Generative Text Steganography Based on LSTM Network and Attention Mechanism with Keywords

Huixian Kang[†], *Hanzhou Wu*[†,‡,*] *and Xinpeng Zhang*[†,‡,*]
[†] ***School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China***
[‡] ***Shanghai Institute for Advanced Communication and Data Science, Shanghai 200444, China***
[*] ***Email: h.wu.phd@ieee.org, xzhang@shu.edu.cn***

Abstract

The widespread use of text over online social networks makes it quite suitable for steganography. Conventional text steganography usually transmits the secret data by either slightly modifying the given text or hiding the secret data through synonym replacement. The rapid development of deep neural networks (DNNs) has led automatically generating the steganographic text to become an important and promising topic. This has motivated us to propose a novel generative text steganographic method based on long short-term memory (LSTM) network in this paper. We apply attention mechanism with keywords to the LSTM network to generate the steganographic text. Experiments show that, compared to the related work, the steganographic text generated by the proposed method is of higher semantic quality and more capable of resisting against steganalysis, which has shown the superiority.

Introduction

As a means to secret communication, steganography allows us to hide a secret message into an innocent object typically called *cover* without arousing suspicion from the channel monitor. The steganographic model can be briefly reviewed as follows. A data hider uses a key to drive a well-designed embedding procedure to hide a message into the cover without significantly distorting the cover. The resulting object called *stego* will be sent to a receiver via an insecure channel such as Internet. If the channel attacker captures the stego and obtains evidence of steganography, he can take action to interrupt the communication and even track the data sender or receiver, meaning that, the steganographic communication is failed. Otherwise, once the receiver receives the stego, he uses the key to fully extract the embedded data.

A straightforward idea to categorize steganographic methods is based on the type of used cover. Digital image is the most popular among all kinds of covers since images are widely distributed over the Internet and easy to be altered for steganography. Audio, video and text are also of increasing attention to the researchers. Recently, there are some approaches [1], [2], reported in the literature that use social behaviors to convey a secret payload. Though many general steganographic algorithms can be applied to different covers, there should be unique treatment associated with each type of cover source since a particular cover always has its own statistical, perceptual and structural characteristics.

Compared with other carriers, the advantage of the text carrier for steganography is its extensiveness and high liquidity in daily social interactions. When the transmission process is disturbed by noise, text is relatively easier to be corrected in terms of semantic analysis. However, the higher degree of information coding of texts makes it not easy to embed extra information into texts, leading text steganography to be a quite challenging topic.

Conventional text steganographic methods could be roughly divided into format based and content based. Format based methods such as [3], [4], [5], [6], are usually visually difficult to detect. However, as they are sensitive to the cover format, it may lead to poor anti-interference ability. Content based methods [7], [8], [9], often hide the secret data by synonym substitution, misspelling, or other techniques. The steganographic text obtained by this type of method is not limited by the format and therefore has relatively stronger anti-interference ability. A drawback is that, it may expose distinguishable difference in statistical characteristics compared with the ordinary text, and steganalysis tools, e.g., [10], may detect steganographic text easily. In addition, the capacity of steganography by cover modification is usually low.

With the development of natural language processing (NLP) technology in recent years, researchers have started to study automatically generating steganographic text to convey a secret payload. Such kind of steganographic method belongs to the category that steganography without intuitive modification of the cover. It works by generating the steganographic text based on the to-be-embedded data during the process of text generation. Compared with previous works, this type of steganographic method usually has higher embedding capacity and stronger anti-interference ability, and is therefore becoming a promising topic. There have been some methods proposed in past years, e.g., [11], [12], [13], [14], [15], [16]. Recently, Yang *et al.* [17] present a method generating the steganographic text based on recurrent neural networks (RNNs). In the work, they use a large-scale text database and LSTM to construct a language model. In order to generate a sentence, the conditional probability distribution of each word is encoded by a binary tree or Huffman tree, to realize secret information hiding. However, their model ignores the semantic relevance between sentences, which may make the generated steganographic text easily arouse suspicion, which has motivated us to present a new text steganographic method to achieve better trade-off between text-semantic quality, embedding capacity and efficiency.

We propose to use long short-term memory (LSTM) network combined with attention mechanism based on a large-scale ordinary text database to construct a language model. Moreover, keywords are taken into account for the attention mechanism. During generating the steganographic text, the present word to be generated is determined according to the conditional probability distribution (of words) calculated by LSTM network and the secret value to be embedded. A receiver is able to retrieve the secret data without error from the steganographic text by using the identical language model to the sender. Experiments have shown that, in terms of context-semantic relevance, the proposed work significantly outperforms the related work. To well present our work, we organize this paper as follows. We first introduce preliminary concepts. We then detail the proposed generative text steganography, followed by experiments for performance evaluation. Finally, we conclude this paper and provide discussion.

Preliminaries

In this section, we present brief introduction about generative text steganography, LSTM network, and attention mechanism.

Generative Text Steganography

Compared with conventional text steganography, generative text steganography does not need a pre-specified cover (text) object, instead, it directly produces the steganographic text according to the secret data and a trained generative model. A text can be represented by a sequence of ordered words. We can denote the ordered word sequence by $S = \{\omega_i\}_{i=1}^{n}$, where ω_i represents the i-th word of the sentence. For most automatic text generation approaches, the word at the i-th position of the sentence can be associated with the conditional probability distribution based on all previous words (if any), leading the entire text to be expressed as the product of n conditional probabilities, which is written as:

$$\begin{aligned} \mathrm{Prob}(S) &= \mathrm{Prob}(\omega_1, \omega_2, ..., \omega_n) \\ &= \prod_{i=1}^{n} \mathrm{Prob}(\omega_i | \omega_{i-1}, \omega_{i-2}, ..., \omega_1). \end{aligned} \tag{1}$$

It is required that $\mathrm{Prob}(S)$ should be kept as high as possible, and $\mathrm{Prob}(S) = 1$ ideally. A straightforward idea to automatically generate S is that, given ω_1, we produce a set of candidate words for ω_2, denoted by $C(\omega_2)$. All elements in $C(\omega_2)$ are mapped to an integer in such a way that we can always set the value of ω_2 as an element in $C(\omega_2)$ that its mapped value matches the secret data and maximizes $\mathrm{Prob}(\omega_2|\omega_1)$. $\mathrm{Prob}(\omega_2|\omega_1)$ can be determined based on a local training dataset. The similar process will be applied to the subsequent symbols in S one by one. Mathematically, an element in $C(\omega_i)$ is selected as the value of ω_i to match the present secret data and maximize $\mathrm{Prob}(\omega_i|\omega_{i-1})$, which, actually, assumes that $\mathrm{Prob}(\omega_i|\omega_{i-1}, \omega_{i-2}, ..., \omega_1) = \mathrm{Prob}(\omega_i|\omega_{i-1})$.

Regardless of the resulting semantic quality of applying the simple method mentioned above, it requires us to design such kind of *word-wise* generative steganographic procedure that two core issues should be addressed for generating the present word ω_i: 1) Collect a list of candidate words prior to word generation, where each candidate is associated with a probability indicating the degree of "fitness" of selecting it as the present word to be generated. 2) Map each candidate to an integer such that one can always find a word from the list that its mapped value matches the secret data.

Accordingly, we need to use an effective statistical generative model for candidate-word generation, where the words can be sorted by their associated probabilities. The sum of the probabilities equals 1, and the "*most appropriate*" word has the largest probability value. On the other hand, we have to build the mapping relationship between the secret data and a word. In this paper, for the former, we use the LSTM network combined with attention mechanism and keywords to generate the candidate words. And, for the latter, we use the methods in [17] for simplicity.

LSTM Network

LSTM [18] has been widely used in sequence problems due to its ability to capture long-term dependence. A LSTM network unit can be described in the following set of formulas:

$$\begin{cases} I_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \\ F_t = \sigma(W_f[h_{t-1}, x_t] + b_f), \\ C_t = F_t C_{t-1} + I_t \tanh(W_C[h_{t-1}, x_t] + b_c), \\ O_t = \sigma(W_o[h_{t-1}, o_t] + b_o), \\ h_t = O_t \tanh(C_t), \\ y_t = f_y(W_y h_t + b_y), \end{cases} \tag{2}$$

where x_t, h_t and y_t indicate the input vector, hidden vector, and output vector at the t-th step. W_* and b_* are to-be-learned matrices and biases. I_t corresponds to the input gate, which controls whether the new information needs to be stored or not in the memory cell. F_t indicates the forget gate, controlling whether the stored information needs to be thrown away. C_t means the memory cell, which is controlled by the input gate and the forget gate. The output gate O_t controls whether the current hidden state needs to be influenced by the memory cell. We need to be aware of that when we calculate the output at time step t, the input we used contains all the information from the beginning. Using formula to more intuitively indicate the output y_t at time step t:

$$y_t = f(x_t | x_{t-1}, x_{t-2}, ..., x_1), \tag{3}$$

we can find that the formula is extremely similar to Eq. (1). This is why LSTM can be used for generative text steganography, which, actually, is a sequence task. For more details, we refer to [18].

Attention Mechanism

Attention mechanism [19], [20], [21], has been a quite popular and useful technique in deep learning in recent years. It can be used for performance improvement in many fields such as natural language processing, computer vision, speech processing, and so on. For example, Bahdanau *et al.* [20] first applied attention mechanism to neural machine translation (NMT), by using a typical Encoder-to-Decoder framework. Suppose that, the source language is represented as $S = \{\omega_1, \omega_2, ..., \omega_N\}$, where ω_k is the k-th word. In the traditional framework without attention mechanism, each ω_k is encoded into a hidden vector h_k using the Encoder, and all hidden vectors $(h_1, h_2, ..., h_N)$ are linearly transformed to get a context vector c_i, which is further decoded into the i-th target word y_i using the Decoder together with previously generated target words. All produced target words form the target language $T = \{y_1, y_2, ..., y_M\}$. During the target-word generation, by combining the attention mechanism to the Encoder-to-Decoder framework, better translation results can be obtained since the attention mechanism allows the model to automatically search relevant context from the source sentence out for better prediction.

The aforementioned attention mechanism takes into account all hidden vectors $(h_1, h_2, ..., h_N)$ of the Encoder during determining c_i. In detail, c_i is determined according to the weighted average of $(h_1, h_2, ..., h_N)$ and the attention weights $\{a_{i,j} | 1 \leq i \leq M, 1 \leq j \leq N\}$. $a_{i,j}$ is computed with the current hidden vector of the target word, denoted by z_i, and hidden vectors $(h_1, h_2, ..., h_N)$.

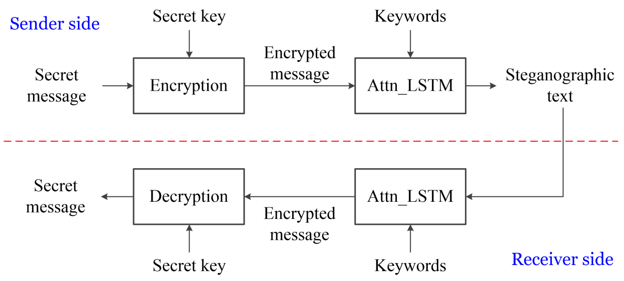

Figure 1. *Sketch for the proposed steganographic method.*

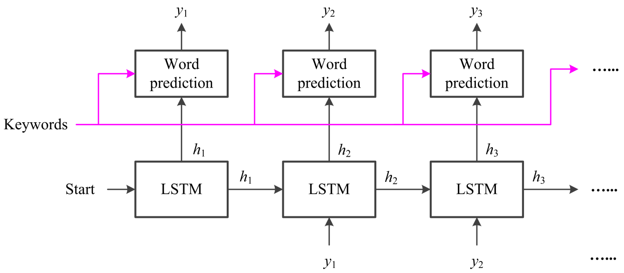

Figure 2. *The structural information of the functional block "Attn_LSTM".*

The entire process can be briefly described by the formula below:

$$c_i = \sum_{j=1}^{N} a_{i,j} h_j, \tag{4}$$

where

$$a_{i,j} = \text{align}(z_{i-1}, h_j) = \frac{\exp(\text{score}(z_{i-1}, h_j))}{\sum_k \text{score}(z_{i-1}, h_k))} \tag{5}$$

and score(z_{i-1}, h_j) is a function that scores how well the inputs around the position j and the output at the position i match [20], [21]. As the attention mechanism can automatically notice words in the source language during the process of decoding the context vector to target language, we apply it to automatically generate the steganographic text expecting to produce long text sentences having high sematic quality, which would be quite helpful for concealing the presence of hidden data, and having high capacity.

Proposed Method

In this section, we introduce the proposed method in detail. Figure 1 shows the sketch for the proposed method. It can be seen that, the secret message is first encrypted as a random bitstream. By feeding the random bitstream and keywords to the LSTM with attention mechanism (denoted by the block "Attn_LSTM"), the steganographic text can be generated, which will be sent to the receiver via an insecure channel. With the steganographic text, a receiver is able to reconstruct the random bitstream with the identical neural network model and keywords, which allows the receiver to further reconstruct the secret message without error by decryption. In this way, the secret communication is realized.

LSTM Network with Attention Mechanism

A core work for the proposed method is to combine the attention mechanism into the LSTM network to generate long steganographic text having high sematic quality. Figure 2 shows the structural information of the functional block "Attn_LSTM". The block has the form of a chain of repeating modules of LSTM unit. For each LSTM unit, it receives the hidden vector of the previous L-STM unit and the word vector of previously generated word. The new hidden vector will be used for word prediction together with keywords, and also fed to the next LSTM unit for subsequent processing. Obviously, the attention mechanism involves the hidden vector and keywords. We take the t-th LSTM unit for example. Figure 3 shows the structural information of the word prediction module. It can be described as follows. First of all, we determine the attention vector attn$_t$ as:

$$\text{attn}_t = W_{\text{attn}}[k_1; k_2; \ldots; k_{n_k}] + b_{\text{attn}}, \tag{6}$$

where k_j is the j-th keyword vector, n_k is the total number of keywords, and W_{attn} and b_{attn} are parameters to be optimized during training. $[k_1; k_2; \ldots; k_{n_k}]$ means that, we construct a new vector by concatenating all the n_k vectors. Then, we determine a new vector depending on attn$_t$ and h_t as:

$$h'_t = [\text{attn}_t; h_t]. \tag{7}$$

By using linear transform, we further obtain

$$\text{output}_t = W_{\text{dec}} h'_t + b_{\text{dec}}, \tag{8}$$

where W_{dec} and b_{dec} are to be optimized, finally resulting in

$$\text{Prob}(y_t | y_{<t}) = \text{softmax}(\text{output}_t). \tag{9}$$

Data Embedding and Extraction

Assuming that, we are now processing the t-th LSTM unit and expecting to embed an integer $d \geq 0$. It can be seen from Eq. (9) that, we can collect a set of candidate words. The candidates can be sorted in a decreasing order according to their probabilities. The $(d+1)$-th word will be selected as the generated word. Thereafter, by processing the subsequent LSTM units with the similar way, we can produce the *stego* text. Obviously, there are many ways to generate an integer d to be embedded, e.g., directly converting a bitstream with a fixed length to a decimal number. The way to generate an integer is not the main interest of this paper. For simplicity, we will use the methods introduced in [17], i.e., fixed-length coding (FLC) and variable-length coding (VLC), for experiments in this paper. However, it is pointed that, one can design other effective methods to build the mapping relationship between the secret bitstream and the candidate words.

The trained neural network block (i.e., "Attn_LSTM") shown in Figure 1 and keywords should be pre-shared between the sender and the receiver so that the secret data can be fully reconstructed. The reconstruction process can be performed in an inverse way to the sender side, which is straightforward.

Remark 1. Unlike previous works, we take into account the keywords for text generation. The first generated word by using the proposed method could also carry the secret data.

Remark 2. It is possible that, two candidate words have the identical prediction probability. In order to ensure that the sender and the receiver produce the identical sorted word list, when two words have the identical prediction probability, the *lexicographically* smaller word can be associated with a smaller index.

Remark 3. Multiple stego texts will be separately generated if the secret data needs to be split to multiple pieces. Every stego text uses a different set of keywords.

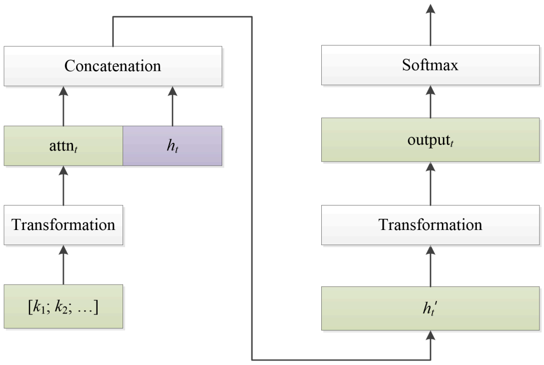

Figure 3. *The structural information of the word prediction module.*

Table 1. Details of the training datasets.

Dataset	ZhiHu	ESSAY
Total number of distinct tokens	97616	238905
Total number of distinct paragraphs	56621	494944
Average length of a paragraph	77.81	64.28

Table 2. Running time (seconds) of generating steganographic text for different models at different data embedding rates.

Data Embedding Rate	LSTM Model	
(bits per word, bpw)	FLC	VLC
1	3.766±0.097	3.767±0.106
2	3.819±0.089	5.094±0.134
3	3.872±0.071	6.978±0.198
4	3.891±0.082	9.058±0.146
5	4.064±0.514	11.959±0.329
Data Embedding Rate	Attn_LSTM Model	
(bits per word, bpw)	FLC	VLC
1	3.687±0.104	3.832±0.324
2	3.693±0.079	5.353±0.262
3	3.747±0.177	6.783±0.224
4	3.760±0.090	8.769±0.519
5	3.770±0.079	11.648±0.300

Experimental Results and Analysis

We present experimental results and analysis in this section.

Setup

Since we expect the proposed method to automatically generate the steganographic text that is sufficiently similar to ordinary text, we need a large-scale dataset of natural text to train our language model so as to mimic/capture the statistical characteristics of natural text. Moreover, the proposed method exploits keywords for generating text with high-level quality, requiring us to prepare datasets providing keyword summary. To this end, according to our best effort, we select two most suitable datasets for model training, i.e., ZhiHu and ESSAY [22]. The statistical information of the two datasets are given in Table 1. For ZhiHu, we use 50,000 paragraphs for training, and the rest for evaluation. For ESSAY, we use 490,000 for training, and the rest for evaluation.

For simulation, we use Word2Vec in [23] to obtain word vectors, whose dimensions are all 100. However, it is always free for us to apply other effective algorithms to obtain the word vectors. The recurrent hidden layer of one-layer LSTM model contains 256 hidden units, namely, the dimension of hidden vector h_i shown in Figure 2 is 256. The attention layer contains 100 hidden units, namely, the dimension of "attn$_t$" shown in Figure 3 is 100. The model is trained with the Adam optimizer [24] and dropout mechanism [25]. The learning rate is initialized as 0.001. The batch size is set as 16 and dropout rate is 0.5. In our experiments, 5 keywords are used for attention mechanism. Each keyword has a dimension of 100 (by Word2Vec).

Data Embedding Efficiency

We use PyTorch 1.0.1[1] as the simulation platform. And, TITAN RTX 24GB GPU x1 and CUDA 10.0 are used for accelerated computing. We use ZhiHu dataset for experiments. We have spent around five days to obtain a trained model. We use the running time of generating the steganographic text with a trained model to represent the data embedding efficiency. And, we use the average number of embedded bits per word (bpw) to represent the data embedding rate. For each data embedding rate, we generate 500 steganographic texts (each containing 100 words) and record the running time. We use the FLC method and VLC method introduced by [17] to build the replacement relationship between the secret bitstream and the word candidates. We evaluated the data embedding efficiency for the LSTM model and Attn_LSTM model. Table 2 shows the results. It is noted that, the only difference between LSTM and Attn_LSTM is that, LSTM does not use attention mechanism. It can be seen from Table 2 that, for both models, the time to generating the steganographic text, averagely, is almost the same, meaning that, the Attn_LSTM model will not significantly increase the computational cost once the training phase has been finished. Besides, as the data embedding rate increases, the VLC method takes more time than the FLC method. The reason is that, the VLC method takes more time to build a Huffman tree than to build a binary tree, which is used in the FLC method. In this comparison, we can see that the average time the FLC method (used by Attn_LSTM model) takes to generate a 100-word steganographic text is around 3.7 seconds, which shows good performance in data embedding efficiency.

Statistical Quality Analysis and Steganalysis

We use *perplexity*, which is a standard metric for sentence quality testing, to measure how well the automatically-generated steganographic text. The perplexity is defined as follows [17]:

$$P_L = 2^{-\frac{1}{n}\cdot\mathrm{logProb}(\omega_1,\omega_2,\ldots,\omega_n)}, \tag{10}$$

where $\{\omega_i\}_{i=1}^n$ represents the generated steganographic text. In most cases, a smaller P_L indicates better quality of the generated steganographic text. We apply the FLC method and VLC method to our Attn_LSTM model and compare it with the corresponding LSTM model. We train each model on two datasets mentioned

[1] https://pytorch.org/

Table 3. The mean and standard deviation of P_L for different models at different data embedding rates.

Used	Data Embedding Rate	LSTM Model		Attn_LSTM Model	
Dataset	(bpw)	FLC	VLC	FLC	VLC
ZhiHu	1	4.852±1.125	7.245±1.984	3.948±1.349	5.234±1.888
	2	8.763±3.814	8.162±1.560	7.146±2.333	6.543±1.942
	3	11.091±6.326	9.659±2.963	10.799±5.585	7.803±1.576
	4	18.872±7.146	11.464±2.559	16.664±7.979	8.164±2.791
	5	24.358±9.557	12.791±2.950	23.270±8.893	8.077±2.894
ESSAY	1	5.157±0.939	8.168±0.355	4.756±0.209	6.209±0.295
	2	10.482±1.581	9.137±0.479	8.181±1.081	7.415±0.689
	3	14.564±5.152	12.694±1.214	12.061±2.098	8.877±1.414
	4	33.181±7.489	14.406±1.680	29.954±7.232	10.607±1.580
	5	55.231±12.569	17.991±2.125	54.769±12.755	12.989±2.725

Table 4. EMD for different models at different embedding rates.

Used	Rate	LSTM Model		Attn_LSTM Model	
Dataset	(bpw)	FLC	VLC	FLC	VLC
ZhiHu	1	10.513	9.655	8.888	8.864
	2	10.937	10.257	9.287	9.351
	3	11.359	10.887	9.953	9.881
	4	11.568	11.428	10.903	10.241
	5	12.014	11.783	11.977	10.326
ESSAY	1	11.822	11.252	8.629	8.549
	2	11.911	11.351	9.031	8.677
	3	11.912	11.361	9.381	9.256
	4	12.290	11.549	10.429	9.900
	5	12.594	11.644	10.604	10.275

Table 5. Steganalysis results for different models at different embedding rates.

Used	Rate	LSTM Model		Attn_LSTM Model	
Dataset	(bpw)	FLC	VLC	FLC	VLC
ZhiHu	1	0.553	0.443	0.460	0.457
	2	0.357	0.413	0.393	0.473
	3	0.650	0.590	0.580	0.537
	4	0.633	0.617	0.623	0.600
	5	0.747	0.663	0.693	0.660
ESSAY	1	0.531	0.477	0.520	0.503
	2	0.587	0.520	0.567	0.517
	3	0.663	0.557	0.577	0.547
	4	0.677	0.601	0.630	0.556
	5	0.703	0.660	0.672	0.601

above, and for each embedding rate, we generate 500 steganographic texts for evaluation. The mean and standard deviation of the perplexity are determined, which are shown in Table 3.

According to Table 3, we can draw out the following conclusions. First of all, for each dataset, as the embedding rate increases, the perplexity will gradually increase. Secondly, with the increase of the embedding rate, the quality of the text obtained by the FLC method drops sharply, the quality of the text obtained by the VLC method slowly decreases. Thirdly, the perplexity of the text obtained by the Attn_LSTM model is lower than the text generated by the LSTM model, which has demonstrated the superiority of the Attn_LSTM model.

We further compare the overall statistical distribution similarity between the steganographic text generated by each method and the real text (a total of 500 pairs). We first use Doc2Vec [26] to map the steganographic text to a high-dimensional space. Then, we use Earth Mover's Distance (EMD) [27] to measure the similarity of the steganographic text and the real text. The results are shown in Table 4. From Table 4, we can get the similar conclusion, that is, the EMD between the steganographic text generated by the Attn_LSTM model and the real text is smaller, indicating that Attn_LSTM can generate the more real-like text than LSTM.

In addition, we use the steganalysis algorithm [28] based on Bayesian Estimation and Correlation Coefficient methodologies to evaluate the proposed work based on the 500 pairs of texts. We determine various statistical indicators for the generated steganographic texts, and then get the average detection accuracy of different statistical indicators. Table 5 records the average detection accuracy for different datasets at different data embedding rates. The closer the average detection accuracy is to 0.5, the superior the steganographic method is. Here, the detection accuracy [17] is defined as the ratio between the total number of correctly classified samples and the total number of all test samples. From Table 5, we can conclude that, the steganographic text generated by the Attn_LSTM model is more semantically relevant and therefore more resistant to steganalysis. In applications, it is difficult for an attacker to access the trained model, enhancing the security.

Steganographic Examples

The above experimental results are based on statistical analysis. For fair evaluation, we further show some examples of the

Attn_LSTM Model with ZhiHu

我是一名大一学生，现在想要去读研，想去做一名律师，想问问大家的想法，是如何安排自己安排作息和效率？我是一个比较认真的学生，上课的时候却总是记不住。记不住的时候就是外文，我就想要好好利用自己，效率很低，所以我也想问下，我该如何做？谢谢。

Google Translation: *I am a freshman student, and now I want to go to graduate school, want to be a lawyer, I want to ask everyone's thoughts, how do you arrange your own schedule and efficiency? I'm a serious student, but I can't remember it in class. When I can't remember it, it's a foreign language. I want to make good use of myself, and the efficiency is very low, so I also want to ask, what should I do? Thank you.*

Youdao Translation: *I am a freshman, now I want to go to graduate school, want to be a lawyer, want to ask your opinion, is how to arrange their own schedule and efficiency? I am a more serious student, but always remember in class. I want to make good use of myself when I can't remember the foreign language. The efficiency is very low. So I also want to ask, what should I do? thank you.*

(a) Embedding rate: 1 bit per word

我想减肥啊!我是高三的学生，因为我想去减肥塑形，但是现在又开始节食，体重110。现在想要减肥的话能吃些什么。不然我不想吃什么。我会不吃。不知道怎么做才会有效果?

Google Translation: *I want to lose weight! I am a junior high school student, because I want to lose weight and shape, but now I start dieting again, weight 110. What to eat now if you want to lose weight. Otherwise I don't want to eat anything. I will not eat. I don't know what to do to have effect?*

Youdao Translation: *I want to lose weight! I am a senior three student, because I want to lose weight and get in shape, but now I am on a diet and weigh 110. What can you eat if you want to lose weight now? Otherwise I don't feel like eating anything. I won't eat it. Don't know what to do to get results?*

(b) Embedding rate: 2 bits per word

最近想练出来,自己动感骑车，想请教各位。该怎么练出来力量?有什么方法?如果能动感练一天话又不是很担心,什么牌子也适合女生，求指导一下!

Google Translation: *I want to practice it recently and ride my own bike. I want to ask you. How can I develop my strength? Is there any way? If you can practice for a day and you are not very worried, what brand is also suitable for girls, ask for guidance!*

Youdao Translation: *Want to practice recently come out, oneself spin bicycle, want to consult everybody. How should practice come out strength? What are the methods? If can move feeling to practice a day word again not very worry, what brand also suits a girl, beg to instruct!*

(c) Embedding rate: 3 bits per word

Attn_LSTM Model with ESSAY

秋天，它是一位魔术师，它把金黄色装扮的五彩缤纷，田野的四周荡漾的脸庞。金灿灿的鲜花开得甜甜的,格外壮观,更加迷人！果园里，一望无际，开满了红色的笑脸，格外热闹，人人都喜爱的季节，它既凉爽又金灿灿。

Google Translation: *In autumn, it is a magician. It is colorful with golden colors and rippling faces around the fields. Jin Canchan's flowers are sweet, extraordinarily spectacular and even more charming! The orchard is endless, full of red smiley faces, especially lively, a season everyone loves, it is both cool and golden.*

Youdao Translation: *Autumn, it is a magician, it put the golden color dress colorful, the fields around the ripples of the face. The golden flowers are sweet, especially spectacular, more charming! Orchard, as far as the eye can see, full of red smiling faces, especially lively, everyone loves the season, it is cool and golden.*

(d) Embedding rate: 1 bit per word

太阳伯伯苏醒过来了，冬姑娘悄悄地走出人间，春光在寒冷人间苏醒，打扮如同一棵仙女，跳起了人间，舞出了一簇簇春光般的笑声，春光变得格外耀眼，让人心温暖。

Google Translation: *The uncle of the sun woke up, and the winter girl quietly walked out of the world. Spring light woke up in the cold world, dressed as the same fairy, jumping out of the world, dancing a bunch of spring-like laughter. warm.*

Youdao Translation: *The sun uncle wake up, winter girl quietly out of the world, spring wake up in the cold world, dressed as a fairy, jump up the world, dance a bunch of spring laughter, spring become particularly dazzling, let the heart warm.*

(e) Embedding rate: 2 bits per word

全身光秃秃树木长着水灵灵的青草。微风一笑，就像一幅画展开。阳台上的小草也都沐浴着大地。池塘里穿着碧绿的一朵朵桃花，娇嫩的嫩芽像细一样可爱，它长得郁郁葱葱。

Google Translation: *The bare trees all over were covered with green grass. A breeze smiled, like a painting unfolding. The grass on the balcony was also bathing the ground. The pond was dressed in green peach blossoms. The delicate tender buds were as lovely as thin, and they grew lush.*

Youdao Translation: *The trees were bare with fresh grass. A smile on the breeze, like a picture spread out. The grass on the balcony is also bathing the earth. The pond is dressed in green peach blossoms, tender buds as lovely as thin, it grows lush.*

(f) Embedding rate: 3 bits per word

Figure 4. *Steganographic examples for the proposed method.*

steganographic texts, which have been shown in Figure 4. Since the used datasets were based on Chinese, we used Google Translator[2] and Youdao Translator[3] to translate Chinese to English for non-Chinese readers. We did not translate Chinese to English by ourselves since we thought it may be not fair due to the subjective bias. Notice that, the used machine translators may not well translate Chinese to English. It is seen that, overall, the proposed work can generate the steganographic text with clear semantic information, showing the applicability. We point that, the semantic quality of steganographic text is dependent of the training dataset. If the dataset is not large-scale, the semantic quality will be poor.

Conclusion and Discussion

In this paper, we propose a novel text steganography method based on LSTM network and attention mechanism with keywords, which can auto-generate high-quality and diverse steganographic text. We have conducted experiments to evaluate the proposed method, and experimental results show that it outperforms related work, which has demonstrated the superiority. With the fast development of online social networking services, moving steganography to social networking environment would be quite suitable. Obviously, text steganography is promising for this scenario since texts are widely used over the social networks, making such covert

[2] `https://translate.google.com/`
[3] `http://fanyi.youdao.com/`

communication be easily concealed by the huge number of normal social activities. Moreover, automatically generating steganographic text further allows us to achieve intelligent steganographic communication over social networks by developing steganographic bots (agents). We believe that, text steganography would become more and more important. In the future, we will combine automatic text steganography into social network environments.

Acknowledgement

It was supported by National Natural Science Foundation of China under grant numbers 61902235, U1636206, U1936214, and 61525203. It was also supported by "Chen Guang" project co-funded by the Shanghai Municipal Education Commission and Shanghai Education Development Foundation.

References

[1] H. Wu, W. Wang, J. Dong, and H. Wang. New graph-theoretic approach to social steganography. In: *Proc. IS&T Electronic Imaging, Media Watermarking, Security and Forensics*, pp. 539-1-539-7(7), San Francisco, CA, Jan. 2019.

[2] X. Zhang. Behavior steganography in social network. In: *Proc. Advances in Intelligent Information Hiding and Multimedia Signal Processing*. pp. 21-23, Kaohsiung, Taiwan, Nov. 2016.

[3] J. T. Brassil, S, Low, N. F. Maxemchuk, and L. O'Gorman. Electronic marking and identification techniques to discourage document copying. *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1495-1504, Oct. 1995.

[4] K. Rabah. Steganography - the art of hiding data. *Information Technology Journal*, vol. 3, no. 3, pp. 245-269, Mar. 2004.

[5] S. Low, N. Maxemchuk, and A. Lapone. Document identification for copyright protection using centroid detection. *IEEE Transactions on Communications*, vol. 46, no. 3, pp. 372-383, Mar. 1998.

[6] W. Bender, D. Gruhl, N. Morimoto, and A. Lu. Techniques for data hiding. *IBM Systems Journal*, vol. 35, no. 3.4, pp. 313-336, 1996.

[7] H. Hu, X. Zuo, W. Zhang, and N. Yu. Adaptive text steganography by exploring statistical and linguistical distortion. In: *Proc. IEEE Second International Conference on Data Science in Cyberspace*, pp. 145-150, Jun. 2017.

[8] Y. Liu, X. Sun, C. Gan, and H. Wang. An efficient linguistic steganography for Chinese text. In: *Proc. IEEE International Conference on Multimedia and Expo*, pp. 2094-2097, Aug. 2007.

[9] M. Topkara, U. Topkara, and M. J. Atallah. Information hiding through errors: a confusing approach. In: *Proc. SPIE, Security, Steganography, and Watermarking of Multimedia Contents IX*, vol. 6505, pp. 321-332, Feb. 2007.

[10] L. Xiang, X. Sun, G. Luo, and B. Ma. Linguistic steganalysis using the features derived from synonym frequency. *Multimedia Tools and Applications*, vol. 71, no. 3, pp. 1893-1911, Aug. 2014.

[11] W. Dai, Y. Yu, Y. Dai, and B. Deng. Text steganography system using markov chain source model and DES algorithm. *Journal of Software*, vol. 5, no. 7, pp. 785-792, Jul. 2010.

[12] H. Moraldo. An approach for text steganography based on markov chains. *arXiv Preprint arXiv:1409.0915*, 15 pages, Sept. 2014.

[13] T. Fang, M. Jaggi, and K. Argyraki. Generating steganographic text with LSTMs. *arXiv Preprint arXiv:1705.10742*, 7 pages, May 2017.

[14] Y. Tong, Y. Liu, J. Wang, and G. Xin. Text steganography on RNN-generated lyrics. *Mathematical Biosciences and Engineering*, vol. 16, no. 5, pp. 5451-5463, Jun. 2019.

[15] F. Dai, and Z. Cai. Towards near-imperceptible steganographic text. *arXiv Preprint arXiv:1907.06679*, 7 pages, Jul. 2019.

[16] Z. Ziegler, Y. Deng, and A. Rush. Neural linguistic steganography. *arXiv Preprint arXiv:1909.01496*, 9 pages, Sept. 2019.

[17] Z. Yang, X. Guo, Z. Chen, Y. Huang, and Y. Zhang. RNN-Stega: linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1280-1295, Sept. 2018.

[18] S. Hochreiter, and J. Schmidhuber. Long short-term memory. *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, Nov. 1997.

[19] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. Recurrent models of visual attention. In: *Proc. Advances in Neural Information Processing Systems (NIPS)*, pp. 2204-2212, Dec. 2014.

[20] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv Preprint arXiv:1409.0473*, 15 pages, Sept. 2014.

[21] M. Luong, H. Pham, and C. Manning. Effective approaches to attention-based neural machine translation. *arXiv Preprint arXiv:1508.04025*, 11 pages, Aug. 2015.

[22] X. Feng, M. Liu, J. Liu, B. Qin, Y. Sun, and T. Liu. Topic-to-essay generation with neural networks. In: *Proc. International Joint Conference on Artificial Intelligence*, pp. 4078-4084, Jul. 2018.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv Preprint arXiv:1301.3781v3*, 12 pages, Sept. 2013.

[24] D. P. Kingma, and J. Ba. Adam: A method for stochastic optimization. *arXiv Preprint arXiv:1412.6980v9*, 15 pages, Jan. 2017.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929-1958, Jan. 2014.

[26] Q. V. Le, and T. Mikolov. Distributed representations of sentences and documents. *arXiv Preprint arXiv:1405.4053v2*, 9 pages, May 2014.

[27] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99-121, Nov. 2000.

[28] S. Samanta, S. Dutta, and G. Sanyal. A real time text steganalysis by using statistical method. In: *Proc. IEEE International Conference on Engineering and Technology*, pp. 264-268, Mar. 2016.