PAPER

A Novel Linguistic Steganography Based on Synonym Run-Length Encoding

Lingyun XIANG[†a)], Xinhui WANG[†], Chunfang YANG[††], *Nonmembers*, *and* Peng LIU[†††], *Member*

SUMMARY In order to prevent the synonym substitution breaking the balance among frequencies of synonyms and improve the statistical undetectability, this paper proposed a novel linguistic steganography based on synonym run-length encoding. Firstly, taking the relative word frequency into account, the synonyms appeared in the text are digitized into binary values and expressed in the form of runs. Then, message are embedded into the parities of runs' lengths by self-adaptively making a positive or negative synonym transformation on boundary elements of two adjacent runs, while preserving the number of relative high and low frequency synonyms to reduce the embedding distortion. Experimental results have shown that the proposed synonym run-length encoding based linguistic steganographic algorithm makes fewer changes on the statistical characteristics of cover texts than other algorithms, and enhances the capability of anti-steganalysis.

key words: *information security, linguistic steganography, synonym substitution, run-length encoding, steganalysis*

1. Introduction

Nowadays, there are more and more digital multimedia transmitted on the network. They can be selected to hide the secret message by steganography for covert communication. Steganography aims to embed secret message into seemly ordinary media such as texts, images, videos, without arousing observer's suspicions. With the prevalence of text-based information, transmitting secret message by the steganography taking the text as a cover will not easily draw attention of suspect. However, text steganography is commonly regarded as a challenging topic in the field of data hiding as the few redundant embedding spaces and sophisticated natural language processing techniques.

Steganography methods for texts can be categorized into two categories: format-based and linguistic methods. The format-based methods always hide message by character spaces assignment, line shifting and word shifting [1], changing font format in compound documents or some sole characteristics in specific documents [2]. Linguistic steganography referred to natural language steganography [3], mainly uses mimicking techniques to directly generate a new cover text [4], or linguistically modifies content of cover text by synonym substitution (SS) [5]–[11], syntactic transformation [12], [13], or translation [14], [15] to camouflage the secret message.

Synonym substitution is the major and famous transformation used in linguistic steganography. The steganography substitutes the selected words with their synonyms so that the updated synonyms sequence with predefined encoded values can represent secret message. The meaning of the stego text can be preserved the same as the cover one in theory. However, the current state-of-the art of synonym substitution based steganography faces several practical problems and is difficult to generate perfect stego text without grammar mistake, syntax error, semantic distortion, etc.

The first problem is that words can have more than one sense, even more than one part of speech. It must ensure that a word can be replaced by its synonyms with right senses and part of speech, and the extraction process can unmistakably recover the embedded secret message. Using absolute synonyms for carrying the secret message is one solution to this problem. Absolute synonyms mean that words are synonymous in any of their senses. Muhammad et al.[7] proposed a method to only adopt two absolute synonyms of each synonym set for embedding message. The method obtained good imperceptibility, but it greatly reduce the embedding capacity and anti-steganalysis capability. Bolshakov [6] also employed the relative synonyms for embedding message, which were previously tested for semantic compatibility with collocations to determine whether synonym substitutions were correct. However, they just gave a manually traced Russian example of their proposed method without automatically generating stego texts. Chang et al.[8] proposed a novel solution which elaborately designed a vertex coding to ensure each synonymous word be encoded as the same value in different senses.

The second problem is how to make the new synonyms are suitable for the contexts. Inappropriate synonyms introduced by substitutions will reduce the quality and readability of stego texts, lead to bad imperceptibility and much suspicion to the existence of the secret. In order to generate unsuspicious stego text, Topkara et al.[9] just chose the prior alternative for every synonym to be replaced according to their semantic similarities. While the steganography in [10] used the disambiguation function to determine which synonym was right to the current context. [8] utilized the Google n-gram corpus to check the acceptability of a syn-

Manuscript received August 22, 2016.
Manuscript revised October 20, 2016.
Manuscript publicized November 8, 2016.

[†]The authors are with the Hunan Provincial Key Laboratory of Intelligent Processing of Big Data on Transportation and School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha, China.

[††]The author is with the Zhengzhou Science and Technology Institute, Zhengzhou, China.

[†††]The author is with the College of Information Science and Engineering in Hunan University, Hunan University, Changsha, China.

a) E-mail: xiangly210@163.com

DOI: 10.1587/transinf.2016EDP7358

onym in context. These methods all relied on the capabilities of the natural language processing techniques. Not only the sender but also the receiver should master advanced NLP techniques and abundant linguistic resource.

The third problem is that most existing linguistic steganographic methods are vulnerable while facing to steganalysis. The stego text should not only have good imperceptibility in the aspect of natural language, but also in the aspect of statistical characteristics. Steganalysis [16], [17] is to discover the existence of hidden message by analysis the statistical characteristics in stego and cover texts. For the synonym substitution based steganography, [18] utilized the statistics of context fitness estimated by context clusters to distinguish stego texts from normal ones. [19] used the relative frequency information from the same synonym substitution sets as a feature vector to classify the stego and cover texts. [20] designed a similar statistical steganalysis using features deriving from the synonymous words' frequency distributions. These steganalysis methods seriously threaten the security of the secret message in the stego texts.

However, little effort has been made to improve the anti-steganalysis capability of synonym substitution based steganography. Yang et al.[11] applied the matrix encoding to synonym substitution-based steganography to reduce the modifications to decrease the possibility of stego texts being discovered by steganalysis. However, this method cannot adaptively embed message with various embedding rate, and still changes the word frequency characteristics of cover texts in the embedding process which will be used by steganalysis.

In this paper, to reduce the difference between the word frequency characteristics of the stego and corresponding cover text, and achieve high security, an efficient synonym run-length encoding method is proposed to represent and embed the secret message. The secret message is embedded by adaptively selecting transformations among relative high and low frequency synonymous words. The proposed run-length encoding method preserves the statistical characteristics of the synonyms in the embedding process, thus offers the capability of resistance against steganalysis techniques.

The rest of this paper is organized as follows. Section 2 introduces the principle of synonym run-length encoding method to encode the synonyms in a text for message embedding. In Sect. 3, the run-length encoding is applied to design a steganographic method. Section 4 details the experimental results and analysis. Finally, the paper ends with conclusions in Sect. 5.

2. Stego Encoding Method Based on Synonym Run-length

2.1 Synonym Representation

The statistical characteristics of the word frequency can provide important clues for the steganalysis because they are always changed by the synonym substitutions. [20] theoretically analyzed the impact caused by synonym substitutions in the steganography, and pointed out that the number of high frequency synonyms may be reduced while that of low frequency synonyms would be increased. Then it extracted some statistical features in the view of the word frequency distributions to capture these changes for SVM to discriminate stego and cover texts. Therefore, in order to improve the anti-steganalysis capability, one should preserve the original natural statistical characteristics of the word frequency across the synonym substitutions.

In this paper, only the absolute synonyms are taken as cover words, and the relative synonyms are excluded. Besides, the impact caused by the context is ignored under an assumption that the used synonyms for embedding message are always suitable for the current context.

The synonymous words in the same synonym set will be represented as a unique binary digital in this paper. Given a synonym set $S = \{s_0, s_1, \ldots, s_{n-1}\}$ including n synonymous words in descending order of their frequency, i.e. $f(s_i) \geq f(s_{i+1})$, where $f(s_i)$ denotes the relative frequency of synonym s_i derived from a huge corpus, and $\sum_{i=0}^{n-1} f(s_i) = 1$, then the synonym s_i is digitized by the rule shown in Eq. (1).

$$d(s_i) = \begin{cases} 0 & i = 0 \\ 1 & else \end{cases} \tag{1}$$

where $d(s_i)$ represents the digitized value of s_i. $d(s_0) = 0$ means s_0 is converted to a digital '0', and its relative frequency is maximum among those of synonymous words in S; otherwise, s_i is represented as a digital '1'. In a word, for the synonymous words, the word with the highest relative frequency is denoted as a digital '0', while all others are denoted as a digital '1'.

Definition 1 Relative high frequency synonym (RHF synonym): a synonym whose digitized value is '0'.

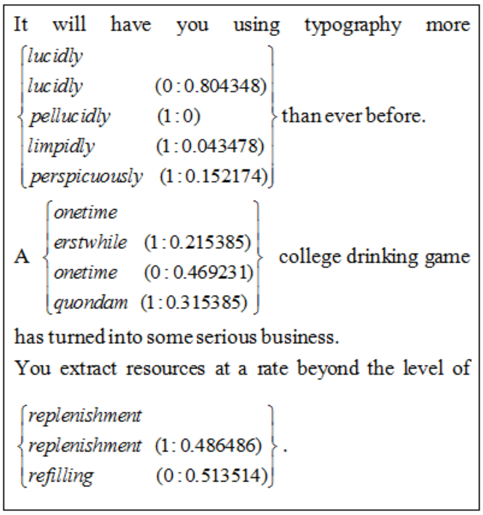
Definition 2 Relative low frequency synonym (RLF synonym): a synonym whose digitized value is '1'.

This representation method implies two meanings. The first is that the synonyms in the same set are just represented to two digital numbers, no matter how many synonyms the set includes. The second is that the digital assigned to a synonym is associated with its relative frequency. Figure 1 shows a fragment of a sample text that contains three synonyms. It lists the synonymous words of the appeared words with their assigned digital values and relative word frequencies, which are in the form of *synonymous word (assigned digital, relative word frequency)*.

In the sample text, there are three synonyms *lucidly*, *onetime*, *replenishment*, which are located in the following synonym sets, respectively.

{*lucidly, pellucidly, limpidly, perspicuously*};
{*erstwhile, onetime, quondam*};
{*replenishment, refilling*}.

Taking the word frequency counted from British National Corpus as an example, we calculate the relative word



It will have you using typography more
lucidly
lucidly (0:0.804348)
pellucidly (1:0)
limpidly (1:0.043478)
perspicuously (1:0.152174)
than ever before.

A
onetime
erstwhile (1:0.215385)
onetime (0:0.469231)
quondam (1:0.315385)
college drinking game
has turned into some serious business.

You extract resources at a rate beyond the level of
replenishment
replenishment (1:0.486486)
refilling (0:0.513514)
.

Fig. 1 A fragment of a sample text with digitized synonyms

frequency of each synonym list in Fig. 1. Since *lucidly* has the highest relative word frequency compared with its other three synonymous words, thus the synonym digitization results are:

$d(lucidly) = 0$; $d(pellucidly) = 1$;
$d(limpidly) = 1$; $d(perspicuously) = 1$;

The synonyms in the other two synonym set can be expressed as digital '0' or '1' according to Eq. (1). Finally, the synonym sequence "lucidly, onetime, replenishment" will be digitized into a binary sequence "001".

2.2 Synonym Run

Suppose the cover text T contains N synonyms $w_0, w_1, \dots, w_{N-1}$. Using the above synonym digitization method, the synonyms are converted into binary sequence D, which can be simply expressed as

$$D = d(w_0) \dots d(w_i) \dots d(w_{N-1}), i = 1, \dots, N-2 \qquad (2)$$

Definition 3 In the binary sequence converted from synonyms, the binary substring with repeating digital values is defined as a run.

A run is recorded by two symbols, run-length and run-value. Run length denotes the number of the digitals or bits in a run. Run value indicates the value of the digital. A run $d(w_k) \dots d(w_{k+L-1})$ satisfies the following equations:

$$\begin{cases} d(w_k) \neq d(w_{k-1}) \\ d(w_{k+j}) = d(w_{k+j-1}) \\ d(w_{k+L}) \neq d(w_{k+L-1}) \end{cases} \qquad (3)$$

where $j = 1, \dots, L-1$, and L denotes the run-length of this run. The corresponding run value equals to the value of the run $d(w_k)$. For example, a binary sequence "00011" contains two runs "000" and "11", their run-lengths are 3 and 2, run values are '0' and '1', respectively.

Suppose D composes of N_1 runs, it can be written as Eq. (4) in form of runs.

$$D = r_1 r_2 \dots r_{N_1}, r_i = d(w_k), \dots, d(w_{k+l(r_i)-1}) \qquad (4)$$

where $i = 1, \dots, N_1$, $l(r_i)$ is the run-length of run r_i, $k = \sum_{j=1}^{i-1} l(r_j)$. Denote the run-value of run r_i as $v(r_i)$.

Definition 4 If run-length is greater than 1, then the run is defined as a modifiable run, otherwise it is an unmodifiable run.

When run-length $L = 1$, an unmodifiable run satisfies the following equations:

$$\begin{cases} d(w_k) \neq d(w_{k-1}) \\ d(w_k) \neq d(w_{k+1}) \end{cases} \qquad (5)$$

It just contains only one element $d(w_k)$. $d(w_{k-1})$ belongs to the previous run, and $d(w_{k+1})$ belongs to the next run.

2.3 Stego Encoding Method Based on Synonym Run-Length

Definition 5 The encoded value of a run is defined as the parity of its run-length. If the run-length of a run is even, then its encoded value is '0'; otherwise, it is '1'.

According to Definition 5, a run can be encoded into a binary value as shown in Eq. (6).

$$E(r_i) = l(r_i) \mod 2 \qquad (6)$$

The encoded value of a run can be flipped by a synonym substitution operation to increase or decrease the run-length. The value of the element in the run will be flipped from '0' to '1', or from '1' to '0'. Namely, a RHF synonym is replaced by its RLF synonymous word or a RLF synonym is replaced by its RHF synonym.

Definition 6 Positive synonym transformation: an operation which substitutes a RHF synonym by randomly choosing one of its RLF synonymous words to flip the digitized value from '0' to '1'.

Definition 7 Negative synonym transformation: an inverse operation of the positive synonym transformation, which substitutes a RLF synonym with its RHF synonymous word to flip the digitized value from '1' to '0'.

Since a text always includes more RHF synonyms than RLF ones, the rate of RHF synonyms to RLF synonyms in the text will keep as a relative high value. Once the steganography disturbs the distributions of RHF and RLF synonyms, they will be taken as the clues to discover the existence of the hidden secret message. Therefore, to preserve statistical characteristics of the cover texts, a modification constraint should be defined to be followed when one alters the parity of the run-length to embedding message.

Modification constraint: The modifications done to change synonym run-lengths for embedding message should keep the number of RHF synonyms and RLF synonyms in a text be unchanged as far as possible.

Theorem 1: The modification constraint can be satisfied by making the same number of positive and negative synonym transformations.

Proof: A positive synonym transformation will cause that the number of RLF synonym is increased by 1, while the number of RHF synonym is decreased by 1. A negative synonym transformation will cause an inverse impact. The number of RLF synonym is decreased by 1, while the number of RHF synonym is increased by 1. Thus, in order to make that the increased number of RHF synonyms is equal to the decreased one after embedding message, there must make the same number of positive and negative synonym transformations.

End;

Theorem 2: The encoded value of a modifiable run can be flipped both by a positive and negative synonym transformation made on boundary elements of the two adjacent modifiable runs.

Proof: Given the two adjacent modifiable runs expressed as follows:

$$r_i = d(w_k), \ldots, d(w_{k+l(r_i)-1}) \tag{7}$$

$$r_{i+1} = d(w_{k+l(r_i)}), \ldots, d(w_{k+l(r_i)+l(r_{i+1})-1}) \tag{8}$$

where $l(r_i) > 1$, and $l(r_{i+1}) > 1$. When the encoded value of the current modifiable run r_i is required to be flipped, the parity of its run-length can be changed by decreasing or increasing the run-length of r_i by 1, namely, $E(r_i') = (l(r_i) \pm 1) \mod 2 = \overline{E(r_i)}$.

If a synonym substitution is made on boundary element $w_{k+l(r_i)-1}$ of r_i, $w_{k+l(r_i)-1}$ is substituted with its synonymous word $w'_{k+l(r_i)-1}$, then $d(w_{k+l(r_i)-2}) \neq d(w'_{k+l(r_i)-1})$, and $d(w_{k+l(r_i)}) = d(w'_{k+l(r_i)-1})$. $w'_{k+l(r_i)-1}$ will be an element of the new next run r'_{i+1}, and run-length of r'_i is decreased by 1 to be $l(r_i) - 1$.

On the other hand, if a synonym substitution is made on $w_{k+l(r_i)}$, which is the first element of run r_{i+1}, $w_{k+l(r_i)}$ is substituted with its synonymous word $w'_{k+l(r_i)}$, then $d(w'_{k+l(r_i)}) = d(w_{k+l(r_i)-1})$, and $d(w'_{k+l(r_i)}) \neq d(w_{k+l(r_i)+1})$. $w'_{k+l(r_i)}$ will be an element of the new current run r'_i, and run-length of r'_i is increased by 1 to be $l(r_i) + 1$. Since $d(w_{k+l(r_i)-1}) \neq d(w_{k+l(r_i)})$ before making a synonym substitution, one of the above two synonym substitutions must be a positive transformation, and the other one is a negative one.

End;

Theorem 3: The encoded value of an unmodifiable run cannot be flipped by a synonym transformation made on its only element.

Proof: Given the current unmodifiable run $r_i = d(w_k)$, its previous run $r_{i-1} = d(w_{k-l(r_{i-1})}), \ldots, d(w_{k-1})$, and next run $r_{i+1} = d(w_{k+1)}), \ldots, d(w_{k+l(r_{i+1})})$. It must be $d(w_k) \neq d(w_{k-1})$ and $d(w_k) \neq d(w_{k+1})$. If a synonym transformation is made to substituted w_k by its synonymous word w'_k, then $d(w'_k) = \overline{d(w_k)}$, thus $d(w'_k) = d(w_{k-1}) = d(w_{k+1})$, which leads to that r_i disappears and its only element is merged with its adjacent runs r_{i-1} and r_{i+1} into a new run. Then, the parity of the merged run's length possibly does not equal to the message bit previously embedded into the run $r_{i-1} = d(w_{k-l(r_{i-1})}), \ldots, d(w_{k-1})$. This will cause the embedded secret message be recovered incorrectly. Therefore, the only element of an unmodifiable run cannot be transformed to change its encoded value.

End;

If the embedded secret bit is inconsistent with the encoded value of the corresponding run, one can alter the run-length by synonym substitutions on boundary elements of the modifiable runs to flip the encoded value. In order to comply with the modification constraint, one must self-adaptively and elaborately make alternative choices of positive or negative synonym transformations to keep the balance of RHF and RLF synonyms. However, there exist unmodifiable runs, whose element cannot be changed. As no matter whether the current run is modifiable or not, the modification on the boundary element will not only change its run-length but also that of its adjacent run, thus we employ the next run r_{i+1} with the current run r_i together to analyze how to embed a secret message bit, when the encoded value of the current run does not equal to the current secret message bit. There are four cases described as follows.

Case 1: the current run r_i is modifiable; the next run r_{i+1} is unmodifiable.

In this case, a synonym transformation could only be made on the last element of the modifiable current run r_i to flip its encoded value when its encoded value does not equal to the current secret message bit.

Since r_{i+1} is an unmodifiable run, the synonym transformation cannot be made on its only element, otherwise it will cause three adjacent runs to be merged. Thus, when $E(r_i)$ does not equal to the current secret message bit m_i, one must make synonym transformation on the last element of r_i to decrease run-length by 1, so that the encoded value of r_i will be flipped.

For example, suppose a digitized synonym sequence containing three adjacent runs is ”001000”, the current run r_i is the first run ”00”, its next run r_{i+1} is an unmodifiable run ‘1’, and the next run of r_{i+1} is ”000”. If the synonym transformation is done on r_{i+1} to make ‘1’ be changed to ‘0’, it will make this three runs merged to a new run ”000000” whose encoded value is the same as it of the current run r_i. In order to avoid this failed embedding, one could only make synonym transformation on the modifiable run r_i and flip its last element to ‘1’. Thus, the updated current run will be ‘0’, and the next run will be ”11”. As a result, the encoded value of the current run will be changed to ‘1’ from ‘0’.

Case 2: the current run r_i is unmodifiable; the next run r_{i+1} is modifiable.

In this case, a synonym transformation could only be made on the first element of the modifiable run r_{i+1} to flip the encoded value of r_i when the encoded value of the current run does not equal to the current secret message bit.

According to Theorem 3, the synonym transformation cannot be made on r_i, thus we must make synonym transformation on the first element of r_{i+1} to change its value. As the changed value equals to the run-value of r_i, the first element of r_{i+1} will belong to r_i after synonym transformation. As a result, the run-length of the updated current run will be increased by 1, and its encoded value will be flipped.

For example, suppose a digitized synonym sequence containing three adjacent runs is ”001000”, the current run r_i is the second run which is a unmodifiable run ‘1’, its next run r_{i+1} is a unmodifiable run ”000”, and its previous run r_{i-1} is ”00” which has be embedded a message bit ‘0’ according to its encoded value. When the current message bit ‘0’ is required to be embedded, which does not equal to the encoded value of r_i, one should do a synonym transformation on the first element of r_{i+1} to make encoded value of r_i be changed from ‘1’ to ‘0’, which will belong to the current run. Thus, the updated current run will be ”11”, and the next run will be ”00”. As a result, the encoded value of the updated current run will be changed to ‘0’ from ‘1’, which will equal to the current message bit.

Since the RLF synonym is much fewer than RHF synonym, the number of unmodifiable runs composing of one RLF synonym is much larger than that of unmodifiable runs composing of one RHF synonym. Therefore, in the above case 1 and 2, the probability of the modifiable run composing of RHF synonyms is higher than that of RLF synonyms. When the synonym transformation must be made on the only modifiable run of the two adjacent runs, there must be made more positive synonym transformations than negative ones, which leads to more RHF synonyms being substituted by RLF synonyms. In order to make the same number of positive and negative synonym transformations in terms of Theorem 1, one should make more negative transformations than positive ones in case 3 and 4.

N_p and N_n are used to record the number of positive, negative transformations, respectively. N_p and N_n are initialized to 0. When a positive transformation is made, N_p is increased by 1; when a negative transformation is made, N_n is increased by 1.

Case 3: the current run r_i is modifiable; the next run r_{i+1} is modifiable.

In this case, when the encoded value of the current run does not equal to the current secret message bit, if there have made more positive transformation, then a negative transformation on the boundary element of the two adjacent runs must be chose to flip the encoded value of the current run; otherwise, a positive transformation is chose.

According to Theorem 2, there are two choices of synonym transformation to flip the encoded value of the current run. In order to satisfy the modification constraint as far as possible, one can self-adaptively employ a positive or a negative transformation to correct the unbalance between the number of RHF synonyms and the number of RLF ones caused by the case 1 and case 2. If there have carried out more positive transformations than negative ones, i.e., $N_p \geq N_n$, then a negative transformation will be made, and N_n will be increased by 1; else if $N_p < N_n$, a positive transformation will be made, and N_p will be increased by 1.

For example, given the current run r_i is ”000”, the next run r_{i+1} is ”111”, the current message bit m_i is ‘0’, $E(r_i) = 1 \neq m_i$.

If $N_p = 10$, $N_n = 9$, $N_p \geq N_n$, one should choose to make a negative transformation on the first element of r_{i+1} to make its value be changed from ‘1’ to ‘0’. Then the updated current run will be ”0000”, the next run will be ”11”, $N_n = 9 + 1 = 10$. Finally, the encoded value of the current run will be changed to ‘0’ from ‘1’, $N_p = N_n$, and the number of RHF synonyms to that of RLF synonyms will keep unchanged in the cover and stego text.

If $N_p = 9$, $N_n = 10$, $N_p < N_n$, one should choose to make a positive transformation on the last element of r_i to make its value be changed from ‘0’ to ‘1’. Then the updated current run will be ”00”, the next run will be ”1111”, $N_p = 9+1 = 10$. Finally, the encoded value of the current run will be changed to ‘0’ from ‘1’, and $N_p = N_n$.

Case 4: the current run r_i is unmodifiable; the next run r_{i+1} is unmodifiable.

In this case, if there have made more positive transformation, then a negative transformation should be chosen to be made on the run with the run-value of ‘1’; otherwise, a positive transformation should be chosen to be made on the run with the run-value of ‘0’. Then a new current run will be formed, and the current message bit should be embedded into the new formed current run again.

Given four runs r_{i-1}, r_i, r_{i+1}, r_{i+2}, r_i is the current run, r_i and r_{i+1} are both unmodifiable, namely, they contain only one element. The run-value of r_{i-1} equals to that of r_{i+1}, while the run-values of r_i and r_{i+2} are the same. In order to alter the encoded value of the current run to make $E(r_i) = m_i$, the following two manners can be adopted to change the length of current run.

(1) Do the synonym transformation on the only element of the current run r_i. Then r_i will be merged with its previous run r_{i-1} and next run r_{i+1} to form a new run r'_{i-1} to replace r_{i-1}. Since $l(r_i) = l(r_{i+1}) = 1$, then $E(r'_{i-1}) = (l(r_{i-1}) + 2) \bmod 2 = E(r_{i-1})$. The mergence will not change the embedded message bit in the run r_{i-1}, but the current bit cannot be embedded into the current run, which should be embedded into a run next to r'_{i-1} by taking account of the cases of the updated two runs next to r'_{i-1}.

(2) Do the synonym transformation on the next run r_{i+1}. Then, the current run r_i will be merged with the two next run r_{i+1} and r_{i+2} to form a new run r'_i, and $E(r'_i) = E(r_{i+2})$. However, $E(r'_i)$ does not always equal to m_i. Thus, it will be required to embed m_i into the new r'_i by analyzing the updated r'_i and r'_{i+1}.

In a word, no matter making transformation on r_i or r_{i+1}, r_i and r_{i+1} will be merged into r_{i-1} or r_{i+2}, and the encoded value of r_{i-1} and r_{i+2} will not be changed. The current message bit should be embedded again into the new formed current run. However, the value of r_i and r_{i+1} is different, thus, one of the two synonym transformations on r_i

and r_{i+1} is a positive transformation, and the other one must be a negative one. According to the recorded number of the made positive transformation and that of the made negative transformation, a positive or negative transformation can be self-adaptively chosen to be made on r_i and r_{i+1}. If a negative transformation is chosen, then it should be made on the run, whose run-value is '1'; otherwise, a positive transformation should be made on the run, whose run-value is '0'. After the transformation, the runs will be updated without embedding the current message bit m_i. Thus, m_i should be operated to be embedded into the new ith run according to cases of the new ith run and $(i + 1)$th run.

3. Linguistic Steganography Based on Synonym Run-length Encoding

3.1 Embedding Algorithm

According the analysis above, a novel linguistic steganographic algorithm is proposed, which contains two parts: embedding algorithm and extraction algorithm. The embedding algorithm is described in details.

Input: Secret message M, cover text C, synonym database SD and relative word frequencies of synonyms.

Output: stego text C'.

Step 1: According to the synonym database SD, traverse cover text C to recognize the appearing synonyms, and digitalize them by looking up their relative word frequencies. Denote the resulting digitalized binary sequence as D, and express it in the form of runs as following:

$D = \{r_i | i = 1, \ldots, N, r_i = d(w_k, \ldots, d(w_{k+1(r_i)-1})\}$

$l(r_i)$ is the run-length of the ith run r_i, whose run-value is denoted as $v(r_i)$, N is the number of the runs.

Step 2: Convert the secret message into a binary sequence $M = \{m_1, m_2, \ldots, m_t\}$.

Step 3: Initial $N_p = N_n = 0$, where N_p, N_n is used to record the number of the positive transformation, and that of the negative transformation, respectively.

Step 4: Set $i = 1$.

Step 5: If $N - 1 < t$, the current embedding capacity cannot satisfy the requirement, then the secret message must be split and embedded into other more suitable cover texts. The embedding fails.

Step 6: If $E(r_i) = m_i$, then go to step 7; otherwise, follow the modification rules to embed m_i.

1) r_i is a modifiable run, r_{i+1} is a unmodifiable run

A synonym transformation is done to the last element of r_i. r_i will be reduced an element, while r_{i+1} is added an element. If $v(r_i) = 0$, then it is a positive transformation, and $N_p = N_p + 1$; else it is a negative one, and $N_n = N_n + 1$.

2) r_i is a unmodifiable run, r_{i+1} is a modifiable run

A synonym transformation is done to the first element of r_{i+1}. r_{i+1} will be reduced an element, while r_i is added an element. If $v(r_{i+1}) = 0$, then it is a positive transformation, and $N_p = N_p + 1$; else it is a negative one, and $N_n = N_n + 1$.

3) r_i is a modifiable run, r_{i+1} is a modifiable run

If $v(r_i) = 0$, and $N_p < N_n$, then a positive transformation is done to the last element of r_i, $N_p = N_p+1$; if $v(r_i) = 1$, and $N_p \geq N_n$, then a negative transformation is done to the last element of r_i, $N_n = N_n + 1$. Update r_i and r_{i+1}. r_i will be reduced an element, while r_{i+1} is added an element.

If $v(r_i) = 0$, and $N_p \geq N_n$, then a negative transformation is done to the first element of r_{i+1}, $N_n = N_n + 1$; if $v(r_i) = 1$, and $N_p < N_n$, then a positive transformation is done to the first element of r_{i+1}, $N_p = N_p + 1$. Update r_i and r_{i+1}. r_{i+1} will be reduced an element, while r_i is added an element.

4) r_i is a unmodifiable run, r_{i+1} is a unmodifiable run

If $i > N - 3$, then the embedding fails. The algorithm ends.

If $v(r_i) = 0$, and $N_p < N_n$, then a positive transformation is done to the only element of r_i, $N_p = N_p + 1$; if $v(r_i) = 1$, and $N_p \geq N_n$, then a negative transformation is done to the only element of r_i, $N_n = N_n + 1$. r_{i-1} is updated to have two more elements including the modified element in r_i and the only element in r_{i+1}, and the run r_k after r_{i+1} is become to the new $(k - 2)$th run r_{k-2}, for example, the original r_{i+2} is the new ith run r_i. $N = N - 2$. Go to step 5.

If $v(r_i) = 0$, and $N_p \geq N_n$, then a negative transformation is done to the only element of r_{i+1}, $N_n = N_n + 1$; if $v(r_i) = 1$, and $N_p < N_n$, then a positive transformation is done to the only element of r_{i+1}, $N_p = N_p+1$. r_i is updated to have more elements including the modified element in r_{i+1} and all elements in r_{i+2}, and the run r_k after r_{i+2} is become to the new $(k - 2)$th run r_{k-2}, for example, the original r_{i+3} is the new $(i + 1)$th run r_{i+1}. $N = N - 2$. Go to step 5.

Step 7: $i = i + 1$. Go to Step 6 until $i = t$; otherwise, go to step 8.

Step 8: Output the stego text C'.

3.2 Extracting Algorithm

The extraction of secret message from the stego texts is the inverse process of the embedding algorithm. The extraction algorithm uses the same way to calculate run-lengths of runs in the stego texts with the embedding algorithm. If the run-length is even, then the secret bit '0' will be extracted; otherwise, the bit '1' will be extracted. But for the bit length of the secret message, it should be embedded into the cover text or be secretly sent to the receiver in a prescribed and covert manner, such that the receiver can know it before extraction. The details are described in the following steps.

Input: Stego text C', synonym dictionary SD, the bit length t, and relative word frequencies of synonyms.

Output: Secret message M.

Step 1: Digitize the synonyms appearing in stego text C', and express them in the following form of runs:

$D = \{r_i | i = 1, \ldots, N, r_i = d(w_k, \ldots, d(w_{k+1(r_i)-1})\}$

Step 2: Let $i = 1$, while $i < N$, do $m_i = E(r_i) = l(r_i) \bmod 2$.

Step 3: Output the secret message:

$M = \{m_1, m_2, \ldots, m_t\}$.

Table 1 The basic information of an example cover text

Cover text	Digitized synonym sequence	Run-length
C	0 0 1 0 0 0 1 1 0 1 0 1 1	2 1 3 2 1 1 1 2

Table 2 The basic information of the cover and generated stego text

Text	Digitized synonym sequence	Run-length
cover	0 0 1 0 0 0 1 1 0 1 0 1 1	2 1 3 2 1 1 1 2
stego	0 0 1 $\underline{\mathbf{1}}$ 0 0 $\underline{\mathbf{0}}$ 1 0 $\underline{\mathbf{0}}$ $\underline{\mathbf{1}}$ 1 1	2 2 3 1 2 3

3.3 An Example of the Proposed Method

For reinforcing the understanding, an example is given to introduce the proposed algorithm. The basic information of the cover text is listed in Table 1.

The synonyms appearing in C are digitized into a binary sequence "0010001101011", thus the corresponding run sequence is $D = \{r_1, r_2, \ldots, r_7, r_8\} = \{00, 1, 000, 11, 0, 1, 0, 11\}$. Initialize $N_p = N_n = 0$. Suppose the secret message is M = "00110". Then the message will be embedded as following:

1) $E(r_1) = l(r_1) \mod 2 = 2 \mod 2 = 0$, $E(r_1) = m_1$, no modification is required.

2) $E(r_2) = l(r_2) \mod 2 = 1 \mod 2 = 1$, $E(r_2) \neq m_2$. r_2 is an unmodifiable run, while r_3 is a modifiable run. Flip the first element of r_3 by a positive synonym transformation since the value of r_3 is '0'. As a result, $N_p = N_p + 1 = 1$, r_2 is updated from '1' to "11", and r_3 is updated from "000" to "00".

3) $E(r_3) = l(r_3) \mod 2 = 2 \mod 2 = 0$, $E(r_3) \neq m_3$. r_3 and r_4 are both modifiable runs. $v(r_3) = 0$, and $N_p > N_n$, then a negative transformation is made to the first element of r_4. As a result, $N_n = N_n + 1 = 1$, r_3 is updated from "00" to "000", and r_4 is updated from "11" to "1".

4) $E(r_4) = l(r_4) \mod 2 = 1 \mod 2 = 1$, $E(r_4) = m_4$, no modification is required.

5) $E(r_5) = l(r_5) \mod 2 = 1 \mod 2 = 1$, $E(r_5) \neq m_5$, r_5 and r_6 are both unmodifiable runs. And $v(r_5) = 0$, and $N_p \geq N_n$, then a negative transformation is made to the only element of r_6. As a result, $N_n = N_n + 1 = 2$, r_5 is updated from "0" to "000", and r_6 is updated to "11". The total number of runs $N = N - 2 = 6$. Currently, $E(r_5) = 3 \mod 2 = 1 \neq m_5$. One should make synonyms transformation to embed m_5 into the new r_5. Since the new r_5 and r_6 are both modifiable runs, $v(r_5) = 0$, and $N_p < N_n$, then we choose to make a positive transformation is made to the last element of the new r_5. As a result, $N_p = N_p + 1 = 2$, r_5 is updated from "000" to "00", and r_6 is updated to "111".

The last run of cover texts is always not utilized to carry secret message. The results of embedding algorithm are displayed in Table 2. It shows that the runs have been changed to be $D' = \{r'_1, r'_2, r'_3, r'_4, r'_5, r'_6\} = \{00, 11, 000, 1, 00, 111\}$ after embedding. Thus, the corresponding run-length sequences are $\{2, 2, 3, 1, 2, 3\}$. Calculate $m_1 = l(r_1) \mod 2 = 0$, $m_2 = 2 \mod 2 = 0$, $m_3 = 3 \mod 2 = 1$, $m_4 = 1 \mod 2 = 1$, $m_5 = 2 \mod 2 = 0$. Finally, the embedded secret message is M = "00110".

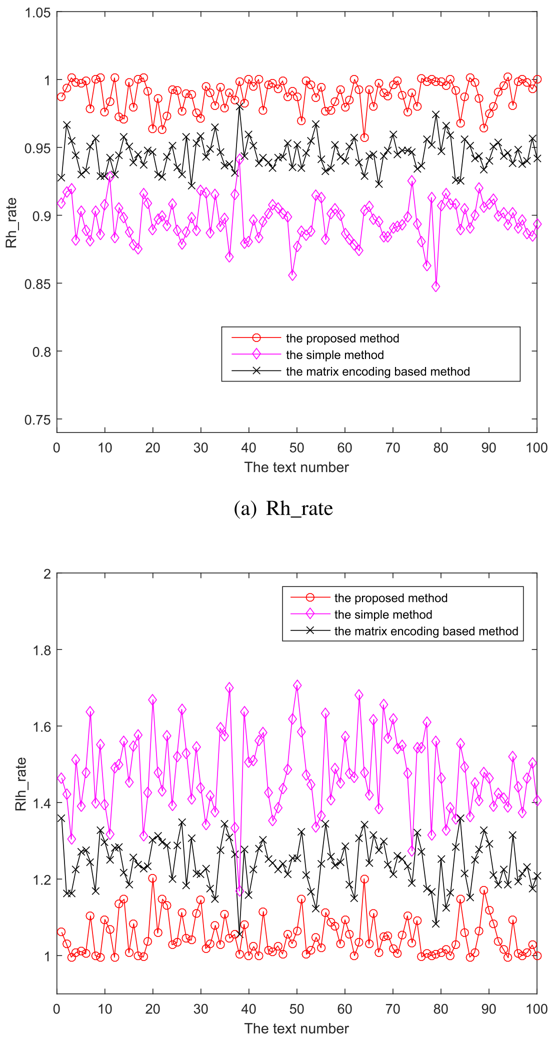
Compared the stego text with the cover one, the different digitized synonym values are marked in underlined and bold font, which have made synonym transformations. It can be found that the number of RHF synonyms and that of RLF ones in the stego text are kept the same as those in the corresponding cover text by self-adaptively making positive or negative synonym transformation. If there have been done more positive transformations, a negative one will have a high priority at the next transformation, and vice versa. Therefore, it is more difficult to discover the very existence of secret message in our stego texts.

4. Experimental Results and Analysis

In this section, the performance of the proposed steganographic method is evaluated by comparing with two existing methods, i.e., the simple synonym substitution based steganographic method, the matrix encoding based linguistic steganographic method [11]. For the reliability of the comparing test, one secret message of each cover text is randomly generated for embedding. The stego texts generated from the same cover text by different steganographic methods were embedded into the same secret message. The simple method directly treats the digital values of synonyms as the secret message. If the digital value does not equal to the secret message bit, the synonym is replaced by its synonymous word assigned the same digital value as the secret message bit. The matrix encoding based method applies the parity check matrix of linear block codes to determine the positions of synonym substitution operations. In this experiment, 1737 cover texts (average 1738 synonyms and 91040 words per cover text) were randomly downloaded from the Internet. The synonym dictionary built by the steganography system in [5] was employed. The synonyms in texts were digitized into binary digitals with the help of frequency lists provided by the electronic book "Word Frequencies in Written and Spoken English: based on the British National Corpus".

4.1 The Distribution of RHF and RLF Synonyms

We randomly choose 100 cover texts and the corresponding stego texts from the three steganographic methods. The numbers of RHF synonyms and those of RLF ones were counted in both cover and stego texts, then the rate of RHF synonyms to all synonyms (shortly as Rh) and the rate of RLF synonyms to RHF synonyms (shortly as Rlh) were calculated in each text. To clearly depict the difference between a stego text and the corresponding cover text, we further calculate two new rates, i.e., *Rh_rate*, *Rlh_rate*. *Rh_rate* is the rate of Rh of the stego text to that of the cover text, and *Rlh_rate* is the rate of Rlh of the stego text to that of the cover text. The *Rh_rates* and *Rlh_rates* of the chosen the stego texts generated by the three steganographic methods are shown in Fig. 2.



(a) Rh_rate

(b) Rlf_rate

Fig. 2 Comparisons of Rh_rates and Rlf_rates of the three methods

As seen from Fig. 2, the statistical characteristics of the stego text generated by the proposed method are closest to those of the corresponding cover text. Thus the proposed method is more capable of preserving word frequencies than other methods. Matrix encoding based method takes the second place, while the simple method performs worst. Note that the smaller difference the stego and cover text has, the less possibility they are distinguished. Generally, it is most difficult to distinguish stego texts generated by the proposed method from the cover ones.

From Fig. 2, it can also be found that the numbers of RHF and RLF synonyms in our stego texts are still slightly different from those of the corresponding cover texts. These differences are caused by the fact that the number of RHF synonyms is generally larger than that of RLF synonyms in cover texts. The RLF synonyms are easier to create unmodifiable runs than RHF ones. If both adjacent runs are modifiable or unmodifiable, the positive and negative transformations are both available. In these two cases, our method can self-adaptively select the appropriate transformation. However, if one of the two adjacent runs is unmodifiable, because the number of RLF synonym is usually less than that of RHF synonym in a cover text, it should be more possible that the value of the unmodifiable run is ‘1’ which is the digital value of the RLF synonym and the value of the modifiable run is ‘0’ which is the digital value of the RHF synonym. According to the modification rules in Case 1 and 2, one should do a transformation on the modifiable run. Thus, it is possible that more positive transformations will be made than negative ones, leading to more RHF synonyms are replaced by RLF ones. Namely, the number of RHF synonyms decreases, while that of RLF synonyms increases. Therefore, the rates of RHF synonyms to all synonyms in our stego texts are often slightly lower than that of cover texts shown in Fig. 2(a), while the rates of RLF synonyms to RHF ones are slightly higher than that of cover texts.

4.2 Capability of Anti-Steganalysis

The anti-steganalysis capability of different stego texts were evaluated using the steganalysis system proposed in [20]. This steganalysis system extracts detection features from synonym frequency to detect the stego texts generated by synonym substitution based steganography.

Assume that a synonym vector $\{s_0, s_1, \ldots, s_{n-1}\}$ is ordered in descending order of the frequencies of the inside synonyms, and $< i, n >$ is the attribute pair of synonym s_i. The detection feature is defined as follows:

$$p(j,n) - p(h,n) = \frac{f(j,n) - f(h,n)}{\sum_{i=0}^{n-1} f(i,n)} \tag{9}$$

where $f(i,n)$ is the number of total occurrences of synonyms whose attribute pairs are $< i, n >$ in the text.

The steganalysis system in [20] only chooses six features. Here, the first and second features, i.e., $p(0,2) - p(1,2)$, $p(0,3) - p(1,3)$ of 100 texts in each category are chosen. In order to clearly depict the difference between a stego text and the corresponding cover text, the rates of the first and second feature of the stego text to those of the cover text, denoted as $feature1\_rate$ and $feature2\_rate$, are illustrated in Fig. 3. It can be found that the features in stego texts of the proposed method can well approximate those in cover texts, while the features in stego texts generated by the two other algorithms are much lower than those in cover texts. This experiment demonstrates that the proposed method can preserve better statistical characteristics than the other two methods.

Furtherly, the SVM classifier [20] is used to predict all the stego texts. The detection results are listed in Table 3.

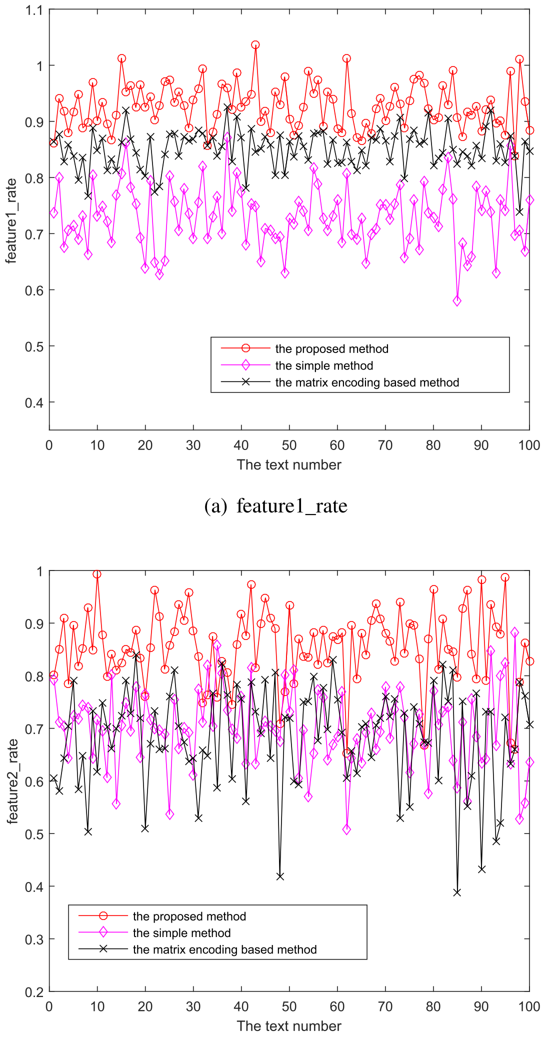
The *recall rate* (rr) is used to measure the performance of the steganalysis system. Recall rate is referred to sensitivity of the steganalysis system, which is defined as:

$$rr = \frac{tp}{tp + fn} \tag{10}$$

where tp denotes the true positive, i.e. the number of stego

Table 3 Detection results for different stego texts by steganalysis in [20]

Stego text type	Total	Detection		*Recall rate*
		stego text	cover text	
Stego text generated by simple method	1737	1311	426	75.41%
Stego text generated by matrix encoding based method	1737	504	1233	29.02%
Stego text generated by the proposed method	1737	169	1568	9.73%



(a) feature1_rate

(b) feature2_rate

Fig. 3 Comparisons of feature rates of the three steganographic methods

texts which are correctly identified as stego ones, fn denotes the false negative, i.e. the number of stego texts which are incorrectly identified as cover ones. The results of the three methods are shown in Table 3.

From Table 3, it can be seen that the recall rate of the proposed method is lowest among those of all the three methods. Namely, the stego texts generated by the proposed method bears the smallest possibility of being detected. This denotes that the proposed method performs better than other two methods on anti-steganalysis.

5. Conclusion

This paper puts forward a novel synonym substitution based steganography using synonym run-length encoding method, which could effectively improve the capability of anti-steganalysis of existing steganographic methods. In the proposed method, the synonyms are digitized into digital ‘0’ or ‘1’ according to their relative frequencies, and represented in forms of runs. To flip the digitized value of a boundary element by a synonym substitution can make the run-lengths of the two modifiable adjacent runs +1 or -1. In other words, the parity of the length of a run can be changed by flipping either its last element or the next adjacent run’s first element. The synonym substitutions on two boundary elements of the two adjacent runs are corresponding to a positive and a negative synonym transformation. Thus, the message bits can be embedded by self-adaptively making positive and negative synonym transformations, so that the distributions of relative high and low frequency synonyms are preserved to ensure secret message security. The experimental results demonstrated that the proposed steganography could achieve a reasonable capability of anti-steganalysis.

There are several interesting works that deserve further studies. Despite the proposed steganography performs well in the aspect of anti-steganalysis, it cannot achieve high embedding capacity and embedding efficiency. The proposed method only embedded at most one bit into a run, which always contains several synonyms. When both two adjacent runs are unmodifiable, according to the modification rule in Case 4, one synonym transformation must be made without being embedded any secret message bit. In the experiments, the proposed method embedded only average 0.3 bits into one synonym, while in theory the simple and matrix encoding based methods can embed 1 and $k/(2^k - 1)$ bits into one synonym respectively, where k is a predefined integer. In addition, we only evaluated the security of linguistic steganography in terms of the statistical undetectability rather than the imperceptibility in terms of natural language. To improve the imperceptibility requires profound experience and knowledge of natural language processing. In the future, more efforts should be devoted to develop more secure, imperceptibility and efficient linguistic steganography.

Acknowledgments

This work has been performed in the Project “Research on Theory and Approach of Secure Text Steganography” supported by National Natural Science Foundation of China (No. 61202439), and partly supported by National Natural

Science Foundation of China(No. 61302159).

References

[1] J.T. Brassil, S. Low, and N.F. Maxemchuk, “Copyright protection for the electronic distribution of text documents,” Proc. IEEE, vol.87, no.7, pp.1181–1196, 1999.

[2] T.-Y. Liu and W.-H. Tsai, “A new steganographic method for data hiding in microsoft word documents by a change tracking technique,” IEEE Trans. Information Forensics and Security, vol.2, no.1, pp.24–30, 2007.

[3] M.J. Atallah, C.J. McDonough, V. Raskin, and S. Nirenburg, “Natural language processing for information assurance and security: an overview and implementations,” Proc. 2000 workshop on New security paradigms, pp.51–65, ACM, 2001.

[4] Y. Liu, X. Sun, Y. Liu, and C.T. Li, “Mimic-ppt: Mimicking-based steganography for microsoft power point document,” Inform. Technol. J, vol.7, pp.654–660, 2008.

[5] K. Winstein, “Lexical steganography through adaptive modulation of the word choice hash,” Unpublished. http://web.mit.edu/keithw/tlex/, 1998.

[6] I.A. Bolshakov, “A method of linguistic steganography based on collocationally-verified synonymy,” Information Hiding, pp.180–191, Springer, 2004.

[7] H.Z. Muhammad, S.M.S.A.A. Rahman, and A. Shakil, “Synonym based malay linguistic text steganography,” 2009 Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA ’09), pp.423–427, IEEE, 2009.

[8] C.Y. Chang and S. Clark, “Practical linguistic steganography using contextual synonym substitution and a novel vertex coding method,” Computational Linguistics, vol.40, no.2, pp.403–448, 2014.

[9] U. Topkara, M. Topkara, and M.J. Atallah, “The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions,” Proc. 8th workshop on Multimedia and security, pp.164–174, ACM, 2006.

[10] L. Yuling, S. Xingming, G. Can, and W. Hong, “An efficient linguistic steganography for chinese text,” Multimedia and Expo, 2007 IEEE International Conference on, pp.2094–2097, IEEE, 2007.

[11] Y. Xiao, L. Feng, and X. Lingyun, “Synonym substitution-based steganographic algorithm with matrix coding,” J. Chinese Computer Systems, vol.36, no.6, pp.1296–1300, 2015.

[12] M.J. Atallah, V. Raskin, M. Crogan, C. Hempelmann, F. Kerschbaum, D. Mohamed, and S. Naik, “Natural language watermarking: Design, analysis, and a proof-of-concept implementation,” Information Hiding, pp.185–200, Springer, 2001.

[13] H.M. Meral, B. Sankur, A.S. Özsoy, T. Güngör, and E. Sevinç, “Natural language watermarking via morphosyntactic alterations,” Computer Speech & Language, vol.23, no.1, pp.107–125, 2009.

[14] C. Grothoff, K. Grothoff, L. Alkhutova, R. Stutsman, and M. Atallah, “Translation-based steganography,” Information Hiding, pp.219–233, Springer, 2005.

[15] R. Stutsman, C. Grothoff, M. Atallah, and K. Grothoff, “Lost in just the translation,” Proc. 2006 ACM symposium on Applied computing, pp.338–345, ACM, 2006.

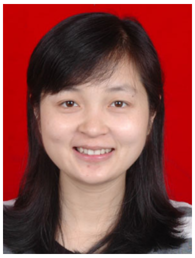
[16] Z. Xia, X. Wang, X. Sun, and B. Wang, “Steganalysis of least significant bit matching using multi-order differences,” Security and Communication Networks, vol.7, no.8, pp.1283–1291, 2014.

[17] Z. Xia, X. Wang, X. Sun, Q. Liu, and N. Xiong, “Steganalysis of lsb matching using differences between nonadjacent pixels,” Multimedia Tools and Applications, vol.75, no.4, pp.1947–1962, 2016.

[18] Z. Chen, L. Huang, H. Miao, W. Yang, and P. Meng, “Steganalysis against substitution-based linguistic steganography based on context clusters,” Computers & Electrical Engineering, vol.37, no.6, pp.1071–1081, 2011.

[19] Z. Chen, L. Huang, and W. Yang, “Detection of substitution-based linguistic steganography by relative frequency analysis,” Digital investigation, vol.8, no.1, pp.68–77, 2011.

[20] L. Xiang, X. Sun, G. Luo, and B. Xia, “Linguistic steganalysis using the features derived from synonym frequency,” Multimedia tools and applications, vol.71, no.3, pp.1893–1911, 2014.



Lingyun Xiang received her BE in computer science and technology, in 2005, and the PhD in computer application, in 2011, Hunan University, Hunan, China. Currently, she is a Lecturer at School of Computer and Communication Engineering, Changsha University of Science and Technology. Her research interests include information security, steganography, steganalysis, machine learning, and pattern recognition.

Xinhui Wang is pursuing his M.E. in communication and information system from Changsha University of Science and Technology. He received his BE in electronic and information engineering from Changsha University, China, in 2011. His research interests include information security and steganography.

Chunfang Yang received the B.S., M.S., and Ph.D. degrees from the Zhengzhou Information Science and Technology Institute in 2005, 2008, and 2012, respectively. Currently, he is a lecturer with Zhengzhou Science and Technology Institute. His current research interests include image steganography and steganalysis technique.

Peng Liu received his B.E. degree in Automation from Xiangtan University, China, in 2006, and M.E. degree in College of Information Science and Engineering, Hunan University, China, in 2011. Since 2011, he has been a Ph.D. candidate in College of Information Science and Engineering, Hunan University, China. During Ph.D. candidate, he has been a visiting student in Electrical and Computer Engineering, University of California Santa Barbara, U.S. His research interests include low power testing, low cost test, test generation, memory test and memristor test.