King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Improved capacity Arabic text watermarking methods based on open word space

Reem A. Alotaibi [a,c,*], Lamiaa A. Elrefaei [a,b,1]

[a] Computer Science Department, Faculty of Computing & Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
[b] Electrical Engineering Department, Faculty of Engineering at Shoubra, Benha University, Cairo, Egypt
[c] Computer Science Department, Faculty of Computing & Information Technology, Taif University, Taif, Saudi Arabia

Received 3 September 2016; revised 10 December 2016; accepted 28 December 2016
Available online 11 January 2017

KEYWORDS

Arabic text watermarking;
Capacity;
Robustness;
Imperceptibility

Abstract Digital watermarking is used to protect text copyright and to detect unauthorized use. In this paper, two invisible blind watermarking methods for Arabic text are proposed. Since the pseudo-space is very small space used to force the connected characters to be isolated, it is added to the word space to hide binary bit "0" or "1". In the first proposed method, the pseudo-space is inserted before and after normal word space based on dotting feature in Arabic text. The second proposed method inserts the pseudo-space and other three small or zero width spaces to increase the capacity, where the presence of them indicates bit "1" and the absence indicates bit "0". The comparative results obtained by testing the proposed methods with some of existing watermarking methods using variable size text samples with different watermark lengths. The experiments show that the proposed methods have the highest capacity and higher imperceptibility than other watermarking techniques from the literature. The robustness of the proposed methods is tested under most of possible text attacks. They are robust against electronic text attacks such as: copying and pasting, text formatting and text tampering for tampering ratio up to 84%.

1. Introduction

Digital watermarking techniques have been widely emerged in order to effectively protect multimedia copyright. Digital watermarking refers to the embedding process that inserts a watermark (i.e. Label, signature, or copyright) into several types of media. These types of media include text, audio, image, and video. The watermarking system involves two main processes: embedding of the watermark into the original data and extracting the watermark from watermarked data or attacked watermarked data. When designing a watermarking system, some of the basic requirements must be taken into account which vary depending on the use of the system

* Corresponding author.
E-mail addresses: reem.a.safran@gmail.com (R.A. Alotaibi), laelrefaei@kau.edu.sa, lamia.alrefaai@feng.bu.edu.eg (L.A. Elrefaei).
[1] Postal address: King Abdulaziz University, Female Campus, Room No. s109, Building No. 61, P.O Box 80221, Jeddah 21589, Saudi Arabia.
Peer review under responsibility of King Saud University.

ELSEVIER Production and hosting by Elsevier

(Stanković et al., 2012; Cox et al., 2007). The key watermarking requirements are: capacity, imperceptibility, robustness and security. Capacity is the total number of hidden bits in an object. Imperceptibility is used to measure the difference between the original and watermarked object by noting any addition to the original object. Robustness is the ability to extract or detect the watermark after the watermarked object has been attacked. Security requirement is the difficulty of extracting the watermark without the destruction of the watermarked object.

Creating a hidden watermark in the text is the hardest kind due to the relative lack of unnecessary information within a text file compared to an image or audio files. The human sensitivity to text changes is higher than the sensitivity to other multimedia. Any text change must reserve the meaning, fluently, writing style and text value (Topkara et al., 2006; Jalil, 2010).

Text watermarking methods can be classified into: line-shift coding, word-shift coding, linguistic methods, open space method and other methods based on the language characteristic (Alotaibi and Elrefaei, 2015). In line-shift coding, the lines are shifted up or down to hide “0” or “1”. In word-shift coding the words are moved horizontally to code secret bits. Linguistic methods aim to change the text structure (syntactic approach) or text content (semantic approach) (Bennett, 2004). Open space method depends on the exploitation of the white spaces by adding a space or more between words, sentences or at the end of lines to indicate on the existence of hidden bits (Bender et al., 1996).

The contributions of this paper are improving the embedding capacity by proposing two Arabic text watermarking methods, Method 1 and Method 2 and testing their robustness. The proposed methods utilize the open space between words but instead of using normal space as in Bender et al. (1996), small spaces or no width ones are used. Arabic letters do not take one form, but their shape varies depending on its location in the word (Alotaibi and Elrefaei, 2015). Pseudo-space is a non printing character when it comes before or after the letter, it forces the letter to take the final or initial form. Using it at the beginning or ending of the word does not change the word shape. The researchers in Alotaibi and Elrefaei (2016) inserted it before the normal space to indicate hidden data. In the proposed Method 1, Pseudo-space is added before and after the regular space to provide capacity as twice as the method presented in Alotaibi and Elrefaei (2016). In the proposed Method 2, four spaces: Pseudo-space, Thin space, Hair space and Zero width space, were chosen to be added to the normal space to give a very large capacity. The existence of these spaces is used to hide bit “1”, and the absence of them is used to hide bit “0”.

Table 2 The used spaces.

Watermarking method	The inserted spaces	Location of insertion
Open space methods (Bender et al., 1996)	NS	Between words, lines or sentences
Pseudo-space method (Alotaibi and Elrefaei, 2016)	PS	Between words
Proposed Method 1	PS	Between words
Proposed Method 2	PS, HS, TS and ZWS	Between words

1.1. Overview of the used spaces

The proposed methods have adopted to embed the watermark in the Arabic text by inserting some spaces to the normal space. The chosen spaces are shown in Table 1 with their Unicode’s (Whitespace character, 2016). Table 1 also shows examples of using these spaces between two words “”سبحان and “”الله to show how wide are they. The vertical pointer indicates where the Unicode space is inserted. Table 2 shows the used spaces in methods (Bender et al., 1996; Alotaibi and Elrefaei, 2016) and in the proposed watermarking methods.

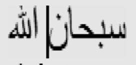
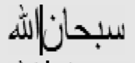
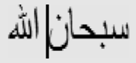
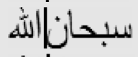
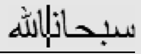
The rest of this paper is organized as follows: Section 2 reviews the related work in Arabic text watermarking. The proposed methods are discussed in Section 3. Experiments are conducted to evaluate the capacity and imperceptibility of the proposed methods and to compare it with five of the Arabic text watermarking methods in Sections 4.1 and 4.2. Robustness of the proposed methods is measured under most of the existing known text attacks in Section 4.3. Section 5 concludes the paper.

2. Related work

Text watermarking methods in any language exploit the characteristics of the writing in that language or the general text characteristics. Arabic script has many characteristics such as: open space, kashida, diacritics and dotting.

Open space is a general characteristic used for data hiding in the host text as in the research work presented in Bender et al. (1996) by exploitation of the white spaces in the text

Table 1 The chosen spaces.

Space name	Space unicode	Description	Example
Normal Space (NS)	U + 0020	Normal space used to separate words	
Pseudo-Space (PS)	U + 200C	Force the connected letters to be separate	
Thin Space (TS)	U + 2009	1/5 (sometimes 1/6) of an em wide	
Hair Space (HS)	U + 200A	Thinner than a thin space	
Zero width Space (ZWS)	U + 200B	No width	

document. The manipulation of the white spaces is done in three different ways: inter-sentence spacing, end-of-line spaces and inter-word spacing. Inter-sentence spacing is encoding secret information in the form of binary string on the text based on spaces between sentences. It encodes "0" by inserting one space or encodes "1" by inserting two consecutive spaces. This method suffers from low capacity because only one or two bits encoded in each sentence. The indiscriminate use of this method reduces its transparency. White Spaces are well utilized in the end-of-line spaces method, because the extra spaces are placed at the end of the line which provides higher imperceptibility to the reader than the previous method. In end-of-line spaces, two spaces are used to embed one bit per line, four spaces are used to embed two bits, eight spaces are used to embed three bits, and so on. There are no restrictions on the use of this method in terms of text structure. However, some programs destroy the hidden data by the removal of the additional spaces.

In inter-word spacing, the extra spaces are inserted between two consecutive words. A single space between words encodes the binary bit "0", while two spaces encode the binary bit "1". Yang and Kot (2004) and Huang and Yan (2001) produced watermarking methods in text images based on modifying the word space. The spaces between characters are combined with a word space based on embedding rules to watermark a text image in Yang and Kot (2004). This method could not be applied in Arabic text because its characters are connected. In Huang and Yan (2001), the lines of text image considered as sampling point of a sine wave. The watermark is inserted in frequency, phase and amplitude of the sine wave. The watermarking methods in text image are not robust to copy and paste or formatting operations. The watermark is lost using OCR programs. Also, they consume time compared to methods which are dealing with text directly.

Arabic language has unique properties used in data hiding. **Dotting** property is one of the most important properties. Points are located up or down some of the Arabic letters. In the research work presented in Shirali-Shahreza and Shirali-Shahreza (2006), points are shifted up referring to hide bit "1" or keeping it in its places to denote bit "0". The dotting property also used with pseudo-space to utilize each open word space in Alotaibi and Elrefaei (2016). The watermarking process includes insertion of pseudo-space before the word space based on the letter before it. The pseudo-space does not change the space between words and the letter shape. It is inserted if the letter before word space is pointed and the embedding bit equals "1". It is also inserted if the letter is un-pointed and the embedding bit equals "0".

Kashida or extension character could be added to the text to indicate the existence of secret data. It is used to adjust the text by extending the words and does not change the meaning. The authors in Gutub et al. (2007) used kashida with dotting property in the watermarking process. They insert kashida before or after letters containing points to indicate the bit one. They also insert it before or after letters that do not contain points to indicate bit zero. The kashida is inserted in a particular pattern in this method, so the authors change the insertion process to increase the security. In Gutub et al. (2010) one Kashida is inserted after any letter can hold it to represent bit zero, and two consecutive Kashidas are inserted to represent bit one.

In Alginahi et al. (2013) kashida is added before 6 letters which accept to connect from the right (Arabic script is written from right to left), provided that the character before these letters accept the insertion of the kashida after it. The researchers in Alginahi et al. (2014) have used the same previous mechanism where the kashida is added before specific sets of letters. They divided the Arabic letters into two sets depending on the occurrences of the letters. Set A containing the higher frequencies while set B containing the lower ones. They have developed two methods, the first used set A and the second used both set A and set B. The main disadvantages of kashida-based methods are the possibility to note the kashida and the capacity is variable depending on the ability of the letter to extend, it is inserted only in certain places in the word.

The Arabic language does not contain vowels, but instead of them, it has diacritics or harakat. **Diacritics** help in the correct pronunciation of the words. They distinguish between two words which composed of the same letters, but having a different voice and meaning. These diacritics were used in the field of steganography and watermarking Arabic text. Some methods (Aabed et al., 2007; Bensaad and Yagoubi, 2011) display diacritics or hide them to denote "0" or "1" bits. Diacritics are utilized in Shah and Memon (2011) in a different way, the direction of the diacritic "Fatha" is reversed. The usual form is used to denote one, and the reversed one to denote zero. This method requires a special font.

The diacritics-based methods are not blind, they require the original text in the detection process. The capacity of them depends on the number of the diacritics in the text. The using of diacritics in Arabic text becomes a little because they could find out by parsing the text. The Persians researchers in Shirali-Shahreza and Shirali-Shahreza (2008,2010) and Shirali-Shahreza (2008), developed some methods to hide data in Arabic and Persian texts based on the Unicode of characters. The Unicode based methods use characters that have special features to be utilized in the hidden process. In Shirali-Shahreza and Shirali-Shahreza (2008) a new steganography approach has been proposed which uses pseudo-space between unconnected characters and pseudo-connection between connected characters to hide bits "0" or "1". These additions does not affect the overall look of the text, but used to hide secret data.

The authors in Shirali-Shahreza and Shirali-Shahreza (2010) use two characters "Ya" and "Kaf" in Arabic and Persian languages with similar appearance in two positions in the word, which means that they have different codes. They use the Persian characters to hide bit "0" and the Arabic characters to hide bit "1". In Shirali-Shahreza (2008) the "0" or "1" bits are hidden based on the use of one of the two forms of the word "La". The "La" word could be written by pressing one button on the keyboard representing one code. The previous implementation is used to hide "0". "La" word is written by inserting the kashida between two characters "Lam" and "Alef" to hide "1". The Unicode based methods have high transparency but very low capacity, except the first method. It has a large capacity, but the secret bits could be affected in the extraction side.

3. Proposed methods

Text watermarking methods based on open space preserve the integrity of the linguistic and grammatical rules, clarity and value of the text. The extra spaces between words, sentences or at the end of the line are often not observed by readers. However, the existing methods based on utilizing white spaces in the text suffer from low data insertion capacity. Also, the using of normal space frequently may draw attention.

Two Arabic text watermarking methods are proposed in this paper by utilizing each word space in the text and this guarantee high insertion capacity. The manipulation of white spaces between words is done by adding small spaces or no width ones instead of using normal space to satisfy imperceptibility requirement. The first method, Method 1 utilizes the dotting feature in a way to improve the capacity of the method presented in Alotaibi and Elrefaei (2016). Pseudo-space is a non printing character used to separate two parts of the same word in Persian language. It is inserted before the normal space in Alotaibi and Elrefaei (2016) and it is inserted before and after normal space in the proposed Method 1 based on the character before the normal space and the character after it whether they are pointed or not. The proposed Method 2 could be applied to Arabic text or any other languages.

3.1. Method 1

The first proposed Arabic text watermarking method, Method 1, improves the capacity of the work presented in Alotaibi and Elrefaei (2016). The letter after the space in addition to the letter before it are checked in the watermark embedding and extraction processes. While the pseudo-space has no width and it is used to make letters appearing isolated, it could be added at the beginning or ending of the Arabic word without any visual effects.

3.1.1. Embedding algorithm

The embedding algorithm takes a watermark in a binary representation as a string of zeros and ones as an input with the text to be watermarked. To get the watermarked text, the checking process is applied from the right to the left at the letter before the space and the letter after it. The pseudo-space is inserted between the letter and the space if the letter matches the insertion condition. The reader could not find any change

Algorithm 1. Method 1 embedding algorithm

```
Input : Original text:OT, watermark in binary format:W, Set of pointed letters:$P$ , and Set of unpointed letters :$UP$
Output : Watermarked text
$j \leftarrow 0$
$L \leftarrow length(W)$
$C \leftarrow length(OT)$
for i=2: C-1 do
  Get $C_i$ , $C_{i-1}$ , $C_{i+1}$
  if Ci is space then
    if $(C_{i-1} \in P \ and \ W[j] = 1)$ then
        insert pseudo space before $C_i$ in Watermarked text
    end
    else if $(C_{i-1} \in UP \ and \ W[j] = 0)$ then
      insert pseudo-space before $C_i$ in Watermarked text
    end
    $j \leftarrow j + 1$
    if $j \geqslant L$then
      $j \leftarrow 0$
    end
    if $(C_{i+1} \in P \ and \ W[j] = 1)$ then
      insert pseudo-space after $C_i$ in Watermarked text
    end
    else if $(C_{i+1} \in UP \ and \ W[j] = 0)$ then
      insert pseudo-space after $C_i$ in Watermarked text
    end
    $j \leftarrow j + 1$
    if $j \geqslant L$ then
      $j \leftarrow 0$
    end
  end
end
return Watermarked text
```

between the original and watermarked texts. The insertion condition depends on the watermark bit and the letter status as illustrated by the embedding algorithm shown in Algorithm 1. Fig. 1 shows an example of Method 1 where the colored letters indicate the existing of pseudo-space after or before it. If we take the second space as an example, the letter before the space is ن which is pointed and the watermark bit equals "0", so nothing is inserted. The letter after the space is ا and the watermark bit equals "1", also nothing is inserted.

3.1.2. Extraction algorithm

Method 1 is a blind watermarking method as the extraction algorithm only needs the watermarked text to return the watermark bits. The extraction process checks the existence of the pseudo space and the previous and next letters to set the watermark bit to "1" or "0". Algorithm 2 shows the extraction algorithm in Method 1.

3.2. Method 2

In Method 2, the embedding algorithm involves the use of four different spaces shown in Table 2, to be mixed with word normal space. Not all spaces are inserted all the time, only the space which matches the watermark bit is inserted.

3.2.1. Embedding algorithm

The watermark bit stream is divided into groups, each of length 4 bits. The first bit is corresponding to pseudo-space. The second bit is corresponding to thin space. The third bit is corresponding to hair space. The fourth bit is corresponding to zero width space. Then, match each bit in each group

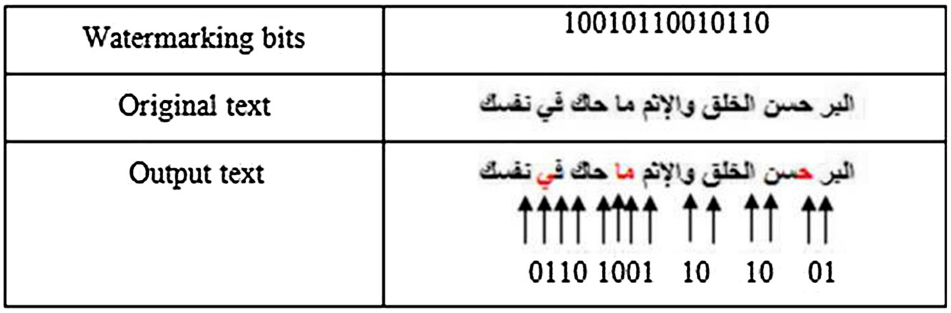
Algorithm 2. Method 1 extraction algorithm

```
Input : Watermarked text: WT , Set of pointed letters:$P$ , and Set of unpointed letters :$UP$
Output: Watermark: W
$j \leftarrow 0$
$C \leftarrow length(WT)$
for i = 5: C-2 do
  Get $C_i, C_{i-1}, C_{i-2}, C_{i+1}, C_{i+2}$
  if $C_i$ is space then
    if $C_{i-1}$ is pseudo-space then
      if $C_{i-2} \in P$ then
        $W[j] \leftarrow 1$
      end
      else if $C_{i-2} \in UP$ then
        $W[j] \leftarrow 0$
      end
    end
    else
      if $C_{i-1} \in P$ then
        $W[j] \leftarrow 0$
      end
      else if $C_{i-1} \in UP$ then
        $W[j] \leftarrow 1$
      end
    end
    $j \leftarrow j + 1$
    if $C_{i+1}$ is pseudo-space then
      if $C_{i+2} \in P$ then
        $W[j] \leftarrow 1$
      end
      else if $C_{i+2} \in UP$ then
        $W[j] \leftarrow 0$
      end
    end
    else
      if $C_{i+1} \in P$ then
        $W[j] \leftarrow 0$
      end
      else if $C_{i+1} \in UP$ then
        $W[j] \leftarrow 1$
      end
      end
      $j \leftarrow j + 1$
    end
end
return Watermark
```

with corresponding space. If the bit equals to “1” the intended space is inserted, else if the bit equals “0” nothing is inserted. For example, if the group bits equal “0000”, no one of the spaces is added. If the group bits equal “1111”, all spaces are added to normal space while maintaining the spaces order. The embedding algorithm steps explained in Algorithm 3. Fig. 2 shows an example of the proposed Method 2.

3.2.2. Extraction algorithm

Method 2 could be used for any language where the space is one of its writing components and not exclusively in the Arabic language. This method works blindly where it does not require the original text to extract secret data. As the previous method, it searches for each space in the text and get the four characters before the space. Each individual word space encodes four bits. If an intended space is founded before word space, it will be



Watermarking bits	10010110010110
Original text	البر حسن الخلق والإثم ما حاك في نفسك
Output text	البر حسن الخلق والإثم ما حاك في نفسك 0110 1001 10 10 01

Figure 1 Example of Method 1.

Algorithm 3. Method 2 embedding algorithm

```
Input : Original text:OT and watermark in binary format:W
Output : Watermarked text
$j \leftarrow 0$
$L \leftarrow length(W)$
$C \leftarrow length(OT)$
for i = 1: C do
  Get $C_i$
  if Ci is space then
    if $(W[j] = 1)$ then
      insert pseudo-space before $C_i$ in Watermarked text
    end
    if $(W[j+1] = 1)$ then
      insert thin space after the previous space in Watermarked text
    end
    if $(W[j+2] = 1)$ then
      insert hair space after the previous space in Watermarked text
    end
    if $(W[j+3] = 1)$ then
      insert zero width space after the previous space in Watermarked text
    end
    $j \leftarrow j + 4$
    if $j \geqslant L$then
      $j \leftarrow 0$
    end
  end
end
return Watermarked text
```

matched with its position in the watermark group. The extraction algorithm is stated in details in Algorithm 4.

Algorithm 4. Method 2 extraction algorithm.

```
Input : Watermarked text: WT
Output : Watermark: W
$j \leftarrow 0$
$C \leftarrow length(WT)$
for i = 5: C do
  Get $C_i$ , $C_{i-1}$ , $C_{i-2}$ , $C_{i-3}$ , $C_{i-4}$
  if $C_i$ is space then
    $col4 = C_{i-4} + C_{i-3} + C_{i-2} + C_{i-1}$ $col3 = C_{i-3} + C_{i-2} + C_{i-1}$
    $col2 = +C_{i-3} + C_{i-2} + C_{i-1}$
    if col4 is PS followed by TS followed by HS followed by ZWS then
      $W[j,4] \leftarrow 1111$
    end
    else if col3 is TS followed by HS followed by ZWS then
      $W[j,4] \leftarrow 0111$
    end
    else if col3 is PS followed by TS followed by ZWS then
      $W[j,4] \leftarrow 1101$
    end
    else if col3 is PS followed by HS followed by ZWS then
      $W[j,4] \leftarrow 1011$
    end
    else if col3 is PS followed by TS followed by HS then
      $W[j,4] \leftarrow 1110$
    end
    else if col2 is PS followed by TS then
      $W[j,4] \leftarrow 1100$
    end
    else if col2 is TS followed by HS then
      $W[j,4] \leftarrow 0110$
    end
    else if col2 is HS followed by ZWS then
      $W[j,4] \leftarrow 0011$
    end
    else if col2 is PS followed by HS then
      $W[j,4] \leftarrow 1010$
    end
    else if col2 is PS followed by ZWS then
      $W[j,4] \leftarrow 1001$
    end
    else if col2 is TS followed by ZWS then
      $W[j,4] \leftarrow 0101$
    end
    else if col1 is PS then
      $W[j,4] \leftarrow 1000$
    end
    else if col1 is TS then
      $W[j,4] \leftarrow 0100$
    end
    else if col1 is HS then
      $W[j,4] \leftarrow 0010$
    end
    else if col1 is ZWS then
      $W[j,4] \leftarrow 0001$
    end
    else
      $W[j,4] \leftarrow 0000$
    end
    $j \leftarrow j + 4$
  end
end
return Watermark
```

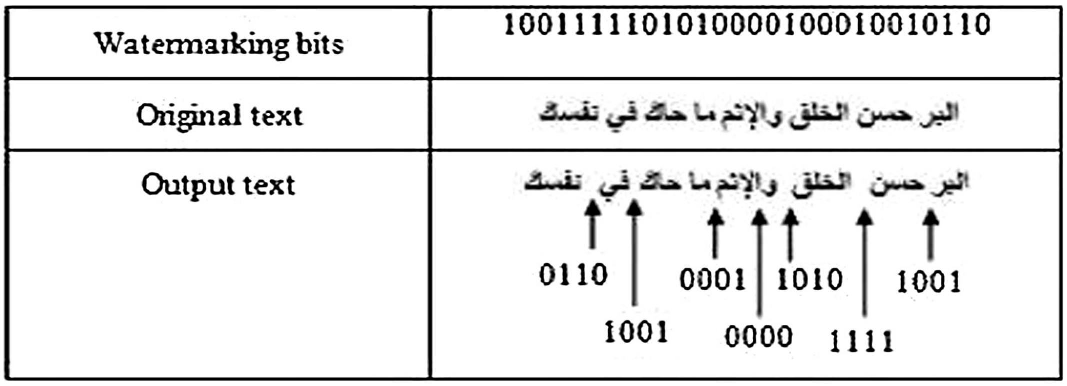


Figure 2 Example of Method 2.

4. Results and discussions

To evaluate the proposed methods in terms of capacity, imperceptibility and robustness, experiments are done using 15 different size text samples. The text samples are chosen from OSAC: Open Source Arabic Corpus (Saad and Ashour, 2010) and AraCorpus (Index of /AraCorpus, 2010) with different sizes from very small texts to very large ones. The text samples are classified based on MS Windows OS file search by size property (Kashyap, 2010). Table 3 shows text categories based on Windows classification. Table 4 shows the specifications of the used text samples.

Table 3 Microsoft Windows OS text file size classifications (Kashyap, 2010).

Text category	File size
Empty	0 KB
Tiny	0–10 KB
Small	10–100 KB
Medium	100 KB–1 MB
Large	1–16 MB
Huge	16–128 MB
Gigantic	> 128 MB

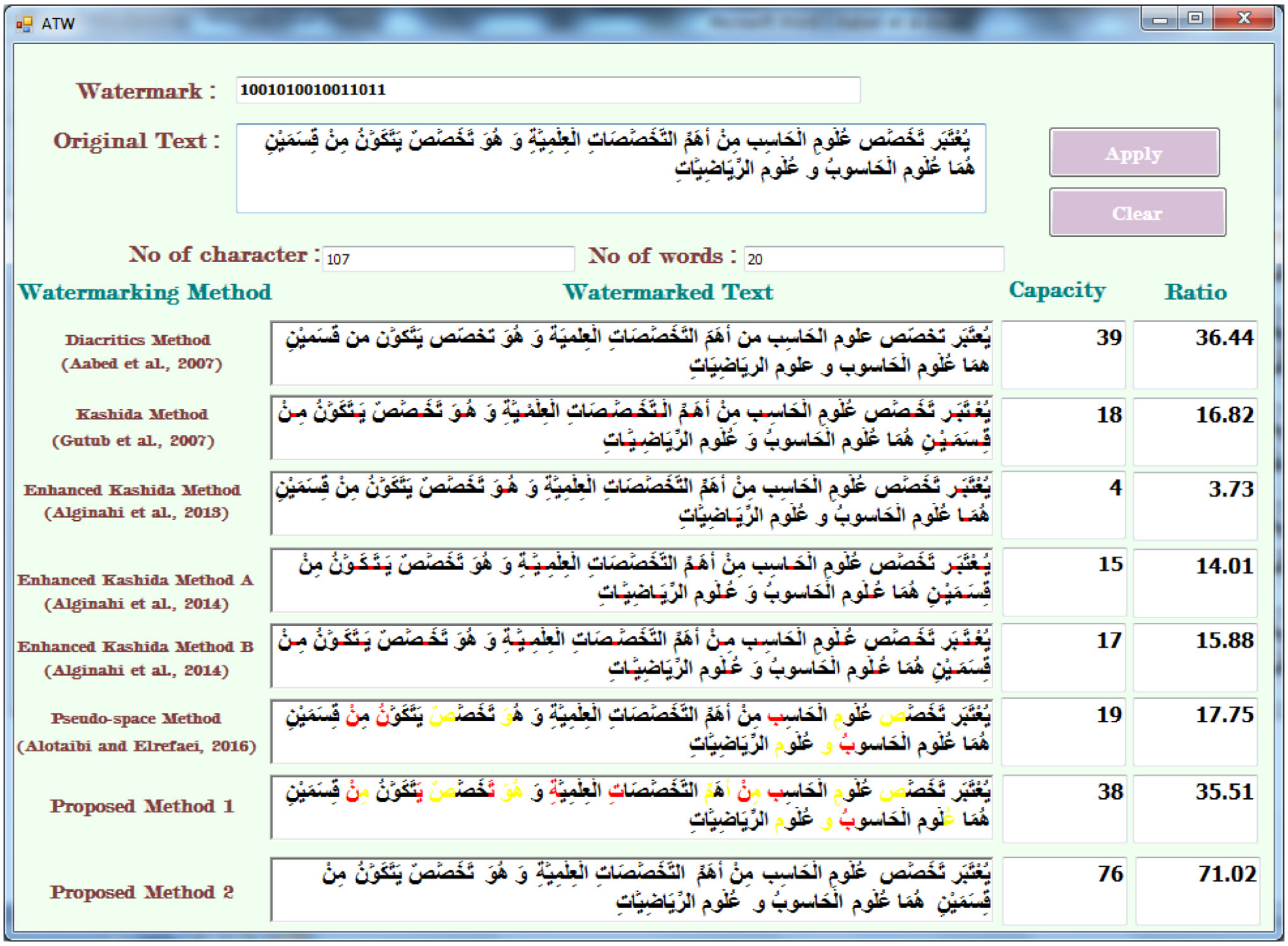
The watermark length has been chosen as follows: 8 bit, 16 bit, 32 bit, 46 bit, 128 bit and 256 bit to observe the relationship between the length of the watermark and the methods' robustness. The watermark is generated randomly in each text sample. The proposed methods have been implemented using C#.net programming language.

4.1. Imperceptibility Evaluation results

Fig. 3 shows the GUI of the Arabic text watermarking system which includes the proposed methods and other six Arabic text watermarking methods: Diacritics method (Aabed et al., 2007), kashida method (Gutub et al., 2007), enhanced kashida method (Alginahi et al., 2013), enhanced kashida method A (Alginahi et al., 2014), enhanced kashida method B (Alginahi et al., 2014) and pseudo-space method (Alotaibi and Elrefaei, 2016). As shown in Fig. 3, kashidas are colored with red, pointed letters that are followed or preceded by pseudo-space are colored with red while un-pointed letters are colored with yellow. The proposed Method 1 and the method presented in Alotaibi and Elrefaei (2016) have higher imperceptibility than others. No one can distinguish between the original and watermarked text because of the use of invisible character pseudo-space. The proposed methods, Method 1 and Method 2, have higher imperceptibility than methods based on kashida and diacritics. Diacritics method (Aabed

Table 4 The text samples specifications.

Text sample	Category	File size in byte	Total characters	No. of words
1	Tiny	260	145	29
2	Tiny	4657	2664	496
4	Small	37,222	20,636	3817
5	Small	82,469	45,647	7707
8	Medium	692,668	437,860	86,084
9	Medium	839,936	399,038	76,036
10	Medium	992,340	489,346	93,565
8	Large	1,603,394	860,748	135,032
9	Large	5,014,295	2,695,581	456,299
10	Huge	33,795,725	18,244,881	2,937,989
11	Huge	50,477,212	27,326,194	4,415,785
12	Gigantic	274,730,158	148,163,622	23,690,877



ATW

Watermark : 1001010010011011

Original Text : يُعْتَبَر تَخَصّص عُلوم الحَاسِب مِنْ أهَمّ التَّخَصّصَات العِلْمِيَّة وَ هُوَ تَخَصّص يَتَكَوَّنُ مِنْ قِسْمَيْن هُمَا عُلوم الحَاسوبُ و عُلوم الرِّيَاضِيَّات

No of character : 107 No of words : 20

Watermarking Method	Watermarked Text	Capacity	Ratio
Diacritics Method (Aabed et al., 2007)	يُعْتَبَر تخصص علوم الحَاسِب من أهَمّ التَّخَصّصَات العِلْمِيَّة وَ هُوَ تخصص يَتَكَوَّن من قسمين هما عُلوم الحَاسوب و علوم الرِيَاضِيَات	39	36.44
Kashida Method (Gutub et al., 2007)	يُعْتَبَر تَخَـصّص عُلوم الحَاسِـب مِنْ أهَـمّ الـتَّخَـصّـصَات العِلْمِـيَّة وَ هُـوَ تَخَـصّص يَـتَكَوَّنُ مِـنْ قِسـمَـيْن هُمَا عُلوم الحَاسوبُ وَ عُلوم الرِّيَاضِـيَّـات	18	16.82
Enhanced Kashida Method (Alginahi et al., 2013)	يُعْتَبَـر تَخَصّص عُلوم الحَاسِب مِنْ أهَمّ التَّخَصّصَات العِلْمِيَّة وَ هُـوَ تَخَصّص يَتَكَوَّنُ مِنْ قِسْمَيْن هُمَـا عُلوم الحَاسوبُ و عُلوم الرِّيَـاضِيَّات	4	3.73
Enhanced Kashida Method A (Alginahi et al., 2014)	يُـعْتَبَر تَخَصّص عُلوم الحَـاسِب مِنْ أهَـمّ التَّخَصّصَات العِلْمِـيَّـة وَ هُوَ تَخَصّص يَـتَـكَـوَّنُ مِنْ قِسـمَـيْن هُمَا عُـلوم الحَاسوبُ وَ عُـلوم الرِّيَـاضِـيَّـات	15	14.01
Enhanced Kashida Method B (Alginahi et al., 2014)	يُعْـتَـبَر تَخَـصّص عُـلوم الحَاسِب مِـنْ أهَمّ التَّخَصّـصَات العِلْمِـيَّـة وَ هُوَ تَخَـصّص يَـتَكَـوَّنُ مِـنْ قِسْـمَيْن هُمَا عُـلوم الحَاسوبُ وَ عُـلوم الرِّيَاضِـيَّـات	17	15.88
Pseudo-space Method (Alotaibi and Elrefaei, 2016)	يُعْتَبَر تَخَصّص عُلوم الحَاسِب مِنْ أهَمّ التَّخَصّصَات العِلْمِيَّة وَ هُوَ تَخَصّص يَتَكَوَّنُ مِنْ قِسْمَيْن هُمَا عُلوم الحَاسوبُ و عُلوم الرِّيَاضِيَّات	19	17.75
Proposed Method 1	يُعْتَبَر تَخَصّص عُلوم الحَاسِب مِنْ أهَمّ التَّخَصّصَات العِلْمِيَّة وَ هُوَ تَخَصّص يَتَكَوَّنُ مِنْ قِسْمَيْن هُمَا عُلوم الحَاسوبُ و عُلوم الرِّيَاضِيَّات	38	35.51
Proposed Method 2	يُعْتَبَر تَخَصّص عُلوم الحَاسِب مِنْ أهَمّ التَّخَصّصَات العِلْمِيَّة وَ هُوَ تَخَصّص يَتَكَوَّنُ مِنْ قِسْمَيْن هُمَا عُلوم الحَاسوبُ و عُلوم الرِّيَاضِيَّات	76	71.02

Figure 3 GUI of the proposed Arabic text watermarking methods.

et al., 2007) draws attentions where some diacritics are present and others are not.

4.2. Capacity Evaluation results

The capacity of the proposed methods: method 1 and method 2 is presented in Table 5 using the first ten text samples shown in Table 4. The capacity of text watermarking methods is the total number of bits which are embedded in the original text. The capacity ratio is computed as:

$$\textit{Capacity ratio} = \frac{\text{Number of hidden bits}}{\text{Total number of characters}} \times 100 \quad \text{(Alginahi etal., 2013)}$$

For comparison purposes, the proposed methods are compared with six methods (Aabed et al., 2007; Gutub et al., 2007; Alginahi et al., 2013, 2014; Alotaibi and Elrefaei, 2016) from the literature, are shown in Fig. 4. The capacity results of the proposed methods are higher than these watermarking methods.

Table 6 shows the average capacity comparison results using ten text samples. These text samples are taken from Table 4. The average is the sum of each text capacity divided by 10. The proposed methods have the highest capacity. Method 1 improved the capacity of Diacritics method (Aabed et al., 2007), kashida method (Gutub et al., 2007), enhanced kashida method (Alginahi et al., 2013), enhanced kashida method A (Alginahi et al., 2014), enhanced kashida method B (Alginahi et al., 2014) and pseudo-space method (Alotaibi and Elrefaei, 2016) by about 1852%, 92%, 535%, 162%, 123%, and 106% respectively. The capacity of Method 2 has improved compared to the previous Arabic watermarking methods about 4405%, 343%, 1366%, 505%, 414% and 377% respectively. These increase percentages are calculated based on formulas in Percentage Change – Percentage Increase and Decrease (2011). Diacritics method (Aabed et al., 2007) has a variable capacity based on the existing of diacritics on the text. Even in case of using full diacritical text as shown in Fig. 3, Method 1 and Method 2 have capacity closer or higher than the capacity of diacritics method. Now, most of electronic Arabic texts are not diacritical.

4.3. Robustness Evaluation results

The proposed methods are tested to evaluate their robustness against most kind of text attacks. Tampering attack is the most common type of text attacks especially in doing academic researches. The attacker tries to change the overall look of

Table 5 The capacity of the proposed methods using 10 text samples.

Text sample	Method 1		Method 2	
	Capacity (bits)	Ratio (%)	Capacity (bits)	Ratio (%)
1	45	31.03	116	80
2	847	31.79	1988	74.62
3	3620	32.27	8128	72.47
4	7094	34.37	15,264	73.96
5	14,353	31.44	30,824	67.52
6	29,012	33.25	64,404	73.81
7	43,089	31.91	93,932	69.57
8	113,293	25.87	304,236	78.74
9	138,696	34.75	344,804	76.24
10	173,453	35.44	374,408	76.51
Average	52350.2	32.212	123810.4	74.344

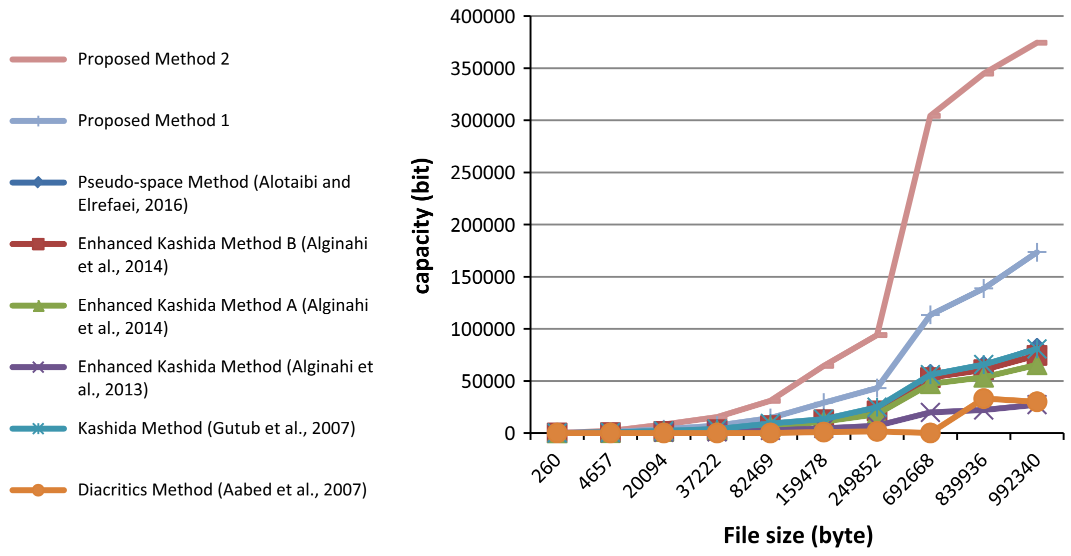


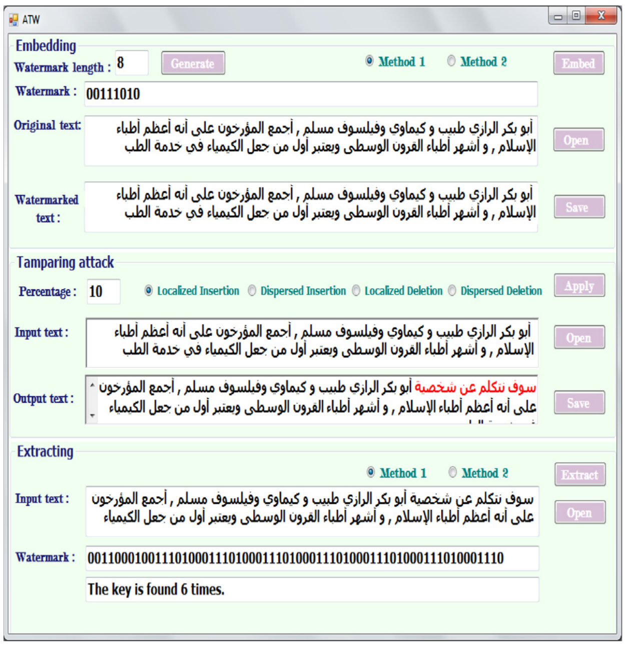
Figure 4 Capacity comparison results in different methods using ten text samples.

Table 6 Average capacity comparison results using 10 text samples.

Watermarking method	The used feature	Capacity (bits)	Capacity ratio (%)
Diacritics Method (Aabed et al., 2007)	Diacritics	7262	1.65
Kashida Method (Gutub et al., 2007)	Kashida	25,603	16.754
Enhanced Kashida Method (Alginahi et al., 2013)	Kashida	8467	5.07
Enhanced Kashida Method A (Alginahi et al., 2014)	Kashida	20,513	12.297
Enhanced Kashida Method B (Alginahi et al., 2014)	Kashida	23,441	14.452
Pseudo-space Method (Alotaibi and Elrefaei, 2016)	PS	24,841	15.585
Proposed Method 1	PS	52,350	32.212
Proposed Method 2	PS, HS, TS and ZWS	123,810	74.344

the text by inserting new words or sentences into the original text or by deleting them. Insertion and deletion processes are done in two ways: localized or dispersed (Jalil, 2010). In our experiments, localized insertion is done in one place within the watermarked text. This place can be at the beginning or the end, or anywhere in the middle of the text. In dispersed insertion, the words or sentences were added randomly in several different parts of the watermarked text. In Localized deletion, the sentences and words were deleted from one random place in the watermarked text. Dispersed deletion is the most distortion attack of the watermark because the deletions are in different random places from the text.

Fig. 5 shows the GUI used to test the robustness of the proposed methods for four different types of attack: localized insertion, dispersed insertion, localized deletion and dispersed deletion. The watermark is embedded in the original text many

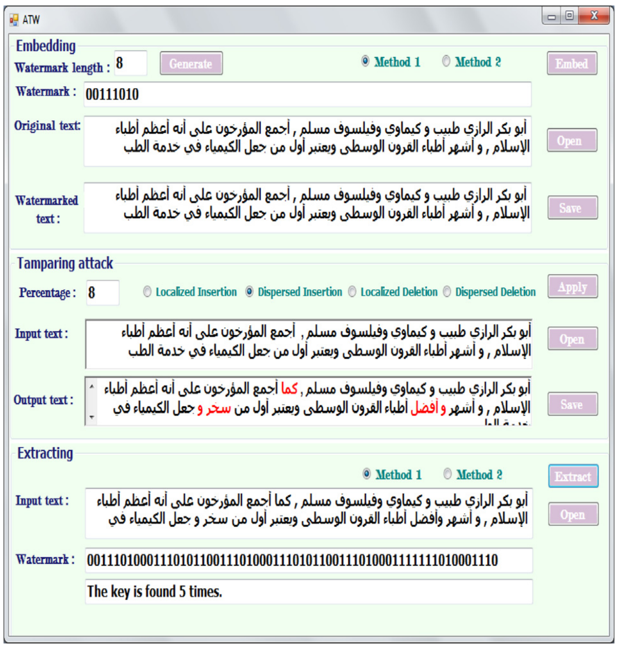


Embedding
Watermark length : 8
[✓ Method 1] Method 2
Watermark : 00111010
Original text: أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب
Watermarked text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب

Tamparing attack
Percentage : 10
[✓ Localized Insertion] Dispersed Insertion Localized Deletion Dispersed Deletion
Input text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب
Output text : سوف نتكلم عن شخصية أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء

Extracting
[✓ Method 1] Method 2
Input text : سوف نتكلم عن شخصية أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء
Watermark : 0011000100111010001110100011101000111010001110100011101000111010001110
The key is found 6 times.

a) Localized Insertion attack (inserted words are colored with red in output text).

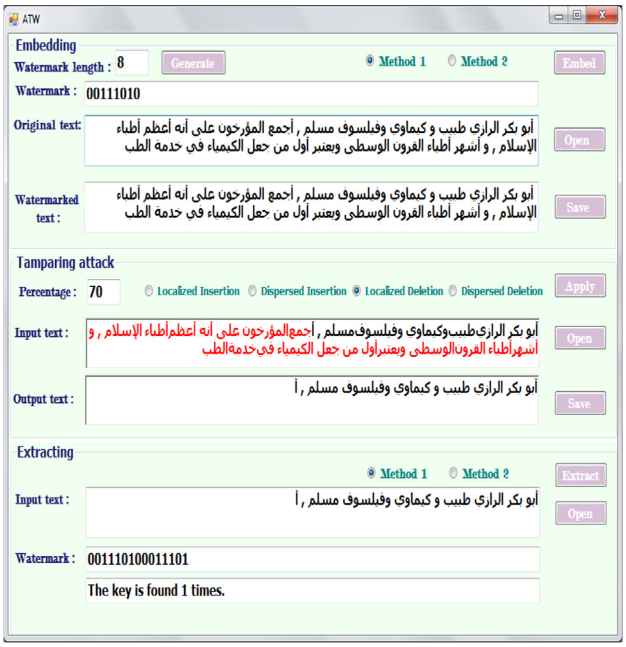


Embedding
Watermark length : 8
[✓ Method 1] Method 2
Watermark : 00111010
Original text: أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب
Watermarked text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب

Tamparing attack
Percentage : 8
Localized Insertion [✓ Dispersed Insertion] Localized Deletion Dispersed Deletion
Input text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب
Output text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , كما أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر و أفضل أطباء القرون الوسطى ويعتبر أول من سحر و جعل الكيمياء في

Extracting
[✓ Method 1] Method 2
Input text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , كما أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر وأفضل أطباء القرون الوسطى ويعتبر أول من سحر و جعل الكيمياء في
Watermark : 001110100011101011001110100011101011001110100011111111010001110
The key is found 5 times.

b) Dispersed Insertion attack (inserted words are colored with red in output text).

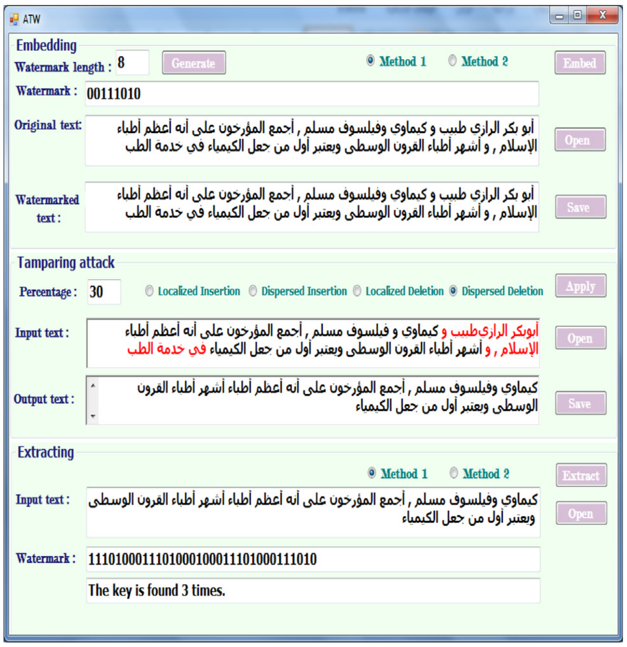


Embedding
Watermark length : 8
[✓ Method 1] Method 2
Watermark : 00111010
Original text: أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب
Watermarked text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب

Tamparing attack
Percentage : 70
Localized Insertion Dispersed Insertion [✓ Localized Deletion] Dispersed Deletion
Input text : أبو بكر الرازيطبيبوكيماوي وفيلسوفمسلم , أجمعالمؤرخون على أنه أعظمأطباء الإسلام , و أشهرأطباء القرونالوسطى ويعتبرأول من جعل الكيمياء فيخدمةالطب
Output text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أ

Extracting
[✓ Method 1] Method 2
Input text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أ
Watermark : 001110100011101
The key is found 1 times.

c) Localized Deletion attack (deleted words are colored with red in input text).



Embedding
Watermark length : 8
[✓ Method 1] Method 2
Watermark : 00111010
Original text: أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب
Watermarked text : أبو بكر الرازي طبيب و كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب

Tamparing attack
Percentage : 30
Localized Insertion Dispersed Insertion Localized Deletion [✓ Dispersed Deletion]
Input text : أبوبكر الرازيطبيب و كيماوي و فيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء الإسلام , و أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء في خدمة الطب
Output text : كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء

Extracting
[✓ Method 1] Method 2
Input text : كيماوي وفيلسوف مسلم , أجمع المؤرخون على أنه أعظم أطباء أشهر أطباء القرون الوسطى ويعتبر أول من جعل الكيمياء
Watermark : 111010001110100010001110100011101 0
The key is found 3 times.

d) Dispersed Deletion attack (deleted words are colored with red in input text).

Figure 5 GUI used for robustness testing of the proposed methods.

times based on the watermark and the text lengths to increase the robustness (Cox et al., 2007). Then, the watermarked text has been attacked and finally, the watermark is extracted from the attacked text.

The robustness is measured depending on the number of times the watermark is found in the attacked watermarked text. The watermark is considered to be survived if it is founded once or more in the attacked text, otherwise it is lost. The robustness of the proposed methods is tested using different 12 text samples listed in Table 4. Each sample is tested with watermark lengths: 8, 16, 32, 64, 128 and 256. Table 7 presents the testing scenario of the tested text attacks and its results.

Table 7 Testing scenario and results of different text attacks.

Attack type	Testing scenario	Testing result
Localized insertion	First, the user enters the insertion percentage and the watermarked text. Then, words/sentences are inserted in one location in the watermarked text based on the insertion percentage. This process is repeated until the number of times the watermark extracted is changed. Fig. 5a illustrates an example for testing Method 1 against a localized insertion attack	The proposed methods are robust against localized insertion attack as the number of times the watermark embedded is equal to the number of times the watermark extracted regardless the insertion percentage
Dispersed insertion	First, the user enters the insertion percentage and the watermarked text. Then, words/sentences are inserted in different locations in the watermarked text based on the insertion percentage. This process is repeated until the number of times the watermark extracted is changed. Fig. 5b illustrates an example for testing Method 1 against a dispersed insertion attack	In Method 1, the watermark is lost if it is embedded 3 times or less. The watermark is destroyed in text samle1 with watermark lengths 16 and more and in text sample 2 with watermark length 256 In Method 2, the watermark is lost if it is embedded only once. This can not be happened except in very tiny text sizes
Localized deletion	First, the user enters the deletion percentage and the watermarked text. Then, words/sentences are deleted from one location in the watermarked text based on the deletion percentage. This process is repeated until the number of times the watermark extracted is changed. Fig. 5c illustrates an example for testing Method 1 against a localized deletion attack	The proposed methods are robust against localized deletion attack using medium and larger text. The watermark is lost only in case of deleting 100% of text which means deleting all the text. The required deletion percentage to destroy the watermark in Method 1 and Method 2 using small size text is between 97% and 100% and 99% and 100% respectively. Text sample No. 1 and 2 are used to represent tiny text. Text sample No. 1 is a very small text includes only 29 words which are nothing important to be watermarked. In tiny text, the required deletion percentages vary between 0 to 100% depending on the number of times the watermark repeated in the watermarked text before it attacked and the watermark length
Dispersed deletion	First, the user enters the deletion percentage and the watermarked text. Then, words/sentences are deleted from different locations in the watermarked text based on the deletion percentage. This process is repeated until the number of times the watermark extracted is changed. Fig. 5d illustrates an example for testing Method 1 against a dispersed deletion attack	Using medium, large, huge and gigantic texts, the dispersed deletion percentage to destroy the watermark is 100% in both methods. In small size texts, the dispersed deletion percentage is ranging from 84% to 100% in Method 1 and from 93% to 100% in Method 2. The deletion percentage is between 0% and 97% in Method 1, 0% and 100% in Method 2 using tiny texts
Copying and pasting	The user copies the watermarked text, then pastes it in another program and saves it. After that the watermark is extracted from the saved file	The proposed methods are robust against copying and pasting attack as the number of times the watermark embedded is equal to the number of times the watermark extracted
Formatting	The user changes the watermarked text style like: font style, text size, coloring, highlighting and any other effects. Then, the watermark is extracted	The proposed methods are robust against formatting attacks as the number of times the watermark embedded is equal to the number of times the watermark extracted
Retyping, Printing, and OCR	The user retypes the watermarked text, or prints it, then scans it. After that OCR program is used to turn the scanned printed text into an electronic one. Finally the watermark is extracted	The proposed methods depend on the existence of pseudo-space. This character does not print and does not have any width, So the proposed methods are not robust against retyping, printing or OCR

5. Conclusion

Two methods for Arabic text watermarking are proposed by utilizing each word space. Method 1 is specialized to be used in the Arabic text or similar languages as it utilizes the Arabic dotting feature. Method 2 could be used for any language uses spaces to separate their words. The contribution of Method 2 is the using of four tiny spaces: pseudo-space, thin space, hair space and zero width space with normal word space. The performance of the proposed methods is tested to evaluate their capacity, imperceptibility and robustness results and compared to some Arabic text watermarking methods using different text samples with six watermark lengths. The proposed methods found to have the highest capacity and imperceptibility. Method 2 has the largest capacity, but slightly lower imperceptibility than Method 1. The proposed methods are robust against electronic text attacks such as: copying and pasting, text formatting and text tampering for tampering ratio up to 84%.

As a future work, text watermark can be used with compression algorithms such as Huffman algorithm instead of using binary bits to represent the watermark. A private key could be used in the proposed watermarking methods to prove the authenticity. The proposed methods also could be combined with other Arabic watermarking methods from the literature.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

Aabed, M.A., Awaideh, S.M., Elshafei, A.R.M., Gutub, A.A., 2007. Arabic diacritics based steganography. In: ICSPC 2007. IEEE International Conference on Signal Processing and Communications, IEEE, pp. 756–759.

Alginahi, Y.M., Kabir, M.N., Tayan, O., 2013. An enhanced Kashida-based watermarking approach for Arabic text-documents. In: 2013 International Conference on Electronics, Computer and Computation (ICECCO), IEEE, pp. 301–304.

Alginahi, Y.M., Kabir, M.N., Tayan, O., 2014. An enhanced Kashida-based watermarking approach for increased protection in Arabic text-documents based on frequency recurrence of characters. Int. J. Comput. Electr. Eng. 6 (5), 381.

Alotaibi, R.A., Elrefaei, L.A., 2015. Arabic Text Watermarking: A Review. arXiv preprint arXiv:1508.01485, pp. 1–16.

Alotaibi, R.A., Elrefaei, L.A., 2016. Utilizing word space with pointed and un-pointed letters for Arabic text watermarking. In: 2016 UKSim-AMSS 18th International conference on Computer Modelling and Simulation, pp. 111–116.

Bender, W., Gruhl, D., Morimoto, N., Lu, A., 1996. Techniques for data hiding. IBM Syst. J. 35 (3.4), 313–336.

Bennett, K., 2004. Linguistic steganography: survey, analysis, and robustness concerns for hiding information in text. Purdue University. CERIAS Tech. Report 2004–13..

Bensaad, M.L., Yagoubi, M.B., 2011. High capacity diacritics-based method for information hiding in Arabic text. In: International Conference on Innovations in Information Technology (IIT), IEEE, pp. 433–436.

Cox, I., Miller, M., Bloom, J., Fridrich, J., Kalker, T., 2007. Digital Watermarking and Steganography. Morgan Kaufmann.

Gutub, A.A.A., Ghouti, L., Amin, A.A., Alkharobi, T.M., Ibrahim, M.K., 2007. Utilizing extension character 'Kashida' with pointed letters for Arabic text digital watermarking. In: SECRYPT, pp. 329–332.

Gutub, A.A.A., Al-Alwani, W., Mahfoodh, A.B., 2010. Improved method of Arabic text steganography using the extension 'Kashida'character. Bahria Univ. J. Inf. Commun. Technol. 3, 68–72.

Huang, D., Yan, H., 2001. Interword distance changes represented by sine waves for watermarking text images. IEEE Trans. Circuits Syst. Video Technol. 11 (12), 1237–1245.

Index of /AraCorpus, 2010. Aracorpus.e3rab.com. http://aracorpus.e3rab.com/argistestsrv.nmsu.edu/AraCorpus/?C = D;O = A (accessed 2016).

Jalil, Z., 2010. Copyright Protection of Plain Text Using Digital Watermarking (Doctoral dissertation). National University of Computer and Emerging Sciences, Islamabad.

Kashyap, V., 2010. Top 7 Windows Search Tricks Used by Pro Users, Makeofuse. http://www.makeuseof.com/tag/top-7-windows-search-tricks-search-ninja (accessed 2016).

Percentage Change – Percentage Increase and Decrease, 2011. Skillsyouneed.com. http://www.skillsyouneed.com/num/percent-change.html (accessed 2016).

Saad, M.K., Ashour, W., 2010. OSAC: Open Source Arabic Corpus. In: 6th ArchEng International Symposiums, EEECS10 the 6th International Symposium on Electrical and Electronics Engineering and Computer Science. European University of Lefke, Cyprus.

Shah, A., Memon, M.S., 2011. A novel text steganography technique to Arabic language using reverse fatah. Pak. J. Eng. Technol. Sci. (PJETS) 1 (2), 106–113.

Shirali-Shahreza, M., 2008. A New Persian/Arabic text steganography using "La" word. In: Advances in Computer and Information Sciences and Engineering. Springer, Netherlands, pp. 339–342.

Shirali-Shahreza, M.H., Shirali-Shahreza, M., 2006. A new approach to Persian/Arabic text steganography. In: 5th IEEE/ACIS International Conference on Computer and Information Science and 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse (ICIS-COMSAR'06), pp. 310–315.

Shirali-Shahreza, M.H., Shirali-Shahreza, M., 2008. Steganography in Persian and Arabic unicode texts using pseudo-space and pseudo connection characters. J. Theor. Appl. Inf. Technol. 4 (8).

Shirali-Shahreza, M.H., Shirali-Shahreza, M., 2010. Arabic/Persian text steganography utilizing similar letters with different codes. Arabian J. Sci. Eng. 35 (1b).

Stanković, S., Orović, I., Sejdić, E., 2012. Multimedia Signals and Systems. Springer, New York, NY.

Topkara, M., Riccardi, G., Hakkani-Tür, D., Atallah, M.J., 2006. Natural language watermarking: challenges in building a practical system. In: Electronic Imaging 2006 International Society for Optics and Photonics, pp. 60720A–60720A.

Whitespace Character, 2016. < https://en.wikipedia.org/wiki/Whitespace > character (accessed 2016).

Yang, H., Kot, A.C., 2004. Text document authentication by integrating inter character and word spaces watermarking. In: Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference, vol. 2, pp. 955–958.