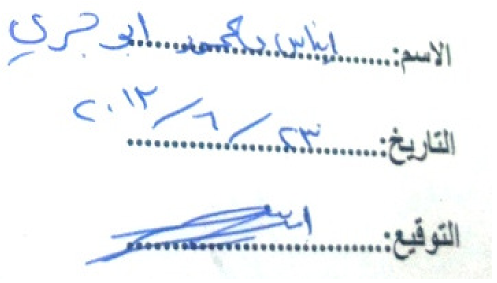
Middle East University

June, 2013

AUTHORIZATION FORM

إقرار تفويض

أنا ايناس محمود ابو جري أفوض جامعة الشرق الأوسط للدراسات بتزويد نسخ من رسالتي للمكتبات، أو الهيئات أو المؤسسات أو الأفراد عند طلبها



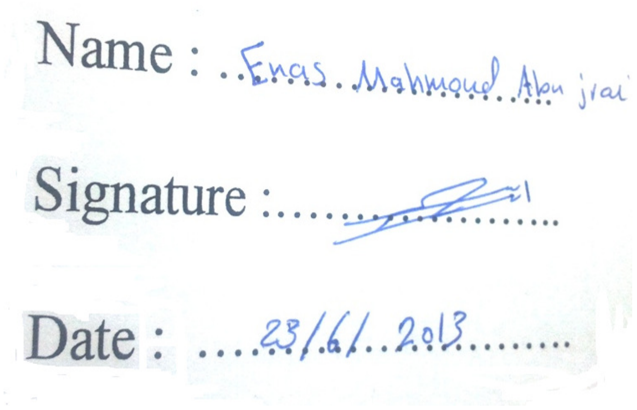
الاسم: ايناس محمود ابو جري

التاريخ: ٢٣/٦/٢٠١٢

التوقيع: [illegible]

AUTHORIZATION FORM

I am Enas Mahmoud Hussien Abu jrai ,Authorize the Middle East University to supply a copy of my Thesis to libraries, establishments or individuals upon their request



Name : ..Enas..Mahmoud..Abu jrai

Signature :..........[illegible]..........

Date : ....23/6/..2013........

Examination Committee Decision

This is to certify that the thesis entitled " Real Time Arabic Translation System For Signboard Images Based on Printed Character Recognition" was successfully defended and approved on 10/6/2013 .

Examination Committee Members signature

Dr .Ahmad Kayed
Department of computer science
Middle East university
[illegible]

Dr. Arafat Awajan
Department of computer science
Princess Sumaya University for Technology
[illegible]

[illegible]
Dr .Jalal Al atom
Department of Management Information Systems
Princess Sumaya University for Technology

DEDICATION

(وَإِذْ تَأَذَّنَ رَبُّكُمْ لَئِنْ شَكَرْتُمْ لَأَزِيدَنَّكُمْ) سورة إبراهيم (7)

Almighty Allah says “And remember! Your Lord caused to be declared (publicly): "If ye are grateful, I will add more (favors) unto you”. So all praise is for Allah, the exalted, for his favors that cannot be counted. I dedicate this work to my parents, my uncle Ahmad, my brothers, my friends, and all those who helped, supported, and taught me.

ACKNOWLEDGMENTS

I would like to thank my father and my mother for their continuous support during my study. I also would like to thank my great supervisor Dr.Arafat Awajan for his support, encouragement, proofreading of the thesis drafts, and for helping me throughout my studies, putting me in the right step of scientific research. I would like to thank the Information Technology Faculty members at the Middle East University, specially Dr. Waled Awajan. I would like also to thank Dr. Al Bara'a Awajan. I would also like to thank all of my family members , and all of my friends specially Shoroq, Dr.Ahed Sharari, Dr.Hadeel Al Khattab, Shaden, Tamador, Ahlam, Maryam, Liqaa, Hajr.

Table Of Contents

List of Tables

List of Figures

Abstract

Compression techniques have gained great importance in the field of communications and information technology in order to reduce the growing size of data and to increase the data transmission speed between computers and over the networks. In addition to these aims, text compression techniques aim at using the compressed text in text oriented application such as searching, summarizing, and information retrieval.

In this work, the morphological feature of Arabic language was used to build a new technique of Arabic Text Compression that satisfies two objectives: obtaining good compression rate and having compressed text structure that can be used to extract information from the compressed file instead of the original file. In addition to reducing the text size and increasing the transmission speed, these techniques speed up the information extraction, terms of searching processes.

Different common compression techniques Lempel–Ziv-Welch (LZW) and Burrows Wheeler Transform (BWT) were tested on Arabic texts and their results were compared in term of compression ratio. LZW was the best one for all categories of the Arabic texts, then BWT techniques. Features of the Arabic language were studied and then exploited to improve the performance of these techniques. The fact that Arabic letters have a single case was used to improve the performance of LZW. Through exploitation of the unused locations of the dictionary, the results showed that the compression ratio for the proposed method

was better than all the other techniques. The morphological features of the Arabic language had been used as a pre-processing step for data compression.

As a result of this study, a new hybrid technique has been suggested with better results in term of compression rate and text researchable files. This technique works in phases. In the first phase, the text file is split into four different files using a Multilayer analyzer. In the second phase, each one of these four files is compressed using BWT. Different compression techniques were investigated and tested at the level of each one of the four files. BWT technique was found to be suitable for all text files in terms compression ratio. The integration of the Multilayer model with LZW to compress all the files reduced the compression time.

الخلاصة

تقنية هجينة لضغط النصوص العربية

اكتسبت تقنيات الضغط أهمية كبيرة في مجال الاتصالات وتكنولوجيا المعلومات للحد من تزايد حجم البيانات وزيادة سرعة نقل البيانات بين أجهزة الحاسوب وشبكات نقل المعلومات. كما تهدف تقنيات ضغط النص لاستخدام النص المضغوط في التطبيقات النصية مثل البحث، والتلخيص، واسترجاع المعلومات.

في هذا العمل، استخدمت الميزة الصرفية للغة العربية لبناء تقنية جديدة لضغط النصوص العربية, و التي تلبي هدفين: الحصول على معدل ضغط جيد , تخزين النص المضغوط وفق بنية يمكن استخدامها لاستخراج المعلومات من الملف المضغوط بدلا من الملف الأصلي . بالإضافة إلى تخفيض حجم النص وزيادة سرعة نقل الملفات، هذه التقنيات هي الأسرع في استخراج المعلومات و عمليات البحث.

اختبرت تقنيات ضغط مختلفة والاكثر شيوعا مثل (LZW,BWT) على النصوص العربية وتم مقارنة نتائجها من حيث معدل نسبة الضغط. حيث أشارت النتائج أن تقنية LZW حققت أفضل نسبة ضغط لجميع فئات النصوص العربية, ثم BWT.

استغلت هذه الدراسة خصائص اللغة العربية لتحسين أداء هذه التقنيات. كما استخدمت هذه الدراسة حقيقة أن الحروف العربية لديها حالة واحده وذلك لتحسين أداء LZW . الطريقة المقترحة المسماة ب(LZWA) تستغل المواقع الغير المستخدمة في القاموس. حيث حققت هذه الطريقة نسبة ضغط أفضل من كل التقنيات السابقة.

ونتيجة لهذه الدراسة، فقد اقترحت تقنية هجينة جديدة تستخدم احد المحللات الصرفية لمعالجة النص قبل تطبيق احد تقنيات الضغط التقليدية, حيث حققت نتائج جيدة من حيث الحصول على نسبة ضغط أفضل وإيجاد ملفات نصية قابلة للبحث. تعمل هذه التقنية على مراحل, في المرحلة الأولى: يتم تقسيم الملف النصي إلى أربعة ملفات مختلفة باستخدام محلل متعدد الطبقات. في المرحلة الثانية: يتم ضغط كل واحد من هذه الملفات الأربعة باستخدام تقنيات مختلفة. تم إجراء العديد من الاختبارات لتحديد الضاغط المناسبة لكل ملف من الملفات الأربعة لتطبق كل منها على حدا. فكانت تقنية BWT الأفضل والمناسبة لجميع الملفات. إن دمج نموذج متعدد الطبقات مع تقنية LZW لضغط ملفات النصوص العربية يخفض من الوقت المستغرق لضغط الملف.

Chapter One

Introduction

1.1. Overview.

Data compression has important applications in the areas of data transmission and data storage. It aims at reducing the size of data in order to improve the speed of transmission and reduce the size that is needed for the storage.

Data compression techniques can be classified into two categories: Lossy and Lossless techniques (Blelloch,2010). Lossy techniques reconstruct an approximation of the original file from the compressed file. These methods are used when you want to get the compression ratio very high, and there is no essential need to have the resulting file after the compression process is identical to the original file. This type of techniques is suitable for multimedia files such as image files, sound files, video files. While lossless techniques are able to reconstruct the original file exactly from the compressed file. Lossless techniques themselves can be classified into two main categories as shown in Figure 1.1: statistical compression techniques and dictionary compression techniques (Lourdusamy and Shanmugasundaram, 2011).

Text compression is a subfield of data compression. It focuses on compressing natural language texts as they occur in real world. Text compression uses mainly the different features of natural languages to improve the compression ratio and performance. Research papers concerning natural language text compression have been published during the last three decades. their main concern were the European languages such as English, French and German (Manber,1999), (Moronfolu and Oluwade, 2009), (Altarawneh and Altarawneh ,2011) (Hasan, 2011), and on other languages such as Japanese and Chinese (Teahan, McNa and Witten, 2000). Few studies and published research papers were focusing on the compressing of Arabic text.

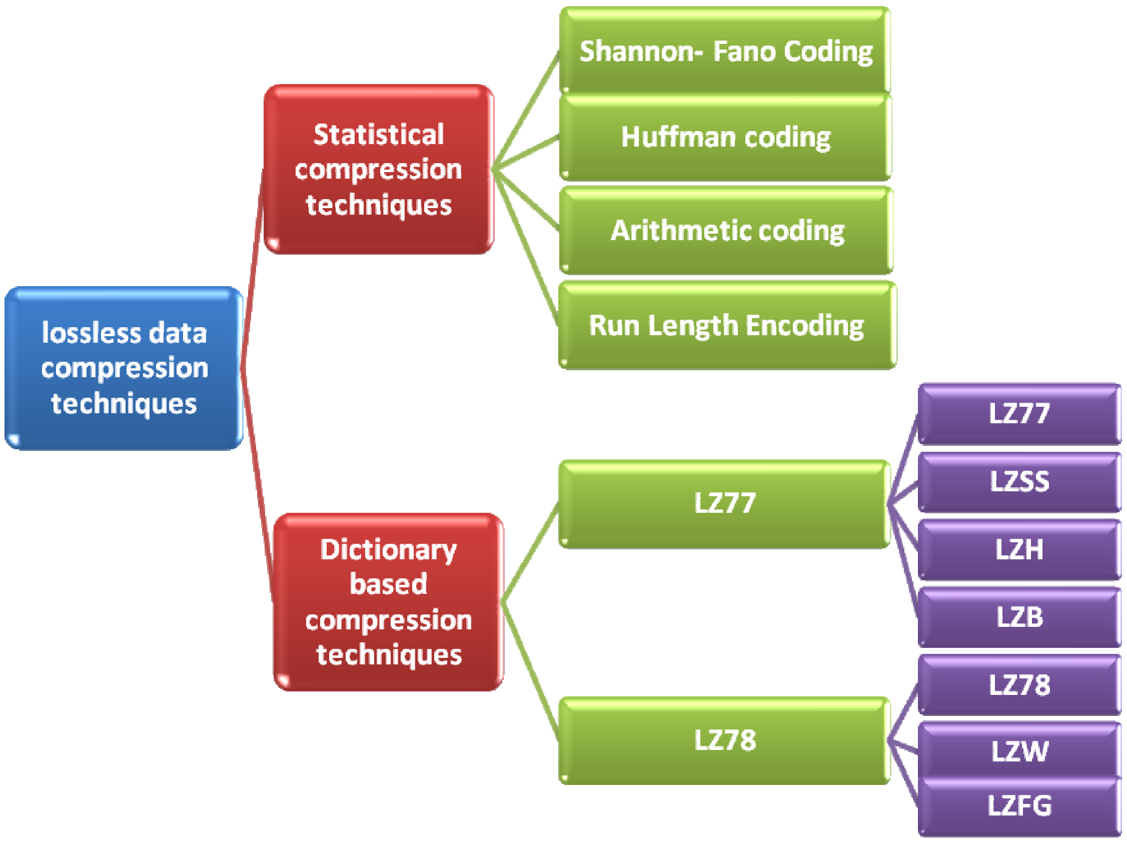

lossless data compression techniques
Statistical compression techniques
Shannon- Fano Coding
Huffman coding
Arithmetic coding
Run Length Encoding
Dictionary based compression techniques
LZ77
LZ78
LZ77
LZSS
LZH
LZB
LZ78
LZW
LZFG

Figure 1.1 Lossless data compression techniques

Arabic and other Semitic languages are complex, rich in terms of and morphological features where tens or hundreds of words can be derived from the same root. These morphological features can be exploited to improve the compressing ratio of Arabic texts, (Soudi, Bosch, and Neumann, 2007). Several researches have been done about Arabic morphological analysis, using variety of approaches at different degrees of linguistic depths. Algorithms and systems for compressing Arabic texts can be divided into four known methods: table lookup based approach, combinatorial approach, pattern based approach, and linguistic based approach, (Al-Sughaiyer and Al-Kharashi, 2004).

1.2. Features Of The Arabic Language.

1.2.1. Structure and Categories of Arabic Words.

An Arabic word is a series of alphabet letters and diacritical marks. Thirty six characters are used in Modern Standard Arabic (MSA): 28 basic letters and eight diacritical marks. The diacritical marks called TASHKEEL are added above or below Arabic letters. The Diacritical marks have three types as shown in Figure 1.2.

These marks are optional so that if each character of an Arabic word has a diacritical mark, the word is called fully vowelized. This kind of text found in the holy Qur'an text, classical poetry, textbooks of children and foreign learners. Figure 1.3 shows the vowelization state of the Arabic word.

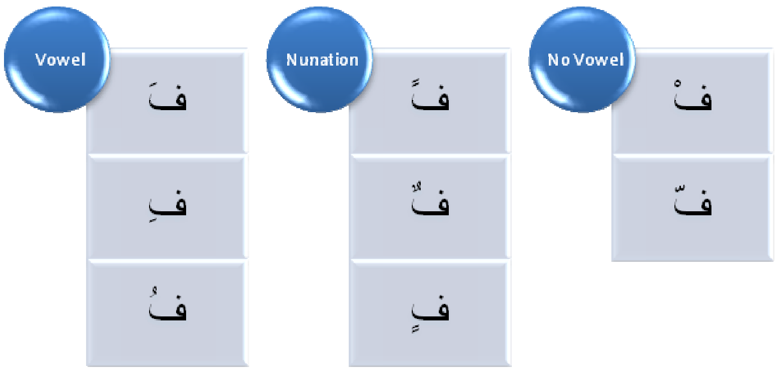


Figure 1.2 Types of Arabic diacritics.

When the diacritical marks are placed to one or two letters of the word, an Arabic word in this case is called partially vowelized. But if the word is without diacritical marks, it is called unvowelized word. This kind of text can be found in newspapers and many of modern Arabic texts. Figure 1.3 shows an example on each of the vowelization state of the Arabic word.

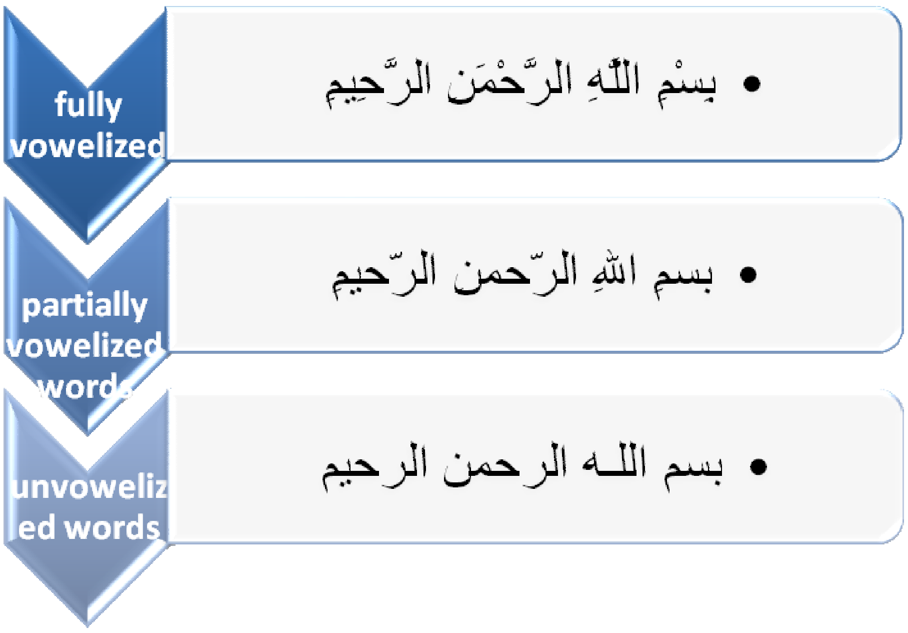

fully vowelized
بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ
partially vowelized words
بسم اللهِ الرّحمن الرّحيم
unvowelized words
بسم الله الرحمن الرحيم

Figure 1.3 The vowelization states of the Arabic Text.

In Arabic language, a word may be derivative or non-derivative: (i) a derivative Arabic word, is generated from a basic Arabic root, Figure 1.4 shows an example of some words that are derived from the root كتب k-t-b 'wrote', (ii) non-derivative word: is nouns borrowed from foreign languages or stop word.

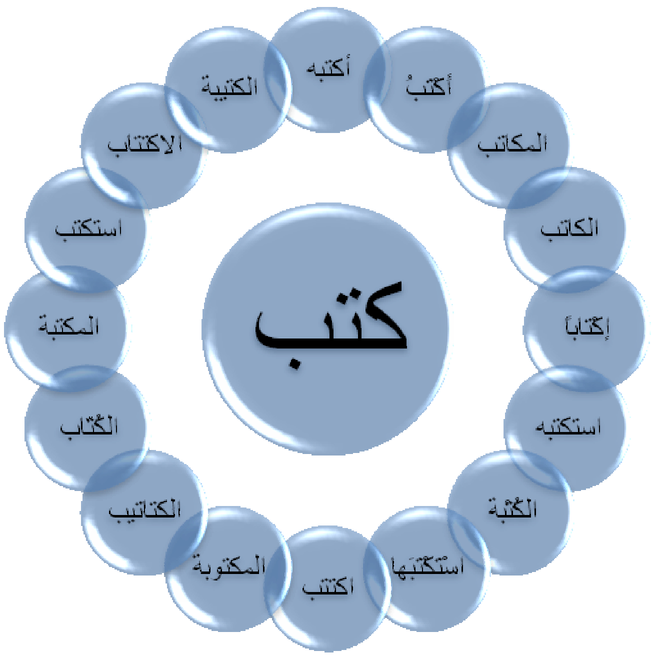
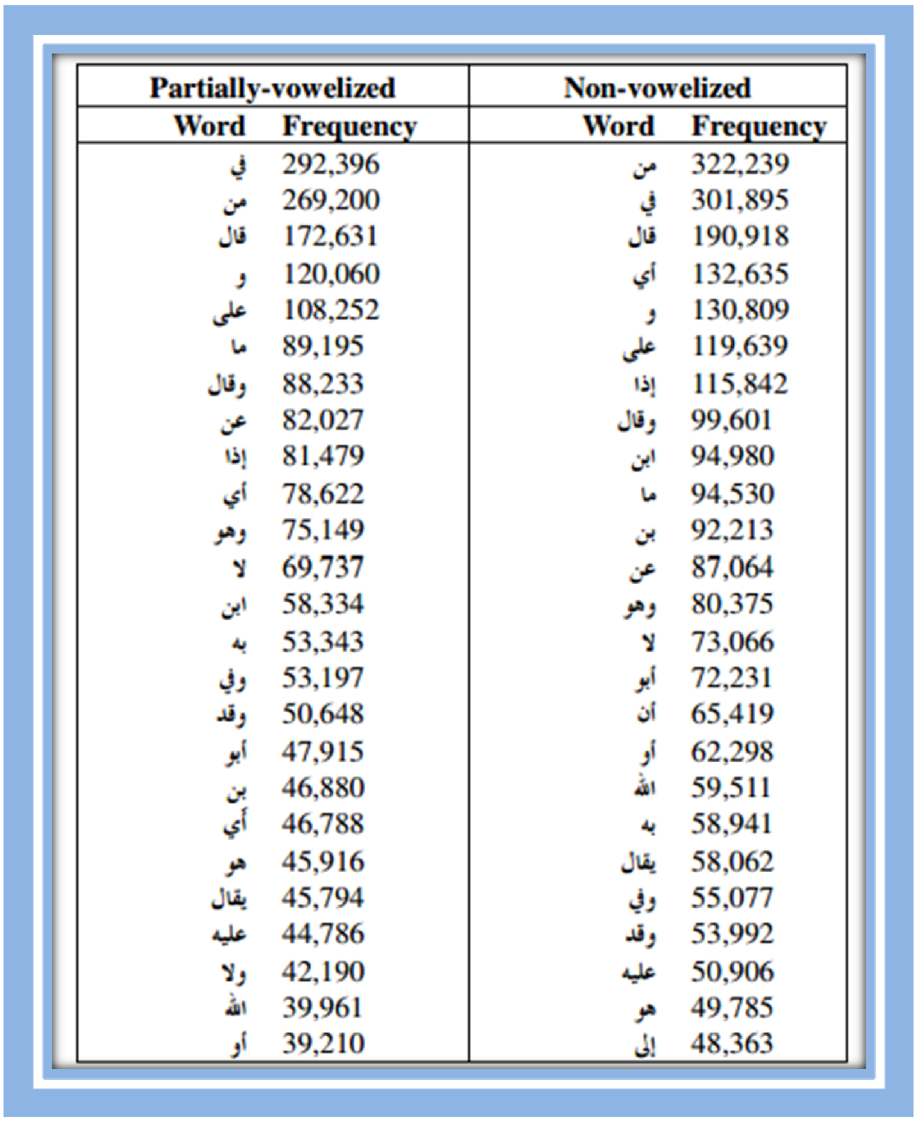


Figure 1.4 Some words derived from the root كتب k-t-b 'wrote'.

Stop words are words that have little semantic meaning. However, they are used to explain grammatical relationships between the words within a sentence. This kind of words consists of: pronouns, prepositions, conjunctions, interjections, pro-sentences. The number of stop words is limited, but their frequency is very high in natural texts. They are represented in Arabic text nearly 40% of the total number of words (Sawalha,2011). Figure 1.5 shows the frequency of these words in real world text which contains 1 million words taken from different genres collected from newspapers and magazines.



Partially-vowelized		Non-vowelized	
Word	Frequency	Word	Frequency
في	292,396	من	322,239
من	269,200	في	301,895
قال	172,631	قال	190,918
و	120,060	أي	132,635
على	108,252	و	130,809
ما	89,195	على	119,639
وقال	88,233	إذا	115,842
عن	82,027	وقال	99,601
إذا	81,479	ابن	94,980
أي	78,622	ما	94,530
وهو	75,149	بن	92,213
لا	69,737	عن	87,064
ابن	58,334	وهو	80,375
به	53,343	لا	73,066
وفي	53,197	أبو	72,231
وقد	50,648	أن	65,419
أبو	47,915	أو	62,298
بن	46,880	الله	59,511
أي	46,788	به	58,941
هو	45,916	يقال	58,062
يقال	45,794	وفي	55,077
عليه	44,786	وقد	53,992
ولا	42,190	عليه	50,906
الله	39,961	هو	49,785
أو	39,210	إلى	48,363

Figure 1.5 Frequency of some stop words (Sawalha and Atwell 2010).

1.2.2. Morphological Analysis

The Morphological analysis is one of the most important techniques used in Natural language processing. Its objective is the analysis of words in order to decompose them into their original morphemes and identify their internal structure. In the case of Arabic words, A word is decomposed into suffix, prefixes, root or stem.

in 2011, Sawalha presented analytical study of Arabic triliteral roots, suffix and prefixes in the Qur'an and traditional Arabic Lexicons. The results show that 376,167 word types are derived from triliteral roots (3-letter root). The number of prefixes and suffixes are limited and they are 215 and 217 respectively. In the case of derivative words, the morphological analyzers may generate the morphological pattern used for the creation of the word in addition to the other components listed before. It is a key step for many applications of natural language processing systems (Al-Sughaiyer and Al-Kharashi 2004; Jurafsky and Martin 2008; Pauw and Schryver 2008), such as:

- Searching the Web.
- Machine translators .
- Lemmatizes.
- Text categorization.
- Text compression.
- Data encryption.
- Vowelization.

Morphological analyzers are different in their approaches and methodologies. The methodologies used in Arabic morphological analysis can be classified as the following:

- **Stemmers**: They aim at extracting the stem of words, Figure 1.6 This kind of analyzer is frequently used by researchers such as (Khoja 2001; Al-Sughaiyer an Al-Kharashi 2002; Al-Shalabi, Kanaan and Al-Serhan 2003; Khoja 2003; Al-Shalabi 2005; AlSerhan and Ayesh 2006; Boudlal et al. 2011).

Word =	Prefix +	Stem +	Suffix
يَعْمَلُونَ =	ي	عمل	ون

Figure 1.6. Example for Stemmers morphological analysis approach.

- **Lexeme-Based Morphology** (Dichy 2001; Al-Shammari and Lin 2008): This category is applying inflectional rules in order to extract a stem, then apply a derivational rule on an inflectional stem output and outputs a derived stem. A compounding rule takes word forms, and similarly outputs a compound stem, words, as shown in Figure 1.7.

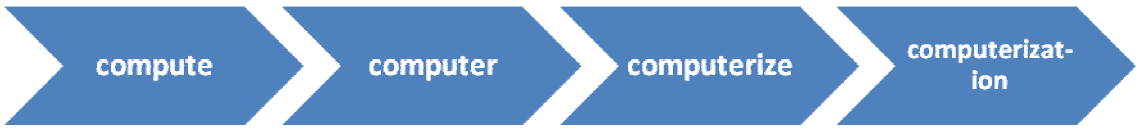


Figure 1.7 Example for Lexeme-based Morphology.

- **Pattern matching algorithms** (Dichy and Farghaly 2003; Al-Shalabi 2005; Alqrainy 2008; Yousfi 2010): These algorithms proceed by extracting the root of the word by matching the word with predefined lists of patterns and affixes. An example of this process is shown in Figure 1.8.

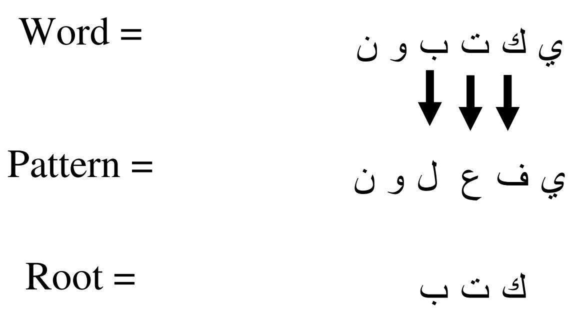


Figure 1.8 Example of pattern matching algorithms.

- **Syllable-Based Morphology (SBM):** This approach depends on analyzing the syllables of the word. For example, the word water is composed of two syllables: wa and ter, (Cahill, 2007).

1.3. Problem Definition.

- Arabic content on the Internet and other digital media is in an exponential increase and the number of Arabs users of these media has been multiplied by more than 20 these last five years. There is a real need to save allocated space for this content as well as allowing more efficient usage, searching, retrieving information operation on this content.
- Few research work and published papers have been dealing with Arabic text compression in spite of the fact that Arabic is one of the major international languages.
- Using techniques borrowed from other language or general data compression techniques ignoring the proper features of Arabic has limited success in term of compression ratio and text representation.

1.4. Objectives.

There is number of data compression algorithms, which are dedicated to compress different data formats. In 2008, Štujber showed that utilizing multiple compression algorithms is good to be used in practice, as a superior alternative to the classic of single-compressor approach.

The main goal of this thesis is to develop a hybrid technique to process the Arabic language, in order to achieve high compression ratio. Such approach is useful to compress just textual files, especially for the Semitic languages. The technique relies on morphological features of the Arabic language as a pre-processing step for data compression. In addition, this thesis aims to a chive the following objectives:

- Studying different methods of data compression techniques on Arabic and English text files such as LZW, BWT and Adaptive Huffman techniques.
- Comparing and evaluating the efficiency of these techniques for different categories of Arabic text files of different sizes.
- Exploiting morphological features of the Arabic language to improve performance of LZW techniques.
- Suggesting a new hybrid techniques to achieve a good compression ratio which is better to store Arabic texts.

1.5. Motivation.

The continual growth of the information highway and the increasing amount of texts stored on and transmitted between computers has increased the need for development of efficient techniques for compressing textual files to make them easier to store on disk and faster to transmit over an Internet connection.

Developing new compression techniques based on the morphological and grammatical features of Arabic language may present a new paradigm which will be able to improve the compression ratio and performance and may produce a new representation of text that can be more appropriate for other applications such as searching text.

1.6. Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 reviews the most important literature related to the data compression. Chapter 3 describes the main elements of our framework design and the implementation details for lossless data compression and the proposed method. Chapter 4 presents the conducted experiments with their used settings and their results. Finally, the conclusions and future work are presented in chapter 5.

Chapter Two

Literature Survey

Two approaches of research on Arabic text compression can be found in the literature. The first approach considers the general purpose techniques and do not take into account the features of Arabic languges. The second one uses the features of Arabic language to develop new compression techniques in order to improve the compression ratio.

2.1. General Purpose Approach.

There are many techniques used to compress data in general and the texts in particular. Some of these techniques proceed at the level of characters. They use the frequency of characters in order to replace the most frequent characters by short codes. Therefore, they are called statistical compression methods. Examples of this category of techniques are the Run Length Coding and Huffman Coding. Other techniques look at strings in the text and put pointers to strings or substrings that have been already appeared. These techniques are called dictionary based techniques. Example of this category of compression techniques is Lempel-Ziv Codes (LZ). There are also techniques which look at the frequency of the character and at the character that occurs at its neighboring when they decide how to encode a character. Examples of the last category are Burrows Wheeler Transform (BWT) and Prediction by Partial Matching (PPM).

Khafagy (2005) had presented a study for a variety of data compression techniques. This study was applied on English or Arabic texts for compression. The best compression ratio had been obtained in case of Neural Compression. Next come the PPM, Lempel–Ziv–Storer–Szymanski (LZSS), LZ77, Lempel–Ziv-Welch (LZW), Lempel–Ziv Ross Williams (LZRW), Huffman techniques. Then finally the RLE program. On the other hand, PPM and Neural technique achieved the best compression time compared to other techniques. In Neural Network, a network structure was suggested with the detailed information which included a number of layers, number of nodes in the layers and the weights between them. Cascading was also observed and it showed negligible advances in performance.

Ghwanmeh, Al-Shalabi, and Kanaan (2006) employed the dynamic Huffman coding with variable length bit coding to compress Arabic texts. They proved that this approach was able to improve the efficiency of compression of Arabic text much better than compression of English text. They showed that the frequency of symbols on the text affected the compression ratio and the average message length.

Alasmer, Zahran, Ayyoub, and Kanan, (2013) had presented a comparison between Static Huffman and LZW techniques on Arabic and English texts. The results had shown that Huffman got better and efficient results in Arabic. An LZW technique was better in English documents especially when it was converted into a binary file.

Most of the published papers studying the efficiency of data compression techniques have been dealing with English texts (Al-laham and El Emary, 2007) (Moronfolu and Oluwade, 2009) (Lourdusamy and Shanmugasundaram, 2011) (Altarawneh and Altarawneh ,2011) (Hasan, 2011). Studies which that deals with Arabic language was more limited despite the fact that Arabic is one of the major international languages. This thesis aims to study, evaluate, and test different methods of data compression algorithms, and to compare their performance on Arabic text files.

2.2. Arabic Language Features Based Approach.

Some research work had used the morphological and grammatical features of Arabic language to improve the compression performance of the data.

Wiseman and Gefner (2007) suggested a suitable technique for Semitic languages. This technique includes two phase: at the first phase, morphological analysis is used to segment the text into two files. The first file includes index values for each pattern. The second file includes the root of the words. The second phase included compressing both files, using traditional BWT. The Hebrew bible file had been used for testing the compression. The size of file was 260 KB. Results showed that using the technique which had been suggested achieved 28.97% of compression rate whereas the traditional BWT compresse this file to 40.13%.

Daoud (2010) had presented a hybrid technique to compressing diacritical Arabic texts, benefiting from Arabic morphological features for split the entered word into root and infix, and exploited the unused 128 location of the ASCII, to represent the most frequent articles, root and affixes. The optimal Huffman compression algorithm was used to measure performance of the proposed techniques which improvements 20% of the traditional technique.

Omer and Khatatneh (2010) used the fact that Arabic letters have a single case to present a new technique which had been applied on Arabic short text. Huffman technique was used to measure the efficiency of the proposed technique. Two passes had been implemented on the input text through: (i) calculating the frequency each of character. (ii) Assigning numerical code between 0 and 127, but checking if the numerical value >=64 encode the character using 7 bits else using 6 bits. This approach outperformd the two pass Huffman compression.

In 2011 there was another technique suggested by (Akman et. Al). It was a suitable technique to compress Semitic languages. It depended on syllable-based morphology. This technique had been implemented for the Turkish language. It is tested of files of sizes ranging from 4.6 to 725 KB; the results had showed that a compression ratio up to 43% was a achieved.

Awajan (2011) used the morphological structure of words to build a multilayer model of the Arabic text. The proposed model worked in two phases: the morphological analysis phase and the encoding phase. Text was split into three categories of words: derivative, non-derivative and functional words. A fourth layer, called the Mask, introduced to aid with the reconstruction of the original text from the three layers in the decoding side. Both the function words and the derivative words (which were divided into pattern and root of the word) had been replaced with their index value. The third category of words has compressed using Huffman coding.

The results of this Multilayer model had showed that its performance was better than many of the traditional compression techniques applied on the whole text. Another advantage of this work was that the layer file reserved for derivative words might be used by the research engine and the information retrieval systems to find out terms or word by looking on their root or stem without the need to decode the compressed text.

Chapter 3

Framework Design and Implementation

3.1.Framework Design.

Each one of the compression methods has some advantages and disadvantages. Dictionaries-based methods are fast but the give smaller compression ratios. On the other hand statistical based methods provide high compression ratios, but they ignore the specificities of natural language texts. Since the morphological features of Arabic may reduce the storage requirements of the Arabic texts and dictionaries, hence they should be exploited to improve the performance of general compression techniques. Thus investigating hybrid approach that combines several of these methods in order to obtain better compression ratio has been proposed.

3.1.1. The Proposed Hybrid Approach.

The Arabic language is rich in morphology, i.e. the origin of the word changed to several forms varying from the original form of the word. For example the word. " قــرأ"-"read", may change to "يقــرأ - read","قـــارئ-reader","readable-مقــروء". and other words. Unlike the English language where the origin of the word is not changed to several forms varying from the original form of the word the increase might be in end or the beginning of the word.

For example, "read", "reads", "reader", "the reader" .... so on (Wiseman and Gefner, 2007).

The BWT technique is very sensitive to the structure of the word, so derivative words are not suitable for compression by this technique. Therefore, we have suggested using one of the morphological analyzers as pre-processing step to implement (BWT) on derivative words**,** using the "root-pattern dictionaries" technique, guided by the proposed method of (Awajan, 2007, Awajan 2011). The main idea of this technique is to replace derived words by index values to their roots and their standard pattern. Then BWT technique is applied to compress the text as shown Figure 3.9.

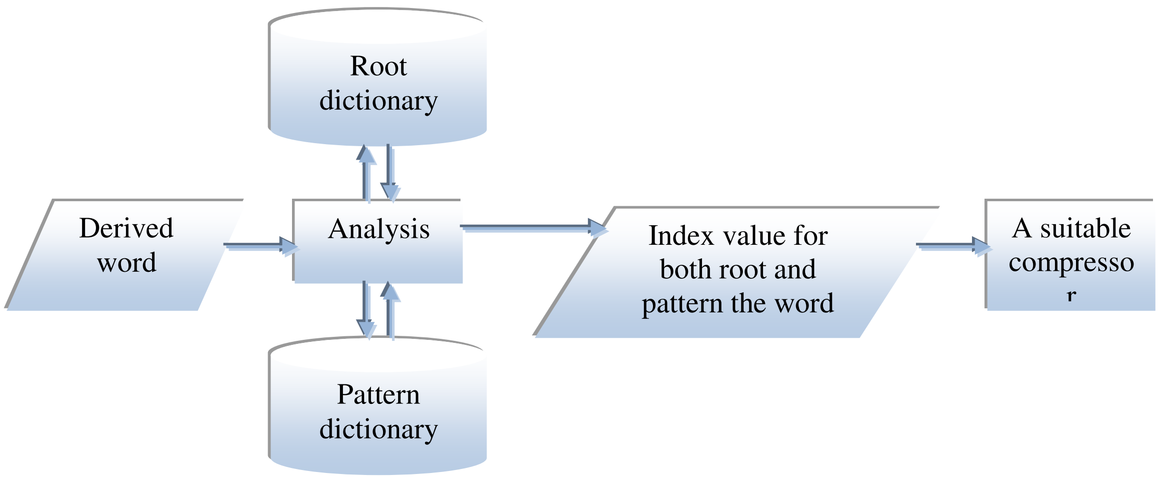


Figure 3.9 The morphological analyzers.

The proposed compression technique consists of two phase, as shown in Figure 3.10:

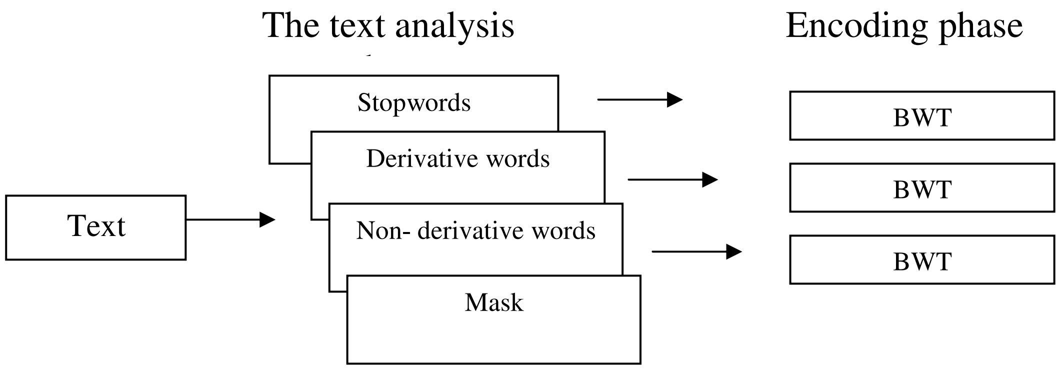


Figure 3.10 The main steps of the hybrid compression approach.

1) Text Analysis Phase:

Much of work has been done at the level of morphological analysis for Arabic text area (Al-shalabi,2005; Al-khalil, 2010; Yousfi, 2010). In 2011 Awajan had developed an efficient approach for the morphological analysis of the Arabic word that is called Multilayer Model. This model employs the features the Arabic language to partition the input text into three layers represent the three categories of words as shown in Figure 3.11: functional derivative, and non-derivative words.

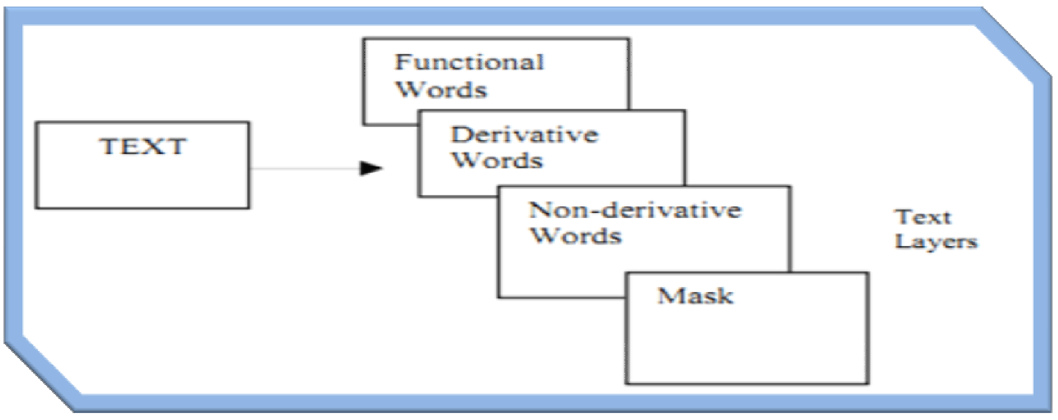


Figure 3.11 Text decomposition into the four layers (Awajan,2011).

Several procedures implemented to split the text into three layers. Figure 3.12 shows Flowchart for Multilayer model based approach.

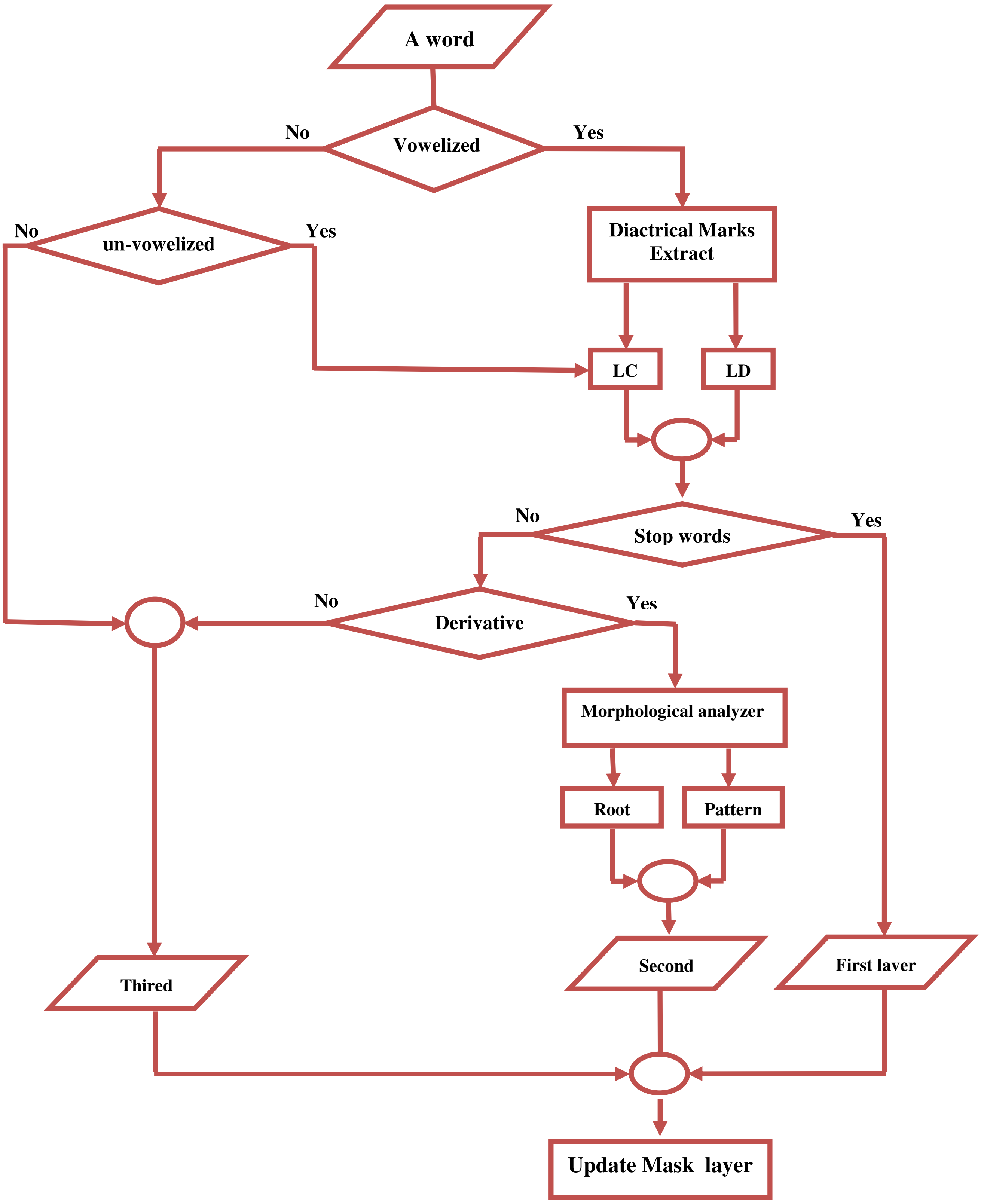


Figure 3.12 Flowchart for Multilayer model based approach.

For each word in the incoming text, the morphological analyzer determines if it is a vowelized word or not. In the case of vowelized word it will decompose it into two lists (LC and LD list) as shown for example Figure 3.13. LC list contains the sequence of consonants forming the word. LD list contains the sequence of diacritical characters of the word as shown in Table 3.1 (Awajan 2007, Awajan,2011).

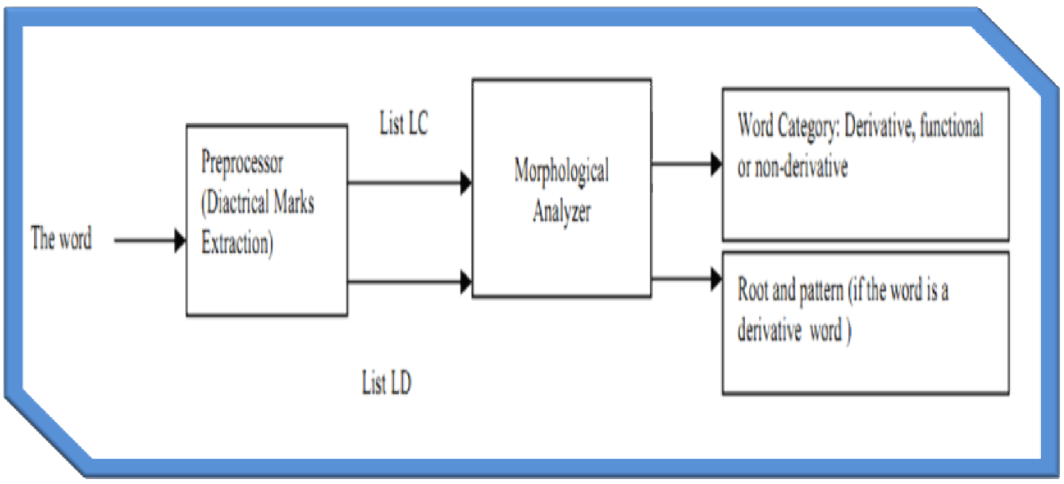

The word
Preprocessor (Diactrical Marks Extraction)
List LC
List LD
Morphological Analyzer
Word Category: Derivative, functional or non-derivative
Root and pattern (if the word is a derivative word)

Figure 3.13. The morphological analyzer (Awajan,2011).

Table 3.1 Decomposition of the word "يذهبون" –" they go "into the two lists (Awajan,2011).

Word	Class of the Word	List of Consonants LC	List of Diacritics LD
يَـذْهَـبُـوْنَ	Vowelized	[ي ذ هـ ب و ن]	[َ ْ ُ َ ْ َ]
يـذهـبـون	Unvowelized	[ي ذ هـ ب و ن]	[ ]

A bit associated with each word is used to indicate if it is vowelized or not. It is set to zero if the word is unvowilized, and it is set to one if the word is vowilized. The list LC of consonants is then tested against a predefined list of functional words. If it is stop word, it will replace with the index value (of 8 bits) taken from the function words table as shown in Figure 3.14, to build the first layer. Figure 3.15 shows construction of the first layer- the functional words layer (Awajan, 2011).

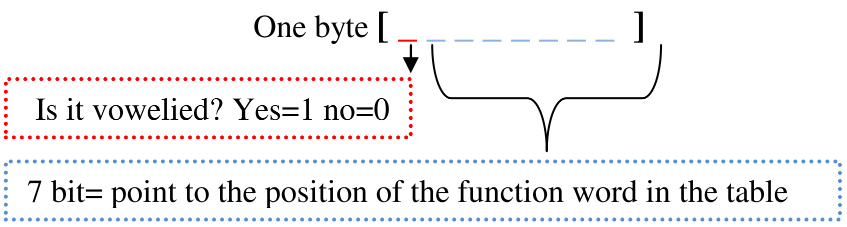

One byte [_ _ _ _ _ _ _ _]
Is it vowelied? Yes=1 no=0
7 bit= point to the position of the function word in the table

Figure 3.14 Structure output the first layer.

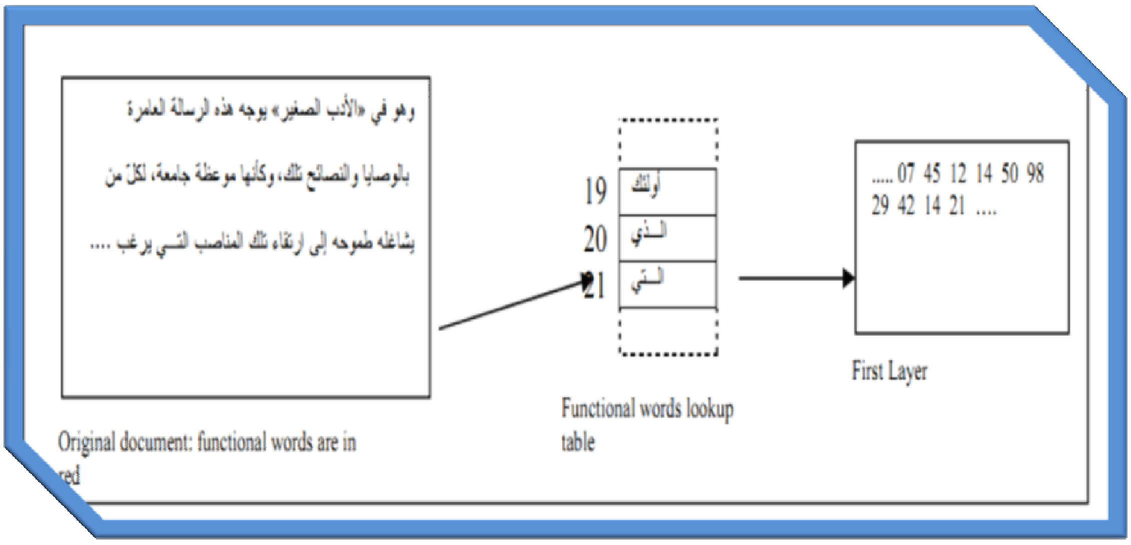

وهو في «الأدب الصغير» يوجه هذه الرسالة العامرة
بالوصايا والنصائح تلك، وكأنها موعظة جامعة، لكلّ من
يشاغله طموحه إلى ارتقاء تلك المناصب التـي يرغب ....
19 أولئك
20 الـذي
21 الـتي
..... 07 45 12 14 50 98
29 42 14 21 ....
First Layer
Functional words lookup table
Original document: functional words are in red

Figure 3.15 Construction of the first layer- the functional words layer (Awajan,2011).

If the word is not detected as stopwords word, the morphological analyzer will determine if it is a derivative word generated from 3-letter-root or not through determining the morphological structure of the word: root and pattern, to replace both of them with their index. A derivative word is encoded using 3 bytes as shown in Figure 3.16.

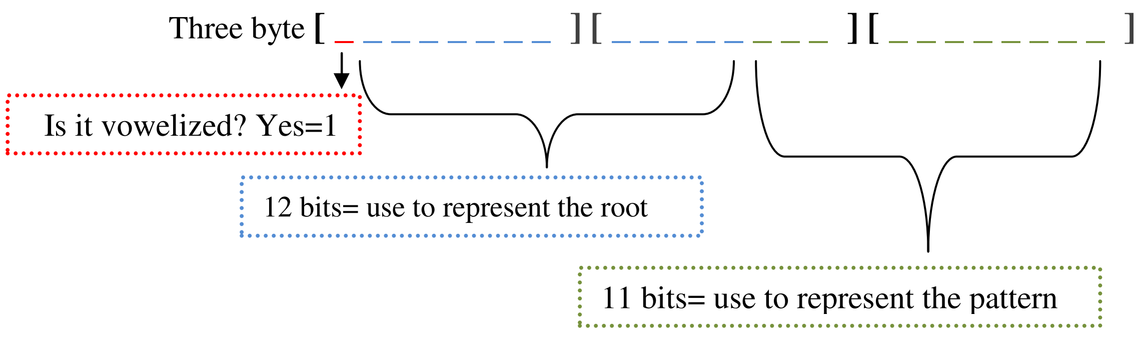

Three byte [_ _ _ _ _ _ _ _] [_ _ _ _ _ _ _ _] [_ _ _ _ _ _ _ _]
Is it vowelized? Yes=1
12 bits= use to represent the root
11 bits= use to represent the pattern

Figure 3.16 Structure output the second layer.

Figure 3.17 shows the reconstruction process of the second layer from the original document using the codes of roots and patterns stored in the appropriate tables.

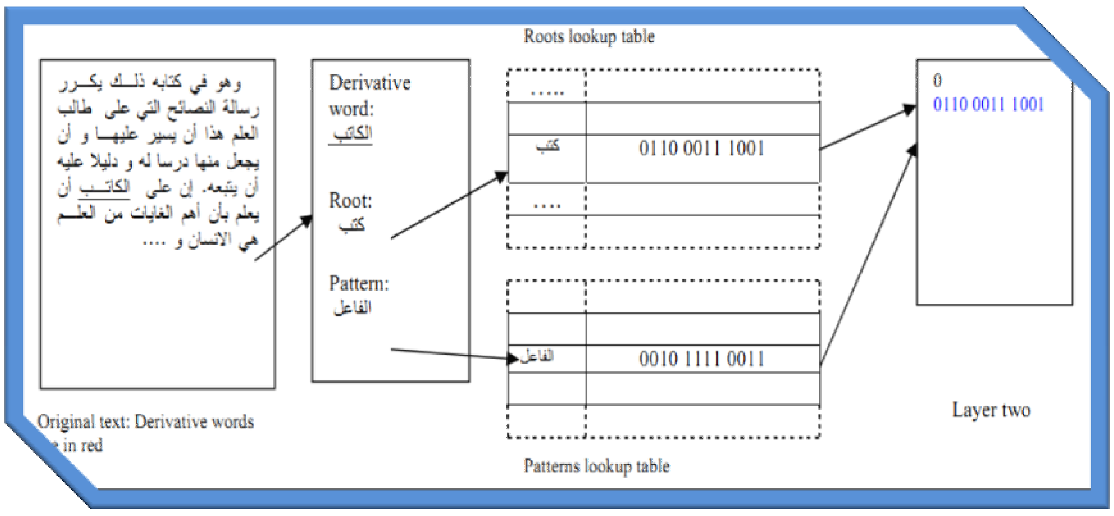

Roots lookup table
وهو في كتابه ذلك يكرر رسالة النصائح التي على طالب العلم هذا أن يسير عليها و أن يجعل منها درسا له و دليلا عليه أن يتبعه. إن على الكاتب أن يعلم بأن أهم الغايات من العلم هي الانسان و ....
Derivative word: الكاتب
Root: كتب
Pattern: الفاعل
.....
كتب
0110 0011 1001
....
الفاعل
0010 1111 0011
0
0110 0011 1001
Layer two
Original text: Derivative words in red
Patterns lookup table

Figure 3.17 Construction of the second layer- the derivative words layer (Awajan 2011).

The Mask layer is used to link the words with their original positions in the input text, for each value in the Mask. If it is equal to zero, the corresponding word is a functional word from the functional words table and encoded in the first layer. If the value is 1, the corresponding word is a derived word, so that the root will be extracted from the pattern of the original word, then encoded in the second layer. If the value is 2, the corresponding word is a word not classified in the previous categories. Therefore it will be encoded in the third layer. Figures [3.18 - 3.20] shows a pseudo-code representation for Multilayer Model. Also the Mask layer is used to reconstruct the original text from the three other layers of the compressed text, Figure 3.21 shows the decoding algorithm.

Finally, each of the four layers is compressed individually using a suitable compressor.

```
    while (!EOF)
{
        Read the word (x)
       if (  vowelizied (x) =true) then
                {
                   Extract the diactrical marks (x, LC,LD)
                    Output=1
                    GOTO:
                       If  ((function_words(x))= true)  then
                             {
                                  AddToOutput (Get.value(x))
                                   First_Layer. Write(output)
                                              Mask_layer.write(0)
                             }
                                  Else if ((pattern match(x))= true) then
                                       {
                                          Morphological Analyzer (x, Pattern, Root)
                                          AddToOutput( Get.value (Root))
                                          AddToOutput (Get.value (Pattern))
                                          Second_files. Write(output)
                                          Mask_layer.write(1)
                                       }
                                        Else
                                             {
                                                Third_Layer. Write (x)
                                                Mask_layer.write(2)
                                              }
                              }

       Else if (un-vowlized(x)=true )  then
                      {
                          LC= x
                          LD=empty
                         Output=0
                         GOTO;
                      }
              Else
                      {
                        Third_Layer.Write(x)
                        Mask layer.write(2)
                       }
 }
```

Figure 3.18 Pseudo-code for Multilayer Model.

```
Morphological Analyzer (x, Pattern, Root)
        PATTERN MATCH (LC (x), LC(Pattern)),
        FIND_ROOT (Pattern, a, b, c),
        EXTRACT_ROOT (LC(x), a, b, c, ROOT).
        \*  where a, b and c are three integers determining the position of the
        letters of the root in the given pattern *\
```

Figure 3.19 Pseudo-code for Morphological Analyzer.

```
- Return all the patterns that have the same length of the testing word.
- Calculating the number of identical letters between the testing word and
  the patterns.
- Choose the pattern(s) which have the maximum number of identical
  letters.(W).
- Replace the letters of W which correspond to the letters :ل ,ع ,ف
  represent the root letters (ل,ع,ف) in the pattern(s) (P) which have the
  maximum identical umber.
- Add the remaining letters in W without change.
- Return the new pattern.
```

Figure 3.20 Pseudo-code for Pattern match.

```
Read value form mask ( c )
While (c != EOF)
        {
        if c = 0 then
                Output=Read Byte from  Layer One
                          FinalOutPut=functional_table.GetValue(Output)
        Else if c = 1 then
                {
                Output=Read three byte from  Layer two  \* indicate to the root
                and the pattern *\
                if (the first bit = 0) then     \*the word is unvowelized*\
                         {
                             Root= Root _table.GetRoot(Output)
                              Pattern=  Pattern _table.GetPattern(Output)
                                FinalOutPut=Add The verbs (Root,Pattern)
                          }
                if (the first bit i= 1) then     \*the word is vowelized\*
                         {
                              Root= Root _table.GetRoot(Output)
                              Pattern=  Pattern _table.GetPattern(Output)
                              FinalOutPut=
                                 Add the Diactrical(Add The verbs (Root,Pattern))
                             Mark(Root,Pattern,LD)
                          }
                }
        Else  if c = 3 then
                FinalOutput=Read from  Layer  three
}
```

Figure 3.21 Pseudo-code for the decoding algorithm at multilayer model.

The fourth layer which is called the mask was used during the decoding stage, to reconstruct the original text from the decoding of other layers.

Suitable compression techniques were applied to the different layers in order to maximize the compression ratio.

2) Encoding Phase.

At this phase, BWT technique is applied for each layer. The first layer was used to store the compressed representation of the stop words in the text. These words were limited in number and their frequency in texts was very high. BWT was good for the compression of the first layer. Our results of the tests have shown that compression ratio was 0.50%, 0.83% for BWT and LZW respectively.

The second layer was used to store derivative word. Two tables had been constructed to represent the roots and the patterns. BWT had been suggested to compress this layer, where several of our experiences had proved that implementing BWT technique provided good results for all categories of Arabic texts when integrated with multilayer model. Compression ratio for the second layer was 0.54 when using BWT and 0.75 when using LZW.

The third layer represented the words that the system failed to classify into either of the first two layers. A compression technique based on the BWT compression was found to be very good for these types of words. Compression ratio for this technique was 0.41%, but it was 0.49% for BWT and LZW respectively.

The Mask layer contain the number "zero" to indicate the position of the word in first layer. If it contains number "one" this means the current word in the second layer, either if it contains number "two", this means the word in the third layer. For compress, this layer we have suggested, represents each number as binary code then reads one byte to store the data.

Decompression processes for both approaches are completely opposite to the compression process. It works by decoding each layer independently using the appropriate decoder then reconstructing the original text using the mask layer.

3.1.2. **Improvement of the Performance of LZW Technique (LZWA**).

LZW is dictionary based technique. The main idea of LZW is to replace strings of characters with single codes, usually the first 256 codes are used as the standard character set (for the case of 8-bit characters).

The most important characteristic of Arabic language characters is that each character has a single case ("ي"−"أ") , unlike the English characters where each character has two cases, the upper case("A"-"Z") and the lower case ("a"-"z"). Therefore we have suggested to take advantage of this feature by representing each character by the index number (0-73), which has the lowest number of bits compared with their ASCII code (170-239). So the compression process applies the following steps:

1- Reading the text and entered it into the memory
2- Reading one character each time from memory.
3- Examining if the character exists in the dictionary (where each character has an index number), replacing the character with it's index value (7-bit) rather than ASCII code (8-bit). If isn't found, add it to the dictionary and assign a new index to it.

Results of the tests showed that this approach improved 0.04 of the performance of LZW.

3.2. Lossless Data Compression Techniques.

The following is a brief description of the Lossless data compression techniques (Blelloch,2010), which were used in the proposed model for Arabic Texts:

3.2.1. BWT Technique.

The BWT technique was invented by Michael Burrows and David Wheeler, 1994. It converts the original blocks of data into a format that is extremely well suited for compression, through a sequence of steps (Blelloch, 2010). These steps are: Burrows–Wheeler transform (BWT), Move to Front transformation (MTF), Run-Length encoding (RLE) and Encoding Coding (EC) as shown in Figure 3.22:

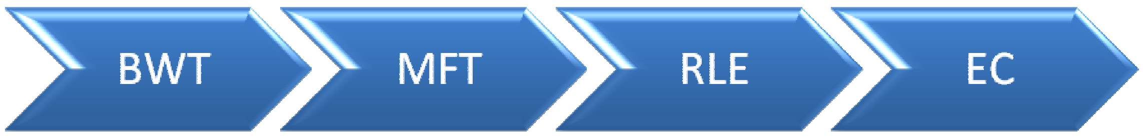


Figure 3.22 Steps of the Burrows-Wheeler Compression Algorithm.

The first step performs the Burrows–Wheeler transform (BWT), which is done by reading blocks of text with predefined size from input and each block is processed separately, Figure 3.23 shows Pseudo-code for BWT. The resulting output block contains exactly the same characters that are started with, differing only in their ordering. This step aims to make blocks of the text easier to code the data with a simple coder.

```
-   Make all cyclic rotations of input text  and number them .
-   Lexicographically sort the resulting phrases and remember
    the number of line containing the input string.
-   Select the last symbols.
```

Figure 3.23 Pseudo-code for BWT.

The decompression of this steps uses the same methods in the reverse order : decompression of the transformed text first and the inverse transform after that.

The second step implements the Move to Front transformation (MTF) to transform the characters into a list of numbers, Figure 3.24 shows Pseudo-code for MTF. This technique doesn't compress data, it is aim is to decrease the redundancy of letters.

```
Insert all symbols of input alphabet into push down store.
while (!EOF)
  {
        Output (position of symbol in buffer).
        Move symbol to front of buffer.
  }
```

Figure 3.24 Pseudo-code for MTF.

The decoder of this step is very easy. It is done through two steps: The first step, reading a code from the compressed file, which represent the position in the table. Then, in the second its corresponding symbol from the table is obtained. Both encoder and decoder should initialize the table with the same symbols in the same positions.

The third step applies RLE on the new text that's has been produced in previous step. RLE is one of the simplest compression techniques dealing with consecutive recurrent symbols (Abel, 2003), which are encoded as a pair: the length of the string and the symbol itself, as shown in the example of Figure 3.25.

- After these steps, we can apply and identify the compression technique. Usually Arithmetic coding or Adaptive Huffman technique is used. We have suggested Adaptive Huffman technique to apply in our work.

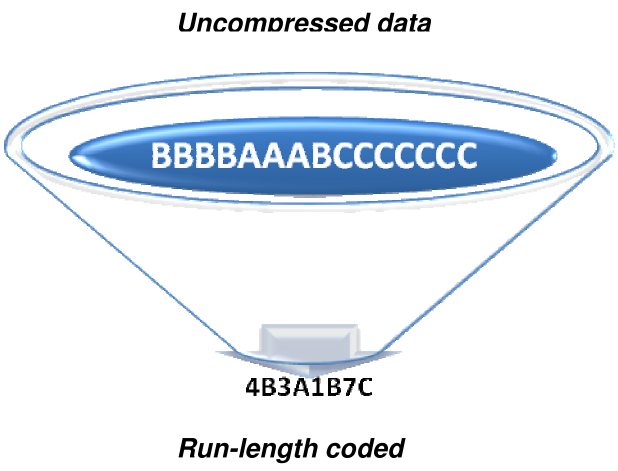


Figure 3.25 Run-Length encoding example.

3.2.2. Adaptive Huffman Technique.

Adaptive Huffman coding or dynamic Huffman coding is based on Huffman coding. It permits representing the code as a tree structure on. Each node of the tree is assigned with a weight and a number. The weight is the count of symbols transmitted. The number is assigned from 1 to ***2n-1***, where ***n***: size of the symbols. Also we need a special control character NYT (Not Yet Transmitted symbol) to indicate that a new leaf is needed in the tree. Figure 3.26 shows the pseudo-code for Adaptive Huffman coding. There are several methods for implementing this technique, The FGK (Faller-Gallager-Knuth) and Vitter algorithm. In our work we adopt the Vitter algorithm since it has been approved that this version produces more efficiency in terms compression ratio. Figure 3.27 shows the Pseudo-code for update tree based on Vitter pattern.

```
  Read Symbol(X)
   while (X!=EOF)
   {
             if (first_read_of(X)) then
              {
               output(code of NYT)
               output(code of node X)
               create new node U with next node NYT and new node
   X
               update_tree(U);
              }
             else
              {
                output(code of node X)
                update_tree(node X)
              }
             readSymbol(X)
           }
```

Figure 3.26 Pseudo -code for Adaptive Huffman coding.

```
procedure update_tree(U)
  {
     while (U!=root)
      {
        b := block of nodes preceding node U
        if ((U is leaf) and (b is block of inner nodes) and
(weight U == weight b))
          or ((U is inner node) and (b is block of leaves) and
(weight U == weight b - 1)) then
         {
          slide U in front of block b
          increment weight of U
               if (U is leaf) then
                      U := parent(U)
              else
               U := parent (U before slide)

         }
        else
              {
               increment weight U
               U := parent(U)
              }
      }
        increment weight U
   }
```

Figure 3.27 Pseudo-code for update tree based.

3.2.3. LZW Technique

The LZW was created by Abraham Lempel, Jacob Ziv, and Terry Welch. This technique is widely used for data compression. It provides a better compression ratio (Kodituwakku and Amarasinghe, 2012). The idea in this technique is to replace strings of characters a single code, where each code is 12 bits.

The encoder begins with a string table that contains all ASCII characters from (0-255). It reads one character each time from the input file, then it checks if the string is in the table. If this is case, we get the fixed code value without any analysis. If the string is not in the table, add it as a new entry at its end, and output the last string. The LZW method provides 4096 entries in the table. Figure 3.28 shows pseudo-code for LZW.

The string table is created during the encoding and decoding process. There is no need to exchange it between them. Decoding is implemented by taking every code from the compressed file and translating it through the code table to find which character it represents.

3.3. Measuring Compression Performances.

There are several criteria to measure the efficiency and quality of lossless compression algorithms. One of the most important and which is used in this thesis is the ratio between the size of the compressed files and the size of the original files (compression Ratio (CR)= Compressed file size / Original file size). In additional, the compression time is measures and compared with other.

```
String ← first input character;

CodeWord ← 256;
Clear out the string table before starting
while(not end of character stream)
{
    Char ← next input character;
    if(String + Char exists in the Dictionary)
        String ← String + Char;
    Else
      {
            Output: the code for String
            insertInDictionary( (CodeWord , String + Char));

            CodeWord++;
            String ← Char;
      }
}
    Output: the code for String;
```

Figure 3.28 Pseudo-code for LZW.

3.4. Implementation Details

In this section, we describe in detail the procedures and implementation phases of the proposed method.

3.4.1. Selection of the Appropriate Compression Techniques.

Several techniques are used to compress data. Figure 1 in chapter one presented the different categories of these techniques. Some of them look at each character in the text and its frequency. They are called statistical techniques (e.g. Run Length Coding, Huffman Coding), whereas other techniques look at strings in the text and put pointers to strings or substrings that have been already appeared. These techniques are called dictionary based techniques (e.g. Lempel-Ziv Codes). There are also techniques which look at the frequency of the character, and also look at the character and it's neighbor when they decide how to encode a character (e.g. BWT).

Our objective was mainly to select the appropriate compression technique from each category of Arabic texts (vowelization state).We limited our work on selecting the best techniques among the three techniques by presenting the main group of compression technique.

3.4.1.1. LZW Technique.

LZW has been selected from the category of the dictionary techniques because it works well for small files and the saving percentage for storage space is high (Kodituwakku and Amarasinghe,2012).

3.4.1.2. BWT Techniques.

Several studies have proved that the compression technique based on BWT provides good results in comparison with the general-purpose compressors (Radescu, 2009), and achieves good compression ratios combined with high speed (Abel, 2003).

3.4.2. Selection of Morphological Analyzer for Arabic Texts.

Several morphological analyzers have been designed for Arabic language in order to deal with the challenges of its morphology and syntax. But there are no united standards to evaluate these studies, which makes the task of comparison between the analyzers very difficult. So they will be compared and evaluated according to advantages and disadvantages each one in terms of deal with categories Arabic texts, and types of Arabic words.

Al-shalabi, (2005) presented a morphological analyzer, which treats the unvowilize text, by removing suffix, prefix of the infected words and finding the suitable pattern to extract the trilateral root of the words, through applying several rules. An advantage of this approach is that it doesn't depend on look up table for searching, but it applies many rules to extract root of a word. This process is time consuming approach.

Yousfi, (2010) presented another approach to analyze Arabic words. His technique depends on segmenting the Arabic verbs to suffix, prefix and root, using only the verbs patterns, instead of using rules during the morphological analysis. This analyzer does not use dictionary of roots and does not use rules during the morphological analysis. It analyzes just the Arabic verbs and ignores other categories of the words either the function words or non-derived words.

Al-khalil, (2010) provided another morphological analyzer for full-vowelized, non-vowelized, partially vowelized for Modern Standard Arabic

(MSA) text. It is based on rules that divide. the words into prefix, stem and suffix, then matches the stem with non-derived words list. The word is a derived word, the analyzer will determine the possible root and pattern, by extracting the clitics and pattern matching algorithm. Most words that fail to be analyzed by this technique are considered to be functional words and proper noun.

Awajan (2011) provided multilayer model for analysis of full-vowelized, non-vowelized, partially vowelized Arabic text. It classifies the text into three categories of words: derived, functional words , and other (i.e. non-derivative word, and the words that the system fails to classify into one of the categories). His approach depends on searching to determine if the word is functional or not, and use two technique to determine the derived word, the first one technique is using dictionary for the patterns and the roots, and the second one is to applying the pattern based algorithm.

This approach attaches all prefixes and suffixs to the dictionary of patterns, to decrease the duration of the morphological analysis. Although it depends on searching to identify functional words, the number of this category of words is limited.

Our aim in this thesis is to integrate more than one algorithm to compress Arabic texts, by taking advantage of the morphological feature of Arabic language. The most important characteristic of a multi-layered model from other analyzers is that it deals with all categories of texts, and all categories of Arabic words, especially as the Arabic text does not consist only of a set of characters, but includes symbols and punctuation marks. So, this approach has been suggested for morphological analysis of words.

3.4.3. Data Preparation.

For testing and evaluating different techniques, a set of text files were selected from multi sources from the internet. They represents stories, holy text from Qur'an, articles from BBC Arabia news, and texts from the traditional Arabic lexicon “Lisān Al-‘Arab”. There files are stored as text files and Unicode encoding. They have different sizes ranging from 1.8 KB to 557 KB. To have more precision in the comparison details, the set of text files represents Arabic and English texts.

3.4.4. Applying Data Compression Techniques on Arabic Texts.

In order to evaluate the efficiency of the algorithms that process the Arabic texts, two experiments were conducted separately for each algorithm. Each experiment includes testing 18 files, which have different size and different content. In first experiment, the algorithms had been applied on three type of texts (vowilized, partial vowilized, non vowilized). In the second experiment, the algorithms had been tested on English texts to measure the efficiency of data compression of Arabic texts in general.

3.4.5. **Measuring the Performance of the Arabic Text compression**.

This step aims to measure the efficiency of data compression algorithms that have been selected on the Arabic texts for all vowelization state.

Compression ratio (CR) and compression time are calculated for measuring the performance of the algorithms (LZW, BWT, Adaptive Huffman, and multilayer model).

3.4.6. Comparing the Arabic Text Compression Techniques.

This steps aims to compare between the Performance of the Arabic Text Compression techniques that have been suggested and selected for each category of Arabic texts separately.

3.4.7. Suggesting a new Hybrid Technique to Compress Arabic Texts.

According to the results of the previous steps, the suitable technique was determined to be integrated them with the morphological analyst, which was selected previously.

Chapter 4

Experimental Evaluation of Arabic Text Compression Techniques

This chapter presents the conducted experiments and their results. In Section 4.1, dataset is presented in detail. In Section 4.2, the development environment is presented. The experiments in Section 4.3 evaluate the performance of the two of the lossless compression techniques (LZW, BWT) for each state of the vowelization states with regard to Arabic language texts in addition to English language texts.

4.1. Data Set.

The first group of experiment include testing 18 files with different sizes varying from 1.8 KB to 82 KB. Each of the techniques to be tested was applied on different categories of the Arabic texts.

The second group of experiments in this study is done on other group of test files with size varying from 200 KB to 558 KB. Three of tables are used in this group of experiments. One for storing the stop words, the second for storing the roots and the third one to represent the standard patterns of Arabic words derived.

A statistical analysis was made to count the stop word's frequency for un-vowelized texts, which were obtained from a corpus representing the BBC and CNN Arabic news (Saad,2011). A stop words table was then constructed. It containd 127 of the most frequently occurring stop words. Table 4.2 shows 10 stop words and of highest frequency.

Table 4.2 Some of the stopwords and their frequency.

Word	Frequency
في	75931
من	50361
على	34058
أن	28020
إلى	25218
ان	16915
عن	13440
التي	12844
مع	9463
الذي	9210

The other two tables were constructed to represent the roots and the patterns. The roots table included the 4095 of the most commonly used 3-letter. The patterns table consists of the 13600 most used patterns (ALESCO, 2008). The later table has two entries for each pattern. One entry represents the list of consonants (LC) and the other entry represents the list of diacritics (LD). Table 4.3 shows a sample of entries in the table of patterns

Table 4.3 A sample of entries of the table of the patterns

1	اسْتِفْعَال	است**ا*	ْ ِ

2	اسْتَفْعَلَ	است***	ْ
3	اسْتُفْعِلاَ	است***ا	ْ َ ُ ِ

4.2. Environment. .

The experiments were carried out on Laptop equipped with a Pentium (R) Core CPU of 2.20 GHz and 2GB RAM, running MS Windows 7 (32-bit) operating system. The source code was written for each technique in vb.net.

4.3. Experiments.

Two groups of experiments were implemented out on Arabic texts. The first group of experiments aimed to compare and evaluate the efficiency of the single data compression techniques for Arabic texts. The second group of experiments aimed to: (i) exploiting morphological features of the Arabic language to improve performance LZW, and (ii) developing a hybrid technique to compress the Arabic texts.

4.3.1. The First Group of Experiments.

In order to evaluate the efficiency of the techniques of compression Arabic texts, two experiments were conducted separately for each algorithm. Each experiment includes testing 18 files, which had different sizes as shown later. These files had been taken from multiple resources from the internet.

In the first experiment, each technique had been applied on different categories of the Arabic texts (vowilized, partial vowelized, unvowelized). In the second experiment, each technique had been applied on both Arabic and English texts.

4.3.1.1.Experiment the LZW Technique

This experiments aimed at measuring the performance of the data Compression techniques on Arabic Language using LZW technique.

A series of experimental tests was performed to compress both Arabic and English texts of different sizes. Table 4.4 and Figure 4.29 showed results of the compression ratio using LZW compression technique for all categories of Arabic texts.

Table 4.4 Compression ratio using LZW for Arabic texts.

File name	origenal size	size of compressed file \ vowelized	size of compressed file \ partially vowelized	size of compressed file\ unvowelized	CR/ vowelized	CR/ partially vowelized	CR/ unvowelized
text 1.txt	1.8 KB	0.93	0.96	0.94	46%	53%	52%
text 2.txt	2 KB	1.07	1.16	1.12	53%	58%	56%
text 3.txt	4 KB	1.78	1.89	1.78	44%	47%	44%
text 4.txt	5 KB	2.36	2.53	2.45	47%	51%	49%
text 5.txt	6 KB	2.44	2.57	2.50	41%	43%	42%
text 6.txt	7 KB	2.97	3.19	3.18	42%	46%	45%
text 7.txt	8 KB	3.13	3.36	3.30	39%	42%	41%
text 8.txt	10 KB	3.75	3.98	3.87	38%	40%	39%
text 9.txt	12 KB	4.34	4.63	4.43	36%	39%	37%
text 10.txt	26 KB	8.40	9.20	8.70	32%	35%	33%
text 11.txt	27 KB	8.62	9.34	9.24	32%	35%	34%
text 12.txt	29 KB	9.09	9.88	9.60	31%	34%	33%
text 13.txt	30 KB	9.31	9.99	9.75	31%	33%	33%
text 14.txt	35 KB	11.41	12.40	12.10	33%	35%	35%
text 15.txt	36 KB	11.27	11.91	11.68	31%	33%	32%
text 16.txt	41 Kb	12.33	13.40	12.87	30%	33%	31%

text 17.txt	**62 KB**	**18.02**	**19.91**	**19.33**	**29%**	**32%**	**31%**
text 18.txt	**82 KB**	**23.60**	**26.04**	**26.62**	**29%**	**32%**	**32%**

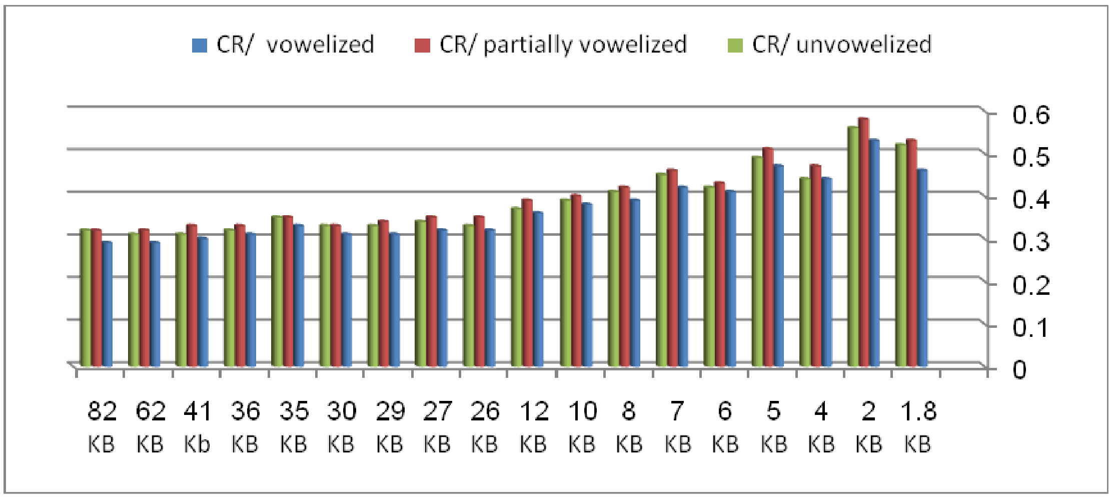


Figure 4.29 Compression ratios (CR) using LZW for Arabic texts.

From the above experiment, it is found that LZW was more suitable to compress vowelized texts than the other categories, followed by compressing unvowelized then partially texts respectively.

Figure 4.30 show the results of comparing the efficiency of compressing Arabic text and English texts.

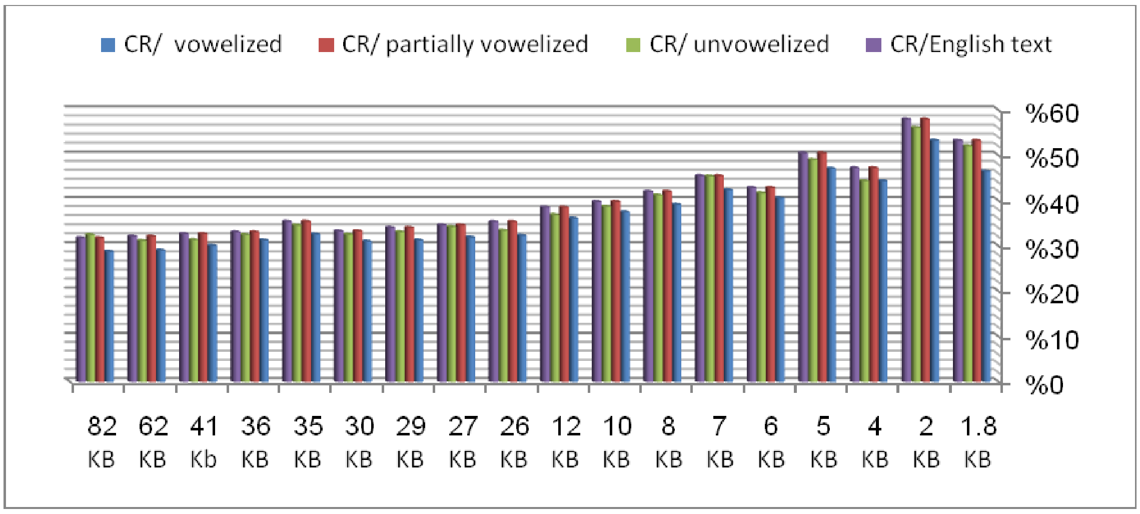


Figure 4.30 Compression Ratios on Arabic and English text using LZW.

From the above mentioned experiments, we can conclude that LZW was the best technique to compress Arabic texts (for all of their categories, vowilized ,non vowilized, and partial vowilized) rather than for English texts.

Figure 4.31 shows that the effect of file size on compression for each category of Arabic texts. When the file size increases, the compression ratio on Arabic text decreases for all the vowelization state. But vowelized texts give better compression ratio than the other state. Comparable results were also obtained for English texts.

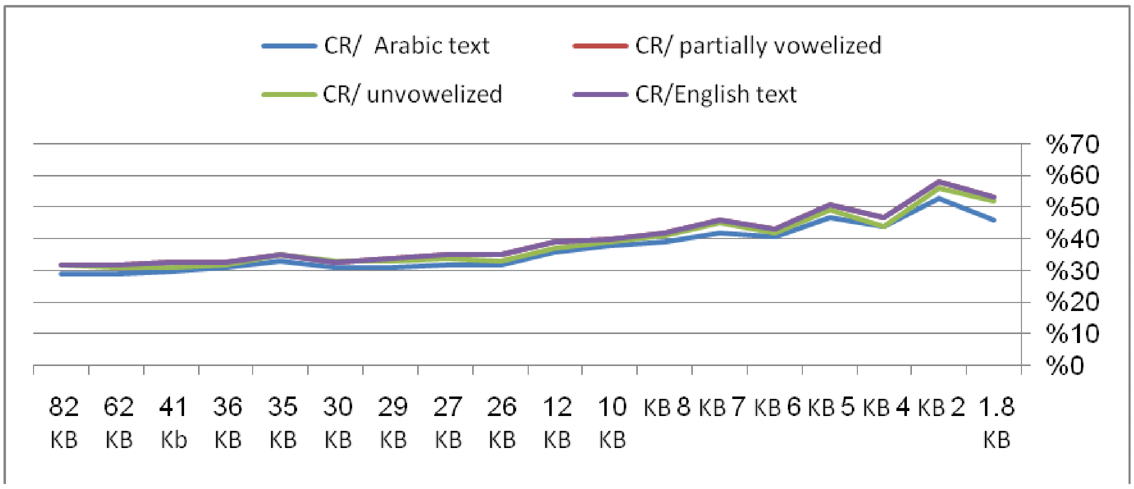


Figure 4.31 The effect of file size on compression for LZW.

4.3.1.2. Experiment the BWT Technique.

For measuring the performance of compressing Arabic Language compression using BWT technique, the previous experiments had been applied on the same files.

A series of experimental tests had been performed to compress both Arabic and English texts of different sizes using BWT technique. Table 4.5 and Figure 4.32 show results of the compression ratio using BWT compression technique for an the categories of Arabic texts.

Table 4.5 Compression ratio using BWT for Arabic texts.

File name	origenal size	size of compressed file \ vowelized	size of compressed file \ partially vowelized	size of compressed file\ unvowelized	CR/ vowelized	CR/ partially vowelized	CR/ unvowelized
text 1.txt	1.8 KB	0.76	0.83	0.82	38%	46%	45%
text 2.txt	2 KB	0.97	1.08	1.04	48%	54%	52%
text 3.txt	4 KB	1.72	1.88	1.71	43%	47%	43%
text 4.txt	5 KB	2.35	2.65	2.61	47%	53%	52%
text 5.txt	6 KB	2.31	2.55	2.53	38%	43%	42%
text 6.txt	7 KB	2.96	3.35	3.44	42%	48%	49%
text 7.txt	8 KB	3.17	3.54	3.55	40%	44%	44%
text 8.txt	10 KB	3.72	4.26	4.27	37%	43%	43%
text 9.txt	12 KB	4.42	5.00	4.92	37%	42%	41%
text 10.txt	26 KB	8.97	10.27	10.04	35%	39%	39%
text 11.txt	27 KB	8.99	10.31	10.45	33%	38%	39%
text 12.txt	29 KB	9.37	10.84	10.99	32%	37%	38%
text 13.txt	30 KB	9.61	10.79	11.25	32%	36%	38%
text 14.txt	35 KB	12.29	14.16	14.78	35%	40%	42%
text 15.txt	36 KB	11.55	13.04	13.29	32%	36%	37%
text 16.txt	41 Kb	12.41	14.11	14.47	30%	34%	35%
text 17.txt	62 KB	19.21	22.03	22.76	31%	36%	37%
text 18.txt	82 KB	25.47	29.36	31.39	31%	36%	38%

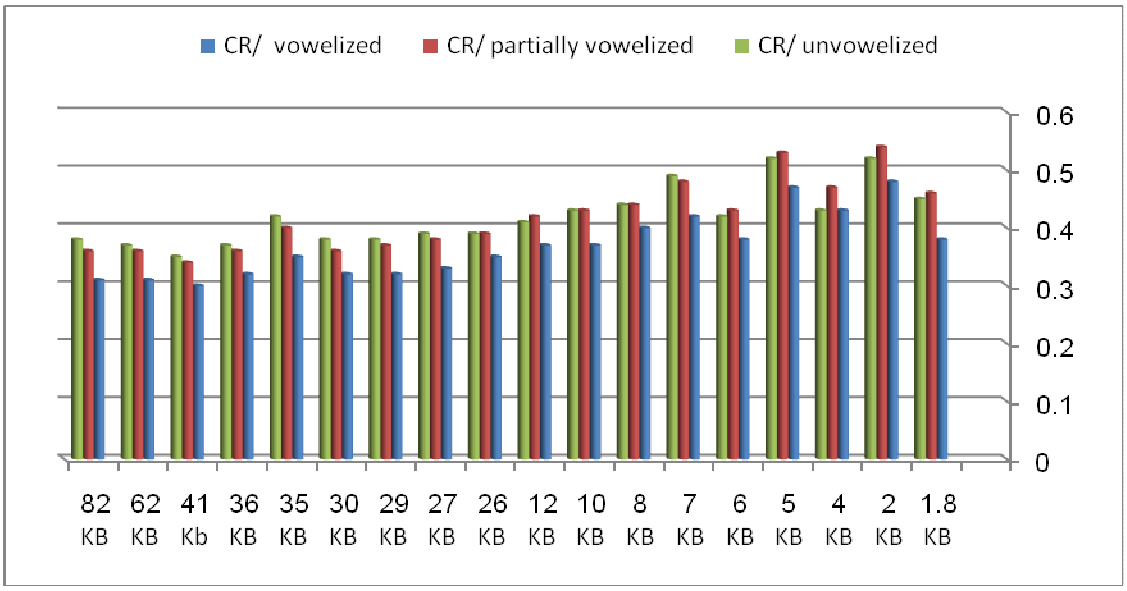


Figure 4.32 Compression ratio using BWT for Arabic texts.

From the above experiment, it was found that BWT was more suitable to compress vowelized texts than other categories. Compression ratio for unvowelized and partially texts were relatively similar.

Figure 4.31 show results of comparing the efficiency of compressing Arabic texts and English texts using BWT to compress the texts files.

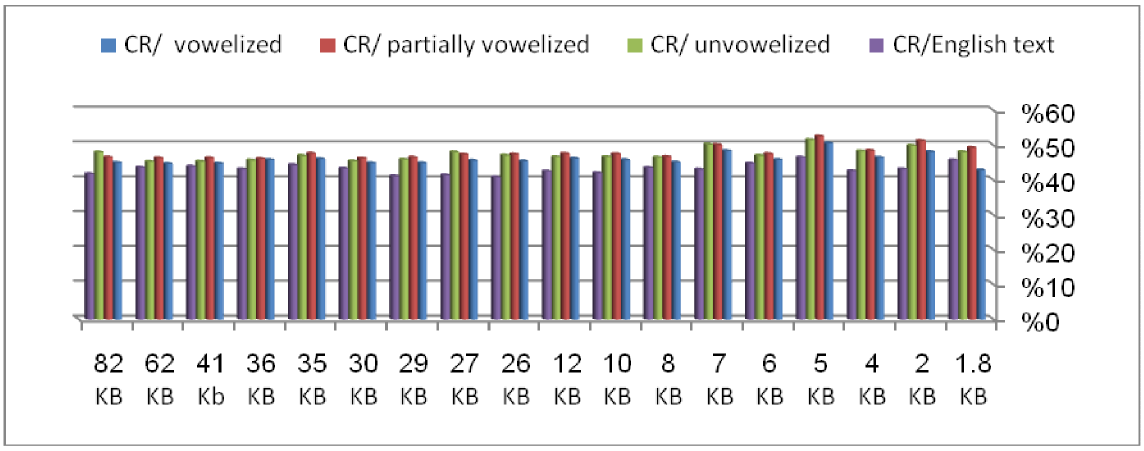


Figure 4.31 Compression Ratio on Arabic and English Text using BWT.

From the above experiments, it can be concluded that BWT was the most suitable technique for compressing vowilized texts compared to other type of text (non vowelized, and partial vowelized), The results also showed better performance for Arabic text than for English texts.

Figure 4.34 shows the effect of file size on compression for each category of Arabic texts. When the file size increases, the compression ratio on Arabic text decreases slightly for all the vowelization state .

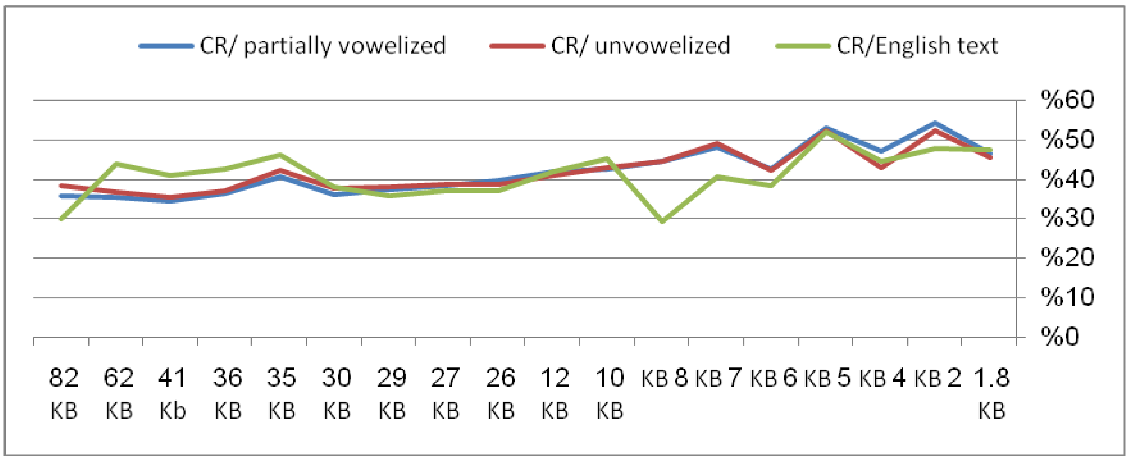


Figure 4.34 The effect of file size on compression using BWT.

Results Analysis

The performance of the data compression depends on: the features of the files, the different symbols contained in it, and symbols frequencies. From the first experiment, it is was noted that compression efficiency of Arabic texts using LZW was very high for all vowelization states, and it give better results than when it was used for compressing English texts.

The most important features of the Arabic language is that it has a single case of each letters, unlike the English language which has 26 letters, and each one has two cases (upper and lower case), In addition, the diacritical marks, which their number is low, just 9 and has high frequencies, hence the ratio of the compression rate of an Arabic text is better than the compression of an English text, Specially when the Arabic text is for vowelized texts.

From the second experiment, it was noted that the BWT technique was suitable just to compress vowelized texts compared with English texts because it contained of the diacritical marks. BWT technique was very sensitive to the structure of the word.

Table 4.6 Compression ratio for LZW and BWT techniques.

Texts type	Text category	LZW	BWT
Arabic texts	vowelized texts	37%	37%
	Un vowelized texts	39%	42%
	Partilly vowelized texts	40%	42%
English texts		40%	41%

According to the Table 4.6 the compression ratio using LZW is best technique to compress Arabic texts especially for vowelized texts, then BWT technique.

The difference in the compression ratio is related to the different mechanisms of the techniques in the compression process; which depends ons the LZW that replaces strings of characters with single codes.

BWT depends on preprocessing the text before the compression process by data compression technique.

4.3.1. The Second Group of Experiments.

Depending on the results of the first tests of techniques, which aimed to evaluated compressing efficiently of Arabic texts, both LZW and BWT will be used to test and evaluate the proposed method. The first proposed method aims to improve performance the LZW to compress the Arabic texts.

Table 4.7 Compression ratio for single techniques.

Text Category	File Size	LZW	BWT	LZWA
Vowelized texts	Text1 (310 KB)	0.30	0.27	0.26
	Text2 (570 KB)	0.30	0.30	0.27
Unvowelized texts	Text3 (200 KB)	0.31	0.30	0.27
	Text4 (558 KB)	0.32	0.36	0.30
Partilly vowelized texts	Text5 (216 KB)	0.33	0.38	0.27
	Text6 (360 KB)	0.35	0.39	0.30

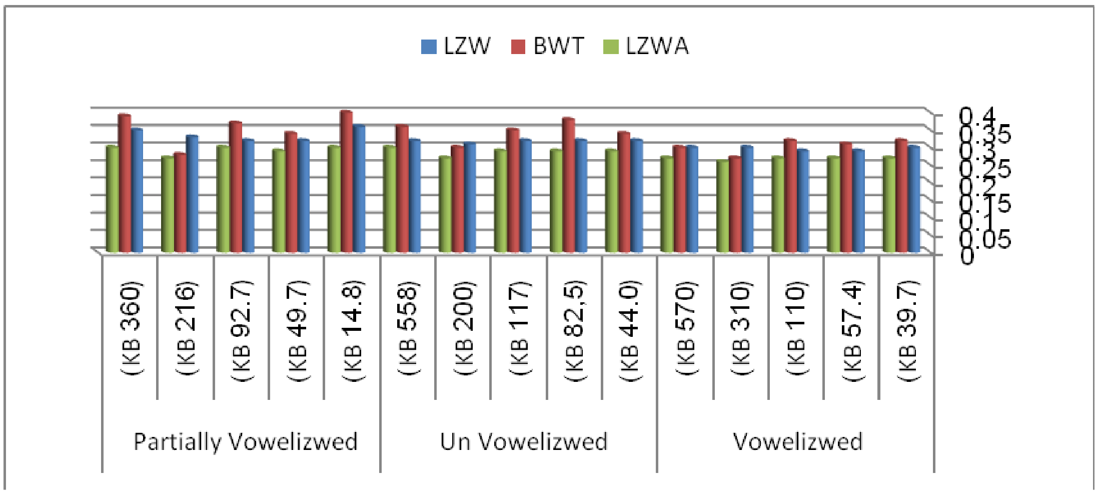


Figure 4.35 Compression Ratios for LZWA, LZW and BWT.

Table 4.7 and Figure 4.35 show the files size and the compression ratio for each file for each technique. It is noted that LZWA was the best for all categories of the Arabic texts, then LZW and BWT respectively. But we must note that, LZWA was suitable just for Arabic texts because the dictionary contains characters of Arabic languages.

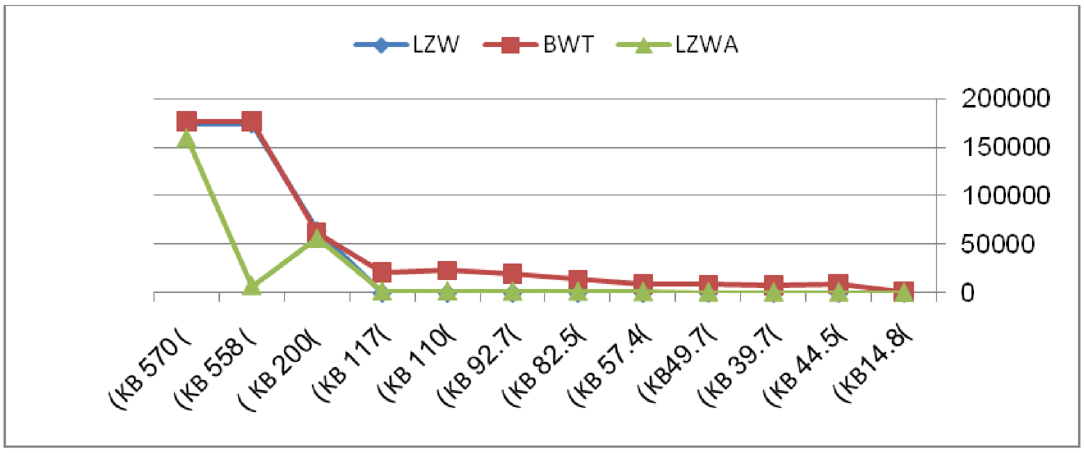


Figure 4.36 Compression times for single technique.

Figure 4.36 depicts the compression times of all the techniques. Compression times of the LZWA and LZW are the same, but compression rate using LZWA is lower than other compression technique.

One of the most important goals of this thesis is to exploit the characteristics of the morphological Arabic language. Multilayer model has been selected to get the less compression ratio as possible. The main idea for this model in that the text is split into smaller linguistically "homogeneous" layers representing the main categories of words. To integrate multilayer with hybrid compression techniques, several experiences had been made to detect which data compression techniques were suitable for each layer separately.

According to Figures [4.37 – 4.39], BWT was the best technique to compress all the layers. Compression ratio for first layer was 50% when BWT was applying, 83% when LZW was applying. Compression ratio for the second layer was 54%,75% for BWT and LZW respectively, as for the third layer was 41%, 49% for BWT and LZW respectively. Table 11 shows results of encoded data and size of the compressed files using multilayer model analyzer and lossless techniques, where they integrate Multilayer model with BWT that gives better results than using it with LZW.

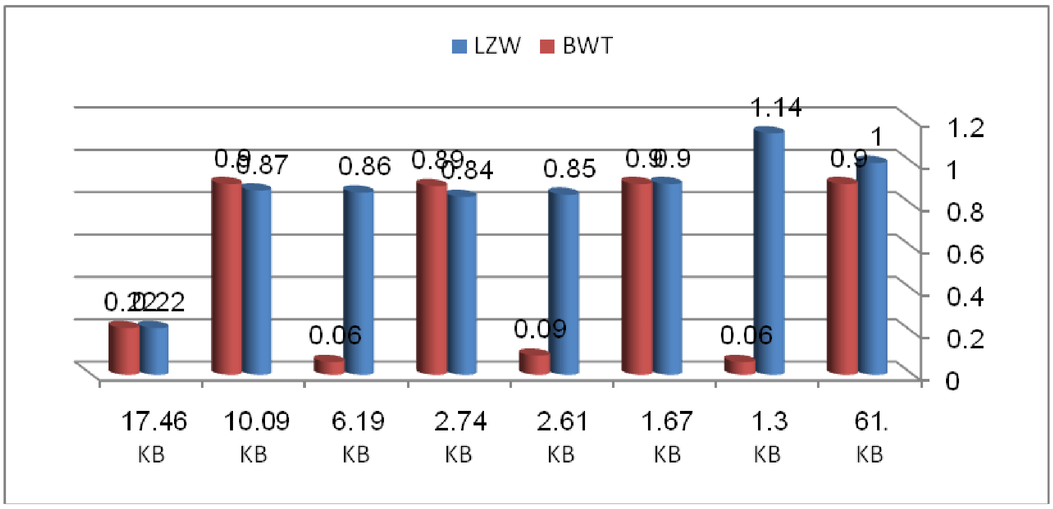


Figure 4.37 Compression ratio for first Layer.

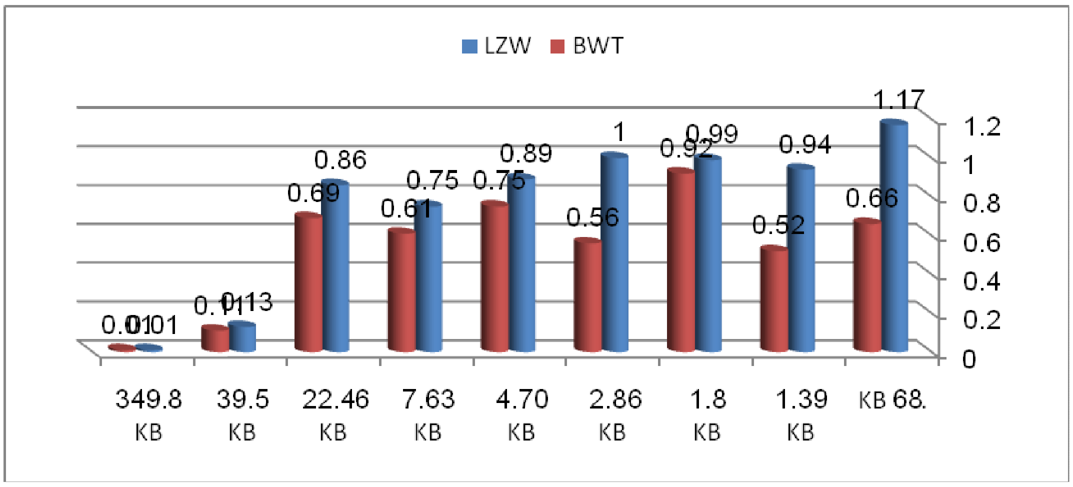


Figure 4.38 Compression ratio for second Layer.

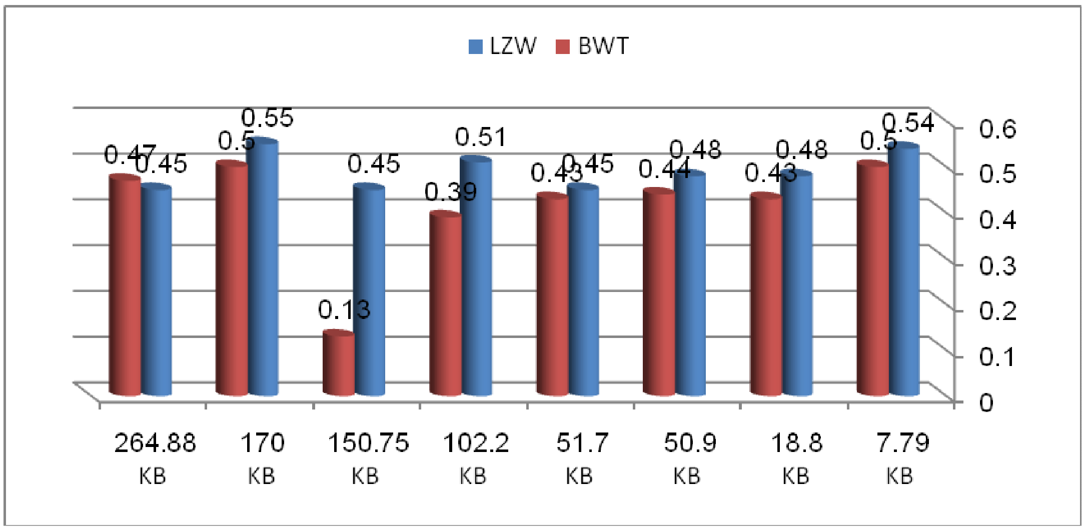


Figure 4.39 Compression ratio for the third Layer.

Table 4.8 Size of the compressed files using Hybrid technique.

Text Category	File Size	Multilayer Model	Multilayer With LZW	Multilayer With BWT
Vowelizwed Texts	Text1 (310 KB)	Layer1 (2.61)	2.24	0.26
		Layer2 (1.39)	1.31	0.73
		Layer3 (150.75)	68.47	20.74
		Mask (4.39)		
	Text2 (570 KB)	Layer1 (6.19)	5.38	0.43
		Layer2 (2.86)	2.89	1.62
		Layer3 (264.88)	121	125
		Mask (7.78)		
Un vowelizwed Texts	Text3 (200 KB)	Layer1 (2.74)	2.32	2.45
		Layer2 (7.63)	5.78	4.66
		Layer3 (88.09)	41.2	31.3
		Mask (4.16)		
	Text4 (558 KB)	Layer1 (10.09)	8.82	9.18
		Layer2 (22.46)	19.42	15.6
		Layer3 (231.81)	11.7	103
		Mask (11.65)		
Partially Vowelizwed Texts	Text6 (215 KB)	Layer1 (152)	1.96	1.79
		Layer2 (349.8)	4.24	3.15
		Layer3 (102.2)	52.5	40.15
		Mask (4.2)		
	Text5 (352 KB)	Layer1 (17.46)	3.85	3.90
		Layer2 (39.5)	5.44	4.51
		Layer3 (170)	95.3	85.4
		Mask (6.52)		

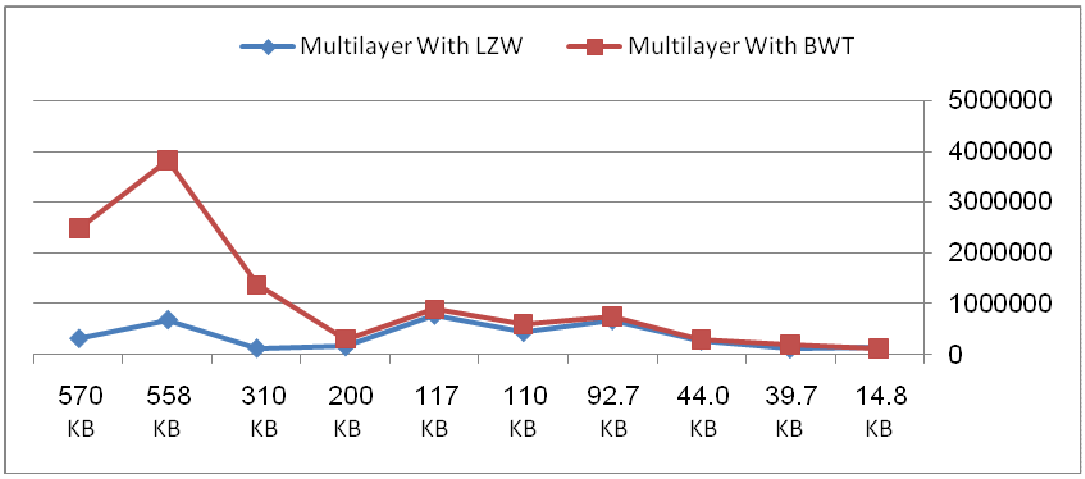


Figure 4.40 Compression times for Multilayer with LZW and BWT.

Figure 4.40 represents the compression time in milliseconds that is needed for encoded and compressed each word in the source files to complete compression in the hybrid techniques. The best (smallest) time is when using LZW with multilayer model and in BWT is the worst (long) time.

Depending on the results of the first group of experiments, which aimed to evaluate compress efficiently Arabic texts, both of LZW and BWT were used in order to evaluate performance of proposed hybrid technique for Arabic texts compression. Table 4.9 shows the compression ratio for the multilayer with BWT was the best.

Text Category	File Size	BWT	LZW	LZWA	Multilayer With LZW	Multilayer With BWT
Vowelizwed	44.0 KB	0.34	0.32	0.29	0.22	0.26
	110 KB	0.32	0.29	0.27	0.25	0.22
	310 KB	0.27	0.3	0.26	0.25	0.08
	570 KB	0.3	0.3	0.27	0.24	0.23
Un vowelizwed	14.8 KB	0.4	0.36	0.3	0.3	0.3
	117 KB	0.35	0.32	0.29	0.28	0.26
	200 KB	0.3	0.31	0.27	0.26	0.23
	558 KB	0.36	0.32	0.3	0.09	0.25
Partially Vowelizwed	215 KB	0.28	0.33	0.27	0.29	0.22
	352 KB	0.39	0.35	0.3	0.31	0.28
Average		**0.33**	**0.32**	**0.28**	**0.25**	***0.23***

Table 4.9 Compression Ratio for single techniques and hybrid.

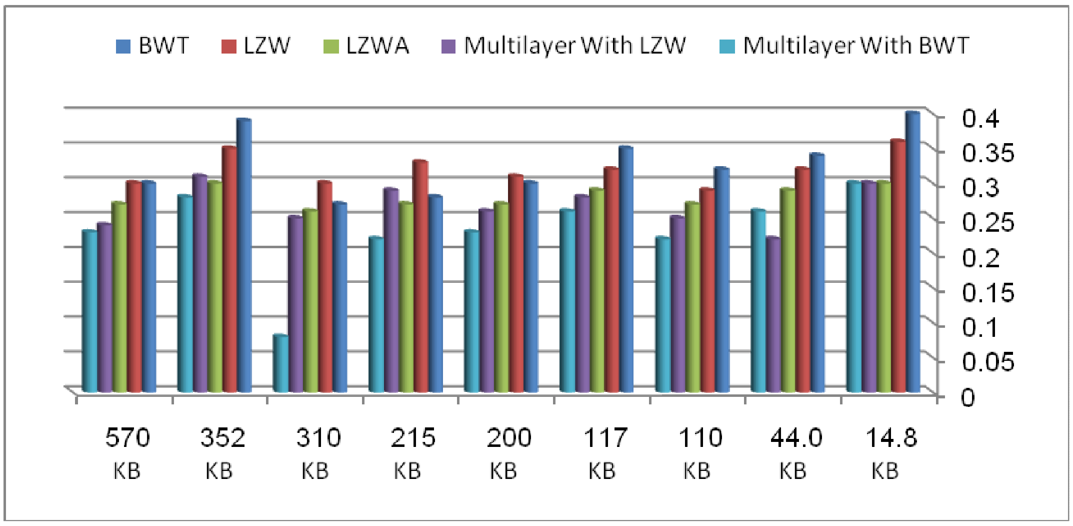


Figure 4.41 Compression Ratios for single and hybrid techniques.

According to the Table 4.9, and Figure 4.41, the compression ratios for hybrid technique (Multilayer with BWT) were the best. The cost to be paid in this case is related to the time efficiency where the compression time is higher with BWT as shown in Figure 4.42.

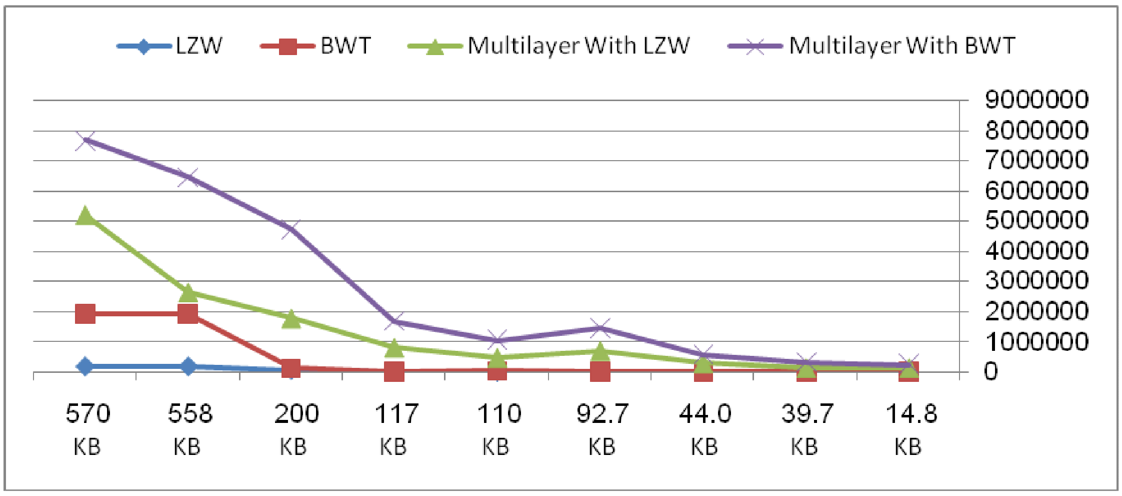


Figure 4.42 Compression times for single and hybrid techniques.

We conducted from all the above experiments, In Hybrid technique for Arabic text data compression, a better compression ratio is achieved compared to single text data compression. Additional to there is a tradeoff between size of the files and the speed of a data compressor and the level of compression it can achieve. As the file size increases the compression times increases for Arabic text, but the compression ratio decreases.

Chapter 5

Conclusion and Future Work

5.1. Conclusion

Two techniques of data compression had been compared and evaluated. They were tested on the different category of Arabic texts (vowelized, partially vowelized and unvowelized). Their performances were studied in term of compression size, compression ratio, and compression time. After testing those techniques on different files of different sizes, we can conclude that LZW was the best one for all categories of the Arabic texts in all compression scales that we had tested, then BWT.

The English texts had been used to compare the performance of the three techniques against their performance with Arabic texts. It was found that as the file size increased, the compression ratio decreased for both Arabic and English texts. Additionally, results had shown that in Arabic text data compression, a better compression ratio was achieved compared to English text compression.

The fact that Arabic letters have single case had been used in improve the performances of LZW. The proposed approach (LZWA) had achieved a better compression ratio compared with LZW and BWT.

A hybrid technique had been developed in order to achieve high compression ratio. It depended on the integration of the Multilayer model of Arabic texts with BWT. Such approach was useful to compress textual files. This technique relied on exploiting the morphological features of the Arabic language to improve the performance of BWT, where the Multilayer model was integrated with BWT. This approach gave better compression ratio than integrating the same model with LZW, or with hybrid technique. The cost to be paid in this case was related to the time efficiency where the compression time was higher with BWT.

5.2. Future Work.

The compression algorithm would be able to take advantage of long-range correlations between words and thus achieve better compression (Horspool and Cormack, 1992). Thus we suggest using word-based technique as pre-processing step to implement lossless data compression techniques on Arabic texts in general, and on derivative words in particular to take advantage of morphological features where tens or hundreds of words can be derived from the same root.

At the multilayer model, we suggest extracting the root of the word from the derivative word pattern, and then storing them in two separate files, in order to replace the pattern with index value, and applying word-based technique using a suitable lossless compression technique to compress roots file, to take advantage of recurrence of the roots.

6. References

Abel, J. ,(2003). "Improvements to the Burrows-Wheeler Compression Algorithm: After BWT Stages", available on www.juergenabel.info/Preprints/Preprint_After_BWT_Stages.pdf Visited on March 2013.

Akman, I. , Bayindir, H. , Ozleme, S. , Akin, Z. and Misra, Sanjay (2011). "Lossless Text Compression Technique Using Syllable Based Morphology". **The International Arab Journal of Information Technology**, VOL.(8) No. (1). pp (66-74).

Alkhalil (2010),"Alkhalil Morpho Sys: A Morphosyntactic analysis system for Arabic texts". available on http://www.itpapers.info/acit10/Papers/f653.pdf Visited on March 2013.

Al-laham, M. , and I. M. El Emary (2007),"Comparative Study Between Various Algorithms of Data Compression Techniques", **Proceedings of the World Congress on Engineering and Computer Science WCECS, October 24-26, 2007, San Francisco, USA.**

Alasmer , Z. M. , Zahran B. M., Ayyoub B. A., Kanan M. A. (2013)."A Comparison between English and Arabic Text Compression". **Journal of Contemporary Engineering Sciences**, Vol. (6), No. (3), pp. (111-119).

Alqrainy, S. (2008). A Morphological-Syntactical Analysis Approach For Arabic Textual Tagging. PhD Thesis, De Montfort University, Leicester, UK.

ALESCO, “Arabic Language Derivation and morphological System,” Published by the Arab League Educational, Cultural and Scientific Organization, http://www.reefnet.gov.sy/ed4-2. htm , Last Visited 2013.

Al-Serhan, H. and Ayesh, A. (2006). "A Triliteral Word Roots Extraction Using Neural Network For Arabic". **IEEE International Conference on Computer Engineering and Systems (ICCES06)**, pp. (436-440). Cairo, Egypt.

Al-Shalabi, R. (2005). "Pattern-based Stemmer for Finding Arabic Roots". **Information Technology Journal**, Vol .(4) No.(1), pp (38-43).

Al-Shalabi, R., Kanaan, G. and Al-Serhan, H. (2003). "New approach for extracting Arabic roots". **in ACIT '2003: Proceedings of The 2003 Arab conference on Information Technology**, Alexandria, Egypt.

Al-Shammari, E. and Lin, J. (2008)."A novel Arabic lemmatization algorithm". In Proceedings of the second workshop on Analytics for noisy unstructured text data, Singapore: ACM,113-118 (On-Line), available: http://dl.acm.org/citation.cfm?id=1390767&dl=ACM&coll=DL&CFID=327509179&CFTOKEN=82258935, Last Visited 2013

Al-Sughaiyer, I. A. and Al-Kharashi, I. A. (2004). “Arabic Morphological Analysis Techniques:A Comprehensive Survey”, **Journal of the American society for information science and technology**, Vol.(55) No.(3),pp (189 – 213).

Al-Sughaiyer, I. A. and Al-Kharashi, I. A. (2004). "Arabic morphological analysis techniques: A comprehensive survey". **Journal of the American Society for Information Science and Technology**. Vol.(55), No(3), pp. (189-213).

Altarawneh .H and Altarawneh .M (2011), "Data Compression Techniques on Text Files: A Comparison Study". **International Journal of Computer Applications**.Vol.(26) No.(5), pp (0975 – 8887).

Awajan A. (2007). "Arabic Text Preprocessing for the Natural Language Processing Applications". **Arab Gulf Journal of Scientific Research.** Vol. (25), No.(4). pp (179-189).

Awajan,.A (2011). "Multilayer Model for Arabic Text Compression", **The International Arab Journal of Information Technology**, Vol.(8) No(2), pp (188-196).

Blelloch, G. E., (2010). Introduction to Data Compression, Computer Science Department Carnegie Mellon Universit. 22-41 (On-Line), available: http://www.cs.cmu.edu/~guyb/realworld/compression.pdf , Last Visited 2013.

Boudlal, A., Belahbib, R., Lakhouaja, A., Mazroui, A., Meziane, A. and Bebah, M. O. A.O. (2011). "A Markovian Approach for Arabic Root Extraction". **The International Arab Journal of Information Technology. Vol.(**8) No(1), pp (91-98).

Cahill, Lynne. (2007). A Syllable-based Account of Arabic Morphology. In Abdelhadi Soudi, Antal van der Bosch and Günther Neumann (eds.) **Arabic Computational Morphology Dordrecht**. Springer. pp. (45-66).

Daoud, A. M. (2010). "Morphological Analysis and Diacritical Arabic Text Compression", **The International Journal of ACM Jordan (ISSN 2078-7952)**, Vol.(1) No (1), pp (41-49).

Dichy, J. (2001). On lemmatization in Arabic, A formal definition of the Arabic entries of multilingual lexical databases. ACL/EACL 2001 Workshop on Arabic NLP, Toulouse, France, 16 July 2001.

Dichy, J. and Farghaly, A. (2003). Roots & patterns vs. stems plus grammar-lexis specifications: on what basis should a multilingual database centred on Arabic be built? MT Summit IX -- workshop: Machine translation for semitic languages, New Orleans, USA. (On-Line), available: http://www.amtaweb.org/summit/WS2/Dichy+Farghaly_paper.pdf, Last Visited 2013.

Ghwanmeh, S., Al-Shalabi, R. , and Kanaan, G. (2006). "Efficient data compression scheme using dynamic Huffman code applied on Arabic language". **Journal of Computer Science**, Vol.(2), pp (885-888).

Hasan, R. (2011), "Data Compression using Huffman based LZW Encoding Technique", **International Journal of Scientific & Engineering Research**, Vol.(2) No(1), pp (1-7).

Horspool, R. N. and Cormack, G. V. (1992), "Constructing word-based text compression algorithms" (On-Line), available: http://pdf.aminer.org/000/146/372/constructing_word_based_text_compression_algorithms.pdf, Last Visited 2013.

Jurafsky, D. and Martin, J. H. (2008). Speech and Language Processing, 2th ed. New Jersey: Prentice Hall. (On-Line), available: http://www.cs.colorado.edu/~martin/SLP/Updates/1.pdf , Last Visited 2013.

Khafagy , M. A. M. (2005)."Arabic Text Data Compression", PhD thesis, Zagazig University.

Khoja, S. (2001). "APT: Arabic Part-of-Speech Tagger". In: Proceedings of the student Workshop at the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL2001), Carnegie Mellon University, Pittsburgh, Pennsylvania. (On-Line), available: http://zeus.cs.pacificu.edu/shereen/NAACL.pdf, Last Visited 2013.

Khoja, S. (2003). APT: An Automatic Arabic Part-of-Speech Tagger. PhD thesis, Lancaster University.

Kodituwakku S.R. , Amarasinghe U. S. (2012), Comparison Of Lossless Data Compression Algorithms for Text Data", **Indian Journal of Computer Science and Engineering**. Vol.(1) No.(4), pp (416-425).

Lourdusamy, R. , Shanmugasundaram, S. (2011) " A Comparative Study Of Text Compression Algorithms", **International Journal of Wisdom Based Computing**, Vol.(1), No.(3),pp (68-76).

Manber U. (1999). "A Text Compression Scheme that Allows Fast Searching Directly in the Compressed File". **Computer Journal of ACM Transactions on Information Systems**, Vol.(15) No.(2), pp (124-136).

Moronfolu, A. , Oluwade, D. (2009), "An enhanced LZW text compression algorithm", **Afr. J. Comp. & ICT** ,Vol.(2) No.(2), pp (13-20).

Omer, E. and Khatatneh, .K (2010). "Arabic Short Text Compression". **Journal of Computer Science**, Vol.(6) No(1), pp (24-28).

Pauw, G. D. and Schryver, G.-M. D. (2008). "Improving the Computational Morphological Analysis of a Swahili Corpus for Lexicographic Purposes". ***the 13th International Conference of the African Association for Lexicography***, Republic of South Africa, 1–3 July 2008.

Radescu, R. (2009). "Transform methods used in lossless compression of text files". **Romanian journal of information science and technology,** Vol.(12) No. (1) , pp (101-115).

Teahan J., McNab R., and Witten H., (2000). "A Compression) based Algorithm for Chinese Word Segmentation" **Computer Journal of Computational Linguistics**,Vol.(26) No.(3), pp (375-392).

Saad M. (2011), Arabic-Corpora, http://sourceforge.net/projects/ar-text-mining/files/Arabic-Corpora/ Last Visited 2013.

Sawalha, M. S. (2011).Open-source Resources and Standards for Arabic Word Structure Analysis: Fine Grained Morphological Analysis of Arabic Text Corpora, The University of Leeds.

Sawalha, .M; Atwell .E, Eric (2010). "Constructing and Using Broad-Coverage Lexical Resource for Enhancing Morphological Analysis of Arabic". ***in: Proceedings of the Language Resource and Evaluation Conference LREC*** 2010, 17-23 May 2010, Valleta, Malta.

Soudi, A. V. Bosch and G. Neuman (eds.). (2007). Arabic Computational Morphology. New York, Springer.

Štujbe, V. (2008). Practical data compression, Master's thesis, Commenius University, Bratislava.

Wiseman Y. and Gefner I. (2007), "Conjugation-based Compression for Hebrew Texts", **Computer Journal of ACM Transactions on Asian Language Information Processing**, vol.(6), No.(1) , pp. (1-10).

Yousfi, A. (2010). "The morphological analysis of Arabic verbs by using the surface patterns". **IJCSI International Journal of Computer Science Issues**. Vol.(7), No.(3), pp (33-36).