J. Vis. Commun. Image R. 61 (2019) 157–169

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

A novel natural language steganographic framework based on image description neural network ☆

Juan Wen [a], Xuejing Zhou [a], Mengdi Li [a], Ping Zhong [b], Yiming Xue [a,*]

[a] College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, China
[b] College of Science, China Agricultural University, Beijing 100083, China

ARTICLE INFO

Article history:
Received 2 January 2018
Revised 19 November 2018
Accepted 17 March 2019
Available online 25 March 2019

Keywords:
Natural language steganography
Image description
Neural network
Embedding capacity
BLEU
Perplexity

ABSTRACT

It is a challenge to conduct natural language steganography on Online interactive platforms such as photo-sharing websites since the stego texts should be consistent with the content of the images. In this paper, a novel natural language steganographic framework based on an end-to-end generative network is proposed. A Convolution Neural Network (CNN) combined with Long Short-Term Memory (LSTM) is trained to generate stego descriptions. Word by Word Hiding (WWH) and Sentence by Sentence Hiding (SSH) schemes are proposed to achieve various embedding capacity under the premise of sharing model between the sender and the receiver. Furthermore, a blind extraction scheme called Hash Hiding (HH) is proposed in case that the model is unavailable for data extraction. Comparative experiments show the superiority of the proposed framework. It is verified that the proposed framework is an effective carrier-less steganographic framework with competitive embedding capacity, considerable text quality, and good reversibility.

1. Introduction

Steganography is to hide secret messages in appropriate carriers without drawing suspicions during covert communication. In steganographic schemes, carriers, such as texts [1–4], images [5–8], videos [9–11], and audio [12–14], are used to embed the secret messages. As a widely used application, the Internet provides plenty of different carriers for steganography. Therefore, more and more people begin to use the Internet as a channel for information hiding. Currently there are many interactive platforms rising on the Internet where the users can post images, videos, and the related descriptive texts, such as social networks, photo-sharing websites, online shopping websites, video-streaming websites which make a feature of danmu ("danmu" is a Chinese word literally means "bullet curtain" and refers to a commentary sharing system in which viewers can post comments directly on top of the video). Benefit from the rich source of carriers and convenient interactiveness, these online interactive platforms can be used as ideal information hiding channels.

On Internet interactive platforms, the secret information can be hidden in images, videos or texts. As the most widely used information carrier, text has advantageous characteristics of various forms, simple coding, convenient storage and fast transmission. Text steganography originates from ancient times, but due to the small redundancy of text, it develops slowly in recent years. In this research, we focus our efforts on conducting a new text steganography scheme on Internet interactive platforms.

Pioneer studies on text steganography are mostly based on format-changing methods, such as word shifting [15], character spacing [16] and font format changing [17]. These approaches are relatively simple and weak in robustness, as they cannot resist the format adjustment or re-typesetting attack. Besides these format-based methods, Huang et al. present a carrier-less steganography framework based on searching a webpage containing the secret message [18]. Sun et al. design a coverless information hiding method based on the Chinese mathematical expression [19]. Along with the improved natural language processing technologies, natural language steganography has become a hot topic in text steganography field, which is no longer based on text format but on syntactic or semantic understanding.

Natural language steganography relies mainly on the content of text in lexical, syntactic or semantic levels. These steganographic schemes can be divided into two categories according to their embedding process, namely, text-modification and text-generation. Text-modification schemes, such as those based on synonym replacement [20], syntactic transformation [21] and

☆ This paper has been recommended for acceptance by Zicheng Liu.

* Corresponding author at: 17East Tsinghua Road, Beijing P.C.:100083, China.
E-mail address: xueym@cau.edu.cn (Y. Xue).

https://doi.org/10.1016/j.jvcir.2019.03.016
1047-3203/

semantic operation [22], hide secret data by modifying the original cover text. These methods rely on complex syntactic or semantic analysis which is not easy to achieve high accuracy. Moreover, they cannot resist the attack by comparing the stego-text with its original one. On the other hand, text-generation schemes automatically generate stego-texts which have the same structure or statistical distribution as natural language. The most typical text-generation systems are TEXTO [23] and NiceText [24] which generate stego-texts according to word dictionary and grammar rules. In addition, Mimicry [25] makes use of context-free grammar to generate stego-text. However, as no semantic information is used in these schemes, the resulting texts lack semantic coherence. Some efforts are done to utilize semantic information to improve the linguistic quality of the generated text. TBS scheme [26] uses a machine translation system to rewrite the original texts. In this way, data are embedded in the modified sentences without changing the meaning of the cover texts. MarkovTextStego [27] uses a Markov chain to generate stego-text so as to maintain the statistical properties of natural language. Ci-stega [28] explores a specific text-generation steganography based on Ci-poetry generation using Markov chain model.

Compared with the traditional text steganography, the biggest challenge to conduct text steganography on the online interactive platforms is that the content of the stego text should be consistent with the content of the corresponding image or video frame. Inspired by the recent researches on neural image description which can automatically train distributions over image features and the descriptive sentences and use them to generate the descriptive sentences, a text-generation steganographic strategy based on image description framework is proposed. An "encoding - decoding" network is used to transfer images to stego texts under the guidance of secret bits. To be more specific, image features are encoded by a Convolutional Neural Network (CNN) and fed to a Long Short Term Memory (LSTM) [29] decoder along with the word embedding vectors, then some strategies will be applied to generate the stego texts according to the secret bits. Based on this framework, we explore a sort of steganographic schemes with different rules to generate stego descriptive sentences which coincide with the content of images.

The main advantages of the proposed neural image description steganography are as follows. First, our work is a new type of carrier-less steganography. For a given image, the traditional way of steganography is to modify image pixels or DCT coefficients, which will more or less result in some distortions. Our approach makes use of images, but instead of modifying images, it generates the stego descriptions of images. As a matter of fact, it does not make any changes in the images. Second, with the online interactive platform, there is no need for the sender to transmit images or videos to the receiver, the sender only need to tell the receiver the link of the chosen images and leave the stego descriptions in the corresponding comment area. Especially in a video sharing website with danmu, multiple stego comments can be sent at different video frames to hide a large amount of secret information on Internet. Besides, benefit from neural networks, there is no need to do any syntactic or semantic analysis which is hard and time consuming as the traditional steganography does. Finally, different schemes have been given to meet different embedding rates. One bit of information can be embedded in one to three words, which makes the embedding rates relatively higher than the traditional natural language steganographic approaches. Experiments show that our algorithms achieve high text quality, reversibility and anti-steganalysis performance.

The rest of the paper is organized as follows. In Section 2, a basic framework of neural image description is reviewed and the way of text prediction is surveyed. Section 3 elaborates the proposed steganographic schemes under different information sharing conditions. Two search algorithms are used to generate stego texts with different embedded capacity. Experimental results and analyses are given in Section 4. The conclusions are drawn in Section 5.

2. Framework of neural image description

Image description, also known as image caption, is a very challenging work concerning the problem of automatically generating a natural language description of images. It should not only detect objects in images, but also obtain the relationship between objects. There are many related work for image description. The previous methods are basically conducted in two ways. One is to use image detection method to predict the image fragments, such as the objects, scenes, and actions, and then a template is used to generate a sentence using these fragments [30]. Sentences generated in this way always lack diversity and seem to be rigid. The other is sort of like a retrieval task. A joint probability of sentences and images is leaned to retrieve the image captions from a sentence database [31]. This kind of methods lack the ability to generate new descriptions when observing new images. Benefit by neural networks and large datasets, the state-of-the-art techniques are based on the structure with a combination of a convolutional neural network to extract a rich representation of the input image and a recurrent neural network (RNN) to generate the descriptive sentence. Google first applied these structure to construct a neural image description model, called Neural Image Caption (NIC) [32], which won a competition in image caption task in 2015. NIC consists of a deep CNN for image feature extraction and a LSTM network for text generation. LSTM is a special kind of RNN which is capable of learning long-term dependencies between words [33]. NIC learns a probability distribution over the space of visual representation and a language model from sentence-image pairs. Compared with the other categories of image caption, NIC can generate diverse sentences according to the content of image with relatively higher text qualities. The framework of NIC and the corresponding LSTM memory block are shown in Fig. 1.

S_t denotes the input word at time t expressed as a one-hot vector with dimension equal to the size of the dictionary, $W_e S_t$ is an operation to map the word S_t to the same space as the image I using word embedding W_e. p_t denotes the probability distribution output from LSTM at time t. N is the number of words in the current training sentence. During training process, the encoder CNN is used to convert an image into a fixed-length vector which is taken as an input of the decoder LSTM. Each word is sent to the corresponding LSTM unit with one-hot representation. All LSTMs share the same parameters and the output of LSTM at time $t-1$ is fed to LSTM at time t. $\ln p_t$ is a log-likelihood function. The related mathematical formula is as follows:

$$x_{-1} = CNN(I) \tag{1}$$

$$x_t = W_e S_t \quad t \in \{0 \ldots N-1\} \tag{2}$$

$$i_t = \sigma(W_{ix} x_t + W_{im} m_{t-1}) \tag{3}$$

$$f_t = \sigma(W_{fx} x_t + W_{fm} m_{t-1}) \tag{4}$$

$$o_t = \sigma(W_{ox} x_t + W_{om} m_{t-1}) \tag{5}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot h(W_{cx} x_t + W_{cm} m_{t-1}) \tag{6}$$

$$p_{t+1} = Softmax(o_t \odot c_t) \tag{7}$$

$$L(I,S) = -\sum_{t=1}^{N} \ln p_t(S_t) \tag{8}$$

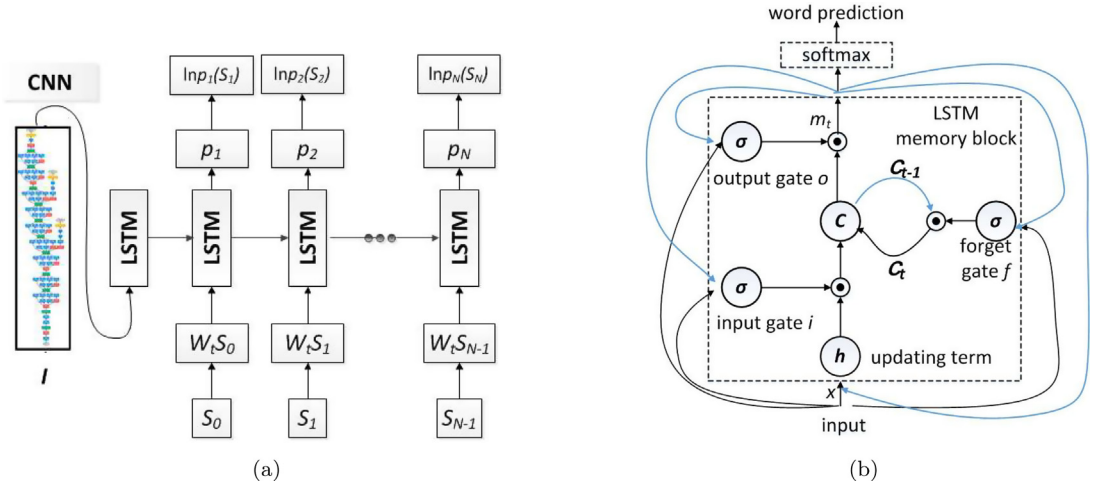

Fig. 1. (a) Framework of NIC (b) LSTM memory block.

where $\odot$ denotes inner product operation. $W_{ix}, W_{im}, W_{fx}, W_{fm}, W_{ox}, W_{om}, W_{cx}$ and W_{cm} are the training parameters. σ denotes sigmoid function and h represents tangent function. x_t denotes the input of LSTM unit at time t, and x_{-1} denotes the initial input of LSTM. c_t, f_t, i_t, and o_t denote the value of cell, forget gate, input gate, and output gate, respectively. m_t is what is used to feed to a Softmax. After the training process, all the training parameters are confirmed. Then we can make an inference for an unknown image by computing p_t for each time t through the trained network using formula (1)–(7). The Softmax function is used to produce the probability distribution p_t over all words. The loss function $L(I, S)$ minimizes the sum of the negative log-likelihood of the correct word S_t at each time t by BPTT (Back Propagation Through Time) and update the various W parameters iteratively.

Currently a variety of models based on NIC with a typical CNN-RNN framework have been proposed. Xu et al. add attention mechanism to the network and automatically select the corresponding features from the input sequence to improve the performance [34]. Wu et al. try to add high-level semantic features to the model using multi-label classification [35]. Karpathy et al. propose the visual semantic alignment model and the multi-modal RNN model which associate images with sentence segments to generate image description in text form for different regions [36]. The main advantage of these approaches is that the entire system can be trained from end to end, and all the parameters can be learned automatically from the training data.

3. Proposed steganographic framework based on image description network

Based on CNN-LSTM networks, a novel neural image description steganographic framework is proposed to generate stego image descriptions. The secret data is embedded during the inference period. There are basically two algorithms to search for the optimal sentence for a test image during the inference phase. One is by sampling, and the other one is called Beam Search. Sampling is to choose the word with the largest probability from the output of LSTM and feed it to the next LSTM. Repeat this process until the end-of-sentence token is generated or the maximum length is reached. Beam Search uses a relevant parameter, namely, *beamsize*, *bs* for short. *bs* is an adjustable parameter. For example, when $bs = 8$, at each time t the model will select 8-best sentence fragments with highest probabilities and feed them to the next LSTM at time $t+1$ and still remain 8-best outputs. When the search process is over, it will output 8-best sentences.

A simple way to apply this image description framework to steganography is to modify the sampling process. Let the sender and receiver come to an agreement on how to choose the corresponding words during sampling process at each moment. For example, at each time t, if the secret bit to be embedded is 1, then choose the word with the largest probability; if the secret bit is 0, then choose the word with the second largest probability. In this way the embedding capacity can achieve one bit per word, we call it Word-by-Word Hiding, WWH for short. See in Fig. 2. BOS is a begin-of-sentence token. To extract the hidden bits for WWH, the receiver has to use the same CNN-LSTM model to repeat the generation process and extract the hidden bits according to the order of probabilities of the received words. WWH is a simple steganographic scheme with high embedding capacity, however, it will lead to a decline in text quality because sampling is not a good solution to find a reasonable description sentence and it will become even worse after modification. In this paper, WWH is served as a baseline.

3.1. An improved steganographic scheme SSH based on Beam Search

To optimize the stego text generation process and get better text quality compared with WWH, we design a scheme based on Beam Search process, called Sentence-by-Sentence Hiding, SSH for short. "Sentence-by-Sentence" means the secret data are embedded at the sentence level. Different from WWH which chooses the corresponding words according to the current secret bit at every time t, SSH embeds data at the last time of generating process when all candidate sentences are generated. By means Beam Search, we can obtain several candidate sentences for each image after generation, which brings redundancy for embedding.

Two issues need to be considered when designing SSH algorithm. The first one is how to match the generated sentences to the binary secret bits. The second one is how to inform the receiver about the length of embed bits.

The main idea of SSH is to encode the generated candidate sentence of a test image at the end of Beam Search and choose the one whose code coincides with the current bit string. SSH does not modify the Beam Search process like WWH. In our scheme, we choose $bs = 2^n$ (n = 1,2,3,...), because in this way, we can apply fixed-length coding to assign each candidate with n binary bits. After embedding, We sort the 2^n candidates according to their probabilities and then give each of them a binary code with fixed

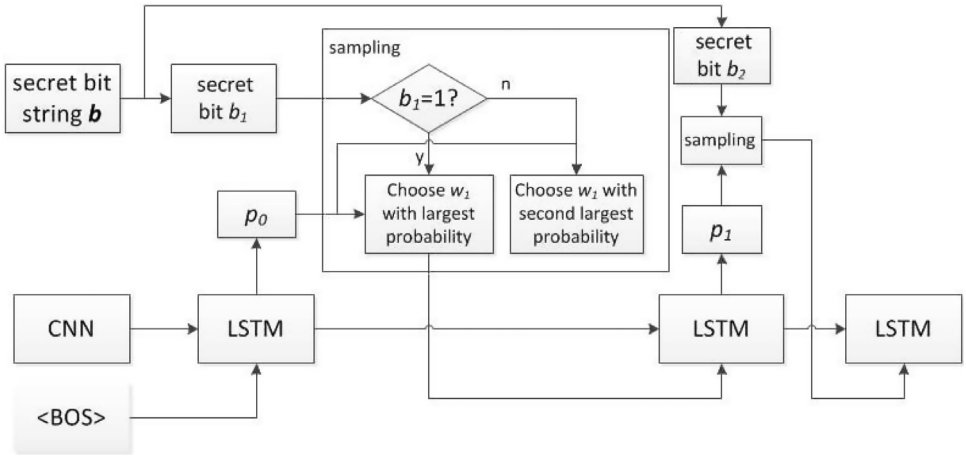

Fig. 2. WWH framework based on sampling.

length n according to its order. In this way, we can build the correlationship between the binary bits and the sentences. The sentence that matches current binary bits will be chosen as the final stego image description. (See in Algorithm 1) For example, let $bs = 2^2$, after the generation process, we will get 4-best candidate sentences which can be encoded as "00", "01", "10", and "11" according to their sentence probability, respectively. Choose the sentence whose binary code coincides with the current two secret bits as the stego description output of the image and send it to the receiver. Assume that the current secret bits are "10", then the third sentence will be chosen as the final output.

For the second problem, it is necessary to let the receiver know the exact length of the embedded data to ensure the correctness of data extraction. To solve this issue, additional bits are added at the beginning of the secret bit string to represent the length of hidden bits before embedding. Moreover, we need an ordered image set that contains enough images to make sure that all of the secret bits can be embedded.

We can easily notice in SSH, because $bs = 2^n$, SSH can hide n bits in a stego sentence because of the encoding rule for the candidate sentences. It means the embedding capacity is closely related to the setting of bs for SSH scheme. If there are x stego sentences, then the maximum number of hidden bits is $x \times n$.

The date embedding process of SSH is described in Algorithm 1 as follows:

Algorithm 1. Date embedding process of Sentence-by-Sentence Hiding (SSH)

Input:
CNN feature matrix M of image set;
$bs = 2^n$ (n is chosen from 1,2,3...);
binary secret bit stream B
maximum length of the generated sentence L;
Output:
stego image description set S;
1 Add 16-bit header data at the beginning of B to represent the length of secret data. Divide B into groups, each group has n bits, add zero at the end if necessary to make sure it can be divided by n;
2 for each group in B **do**
3 extract the current image CNN feature vector f from matrix M;
4 at time $t = 0$, input f and the START symbol to LSTM and choose the 2^n-best words from the probability vector p_1 as the candidate words at time $t = 1$;
5 for the end-of-sentence token is not reached and length of the fragment < L **do**
6 feed 2^n-best fragments to LSTM respectively to obtain 2^n-best words for each fragment;
7 connect each word to the corresponding sentence fragment to form a new sentence fragment;
8 keep the best 2^n sentence fragments;
9 end
10 the best 2^n candidate sentences are sorted and encoded according to their probabilities, select the corresponding sentence according to the binary string of current group and push it into S;
11 end

Algorithm 2. Data extraction process of SSH

Input:
ordered image set I (or URLs of images);
pre-trained CNN-LSTM model shared by sender;
stego image description set S;
$bs = 2^n$ (n is chosen from 1,2,3...);
Output:
binary secret bit stream B;}
1 for $i = 1;\ i < |I|; i++$ **do**
2 select the ith image from I and the corresponding ith image description d_i from S;
3 calculate image CNN vector f using CNN network;
4 do lines 4–9 shown in Algorithm 1 to obtain the best 2^n candidate sentences;
5 the best 2^n candidate sentences are sorted and encoded according to their probabilities, select the one that is the same as d_i, add its code to the binary secret bit stream B;
6 end
7 extract the first 16 binary bits, convert them to decimal number to get the length of the secret bits, extract the corresponding length of secret bits to form the final binary secret bit stream B;

Algorithm 2 describes the data extraction process. The receiver extracts messages by the same CNN-LSTM network and bs to simulate the generation process. The first 16 bits indicate the length of bit string.

Compared to WWH, SSH takes advantage of the better optimized Beam Search algorithm, so the quality of the generated texts will be better. It is worthwhile to notice that although the original test images are required, there is no need to send these images because no modification is made on the carrier image. Instead, the sender only needs to tell the receiver the address or link of the images on the Internet that used to do the embedding, such as the images from the image-sharing website and video frames from video-streaming website.

3.2. HH scheme for model-free extraction with Hash Function

SSH can generate stego image description according to the content of images without modifying images themselves. However, the extraction process of SSH requires many prior knowledge shared by the sender. The receiver has to follow the embedding process with the same trained CNN-LSTM model used by the sender to get the embedded information. Specifically, SSH extractor requires the original images, the CNN-LSTM network structure, and network parameters to extract the secret data. If the model and parameters change during the embedding process, the receiver should be informed of the new model and parameters. (It is necessary to train several CNN-LSTM models to fit different kinds of images). As a result, there will be some limitations to apply SSH in a practical steganographic system because it will apparently cause extra transmission bandwidth. In this subsection, we further propose a Hash Hiding (HH) scheme that can achieve model-free extraction.

Table 1
Word categories obtained under a certain key ($k = hello$).

Word w	$v(w,k)$	Category	Word w	$v(w,k)$	Category
The	0	C_0	there	1	C_1
A	1	C_1	on	1	C_1
In	0	C_0	two	0	C_0
Doggies	1	C_1	dogs	0	C_0
Black	1	C_1	brown	0	C_0
Are	1	C_1	fast	0	C_0
Small	0	C_0	tiny	0	C_0
Cute	1	C_1	thin	1	C_1
Happily	1	C_1	chase	1	C_1
Jumping	0	C_0	running	1	C_1
Grass	0	C_0	grassland	1	C_1

To achieve this, at the sender set, before embedding, a mapping function is needed to match each word in the word dictionary to a binary bit "0" or "1". The word dictionary is the set of all the words that appear in the training samples. In order to get high security, we apply a Hash Function with a private key to convert each word into a binary digit (0 or 1). The mapping function $v(w,k)$ is designed as follows:

$$v(w,k) = (int(hash(w+k))) \quad mod2 \tag{9}$$

where w represents the word, k is the private key taken the form of string and shared between the sender and the receiver. "+" operator is to concatenate word w and key k to form a new string. $hash(\cdot)$ denotes a Hash Function used for encryption. $int(\cdot)$ is a function to convert a string into a decimal number. Using $v(\cdot)$, all the words can be classified into two categories, denoted as C_0 and C_1.

$$C_0 = \{w_i | v(w_i,k) = 0,\ i = 1 \ldots |V|\} \tag{10}$$

$$C_1 = \{w_i | v(w_i,k) = 1, i = 1 \ldots |V|\} \tag{11}$$

where $|V|$ represents the length of vocabulary. It is worth mentioning that for the HH algorithm, in order to enhance security, multiple keys can be used. For example, the two parties can specify the first word of each sentence as the private key. Table 1 shows a small part of the word categories obtained by v that applies md5 Hash Function with the key "hello". Under a specific key, every word from the training corpus belongs to a certain category (C_0 or C_1) according to the mapping function v.

Embedding process: Fig. 3 gives an example to show the embedding process of HH with $bs = 2$ under the key "hello". The secret bits to be embedded are "0010". At time 0, the image representation extracted by CNN is fed to LSTM to get p_0. According to p_0, we sort all the words based on their conditional probability. For example, the top 4 words are "two", "there", "a", and "the". So far, it is exactly the same as SSH. Next, since the first secret bit is 0, only the words that belong to C_0 can be chosen as the candidates. As a result, the words "two" and "the" are chosen because they are the words with the highest probabilities that belong to C_0. This step is different from SSH, since SSH chooses the best words from all the words, it will choose "two" and "there".

At time 1, "two" and "the" are fed into LSTM respectively, then we repeat the sorting and selecting steps and choose the best 2 words from C_0 for each case (note that the second secret

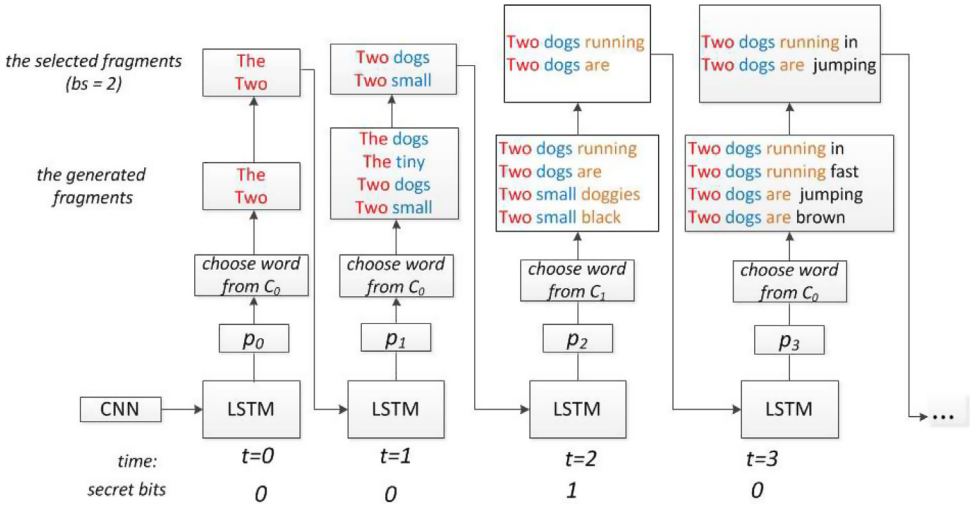

Fig. 3. The embedding process of HH with $bs = 2$.

bit is still 0). The generated words are connected to their previous words to form sentence fragments, thus four sentence fragments are obtained. Choose the best 2 sentence fragments according to their probabilities because we have to keep *bs* inputs at each time.

At time 2, the two selected sentence fragments, "two dogs" and "two small", are fed into LSTM respectively. Since the third secret bit is 1, we choose the best 2 words from C_1 for each sentence fragment. In this way, 4 new longer sentence fragments are obtained and the best 2 words are kept.

Repeating these steps, finally, we can get 2 sentences when the end-of-sentence token is generated. Choose the one with the biggest probability as the final stego sentence. So we get "two dogs are jumping on the grassland".

From the embedding process, we can find that the difference between HH and SSH is that HH needs to choose the word from a certain category that matches the current secret bit at each time. This is actually the way how the bits are embedded in HH.

From the above embedding process, a secret bit is embedded at each time t. In order to balance the embedding capacity and the text quality, besides *bs*, we use another parameter *bpw*, bits per word, to control the embedding frequency. During the generating process of HH, some positions of the sentence can be set as non-stego positions. For these positions, we only need to follow the normal Beam Search of SSH and select *bs*-best words from all the words, regardless of the categories of words. Therefore, $bpw = 1$ means every generated word is used as stego word and $bpw = 1/2$ means every stego word is followed by a non-stego word. That is to say, the secret data should only be embedded into the odd times during generation. In the case of $bpw = 1/p$, $(p \in N+)$, the secret bits are read every p times and embedded in the first word of every p successive words. Under the maximum capacity, every word in the generated sentences should be served as stego word. It is worth noting that each stego word is denoted by one bit using the mapping function $v(w,k)$, which means the maximum *bpw* in HH is 1 bit per word. Therefore, *bpw* should not be set to greater than 1 in HH scheme.

In this paper, *bpw* is set to be 1, $1/2$, and $1/3$, the corresponding algorithms are denoted as HH-1, HH-2, HH-3, respectively. The embedding process of HH-1 are described in Algorithm 3.

Extraction process: The extraction process of HH is totally different from that of SSH. The receiver only needs to know *bpw* and the mapping function $v(w,k)$. Secret bit string can be computed directly from the generated texts using formula (9). *bpw* determines the extraction frequency. For example, assuming v is shared by the sender and the key $k = hello$, when receiving the stego sentence "two dogs are jumping on the grassland", The receiver can use $v(k,w)$ to compute: $v(two, hello) = 0$, $v(dogs, hello) = 0$, $v(are, hello) = 1$, $v(jumping, hello) = 0$, $v(on, hello) = 1$, $v(the, hello) = 0$, $v(grassland, hello) = 1$. If $bpw = 1$, it means every word is a stego word, then the extracted secret bits are "0010101". If $bpw = 1/2$, it means the secret bits are embedded in odd positions. In this case, words in the even position, such as "dogs", "jumping", are non-stego words and chosen from all the word instead of word categories, so they do not hold any secret bit. Because of this, we only extracted secret bits from the words with odd positions, then the extracted secret bits are "0111". If $bpw = 1/3$, only "two", "jumping", and "grassland" are stego words, so we get "001". Since there is no need for the receiver to use the original images, generation model, and word dictionary during the extraction process, we call HH a blind extraction scheme.

Algorithm 3. Date embedding process of Hash Hiding with $bpw = 1$ (HH-1)

Input:
CNN feature matrix M of image set;
$bs = 2^n$ (n is chosen from 1,2,3...);
binary secret bit stream B
maximum length of the generated sentence L;
mapping function $v(k,w)$;
Output:
stego image description set S;}
add $16 - bit$ header data at the beginning of B to represent the length of secret data;
get the category value of each word w for all words in the training corpus according to formula (9);
for each image vector f in matrix M **do**
  at time $t = 0$, obtain the current secret bit m (0/1) from B, input f and the START symbol to LSTM and select the best 2^n words from word category C_m as the candidates;
  for the end-of-sentence token is not reached and length of the fragment $<$ L **do**
    get the corresponding secret bit m from B;
    if $m = none$ **then**
      return;
    end
    feed the 2^n best candidate fragments to LSTM respectively to obtain 2^n best words from word category C_m;
    connect each word to the corresponding sentence fragment to form the new sentence fragments;
    keep the best 2^n sentence fragments;
  end
  select the candidate with the biggest probability as the description of current image and push it into S;
end

Algorithm 4. Data extraction process of HH with $bpw = 1$ (HH-1)

Input:
stego image description ordered set S;
mapping function $v(k,w)$;
$bpw = 1$;
Output:
binary secret bit stream B;}
connect the descriptive sentences successively to form a word string s_w;
extract the first 16 words, calculate their corresponding binary bits by formula (9), convert this binary string into decimal number to get the length of secret message l_s;
for $i = 1;\ i <= l_s; i++$ **do**
  extract the $(i+16)th$ word from s_w, the corresponding binary bit is calculated by formula (9) and pushed into B;
end

4. Experimental results and analyses

In this paper, we choose NIC as the basic image caption model to conduct our experiments since it is simple and easy to be applied. We coduct a series of experiments to test the performance of the proposed natural language steganographic schemes. We use

Flickr8k [37] and COCO [38] datasets which are widely applied in computer vision field and contain 8,000 and 123,000 images, respectively. Most of these images are shown the scenes about humans or animals doing certain activities. Each sample consists of an image and 5 manual annotated sentences which describe the content of the image. The number of training samples, verification samples and test samples in the experiments are shown in Table 2.

To extract the image features, a 16-layer VGGNet is used as a pre-training model. VGGNet is a very effective, classic convolutional neural network that has achieved very good results in major competitions. The architecture of VGGNet applied in our experiments is shown in Fig. 4. After the full connection layer of CNN, each image will transfer to a 4096-dimensional vector. All the vectors of the test images form the CNN feature matrix M, which will be fed into LSTM to generate image description. The LSTM structure used in our experiments is shown in Fig. 1(b). The source code of NIC is available at https://github.com/karpathy/neuraltalk.

4.1. Evaluation metrics

There are two main ways to evaluate the performance of image description system. One is BLEU (Bilingual Evaluation understudy) [39] and the other is PPL (perplexity) [40].

BLEU is a measurement widely used in machine translation to evaluate the quality of texts which have been machine-translated from one natural language to another. BLEU can also be used in image description tasks to evaluate the quality of generate sentences because it can check the similarity of word n-grams between the generated sentence and the reference sentence. The coincidence precision of the generated image description and the reference sentence can be calculated as follows:

$$CP_n(C,S) = \frac{\sum_i \sum_k min(h_k(c_i), max_j h_k(s_{ij}))}{\sum_i \sum_k h_k(c_i)} \tag{12}$$

where C denotes the set of generated sentences, S denotes the set of references manually annotated. c_i is the ith generated sentence to be evaluated, s_{ij} denotes the jth references of c_i. n is a parameter comes from n-gram. When n is set to be 1, 2, and 3 means that the formula evaluates uni-gram, bi-gram, and tri-gram in the sentences. $h_k(c_i)$ denotes the number of occurrences of the kth n-gram in c_i. $h_k(s_{ij})$ denotes the number of occurrences of the kth n-gram in reference sentence s_{ij}. By this metric, a shorter sentence turns out to have a bigger value. To avoid this problem, a punitive factor $b(C,S)$ is added:

$$b(C,S) = \begin{cases} 1 & if\ l_c \geqslant l_s \\ e^{1-l_s/l_c} & if\ l_c \leqslant l_s \end{cases} \tag{13}$$

where l_c denotes the length of the generated sentence, l_s denotes the length of reference sentence. BLEU is a weighted geometric average of $CP_n(C,S)$ among all n-grams:

$$\text{BLEU}_N = b(C,S)\exp(\sum_{n=1}^{N} \omega_n \ln CP_n(C,S)) \tag{14}$$

where $\omega_n = 1/N$ and N is selected as 1, 2, 3, 4 in our experiments to evaluate the generated text quality according to uni-gram, bi-gram tri-gram and 4-gram, respectively.

On the other hand, PPL is a common measure derived from the field of information theory [41] to evaluate language model. Since language model reflects a probability distribution over training texts, PPL provides a method to measure how well the trained model reflects the probability distribution of training corpus. Lower perplexities represent better language models. PPL is calculated as follows:

$$\text{PPL} = 2^{-\frac{1}{m}\sum_{i=1}^{m} \ln p(s_i)} \tag{15}$$

where m denotes the number of generated sentences, s_i is the ith sentence, $p(s_i)$ refers to the tri-gram probability of s_i.

In this paper, we use BLEU to verify the text quality and semantic consistency, and PPL to test the statistical performance of the generated text.

4.2. Text quality results based on BLEU score evaluation

To evaluate the quality of stego texts generated by the proposed steganographic schemes, we carry out the BLEU score evaluation on Flickr8k and COCO image sets. The Beam Search parameter bs is set to be 2^n, $n = (1, 2, \ldots 5)$. NIC is the standard image description model without hiding secret information [32]. For NIC, we test the case of $bs = 1$ to see the result of sampling. We also test the embedding capacity of each scheme under different bs. Embedding capacity is a measurement calculated as the ratio of the embedded bits over the number of bits of the generated texts. WWH is served as the baseline. In Flickr8k experiment, BLEU 1,2,3,4 of WWH scheme achieve 49.5, 28.8, 15.7 and 8.2, In COCO dataset experiment, BLEU 1,2,3,4 of WWH achieve 54.1, 32.8, 18.9 and 10.9, respectively. As WWH can hide one bit of information in each word, the embedding rate of WWH achieves 2.75% bpb (bits per bit) on Flickr8k and 2.68% bpb on COCO dataset. The experimental results on Flickr8k and COCO by SSH and HH are listed in Tables 3 and 4, respectively. The values in boldface are the best results (maximum BLEUs, embbeding bits and capacities) obtained by each model among all *Beamsizes*.

Table 2
The description of dataset.

Dataset	Train	Valid	Test
Flickr8k	6000	1000	1000
COCO	112887	5000	5000

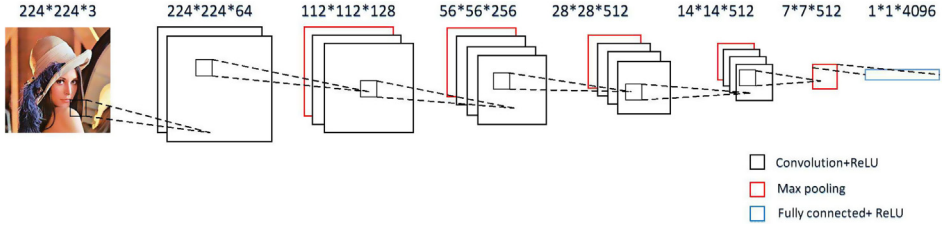

Fig. 4. VGGNet architecture used to extract image representation.

Table 3
BLEU score evaluation on Flickr8k.

	Beamsize (*bs*)	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Embedding bits	embedding capacity(%)
NIC [32]	1	55.7	37.3	24.0	15.7	0	–
	2	57.2	37.3	25.4	16.8	0	–
	4	58.3	39.3	26.0	**17.3**	0	–
	8	58.7	39.4	25.7	16.8	0	–
	16	**59.1**	39.8	26.0	16.9	0	–
	32	**59.1**	**40.0**	**26.3**	17.0	0	–
SSH	2	55.5	37.4	24.4	16.0	1000	0.27
	4	56.1	37.6	**24.8**	**16.6**	2000	0.53
	8	56.4	37.8	24.7	16.1	3000	0.81
	16	**56.6**	**37.9**	24.7	16.1	4000	1.08
	32	56.0	37.2	24.1	15.6	**5000**	**1.34**
HH-1 ($bpw = 1$)	2	46.3	26.8	14.7	8.2	**11118**	2.74
	4	49.5	29.1	16.1	8.8	9771	**2.75**
	8	50.3	30.1	17.5	10.1	9004	2.74
	16	**50.9**	**30.8**	**17.7**	**10.2**	8015	2.70
	32	50.7	30.3	17.0	9.3	6742	2.70
HH-2 ($bpw = \frac{1}{2}$)	2	52.6	33.3	20.1	11.9	**5149**	**1.34**
	4	53.9	33.6	20.3	12.2	4747	**1.34**
	8	55.4	34.5	20.9	12.8	4227	1.33
	16	56.0	**35.9**	**22.7**	**14.3**	3705	1.32
	32	**56.8**	35.7	21.5	12.6	3235	1.32
HH-3 ($bpw = \frac{1}{3}$)	2	54.9	35.4	22.0	13.7	**3632**	0.95
	4	55.3	35.2	22.7	14.3	3357	0.95
	8	54.8	34.6	21.1	12.9	2975	0.95
	16	56.8	36.4	22.5	14.0	2709	**0.96**
	32	**56.9**	**37.0**	**23.3**	**15.0**	2447	0.95

As we can see from Table 3, without data hiding, BLEU scores of NIC reach the highest value among all models under the same condition, while BLEU scores of SSH and HH are slightly decreased compared with NIC. This is obvious because under the same *bs*, NIC model without hiding any data should be more accurate than the other steganography algorithms which choose suboptimal path during generation process to hide data. But it can be seen that the decrease caused by data hiding is not very obvious in many cases. For example, the BLEU-1 score of HH-3 with $bs = 32$ is reduced by about 3.72% compared with non-stego scheme NIC. By increasing *bs*, we can further improve the BLEU performance in most cases. However, the growth rate of BLEU becomes slower when *bs* is set above 16, and BLEU scores even decreased at $bs = 32$ in some cases. This is reasonable. If *bs* is too small, the optimal sentence fragments may not be included among all the candidates, whereas if *bs* is too big, some noise would be introduced in the beam

Table 4
BLEU score evaluation on COCO.

	Beamsize (*bs*)	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Embedding bits	embedding capacity (%)
NIC [32]	1	65.1	46.4	32.1	22.4	0	–
	2	**65.4**	**47.1**	33.6	24.3	0	–
	4	64.9	46.8	**33.9**	**24.9**	0	–
	8	64.2	46.0	33.1	24.3	0	–
	16	63.4	45.3	32.5	23.8	0	–
	32	63.1	44.8	32.0	23.5	0	–
SSH	2	**64.4**	**46.0**	**32.4**	23.1	5000	0.28
	4	63.0	44.9	32.0	**23.2**	10000	0.56
	8	62.5	44.3	31.5	22.7	15000	0.85
	16	61.1	43.0	30.2	21.6	20000	1.13
	32	61.5	42.8	29.8	21.0	25000	**1.41**
HH-1 ($bpw = 1$)	2	51.8	31.1	18.2	10.6	54290	2.70
	4	54.1	33.0	19.8	11.8	49513	**2.71**
	8	**55.0**	**34.0**	20.7	12.6	46682	2.70
	16	54.8	**34.0**	**20.8**	**12.6**	44737	2.68
	32	54.7	**34.0**	20.4	12.4	43312	2.67
HH-2 ($bpw = \frac{1}{2}$)	2	58.9	38.8	25.2	16.4	23672	**1.32**
	4	60.6	40.5	26.9	17.8	22391	1.31
	8	**60.9**	**40.9**	**27.2**	17.8	21688	1.31
	16	60.5	40.5	27.1	**18.1**	21379	1.31
	32	60.1	40.1	26.8	18.0	21215	1.31
HH-3 ($bpw = \frac{1}{3}$)	2	60.8	40.9	27.5	18.6	16453	0.93
	4	61.7	42.0	28.6	19.5	15823	0.93
	8	**62.0**	**42.6**	**29.2**	**20.2**	15410	**0.94**
	16	61.5	41.9	28.6	19.7	15304	**0.94**
	32	61.5	42.0	28.7	20.0	15244	**0.94**

searching process and make the choice more difficult at every time. Theoretically, local optimum does not mean global optimum. BLEU 1, 2, 3, 4 are proportional to each other for most cases.

For SSH, *bs* determines the number of generated sentences, the more sentences it generated, the more data it can embed. As a result, bigger *bs* leads to higher embedding rates. The optimal results of SSH in general appear at $bs = 16$. By increasing *bs*, higher embedding capacity can be obtained, however, the corresponding BLEU may be reduced. Suppose the average length of the sentence is l characters, since $bs = 2^n$, the embedding capacity (bits per bit) of SSH can be calculated by:

$$C_{\mathrm{SSH}} = \frac{n}{8 \times l} \tag{16}$$

For HH scheme, *bpw* is the key factor to control embedding rates, it controls how many bits are embedded in one word. HH-1 model can achieve higher embedding rate than HH-2 and HH-3 because HH-1 embeds one secret bit per word, however, higher embedding rate will result in lower BLEU value inevitable. Also, a lower *bpw* leads to better text quality and lower embedding rate. It is worth noting that the number of embedding bits reduces with the increase of *bs* for HH scheme with the same *bpw*. That is because *bpw* only controls the number of bits hidden in one word. As *bs* increases, the length of generated sentences becomes shorter, which can lead to the reduction of total number of hidden bits. The reason for this length variation is that HH makes prediction only through half of the words in the dictionary, so it turns out to reduce the cost of prediction by cutting down the length of generated sentences. Nonetheless, the embedding capacity of HH is only dependent on *bpw* and has no relation to *bs*. Suppose n_w denotes the total number of words and n_c denotes the total number of characters in the generated sentences. The embedding capacity of HH with $bpw = 1/p$ can be calculated by the formula below:

$$C_{\mathrm{HH}-p} = \frac{n_w}{8 \times n_c \times p} \tag{17}$$

The BLEU results on COCO dataset from Table 4 have a lot in common with the results on Flickr8k, as their changing regularity is quite similar. The BLEU scores on COCO are larger compared with those on Flickr8k under the same experimental setup. It means we can enhance our model performance by simply increasing the size of the training data. With COCO dataset, The best BLEU scores by NIC and SSH scheme occur at $bs = 2$. It is because when the size of training data increases, the description ability of the trained language model is improved, so the model has a good generalization ability to get the best description from small *bs*. For HH scheme, embedding capacity increases as *bpw* increasing, but it also causes some loss on BLEU scores.

Compared with the baseline results of WWH, SSH and HH perform much better on the BLEU evaluation. It indicates that beam searching is a more advanced and effective way to find the optimal sentence than sampling.

Compare the performance between SSH and HH in both Flickr8k and COCO datasets, we can find that under the similar embedding capacity, SSH achieves higher BLEU than HH. For example, when embedding bits is capped at 5000 bits, SSH achieves better BLEU score than all HH methods with similar embedding capacity. That is because HH introduces extra noise during the embedding process by choosing the optimal words from one word category C_0 or C_1. In practical applications, we can choose a suitable scheme according to the our requirement.

4.3. Text statistical performance based on PPL evaluation

The PPL evaluation is carried on COCO dataset to evaluate the statistical performance of the generated texts. PPLs of NIC are used to make comparisons. The baseline of PPL provided by WWH is 26.58. The experimental results are shown in Table 5. The values in boldface show the smallest PPLs for each model among all *Beamsizes*.

As we can see from Table 5, NIC and SSH achieve their lowest PPL when $bs = 16$, while HH achieves its lowest PPL when $bs = 8$. SSH has similar PPL performance as NIC. PPLs by SSH are only a little bit higher than PPLs by NIC. Under the similar embedding capacity, SSH achieves better PPLs than HH. For the HH algorithm, with the growth of *bpw*, PPLs are getting smaller and smaller. When $bpw = 3$, HH outperforms SSH. HH-3 with $bs = 8$ achieves the best result on PPL evaluation among the proposed schemes, almost the same as NIC. Therefore, our algorithms can preserve the probabilistic characteristics of the texts generated by normal image caption model.

4.4. Comparison with other natural language steganographic schemes

A comparison between our framework and several other existing natural language schemes is provided in Table 6. The methods are compared on different perspectives, such as the way of embedding, requirement of cover texts, the knowledge base, embedding capacity and so on. The column of semantic and syntactic coherence shows whether the stego texts meet the grammatical or semantic requirements of the natural language. Blind extraction column is to see whether the original images or texts are needed in data extraction process. The embedding capacities are taken approximate values.

From Table 6, we can see the MarkovTextStego scheme achieves the highest embedding capacity. However, the generated sentences by Markov steganographic do not meet the semantic requirement. Like, "A stick in the little girl is blazing through a man stands on a blue coat fishes", and "brown and people sit on to dried cattails in front points at the lacrosse ball". Therefore, despite its high capacity, Markov lacks practicability.

Synonym substitution scheme is a widely used steganographic scheme. The embedding capacity of Synonym substitution scheme

Table 5
PPL evaluation on COCO.

	Beamsize (*bs*)	PPL
NIC [32]	1	12.05
	2	10.86
	4	10.10
	8	9.76
	16	**9.75**
	32	9.94
SSH	2	10.92
	4	10.23
	8	9.98
	16	**9.96**
	32	10.36
HH-1 ($bpw = 1$)	2	19.83
	4	17.49
	8	**16.59**
	16	16.64
	32	16.74
HH-2 ($bpw = \frac{1}{2}$)	2	14.11
	4	11.52
	8	**11.23**
	16	11.25
	32	12.30
HH-3 ($bpw = \frac{1}{3}$)	2	12.30
	4	10.18
	8	**9.82**
	16	9.91
	32	10.25

Table 6
Comparison of the existing techniques.

Algorithm	Embedding method	Cover-required	Statistic or rule based	Semantic coherence	Syntactic coherence	Embedding rate (*bits*/*bit*)	Knowledge for extraction	blind extraction	paper
TBS	text-generation	y	Statistic	y	y	0.52	Translation model	n	[26]
Markov	text-generation	n	Statistic	n	y	6.5	Markov model	y	[27]
Nicetext	text-generation	n	rule	n	y	0.29	template	y	[24]
syntactic	text-modification	y	rule	y	y	0.52	parsing	n	[21]
Synonym Substitution	text-modification	y	rule	y	y	0.68	synset	y	[20]
WWH	text-generation	n	Statistic	y	y	2.75	CNN-LSTM model	n	our scheme
SSH	text-generation	n	Statistic	y	y	1.41	CNN-LSTM model	n	our scheme
HH-1	text-generation	n	Statistic	y	y	2.74	hash function	y	our scheme
HH-2						1.34			
HH-3						0.95			

can not set to be too high because of the distortions caused by modification.

In terms of the blind extraction capability, TBS needs the original untranslated sentences as references. Syntactic algorithm needs the original sentences before syntactic transformation. WWH and SSH need the original images to duplicate stego generation process. Therefore, these algorithms can not achieve blind extraction. For the rest of the algorithms, the original texts or images are not required for extraction, so we call them blind extraction algorithms.

From comparison, we can draw the conclusion that our proposed models are superior models with desirable embedding capacities. Moreover, HH is a blind extraction steganography algorithm with minimum knowledge required.

4.5. Security analysis

The security analyses are conducted from both subjective aspect and objective aspect.

In subjective aspect, some experiments are done to test the performance based on human evaluation to compare our schemes with TBS, Synonym, and Markov based steganographic algorithms. 100 candidates are chosen to take part in the human evaluation tests. Each candidate evaluates a text file contains 200 sentences stem from two sources, one is generated by a specific steganography algorithm, say, TBS, Synonym substitution, MarkovTextStego, SSH ($bs = 16$), and HH-3 ($bs = 16$). The other one is from normal image description sentences which are manually annotated. Each type covers about 50% of the total amount. All of the steganography methods are trained on the same image description corpus to make a unified language style of the generated sentences. Except TBS, the embedding rates of these schemes are set as 0.8 bpb. For TBS, one secret bit needs to be embedded in one sentence, which makes the embedding capacity to be a fixed value. We examine two aspects of mistakes, average false alarm rate(P_{fa}) and average miss detection rate (P_{md}). P_E is the mean value of P_{fa} and P_{md}. The formulas of P_{fa}, P_{md}, and P_E are shown as follows.

$$P_{fa} = FP/(TP + FP) \tag{18}$$

$$P_{md} = FN/(TP + FN) \tag{19}$$

$$P_E = (P_{fa} + P_{md})/2 \tag{20}$$

where TP (true positives) refers to the number of stego sentences that are correctly labeled by the volunteer or classifier. TN (True negatives) refers to the number non-stego sentences that are correctly labeled by the volunteer. FP (False positives) is the number of non-stego sentences that are incorrectly labeled as stego. FN (False negatives) is the number of stego sentences that are mislabeled as non-stego. P_{fa} is the rate that considers the normal sentences as the stego ones, while P_{md} is the rate that considers the stego sentences as normal sentences. Bigger P_E means higher anti-steganalysis performance. The results are shown in Table 7. The values in boldface show the best anti-subjective-detection performance among all the steganography algorithms.

From Table 7, we can see that SSH with $bs = 16$ has the best performance. The performance of HH-3 is very close to SSH. P_{fa} of the proposed schemes are higher than other algorithms, that means the stego sentences by SSH and HH are so similar to the normal ones that human eyes can hardly distinguish them. P_E of SSH is 0.57%, it is similar to the result of random guessing.

In order to test the objective security, an SVM model with RBF kernel is trained as a classifier to distinguish stego texts from cover texts. The features utilized by [42,43] as well as PPL are merged together to form a feature vector. These features reflect the statistical characteristics of texts from different aspects.

In our experiments, 4960 sentences generated by NIC with different *bs* based on COCO dataset are served as cover texts. Besides, 4960 stego sentences are obtained for each steganographic algorithm based on COCO. Then we take 480 cover sentences from cover texts and 480 stego sentences from stego texts to form the test corpus, the rest are used as training samples. Because one sentence is too short to extract features, during training and testing process, 40 sentences are grouped as one sample. The detection results of different steganographic algorithms are displayed in Table 8. The value in boldface means the lowest detection rate among all steganograpy algorithms.

We can see from the Table 8 that SSH has the lowest detection rates compared with other natural language steganographic methods. For HH algorithm, a larger *bpw* can get better anti-steganalysis performance. Overall, experiments show that the proposed schemes get considerable results in both subjective evaluation by human eyes and objective evaluation by steganalysis.

Table 7
Comparison of text quality results.

	TBS [26]	Synonym [20]	Markov [27]	SSH($bs = 16$)	HH-3($bs = 16$)
Mixed Text size(kb)	20	24	27	21	20
Embedding rate	0.26	0.8	0.8	0.8	0.8
Average P_{fa}	0.26	0.30	0.26	**0.40**	0.37
Average P_{md}	0.66	0.49	0.38	**0.74**	0.69
Average P_E	0.46	0.40	0.32	**0.57**	0.53

Table 8
Comparison of steganalysis results.

	TBS 26]	Synonym [20]	Markov [27]	SSH	HH-1	HH-2	HH-3
$P_E(\%)$	70.08	79.51	56.10	**50.83**	57.50	55.83	54.16

Moreover, by means of the Hash Function in HH, the security performance can be further ensured. For example, some attackers may conduct statistical analyze by detecting the statistical law of the changes of 0 and 1. With the help of Hash Functions, it becomes impossible to make an accurate analysis for the attackers because of the randomness brought by the Hash Function.

4.6. Reversibility analysis

For a carrier-less steganographic scheme, reversibility is a very important evaluation standard to test the accuracy of data extraction. In this subsection, we conduct data extraction experiments to check the correctness of data extraction for every proposed scheme. Experiments prove that all the proposed algorithms, including WWH, SSH, and HH, can achieve 100% accuracy rate in data extraction.

Actually, for WWH and SSH, the key factor to correct extraction is that the maximum probability and the second largest probability are different at each time. If they are the same, then the model may randomly choose one word from the best two words, which will cause confusion. We also carry out an experiment that use 10,000 probability distributions produced in different times during the generation. The histogram is shown in Fig. 5.

Here the abscissa shows the ratio of largest probability to second largest probability for a given probability distribution, the ordinate represents the number of ratios. As we can see from the graph, since the ratios are all bigger than 1, the largest probability and the second largest probability for every time are not the same. Therefore, WWH and SSH can achieve correct extraction.

For HH, the data extraction is based on formula (9). For a word w, $v(w,k)$ calculated by formula (9) with a certain key k is a fixed value, so HH can achieve correct extraction.

4.7. Some examples of the proposed schemes

In this subsection,we present some descriptive sentences generated by NIC and the proposed steganographic schemes for two test images. The test images we choose are shown in Fig. 6. The non-stego and stego sentences are generated through different algorithms and beam sizes.

Test image 1:	Test image 2:
NIC($bs = 32$): two dogs play in the grass.	NIC($bs = 32$): a man in a blue shirt is standing in front of a building.
WWH: two brown dogs are running in the grass.	WWH: two people are standing in front of a building with a large painted bag.
SSH($bs = 4$): two dogs are playing in the grass.	SSH($bs = 16$): a group of people walking down a street.
SSH($bs = 16$): two dogs play in the grass.	SSH($bs = 32$): a group of people are standing in front of a red building.
HH-1($bs = 4$): two black dogs running on grass.	HH-1($bs = 4$): a person is standing on a sidewalk.
HH-1($bs = 16$): a black dog is chasing a brown dog in the grass.	HH-1($bs = 16$): a group of people are walking in front of a white building.
HH-2($bs = 4$): two black dogs are running through a field.	HH-2($bs = 4$): a group of people stand on a sidewalk.
HH-2($bs = 16$): the two dogs are playing in the grass.	HH-2($bs = 16$): a man and woman walk in a city street.
HH-3($bs = 4$): two dogs are playing in a grassy area.	HH-3($bs = 16$): a group of people walking down a city street.
HH-3($bs = 16$): a dog runs through the grass.	

The generation results intuitively demonstrate that the stego sentences generated by the proposed schemes are as good as the sentences generated by the non-hiding scheme NIC. In most of the cases, WWH has a tendency to a generate sentences with more characters. As we can see from the results, the stego sentences generated by SSH and HH have very good syntactic and semantic performances.

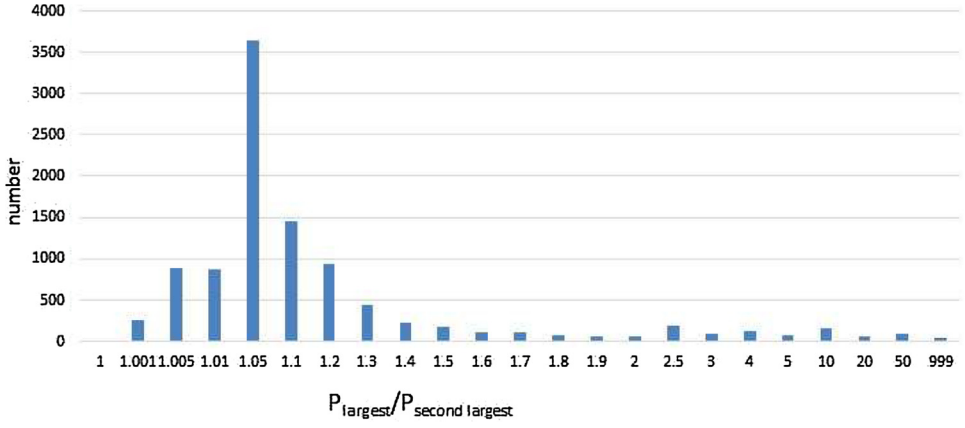

Fig. 5. Histogram of maximum probability divided by secondary probability.

(a) (b)

Fig. 6. (a) Test image 1 (b) Test image 2.

5. Conclusions and future work

In this paper, we have presented a novel natural language steganographic framework based on neural image description. CNN and LSTM are combined to generate stego natural language sentences that accord with the content of test images. Two specific methods, SSH and HH, are proposed to meet different requirement. When the CNN-LSTM model is available for data extraction, SSH is an ideal scheme to conduct steganography. When the model and images can not be given to the receiver for data extraction, HH is an effective scheme. We can get different embedding rates by varying beam size *bs*. *bpw* is used to balance the relationship between text qualities and embedding rates. A wide spectrum of experimental results show that the stego texts generated by our scheme are hard to be detected by human eyes and steganalysis.

The proposed method belongs to "carrier-less" natural language generation. This framework can be applied in the online interactive platforms such as the photo-sharing websites and video-streaming websites with danmu.

In the future research, we will apply the framework to other image description models, some methods such as text alignment can be used to obtain more than one descriptions for each image in order to increase the amount of hidden information for a test image.

Acknowledgement

We would like to thank Andrej Karpathy for his contribution to the demo code of image caption *neuraltalk*. We would like to appreciate the efforts spent by the reviewers. This work is supported by the National Natural Science Foundation of China (Nos. 61802410 and 61872368), and the Chinese Universities Scientific Fund (2017QC003 and 2018QC024).

References

[1] M. Khairullah, A novel text steganography system using font color of the invisible characters in microsoft word documents, 2009 Second International Conference on Computer and Electrical Engineering, vol. 1, 2009, pp. 482–484.
[2] S. Bhattacharyya, P. Indu, G. Sanyal, Hiding data in text using ASCII mapping technology (AMT), Int. J. Comput. Appl. 70 (18) (2013) 29–37.
[3] R. Saniei, K. Faez, The capacity of arithmetic compression based text steganography method, in: 2013 8th Iranian Conference on Machine Vision and Image Processing (MVIP), 2013, pp. 38–42.
[4] S. Chaudhary, M. Dave, A. Sanghi, Aggrandize text security and hiding data through text steganography, in: 2016 IEEE 7th Power India International Conference (PIICON), 2016, pp. 1–5.
[5] D.R. Huang, J.F. Liu, J.W. Huang, Embedding strategy and algorithm for image watermarking in dwt domain, J. Software 13 (7) (2002) 1290–1297.
[6] T. Filler, J. Judas, J. Fridrich, Minimizing additive distortion in steganography using syndrome-trellis codes, IEEE Trans. Inform. Forens. Secur. 6 (3) (2011) 920–935.
[7] L. Guo, J. Ni, Y.Q. Shi, Uniform embedding for efficient jpeg steganography, IEEE Trans. Inform. Forens. Secur. 9 (5) (2014) 814–825.
[8] C. Wang, J. Ni, An efficient jpeg steganographic scheme based on the block entropy of dct coefficients, in: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2012, pp. 1785–1788.
[9] Hong Zhang, Yun Cao, Xianfeng Zhao, Motion vector-based video steganography with preserved local optimality, Multimedia Tools Appl. 75 (21) (2016) 13503–13519.
[10] Tseng-Jung Lin, Kuo-Liang Chung, Po-Chun Chang, Yong-Huai Huang, Hong-Yuan Mark Liao, Chiung-Yao Fang, An improved dct-based perturbation scheme for high capacity data hiding in h.264/avc intra frames, J. Syst. Softw. 86 (3) (2013) 604–614.
[11] S. Jangid, S. Sharma, High psnr based video steganography by mlc(multi-level clustering) algorithm, in: 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), 2017, pp. 589–594.
[12] Hui Tian, Ke Zhou, Hong Jiang, Dan Feng, Digital logic based encoding strategies for steganography on voice-over-ip, in: Proceedings of the 17th ACM International Conference on Multimedia, MM '09, ACM, New York, NY, USA, 2009, pp. 777–780.
[13] C. Pun, X. Yuan, Robust segments detector for de-synchronization resilient audio watermarking, IEEE Trans. Audio Speech Language Process. 21 (11) (2013) 2412–2424.
[14] M. Fallahpour, D. Megías, Audio watermarking based on fibonacci numbers, IEEE/ACM Trans. Audio Speech Lang. Process. 23 (8) (2015) 1273–1282.
[15] J.T. Brassil, S. Low, N.F. Maxemchuk, Copyright protection for the electronic distribution of text documents, Proc. IEEE 87 (7) (1999) 1181–1196.
[16] Prem Singh, A novel approach of text steganography based on null spaces, IOSR J. Comput. Eng. 3 (4) (2012) 11–17.
[17] Bala Krishnan Ramakrishnan, Prasanth Kumar Thandra, A.V. Satya Murty Srinivasula, Text steganography: a novel character-level embedding algorithm using font attribute, Secur. Commun. Networks 9 (18) (2017) 6066–6079.
[18] S. Shi, Y. Qi, Y. Huang, An approach to text steganography based on search in internet, in: 2016 International Computer Symposium (ICS), 2016, pp. 227–232.
[19] Xianyi Chen, Huiyu Sun, Yoshito Tobe, Zhili Zhou, Xingming Sun, Coverless information hiding method based on the chinese mathematical expression, in: Zhiqiu Huang, Xingming Sun, Junzhou Luo, Jian Wang (Eds.), Cloud Computing and Security, Springer International Publishing, Cham, 2015, pp. 133–143,.
[20] H.A. Yajam, A.S. Mousavi, M. Amirmazlaghani, A new linguistic steganography scheme based on lexical substitution, in: 2014 11th International ISC Conference on Information Security and Cryptology, 2014, pp. 155–160.
[21] Zu Xu Dai, Hong Fan, Guo Hua Cui, F.U. Min, Watermarking text document based on statistic property of part of speech string, J. Commun. 28 (4) (2007) 108–113.
[22] Mikhail J. Atallah, Victor Raskin, Michael Crogan, Christian Hempelmann, Florian Kerschbaum, Dina Mohamed, Sanket Naik, Natural language watermarking: design, analysis, and a proof-of-concept implementation, in: Proceedings of the 4th International Workshop on Information Hiding, IHW '01, Springer-Verlag, London, UK, UK, 2001, pp. 185–199.
[23] K. Maher, Texto, 2007. <ftp://ftp.funet.fi/pub/crypt/steganography/texto.tar.gz>.
[24] M. Chapman, G. Davida, Nicetext, 2017. <http://www.securityfocus.com/tools/1183>.

[25] Peter Wayner, Mimic functions, Cryptologia 16 (3) (1992) 193–214.
[26] Ryan Stutsman, Christian Grothoff, Mikhail Atallah, Krista Grothoff, Lost in just the translation, in: Proceedings of the 2006 ACM Symposium on Applied Computing, SAC '06, ACM, New York, NY, USA, 2006, pp. 338–345.
[27] H. Hernan Moraldo, An approach for text steganography based on markov chains, CoRR, abs/1409.0915, 2014.
[28] Yubo Luo, Yongfeng Huang, Fufang Li, Chinchen Chang, Text steganography based on Ci-poetry generation using markov chain model, Ksii Trans. Internet Inform. Syst. 10 (9) (2016) 4568–4584.
[29] Sepp Hochreiter, Jürgen Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[30] Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, Tamara L. Berg, Baby talk: understanding and generating image descriptions, in: Proceedings of the 24th CVPR. Citeseer, 2011.
[31] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, David Forsyth, Every picture tells a story: generating sentences from images, in: Kostas Daniilidis, Petros Maragos, Nikos Paragios (Eds.), Computer Vision – ECCV 2010, Springer, Berlin, Heidelberg, 2010, pp. 15–29.
[32] Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan, Show and tell: a neural image caption generator, CoRR, abs/1411.4555, 2014.
[33] Qianrong Zhou, Liyun Wen, Xiaojie Wang, Long Ma, Yue Wang, A hierarchical LSTM model for joint tasks, in: Maosong Sun, Xuanjing Huang, Hongfei Lin, Zhiyuan Liu, Yang Liu (Eds.), Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, Springer International Publishing, Cham, 2016, pp. 324–335.
[34] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, Yoshua Bengio, Show, attend and tell: neural image caption generation with visual attention, CoRR, abs/1502.03044, 2015.
[35] Qi Wu, Chunhua Shen, Anton van den Hengel, Lingqiao Liu, Anthony R. Dick, Image captioning with an intermediate attributes layer, CoRR, abs/1506.01144, 2015.
[36] A. Karpathy, L. Fei-Fei, Deep visual-semantic alignments for generating image descriptions, IEEE Trans. Pattern Anal. Mach. Intell. 39 (4) (2017) 664–676.
[37] Micah Hodosh, Peter Young, Julia Hockenmaier, Framing image description as a ranking task: data, models and evaluation metrics, J. Artif. Intell. Res. 47 (1) (2015) 853–899.
[38] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, C. Lawrence Zitnick, Microsoft COCO: common objects in context, CoRR, abs/1405.0312, 2014.
[39] Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu, BLEU: a method for automatic evaluation of machine translation, in: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02, Association for Computational Linguistics, Stroudsburg, PA, USA, 2002, pp. 311–318.
[40] F. Jelinek, Continuous speech recognition by statistical methods, Proc. IEEE 64 (4) (1976) 532–556.
[41] C.E. Shannon, A mathematical theory of communication, Bell Syst. Techn. J. 27 (3) (1948) 379–423.
[42] Lingyun Xiang, Xingming Sun, Gang Luo, Bin Xia, Linguistic steganalysis using the features derived from synonym frequency, Multimedia Tools Appl. 71 (3) (2014) 1893–1911.
[43] Zhili Chen, Liusheng Huang, Peng Meng, Wei Yang, Haibo Miao, Blind linguistic steganalysis against translation based steganography, in: Hyoung-Joong Kim, Yun Qing Shi, Mauro Barni (Eds.), Digital Watermarking, Springer, Berlin Heidelberg, 2011, pp. 251–265.