Multilayer Model for Arabic Text Compression

Arafat Awajan
Department of Computer Science, Princess Sumaya University for Technology, Jordan

Abstract: *This article describes a multilayer model-based approach for text compression. It uses linguistic information to develop a multilayer decomposition model of the text in order to achieve better compression. This new approach is illustrated for the case of the Arabic language, where the majority of words are generated according to the Semitic root-and-pattern scheme. Text is split into three linguistically homogeneous layers representing the three categories of words: derivative, non-derivative and functional words. A fourth layer, called the Mask, is introduced to aid with the reconstruction of the original text from the three layers in the decoding side. Suitable compression techniques are then applied to the different layers in order to maximize the compression ratio. The proposed method has been evaluated in terms of the rate of compression it provides and its time efficiency. Results are shown along with real texts to illustrate the performance of the new approach. The novelties of the compression technique presented in this article are that (1) the morphological structure of words may be used to support better compression and to improve the performances of traditional compression techniques; (2) search for words can be done on the compressed text directly through the appropriate one of its layers; and (3) applications such as text mining and document classification can be performed directly on the compressed texts.*

Keywords: *Text compression, morphological analysis, multilayer based compression, word lexical form, word based compression, and Arabic language.*

Received March 12, 2009; accepted November 5, 2009

1. Introduction

The primary aim of using data compression algorithms is to reduce the size of data. This saves both storage space and transmission time. The continual growth of the information highway and the increasing amount of texts available in electronic libraries and on the internet make the development of efficient techniques for compressing textual documents written in natural languages one of the main areas of research in data compression.

There are two main categories of text compression algorithms. The first one is the dictionary-based compression algorithms. These algorithms are generally of the Ziv-Lempel type and they replace a string with a pointer to an earlier occurrence of the same string. The second category is the statistics-based compression algorithms. These algorithms are in general based on Huffman encoding or arithmetic coding and they exploit the uneven frequency distribution of symbols, especially the dependence of symbols on their neighboring context [11, 15]. The symbol-level redundancy is then reduced via the usage of variable-length coding methods such as that of Huffman. These methods assign the shortest code to the most frequently used characters.

Most natural language text compressors use general-purpose data compression techniques and perform compression at the character-symbol level [16]. Some text compressors are word-based compressors that use words as the basic units for performing compression [9, 13, 14, 18]. Some word-based compressors, like the word-based Huffman compression, consider words as symbols of a new alphabet and then apply a statistical-based compressor [25]. There are two main problems related to this approach: the number of distinct words is unbounded, and the symbols of the generated alphabet must be transmitted along with the output of the compression algorithm, which reduces their performances. Other word-based compressors are developed using a dictionary-based compression scheme [13]. The main problem related to this approach is that such a dictionary is text-dependent and must be transmitted along with the compressed text.

2. Related Work

The majority of the available literature and works on text compression deal with English and other European languages [6, 13, 17, 24]. Some solutions are available for other languages, such as Japanese and Chinese [8, 19, 23] However, although Arabic is one of the most widely-used languages in the world, with more than 300 million speakers, there are few studies and published research papers on the compression of Arabic text. Available methods apply either a direct adaptation of techniques used for general-purpose data compression [12] or special techniques dedicated to specific categories of Arabic text such as unvowelized words, or words generated from a limited number of roots [2, 4].

The work presented in this article is based on the idea that the knowledge of the features of natural languages could be used to achieve high compression ratios. In some natural languages, the lexical forms of words can be used to represent the words differently. This new representation may be used to support a better compression or to improve the traditional compression technique [1, 10, 21, 22].

Arabic language in particular, and Semitic languages in general, provide good examples for illustrating this approach [2, 21]. In these languages, morphology stands as a very interesting paradigm on which new compression techniques may be developed. Most Arabic words are generated according to the Semitic root-and-pattern scheme. From one root, tens or hundreds of words (surface form) can be derived according to a few morphological patterns, called awzan in Arabic, using a limited number of affixes. A word may then be represented by its root along with its morphological pattern [3, 7].

The proposed compression technique proceeds in two phases. The first phase consists of the application of a word morphological analyzer on the input text. The morphological structures produced by the analyzer are then used to split the text into three layers representing the three main categories of Arabic words: derivative words, non-derivative words and functional words. In the second phase, since the structures of the three layers are different, different compression algorithms are used to compress them separately in order to obtain a better compression ratio. The compression techniques are chosen in a way that maximizes the compression ratio. A fourth layer, called Mask, is introduced to aid with the reconstruction of the original text from the other three layers during decoding.

Background concepts and an overview of the main features of Arabic language are provided in section 2. The impact of using diacritical marks in Arabic texts and the representation of words and patterns are presented in section 3. Section 4 presents a full description of the system, including the construction of the multilayer model of the text, the compression techniques and the decoding algorithm. Experiments and results showing the performances of the proposed approach are discussed in section 5.

3. Features of the Arabic Language

3.1. Structure of Words

A word is a sequence of letters from an alphabet, herein the Arabic alphabet, terminated with a separator. Each letter in an Arabic word may be followed by a special mark called a diacritical mark. The diacritical marks are used to mark the short vowels. According to the Unicode standard, the Arabic alphabet is represented by 44 characters: 28 basic letters, 8 duplicated letters and 8 diacritical marks [20].

3.2. Categories of Words

Words in the Arabic language are classified into two categories: derivative words, and non-derivative words. Derivative words are generated from basic entities called roots, according to a set of derivation rules or morphological patterns. Non-derivative words are those words that do not obey the standard derivation rules. Functional words and words borrowed from foreign languages are examples of the non-derivative words.

Functional words are the most common words in texts. They include pronouns, prepositions, conjunctions, question words, etc. The most frequently used functional words are limited in number (there are about 100 functional words), but their frequency in texts is very high. For the purposes of this work, we consider the functional words, generally called tools, to comprise the third category of Arabic words.

3.3. Morphological Patterns

The majority of Arabic words are derivative words. All kinds of words (verbs, nouns, adjectives and adverbs) are generated from roots based on a limited number of standard patterns or templates. The pattern used to generate a word determines its various attributes, such as gender (masculine/feminine), number (singular/dual/plural), and tense (past/present/imperative). The larger part of Arabic derivative words is generated from three-letter roots and four-letter roots.

Prefixes and suffixes can be attached to a word to build more complicated derivations. For the purpose of this work, additional morphological patterns, called extended patterns, are introduced. They are built by adding the most common additive parts (suffixes or prefixes) to the standard morphological patterns used for generating names and adjectives. Table 1 shows examples of the standard patterns and extended patterns.

Based on the above, a derivative Arabic word can be represented lexically by its root and its morphological patterns. The most commonly used roots in the modern Arabic language number fewer than 5000 [7]. The morphological patterns, including the extended patterns, form a countable set. According to the system published by the Arab League Educational, Cultural and Scientific Organization [5], about 3500 patterns and extended patterns are used to generate words from the 3-letter roots and 4-letter roots. The number of patterns is significantly reduced in the case of unvowelized texts, and less than 50% of the patterns are commonly-used in modern texts.

4. Representation of Words and Patterns

With regard to the use of diacritical marks, Arabic texts may be classified into two main groups: vowelized and unvowelized texts. The first group represents the fully vowelized Arabic texts, in which every consonant is followed by a diacritical mark that may be placed below, above, or to its side [7]. This kind of text is found in school textbooks and in the holy Qur'an to ensure an absolutely correct reading. The second group consists of texts without diacritical marks. The majority of texts available in real-world applications such as books, newspapers, and texts on the internet belong to this category. In some cases, partially vowelized words may be found in the unvowelized texts, where some diacritical marks, generally one or two, are added in order to eliminate ambiguity in the meaning of these words. Since the partially vowelized words in real-world texts are limited in number, they will be compressed in this work as non-derivative words.

In order to deal with the two possible situations of Arabic words (vowelized, and unvowelized words), a word is split into two lists of characters: LC and LD. The first list (LC) contains the sequence of consonants forming the word [C1, C2, … Cn]. The second list (LD) contains the sequence of diacritical characters of the word [D1, D2, … Dn]. Table 2 shows examples of the decomposition of the words into the lists LC and LD for a vowelized word and an unvowelized word. An unvowelized word has an empty LD component. This representation allows for the processing of vowelized and unvowelized words using the same algorithm.

Table 1. Examples of standard patterns and extended patterns.

Group of Patterns	Examples of Patterns
Patterns used to generate verbs from 3-letter roots	يَفْعُلَ يَفْعُلُ يَفْعِلْ يَفْعِلَ يَفْعَلَنَّ يَفْعَلْ يَفْعَلَ يَفْعَلُ فَعِلَ فَعَلَ
Patterns used to generate verbs from 4-letter roots	يُفَعْلِلْ يُفَعْلِلَ يُفَعْلِلُ فَعْلَلَ
Patterns used to generate names from 3-letter roots	الفَاعل مُفْعِلٌ مُفَاعِلٌ مُنْفَعِلٌ مُفْتَعِـلٌ مُتَفَاعِـلٌ مُسْتَفْعِلٌ
Patterns used to generate adverbs from 3-letter roots	مَفْعِل مَفْعَلَة
Patterns used to generate adjective from 3-letter roots	فَعْلان فَعْلَى فَعْلانَة فَعِيل فَعِيلَة
Extended patterns used to generate verbs	تَفَاعَلْتُ تَفَاعَلْنَا تَفَاعَلْتَ تَفَاعَلْتُمَـا تَفَـاعَلْتُمْ تَفَاعَلُوا سَيَفْعَل فَعَلَهاَ فَعَلُوها
Extended patterns used to generate names and adjectives	الفَاعلُون الفَاعلَات المُفْعَلَة الفَعَّال فَعِيَل

Table 2. Decomposition of words into the two lists: LC and LD.

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]	[ ]

Table 3. Decomposition of patterns into the two lists: LC and LD.

Pattern (Standard/Extended)	List of Consonants LC	List of Diacritics LD
فَـعَـلَ	[٠٠٠]	[َ َ َ]
فُـعِـلَ	[٠٠٠]	[ُ ِ َ]
يَـفْـعَـلُـوْنَ	[ي ٠٠٠ و ن]	[َ ْ َ ُ ُ]
الـمُـتَـفَـاعِـلُـوْنَ	[ا ل ٠ ٠ ٠ ٠ و ن]	[ُ َ ِ . . .]

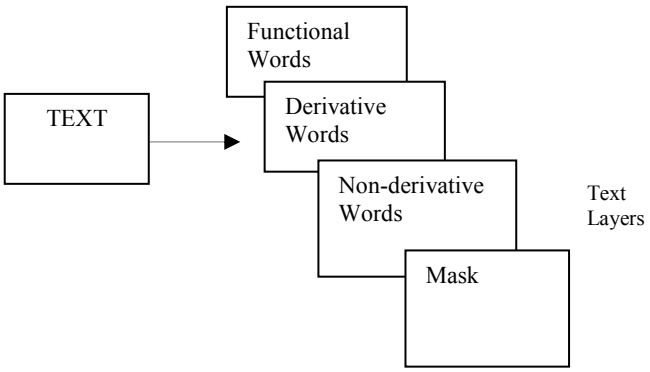

Figure 1. Text decomposition into the four layers.

The same idea is adopted for the morphological patterns, where each pattern is represented by two lists (LC) and (LD). The character '*' is used to represent slots whereby the consonants of a root can be inserted in the list LC to form a real word generated from that root. Table 3 shows examples of the decomposition of patterns (for both the standard pattern and extended pattern) into the LD and LC lists. Since groups of patterns share the same sequence of consonants and differ only in their list of diacritical marks, this representation significantly reduces the number of patterns to be represented and tested during the morphological analysis phase.

5. System Description

The proposed compression technique splits a text into three different layers which correspond to the different types of words, and then compresses them separately with different compressors in order to achieve a better compression. It works in two phases: the word morphological analysis phase and the encoding phase.

In the first phase, a word morphological analyzer is used to identify the category of each word in the input text. Therefore, the input text is decomposed into three basic layers representing the different categories of words: functional words, derivative words, and other words (i.e. non-derivative words, partially vowelized words, the separators except the space, and the words that the system fails to classify into one of the previous categories). A partially vowelized word is a word where one of its diacritical marks at least is omitted. A fourth layer, called the Mask, is introduced to link the words with their original positions in the input text. The Mask is very important for the decoding phase as

it provides the information needed to restore the original text from the three first layers as shown in Figure 1. In the second phase, each of the four layers is compressed individually using the compression method appropriate for the structure and nature of that layer.

5.1. The Word Analyzer

The word morphological analyzer takes each word of the input text and produces its type: derivative word, non-derivative word or a functional word. First, it determines if the word is vowelized or not, and then it decomposes the vowelized word into its LC and LD lists. The list LC of consonants is then tested against a predefined list of functional words. If it is not a functional word, the morphological analyzer determines if it is a derivative or non-derivative word. If the word is derivative, the word morphological analyzer identifies its morphological structure by decomposing it into its lexical components: root and pattern as shown in Figure 2.

The identification of the morphological structure of a word is based on a template-matching scheme. Two main tables are defined: The first table, "ROOTS," represents the most frequent roots of the Arabic language. The second table, "PATTERNS," includes the most commonly-used morphological patterns [5].
Each word of the input text is tested against the predefined set of standard and extended morphological patterns. This operation is performed at the level of the lists LC of the input word and of the patterns. The identification of the word pattern can be described by the following Prolog style recursive rule MATCH, where the character '*' is used to indicate the slots for the root letters in the pattern:

```
MATCH (LC(word), LC(Pattern)):
    MATCH ([],[]).
    MATCH ([Head|tail1], [Head|tail2]):
        MATCH (Tail1, Tail2).
    MATCH ([Head|tail1], [*|tail2]):
        MATCH (Tail1, Tail2).
```

The identification of the root is done by applying the rules that relate the patterns to their root. It uses the procedure FIND_ROOT (Pattern, a, b, c), where a, b and c are three integers determining the position of the letters of the root in the given pattern. The procedure EXTRACT-ROOT identifies the root by extracting its letters from the word.
The decomposition of the word into its root and pattern components is realized by the following rule DECOMPOSE, which calls the MATCH and FIND_ROOT rules:

```
DECOMPOSE (word, Root, Pattern):
    MATCH (LC (word), LC(Pattern)),
      FIND_ROOT (Pattern, a, b, c),
      EXTRACT_ROOT(LC(word), a, b, c, ROOT).
```

It should be noted that the analysis of the words does not have to be correct as it has to be in other works, which are aimed at achieving such objectives as text understanding, automatic translation or text summarizing. Since our aim is simply to achieve better compression, if the analysis yields that a given word is a derivative, when it is actually a non-derivative word, the compression will not be harmed and the reconstruction of the word in the decoding side will still be completely correct (Wiseman & Gefner, 2007).

Table 4. The most frequently used functional words in Arabic texts.

Categories of Functional Words	Examples
Personal Pronouns	أنا، نحن، أنت ، أنتما، أنتم، أنتن، هو، هي، هما، هم
Demonstrative Pronouns	هذا، هذه، ذلك، تلك، هذان، هاتان، هذين، هؤلاء، أولئك
Relative pronouns	الذي، التي، اللذان، اللتان، اللتين، الذين، اللائي، اللاتي، أي
Questions words	من، ما، متى، أيان، أين، أنى، كيف، كم
Conditional tools	إذا، لما، كلما، مهما
Prepositions	من، إلى، عن، على، في
Combined tools	لكنه ، لكنهما، لكنهم، كأنه،كأنها، عنه، فيه، منه، له ، به، بهما، بهم، بهن، ....

5.2. The Encoder

The encoder takes the output of the word morphological analyzer and builds the four layers of the text. Since the majority of words are separated by a space, an efficient way to handle words and separators is to use spaceless words (Ziviani et al. 2000). When a single space follows a word, the compression algorithm encodes only the word; otherwise, the algorithm encodes the word, and then encodes the separator. On the decoding side, a space character will be appended after each word unless it is followed by another separator. Therefore, the redundancy of space can also be compressed.

5.2.1. First Layer (Functional Words)

This layer is used to store the compressed representation of the functional words in the text. As noted earlier, the functional words are limited in number and their frequency in texts is very high. Tests on a large amount of texts of more than 5 Giga Bytes proved that more than 55% of words are functional. The most frequently used functional words found in this collection of texts are shown in Table 4.

A table for the most frequently occurring functional words was constructed. It is built based on the results obtained from tests on a huge amount of texts from the different categories (vowelized and non-vowelized), and contains 128 of the most frequently occurring functional words. Each functional word in the first layer is encoded using one byte: one bit indicates if the functional word is vowelized or not and the other 7 bits point to the position of the functional word in the table. The length of a functional word varies from 2 to 6

characters for unvowelized words and from 4 to 12 characters for the vowelized ones. The compression ratio produced by this representation is very good and the ratio of the compressed size to the original size is, on average, 0.25 for unvowelized functional words and 0.125 for vowelized functional words. Figure 3 shows the building process of the first layer.

5.2.2. Second Layer (Derivative Words)

The second layer represents the derivative words. Two tables are constructed to represent the roots and the patterns. We attached two pointers to each derivative word: one pointer points to its root and the second pointer points to its morphological pattern. The table of roots includes the 4096 most commonly used 3-letter and 4-letter roots. The table of patterns consists of the 2048 most used patterns and extended patterns. It has two entries for each pattern. One entry represents the list of consonants (LC) and the other entry represents the list of diacritical marks (LD). A derivative word is encoded using 3 bytes according to the following scheme:

- The first bit is used to indicate whether the word is vowelized or not.
- 12 bits are used to represent the root, making it possible to represent the most frequent 4096 roots.
- 11 bits are used to represent the pattern, making it possible to represent the most frequent 2048 standard and extended patterns.

Figure 4 shows the reconstruction process of the second layer from the original document using the codes of roots and patterns stored in the appropriate tables.

5.2.3. Third Layer (Non-Derivative Words)

The third layer represents the words that the system failed to classify into either of the first two layers. This includes: non-derivative words, unvowelized derivative words with a length less than or equal to 3 letters, misspelled words, and separators (with the exception of the space). These words will be stored in the third layer. A compression algorithm based on the word-based Huffman compression [25] is found to be very good for these types of words. The words of this layer are considered to form the symbols of an alphabet, and then a statistical-based compressor Huffman encoding is applied.

5.2.4. The Mask (Fourth Layer)

The Mask layer values are associated with each word of the original text and represent its layer type. When a value is 0, then the corresponding word is a functional word encoded in the first layer. When it is 1, then the corresponding word is derived from a root according to a given standard pattern and encoded in the second layer. When it is 2, the corresponding word is a non-derivative word, misspelled word, separator other than the space, or a word not classified in the previous categories and therefore, encoded in the third layer.

5.3 The Decoder

Decoding the compressed text is a direct process that exploits the information in the Mask layer to reconstruct the original text from the three other layers: the functional words layer, the derivative words layer and the non-derivative words layer. The Mask layer plays an important role in providing the structure of the original document. It provides the decoder with the information needed to reconstruct the original text from the other three layers of the compressed text. The decoding algorithm can be summarized as follows:

1. For each value c in the Mask
   a. if c is equal to 0, read layer one and restore the corresponding functional word from the functional words table.
   b. if c is equal to 1, read the root and the pattern from layer two (three bytes),
      i. if the first bit is 0 (the word is unvowelized), rebuild the original word from the root and pattern according to the procedure CreateWord1.
      ii. if the first bit is 1 (the word is vowelized), rebuild the original word from the root and pattern according to the procedure CreateWord2.
   c. if c is equal to 3, read the code from layer three and decode it using the appropriate decoder (the Huffman statistical based decoder).
2. End

The two procedures CreateWord1 and CreateWord2 receive two pointers indicating the root and the pattern. The first procedure rebuilds the word by inserting the letter of the root in the slots marked by the character '*' in the pattern. Since this procedure is called for the unvowelized words, only the list of consonants LC of the pattern is used in this operation. The procedure CreateWord2 is called for the vowelized words. It reconstructs the word by inserting the letter of the root in the appropriate slots marked by the character '*' in the pattern. It adds the diacritical marks of the list LD associated with the pattern such that each consonant is followed by a diacritical mark.

6. Experiments and Performance

In order to evaluate the performance of the proposed approach, we designed experiments on three types of texts: vowelized, unvowelized and partially vowelized texts. For each type, the empirical results are given at the word level and at the level of the whole text. The vowelized text excerpts were a combination of documents from multiple resources from the internet and e-books.

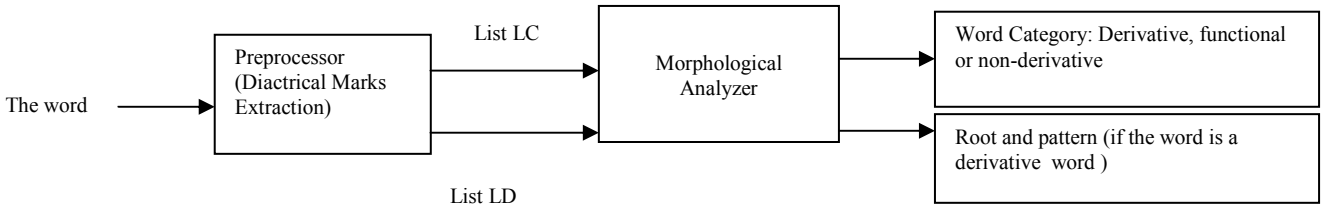

Figure 2. The morphological analyzer.

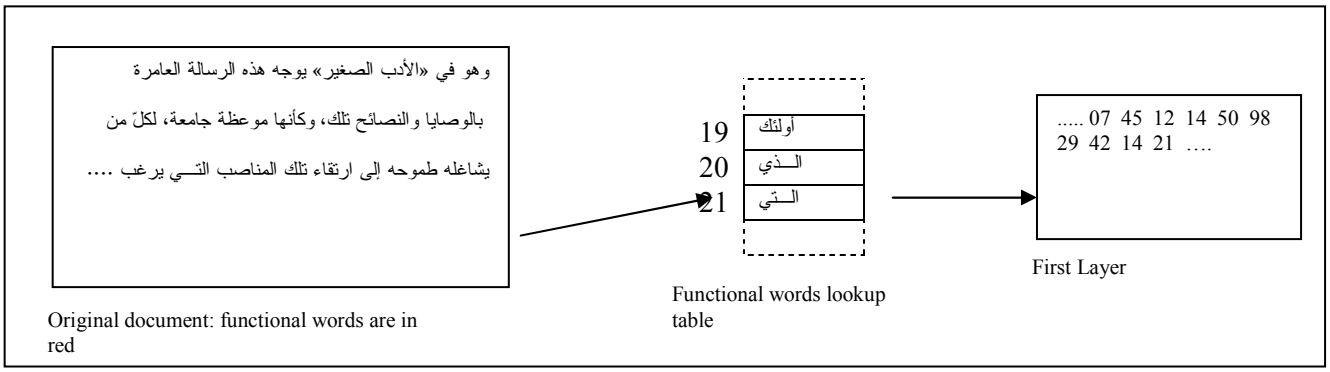

Figure 3. Construction of the first layer – the functional words layer.

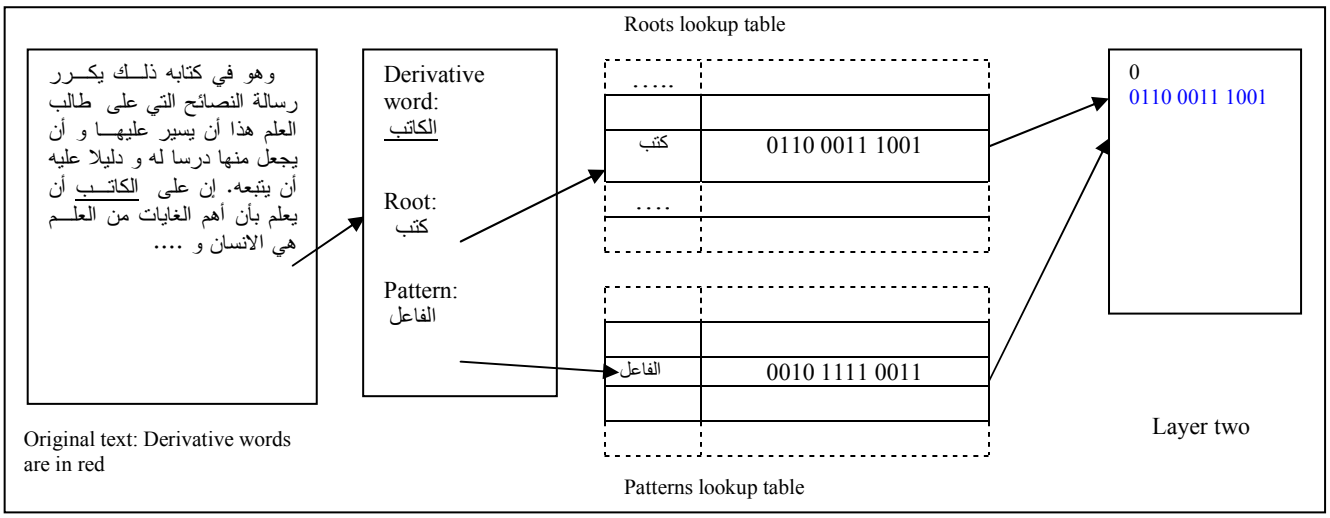

Figure 4. Construction of the second layer-the derivative words layer.

The unvowelized texts consisted of documents downloaded from the AL-JAZEERA news network and the Jordanian newspaper AL-RAI websites. The partially vowelized texts are mainly collected from Kuwaiti monthly magazine AL-ARABI.

Three factors are considered in the analysis of the performance of the proposed approach: the Compression Ratio (CR); the overhead of each component of our solution, especially the compression time; and the failure ratio of the morphological analyzer.

6.1. Compression Ratio

Several common measures of compression have been suggested in literature [12, 15]. The compression ratio CR was computed by dividing the size of the compressed text by the size of original text. The lower the compression ratio, the more the method is considered efficient. However, the efficiency of the compressor depends on the nature of the input texts.

In Table 5, we present examples of the compression ratio calculated for words from different categories and different vowelization cases. It is evident that the results depend on the words tested. The compression ratio is much better for long words than shorter ones, for functional words than for the words in the other two categories, and for the vowelized words than for unvowelized ones.

The ranges as well as the averages of the compression ratio obtained for each category are shown in Table 6. Since the majority of words are followed by space, which is considered as the default end mark and embedded inside the code of the word,

the real results are, on average, better than those shown in Tables 5 and 6.

The tests on real world texts were done with a collection of 183 MB of vowelized texts, 372 MB of unvowelized texts and 168 MB of partially vowelized texts. In the latter category, about 5% of the words were partially vowelized and the other words were unvowelized. Table 7 shows the results of our approach for the case of a statistical compressor based on the Huffman encoding algorithm applied to the third layer. The results at the file level are better than those calculated at the word level mainly because of the elimination of spaces after the majority of words, which improves the compression ratio. It should be noted that the best results are obtained for large-sized documents of vowelized texts.

Overall, the results show that the compression ratio is, on average, less than 0.26. This represents a better solution when compared with compression techniques based on dynamic Huffman coding, which yield a compression ratio of more than 0.55 for Arabic texts [12]. In Table 8, we show the compression performance of the proposed technique compared to zip, gzip and bzip2 compression techniques. The performances of the proposed technique were better for larger files than for small ones and for vowelized texts than for unvowelize texts.

These good results can be attributed to various aspects. On one hand, the use of different algorithms for compressing the different layers of the text proved to be efficient. On the other hand, the three lookup tables used (for roots, functional words, and patterns) are static, and therefore do not need to be transmitted with the compressed file.

Table 5. Examples of the compression ratio for words from different categories.

Word	Category	Vowelization	Size Before Compression (in bits)	Size After Compression (in bits)	Compression Ratio
المعلمون	Derivative Word	Unvowelized	64	24	0.38
الْمُعَلِّمُوْنَ	Derivative Word	Vowelized	128	24	0.19
الذي	Functional Word	Unvowelized	32	8	0.25
الذي	Functional Word	Vowelized	64	8	0.13

Table 6. Compression ratio for the different categories and cases of words.

Category	Vowelization	Size Before Compression (bytes)	Size After Compression bytes)	Range of CR	Average of CR
Derivative Word	Unvowelized	4 – 16	3	0.75 – 0.19	0.36
Derivate Word	Vowelized	6 – 32	3	0.5 – 0.09	0.17
Functional Word	Unvowelized	2 – 6	1	0.5 – 0.17	0.22
Functional Word	Vowelized	4 – 12	1	0.25 – 0.08	0.12

Table 7. Compression ratio for the different categories of texts.

Text Category	ASCII File Size (in MB)	File Size (in Words)	Compression Ratio
Vowelized Texts	183	15,188,420	0.12
Partially Vowelized Texts	168	26,615,632	0.26
Unvowelized Texts	372	59,817,846	0.23

Table 8. Comparative compression performance.

Text Category	ASCII File Size (in MB)	Zip	Gzip	bzip2	Proposed Technique
Vowelized Texts	45	0.22	0.21	0,19	0.15
	183	0.28	0,27	0.20	0.12
Partially Vowelized Texts	38	0.19	0.20	0.17	0.27
	168	0.27	0.25	0.20	0.26
Unvowelized Texts	42	0.21	0.21	0.17	0.24
	372	0.29	0.28	0.22	0.23

6.2. Compression Overhead

Since the tables of roots, functional words and patterns are predefined, only one pass over the whole text is required. The fact that there is no need to transmit these tables along with the compressed texts has a positive impact on the transmission time.

Our approach is asymmetric in the sense that the time consumed in the encoding phase is larger than the time needed in the decoding phase. It requires a larger amount of time on the encoding side to analyze the

morphology of words and to build the four layers. The time efficiency of our approach is further highlighted by the fact that the compression is done only once-while building the layers - whereas decompression is needed more often.

6.3. Failure Ratio of the Analyzer

The main objective of the morphological analyzer is to determine a new means to represent text documents that may be used to achieve better compression. Accurate analysis of the words is not crucial for our system. The main drawback of the analyzer is its overhead in terms of the time needed for the morphological analysis.

The functional words constitute the most substantial part of the compression ratio. Since their number is limited and well known, our system is able to capture all of these words. We therefore conclude that the failure rate of the morphological analyzer has a very limited impact on the performance of the system. Instead, it depends mainly on the input text, in terms of the number of misspelled words in the text, the number of words derived from roots that are not included in the table of roots, as well as the number of partially-vowelized words.

7. Conclusions

In this paper, a new approach for text compression has been described and the results of its implementation have been discussed. The proposed scheme effectively exploits the features of the Arabic language to build a multilayer model of the text. The text is split into smaller linguistically "homogeneous" layers representing the main categories of words: derivative, non-derivatives and functional words. Although the model has been developed and applied to Arabic texts, its basic idea could be investigated for other languages, especially for languages that have regular systems of derivation.

The proposed multilayer model achieves a very good compression ratio through the application of different compression algorithms on the different layers of the text. Evaluation of the new method in terms of the amount of compression it provides showed that its performance is better than many of the traditional compression techniques applied on the whole text. The selection of suitable compression techniques for the different layers is important for better compression.

One of the main advantages of the proposed text compression technique over statistical compression techniques is that there is no need to read all of the text and to wait to define the symbols to begin the compression. The advantage over other dictionary-based techniques is that there is no need to rebuild the tables for each new text and transmit them with the compressed files.

The output is not only a compressed text with a good compression ratio, but it also contains information about the lexical forms of the words. This information is very useful for different applications in natural language processing such as information retrieval, text understanding, document classification and text mining. A fast and flexible word search can be performed on the compressed file, whereby we can replace the search for a word in the entire text by a search for its root in the appropriate layer.

References

[1] Akman I., "A New Text Compression Technique Based on Language Structured," *Computer Journal of Information Science*, vol. 21, no. 2, pp. 87-94, 1995.

[2] Al-Fedaghi S. and Al-Sadoun B., "Morphological Compression of Arabic Texts," *Computer Journal of Information Processing & Management*, vol. 26, no. 2, pp. 303-316, 1990.

[3] Al-Sughaiyer I. and Al-Kharashi I., "Arabic Morphological Analysis Techniques: A Comprehensive Survey," *Computer Journal of the American Society for Information Science and Technology*, vol. 55, no. 3, pp. 189-213, 2004.

[4] Ali N. and Abed E., "A Morphology Based Data Compression Technique for Arabic Text," *in Proceedings of the AFRICOM'84*, pp. 241-251, 1984.

[5] 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 2007.

[6] Bach J. and Witten H., "Lexical Attraction for Text Compression," *in Proceedings of Data Compression Conference*, pp. 516-516, 1999.

[7] Beesley R., "Arabic Finite-State Morphological Analysis and Generation," *in Proceedings of COLING-96, the 16th International Conference on Computational Linguistics*, Copenhagen, pp. 89-94, 1996.

[8] Cheng S. and Wong F., "A Study on Word-Based and Integral-bit Chinese Text Compression Algorithms," *Computer Journal of the American Society for Information Science and Technology*, vol. 50, no. 3, pp. 218-228, 1999.

[9] De Moura S., "Fast and Flexible Word Searching on Compressed Text," *Computer Journal of ACM Transactions on Information Systems*, vol. 18, no. 2, pp. 113-139, 2000.

[10] Diri B., "A Text Compression System Based on the Morphology of Turkish Language," *in Proceedings of the 15th International Symposium*

on Computer and Information Sciences, Turkey, pp. 156-159, 2000.

[11] Fenwick P., "Differential Ziv-Lempel Text Compression," *Computer Journal of Universal Computer Science*, vol. 1, no. 8, pp. 591-602, 1995.

[12] Ghwanmeh S., Al-Shalabi R., and Kanaan G., "Efficient Data Compression Scheme using Dynamic Huffman Code Applied on Arabic," *Journal of Computer Science*, vol. 2, no. 12, pp. 887-890, 2006.

[13] Horspool N. and Cormack V., "Constructing World-based Text Compression Algorithms," *in Proceedings of the IEEE Data Compression Conference*, pp. 62-81, 1992.

[14] Isal R. and Moffat A., "Word-Based Block-Sorting Text Compression", *Proceedings of the 24th Australasian Conference on Computer Science*, pp. 92-99, 2001.

[15] Lelewer A. and Hirschberg S., "Data Compression," *Computer Journal of ACM Computing Surveys*, vol. 19, no. 3, pp. 261-296, 1987.

[16] Long M., Natsev I., and Vitter S., "Text Compression via Alphabet Re-Representation," *Computer Journal of Neural Networks*, vol. 12, no. 4, pp. 755-776, 1999.

[17] Manber U., "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, 1997.

[18] Ng S., Cheng M., and Wong H., "Dynamic Word Based Text Compression," *in Proceeding of the 4th International Conference Document Analysis and Recognition*, pp. 412-412, 1997.

[19] Teahan J., McNab R., and Witten H., "A Compression-based Algorithm for Chinese Word Segmentation," *Computer Journal of Computational Linguistics*, vol. 26, no. 3, pp. 375-392, 2000.

[20] Unicode Standard 5.1, 1991-2008, http://unicode.org/charts /pdf/U0600.pdf, Last Visited 2008.

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

[22] Yaghi J., Titchener R., and Yagi S., "T-Code Compression for Arabic Computational Morphology," *in Proceedings of the Australasian Language Technology Workshop*, Australia, pp. 425-465, 2003.

[23] Yoshida S., Morihara T., Yahagi H., and Satoh N., "Application of a Word-based Text Compression Method to Japanese and Chinese Texts," *in Proceedings of Data Compression Conference*, pp. 561-561, 1999.

[24] Yuret D., "Discovery of Linguistic Relations using Lexical Attraction," *PhD Thesis*, Massachusetts Institute of Technology, 1998.

[25] Ziviani N., De Moura S., Navarro G., and Baeza-Yates R., "Compression: A Key for Next Generation Text Retrieval System," *Computer Journal of IEEE Computer*, vol. 33, no. 11, pp. 37-44, 2000.

Arafat Awajan is associate professor at Princess Sumaya University for Technology (PSUT) and director of the Information Technology Center in the Royal Scientific Socity, Amman, Jordan. He received his PhD degree in computer science from the University of Franche, Comte, France in 1987. He held different academic positions at the Royal Scientific Society and Princess Sumaya University for Technology. He was appointed as the chair of the Computer Science Department (2000-2003) and the chair of the Computer Graphics and Animation Department (2005-2006) at PSUT. He had been the dean of the King Hussein School for Information Technology from 2004 to 2007. His research interests include e-learning, e-business, natural language processing, text compression, and image processing.