TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 08/10 pp516–527
DOI: 10.26599/TST.2019.9010027
Volume 25, Number 4, August 2020

A Generative Method for Steganography by Cover Synthesis with Auxiliary Semantics

Zhuo Zhang, Guangyuan Fu, Rongrong Ni, Jia Liu*, and Xiaoyuan Yang

Abstract: Traditional steganography is the practice of embedding a secret message into an image by modifying the information in the spatial or frequency domain of the cover image. Although this method has a large embedding capacity, it inevitably leaves traces of rewriting that can eventually be discovered by the enemy. The method of Steganography by Cover Synthesis (SCS) attempts to construct a natural stego image, so that the cover image is not modified; thus, it can overcome detection by a steganographic analyzer. Due to the difficulty in constructing natural stego images, the development of SCS is limited. In this paper, a novel generative SCS method based on a Generative Adversarial Network (GAN) for image steganography is proposed. In our method, we design a GAN model called Synthetic Semantics Stego Generative Adversarial Network (SSS-GAN) to generate stego images from secret messages. By establishing a mapping relationship between secret messages and semantic category information, category labels can generate pseudo-real images via the generative model. Then, the receiver can recognize the labels via the classifier network to restore the concealed information in communications. We trained the model on the MINIST, CIFAR-10, and CIFAR-100 image datasets. Experiments show the feasibility of this method. The security, capacity, and robustness of the method are analyzed.

Key words: information hiding; steganography; steganography without modification; Steganography by Cover Synthesis (SCS); generative adversarial networks

1 Introduction

Image steganography is a technique of hiding secret messages in a cover image by advanced methods to prevent the messages from being discovered by an adversary. Traditional image steganography usually uses a cover-modified method to embed messages in the image. Common spatial domain-based advanced methods include the Highly Undetetable steGO (HUGO) algorithm[1], Wavelet obtained Weights (WoW) algorithm[2], and the Spatial UNIversal WAvelet Relative Distortion (S-UNIWARD) algorithm[3]. The JPEG domain UNIversal WAvelet Relative Distortion (J-UNIWARD) algorithm[3] is a transform domain-based method. The modification-based methods are also naturally applied to video steganography[4,5]; however, these approaches, based on modification, inevitably introduce detectable anomalies to the original cover. Traditional steganography needs to be able to withstand current techniques based on deep learning steganography analyzers[6–8]. Some scholars have proposed the concept of Steganography Without Modification (SWM)[9]. The SWM methods usually

• Zhuo Zhang and Guangyuan Fu are with Rocket Force University of Engineering, Xi'an 710025, China. Zhou Zhang is also with the Key Laboratory of Network and Information Security of PAP, Engineering University of PAP, Xi'an 710086, China. E-mail: adam_zz01@163.com; dr-f@21cl.com.
• Rongrong Ni is with the Institute of Information Science, Beijing Jiaotong University, Beijing 100044, China. E-mail: rrni@bjtu.edu.cn.
• Jia Liu and Xiaoyuan Yang are with the Key Laboratory of Network and Information Security of PAP, Engineering University of PAP, Xi'an 710086, China. E-mail: liujia1022@gmail.com; wgd_yxy@163.com.
∗ To whom correspondence should be addressed.
Manuscript received: 2019-04-11; revised: 2019-06-03; accepted: 2019-06-05

employ two methods for information hiding. First, secret messages can be delivered by selecting a natural image[10–12], however, this type of method requires constructing a large natural image database and building complex index structures to establish a mapping relationship between images and information through complex mathematical functions. Second, the sender tries to embed secret messages in the image by building a carrier. Due to the difficulty in constructing natural images, some researchers have realized steganography by synthesizing texture images[13, 14]. However, the steganographic capacity of the synthesized texture image is low. Moreover, since the synthesized image is not a "natural image", it easily causes suspicion of the warder in the real world.

In 2014, Goodfellow et al.[15] proposed a deep-generative network model, called the Generative Adversarial Network (GAN). Recently, the GAN model has been widely used for image generation[16, 17]. The emergence of GAN also provides a new opportunity for the development of information hiding technology. Some scholars have applied GAN to the information hiding field[18–23] and applied the game theory-based generative model to the various elements of the steganography model. Most of these methods[18–23] attempt to use an adversarial strategy in GAN to design a cover-modified steganography for improving the resistance to steganalysis. Currently, the GAN-based method is seldom used in the SWM field. Some methods attempt to construct the stego image by a direct generator[21–23]. The ability to generate natural images based on GAN has gradually improved. It provides a powerful tool for SCS. Inspired by these methods[21–23], we propose a novel cover-synthesis steganography technique based on a generative model, and we call it Synthetic Semantics Stego Generative Adversarial Network (SSS-GAN). In our method, relationship maps between secret message segments and image semantic information are first built, and then semantic information is used to generate stego images through the trained generative model. Simultaneously, the extractor (a subnetwork of the model) can recognize the semantic label of the generated stego image to achieve message extraction. Compared with previous works, the main contributions of this paper are as follows:

(1) With SSS-GAN adversarial learning, our generative model can quickly train to converge on different image datasets. Then, the trained generator can directly generate stego images from the secret message, and the message extractor can extract the secret information in these images.

(2) Through the optimized training of the message extractor, our method can achieve a 100% message extraction accuracy in fixed-length image sequences, which makes the method more practical.

(3) Since there is no modifying process in our method, our method can resist detection by the state-of-the-art steganalysis algorithm.

(4) Because messages actually exist in the semantic information of images, the dense images using the SSS-GAN model have the robustness to resist image processing.

The rest of the paper is structured as follows. In Section 2, we present the related work on SWM and some proposed GAN-based methods of steganography. In Section 3, we describe the proposed method in detail. In Section 4, we present the experimental results and analysis. In Section 5, we state our conclusions.

2 Related Work

2.1 Traditional steganography without modification

Steganography without modification means to complete communication without rewriting the carrier image (cover image). In general, there are two kinds of SWM methods widely used for images: cover selection and cover synthesis[9]. Cover selection steganography uses natural images to convey secret messages. The main idea of this method is to establish a mapping relationship between secret messages and natural images and then pass secret information by these natural images so that the statistical-based steganalysis algorithm can be avoided. Zhou et al.[10] proposed a robust hashing algorithm. They used this algorithm to calculate the hash sequence of the images. Then, these images were indexed into a dataset according to their hash sequences. For the binary secret data segment, the method selects the image from the database with the hash value equal to the value of the segment. Zhou et al.[11] also proposed another SWM method using the bag-of-words model to establish a mapping relationship between secret data segments and images. However, this approach often requires building a large image library, and numerous calculations are required to establish a correspondence between the secret information and the images. The idea of the cover-synthesis approach is to artificially synthesize

a secret message into a texture image. Xu et al.[13] presented a texture image synthesis approach in which Local Binary Pattern (LBP) encoding is first used to draw dots on a white paper, representing secret information, and then the texture sample is selected to synthesize a texture map according to the LBP dot map. Wu and Wang[14] presented a novel method for texture synthesis. They transformed an input image or a text message into an intricate texture by combining several reversible functions provided in the system. The input image or message could be recovered by reversing the process of these functions. However, this kind of texture synthesis steganography is based on the premise that the cover may not represent the content in the real world, which easily causes warden suspicion.

2.2 Steganography with GAN

Recently, with the development of artificial intelligence technology, some new steganographic modes have gradually attracted attention. Different from the traditional modification-based steganography, many scholars have begun to pay attention to image steganography methods based on the generation model.

Cover-modified steganography with GAN. Volkhonskiy et al.[18] proposed a Steganographic Generative Adversarial Network (SGAN) model based on Deep Convolutional Generative Adversarial Network (DCGAN). The SGAN model consists of a generator network for generating cover images, a discriminator network for discriminating generated images from real images, and a steganalyzer network for steganalysis. The experiment proves that hiding information in the SGAN-generated cover images is more secure than hiding information in natural images. Tang et al.[19] proposed the framework of Automatic Steganographic Distortion Learning (ASDL). They used adversarial networks to learn a generator G, obtained the image pixel modification probability matrix by sampling G, and then used the STC method[24] to embed the information. This scheme is called ASDL-GAN. However, the performance of this method is still inferior to the conventional steganography algorithm S-UNIWARD. Yang et al.[20] proposed a GAN-based scheme UT-SCA-GAN. In this study, a generator based on U-NET has been proposed to translate a cover image into an embedding change probability. Compared with the ASDL-GAN method[19], this framework can dramatically increase the security performance, and it performs better than the hand-crafted steganographic algorithm S-UNIWARD. These GAN-based methods described above are embedding-based methods.

Cover-synthesis steganography with GAN. There are some novel methods that use GAN to learn how to directly generate cover (stego) images. Hayes and Danezis[21] defined a game confrontation between three networks, Alice, Bob, and Eve, while training the steganography and steganalysis methods. Finally, the generator (Alice) learns how to generate images. Hu et al.[22] recently proposed a GAN-based SWM method. First, the mapping rules of noise and messages are established, and the image is directly generated by the trained DCGAN. Afterward, they redesigned and retrained a new network to extract the messages in the generated image. However, the image generated by this method is of poor quality. In addition, a separately trained message extractor is needed to extract the noise, but the extraction accuracy is low. In particular, as the number of embedded messages increases in the stego image, the accuracy of message extraction decreases significantly. Liu et al.[23] proposed a GAN-based SWM framework for the construction of a Digital Cardan Grille (DCG) for information hiding. The message is written to the corrupted region of an image that needs to be filled in advance according to the DCG. Then, the corrupted image with the secret message is fed into a DCGAN model for semantic image inpainting. The GAN reconstructs the corrupted image but also generates a stego image that contains the logic rationality of the image content. The DCG is the key for extracting messages in the secret communication. Some scholars have also used neural network models to realize the cover-synthesis steganography with other media carriers (covers), such as text and audio. Fang et al.[25] proposed a steganographic model based on the existing long short-term memory language model and demonstrated that the model can produce realistic tweets and emails while hiding information. Yang et al.[26] recently proposed an automatic audio generation-based steganography model based on lookback recurrent neural network; the model can automatically generate audio covers based on secret information.

In this paper, we propose a novel cover-synthesis steganography framework via generative adversarial networks (SSS-GAN). In our method, the model needs to be trained only once on an image dataset. By mapping the secret message segment into the semantic

information of the image, the model can directly generate the stego image via our model. Meanwhile, the extractor network of the model can directly recognize the semantic information of the stego image with high accuracy. In addition, our model can quickly train to converge on different datasets.

3 Our Method

As illustrated in Fig. 1, the proposed steganography framework consists of three phases.

In the first phase, we train the proposed SSS-GAN model with semantic labels (we use image category labels as semantic information in this paper) on an image dataset that is constructed by collecting a large number of images with different semantic labels from the Internet. When the model converges, we obtain the image generator (G network) and auxiliary classifier (C network).

In the second phase, we use the trained G network combined with random labels and noise to generate a large number of images to form a stego image dataset and further train the $\hat{C}$ network (obtained by connecting the D network to the C network) to improve the extraction accuracy of the message extractor.

In the third phase, the sender segments the secret information and maps the information segmentation to the semantic label and then uses G to generate the stego image. The receiver uses $\hat{C}$ as an extractor to extract the semantic information in the received stego image and then reverse-maps the semantic information into a secret information segment. Table 1 presents the notations used in this paper.

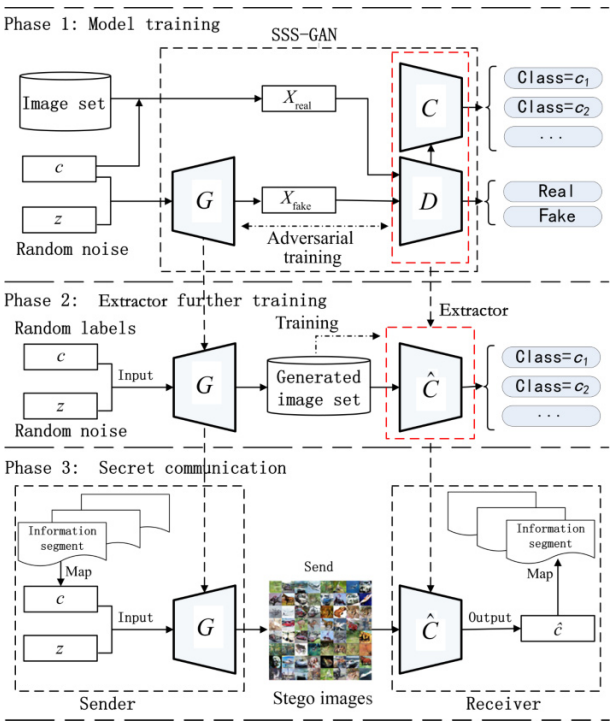


Fig. 1 Proposed steganography framework using SSS-GAN.

3.1 Mapping rule for information to semantic labels

Mapping the information to image semantic labels is preparation before steganography. We can segment the binary data of the m-bit from the binary data of secret information S and then map it to the j-th image semantic label according to the value of the m-bit information (e.g., we map every 6-bit to one of 64 labels), $n = 2^m$, as shown in Fig. 2.

After obtaining the semantic labels, we use the SSS-GAN model to generate the stego image. The model is described in detail below.

3.2 SSS-GAN model design

The SSS-GAN model contains three networks: a

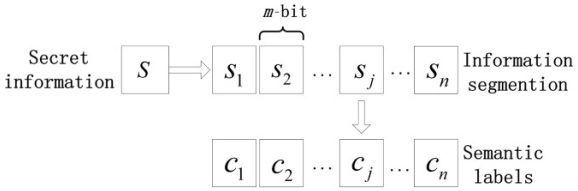


Fig. 2 Diagram for mapping secret data into semantic labels.

Table 1 Notations used in this paper.

Symbol	Description	Symbol	Description
G	Generator network	D	Discriminator network
C	Auxiliary classifier network	$\hat{C}$	Extractor
c	Real semantic label	$\hat{c}$	Semantic label extracted by $\hat{C}$
z	Random noise	*stego*	Images generated by G
X_{real}	Real image from the image dataset	X_{fake}	Generate image
S	Secret information	s_j	j-th segment of S
$\hat{c}_j$	j-th semantic label	z_j	j-th random noise
α	Hyperparameter	V	0 or 1

generator network (G), a discriminator network (D), and an auxiliary classifier network (C). The model structure is shown in Fig. 1. The generative network G is used to combine the noise z and the semantic label c of the image to generate a picture X_{fake}, $X_{\text{fake}} = G(c, z)$; D is used to judge the true and false probabilities of the input image, which can be expressed as $D(X) = P(V|X)$, $X \in \{X_{\text{real}}, X_{\text{fake}}\}$, $V \in \{0, 1\}$; the auxiliary classifier C is used to determine the category label implied by the image, which can be expressed as $C(X) = P(I|X)$, $X \in \{X_{\text{real}}, X_{\text{fake}}\}$, $I \in \{c_1, c_2, \ldots, c_n\}$.

The detailed configurations of the three networks are as follows:

In Tables 2–4, Y_dim represents the dimension of the label, and Z_dim represents the dimension of the noise; FC represents a fully connected layer; Conv2d represents a 2-D convolutional layer; BN represents batch normalization; ReLU represents the rectified linear unit; and leaky ReLU represents a leaky rectified linear unit. The kernel configurations of the convolutional layers are given in the following format: filter/stride (kernel width×kernel height/stride); output size (feature map width×feature map height×numbers of output feature maps). The settings in the table are for the MNIST training set[27] (the image is a 28×28 grayscale image), and the settings in parentheses are for the CIFAR dataset[28] (the image is a 32×32 color image).

Table 2 Detailed architecture of the *G* network.

Layer	Fliter/Stride	Output size	Process
1	-	Z_dim+Y_dim	-
2	-	1024	FC-BN-Relu
3	-	128×7×7 (128×8×8)	FC-BN-Relu
4	-	7×7×128 (8×8×128)	Reshape
5	4×4 / 2	14×14×16 (16×16×64)	Dconv2d-BN-Relu
6	4×4 / 2	28×28×1 (32×32×3)	Dconv2d-Sigmoid

Table 3 Detailed architecture of the *D* network.

Layer	Fliter/ Stride	Output size	Process
1	-	28×28×1 (32×32×3)	-
2	4×4 / 2	14×14×16 (16×16×64)	Conv2d-Leaky Relu
3	4×4 / 2	7×7×128 (8×8×128)	Conv2d-BN-Leaky Relu
4	-	6272 (8192)	Reshape
5 (Input for *C*)	-	1024	FC-BN-Leaky Relu
6	-	1	FC-Sigmoid

Table 4 Detailed architecture of the *C* network.

Layer	Fliter/Stride	Output size	Process
1	-	1024	-
2	-	128	FC-BN-Leaky Relu
3	-	Y_dim	FC
4	-	1	Softmax
5	-	1	Arg_max

3.3 Training strategy for the SSS-GAN

The aim of the generative model G is to generate an image according to the semantic label c, which cannot be distinguished by D, and the semantic label of the generated image can be recognized by the auxiliary classifier. To achieve this goal, we draw on the design method of the loss function of the auxiliary classifier GAN[29] to define the loss function of the model as follows:

$$\max_{G\_C} \max_{D\_C} L^{\text{SSS-GAN}} = L_{G\_C}^{\text{SSS-GAN}} + L_{D\_C}^{\text{SSS-GAN}} \tag{1}$$

where

$$L_{G\_C}^{\text{SSS-GAN}} = (1-\alpha) \cdot E\left[\log P\left(V = 1|X_{\text{fake}}\right)\right] + \alpha \cdot E\left[\log P\left(I = c|X_{\text{fake}}\right)\right] \tag{2}$$

$$\begin{aligned} L_{D\_C}^{\text{SSS-GAN}} =& (1-\alpha)\left\{E\left[\log P\left(V = 1|X_{\text{real}}\right)\right] + E\left[\log P\left(V = 0|X_{\text{fake}}\right)\right]\right\} + \\ & \alpha \cdot \left\{E\left[\log P\left(I = c|X_{\text{real}}\right)\right] + E\left[\log P\left(I = c|X_{\text{fake}}\right)\right]\right\} \end{aligned} \tag{3}$$

$L_{G\_C}^{\text{SSS-GAN}}$ and $L_{D\_C}^{\text{SSS-GAN}}$ are linear combinations of the G network and C network and D network and C network loss, respectively. We use a hyperparameter $\alpha \in (0, 1)$ to control the tradeoff between the importance of the generated image quality and information extraction accuracy.

Update rules. Model training uses a gradient descent method to update network weights according to the following rules:

Model training uses a gradient descent method to update network weights according to the following rules:

- Keeping the G fixed, update the D and C by

$$W_{D,C} \leftarrow W_{D,C} + \gamma_{D,C} \nabla_{D,C} L_{D\_C}^{\text{SSS-GAN}} \tag{4}$$

- Keeping the D and C fixed, update the G by

$$W_G \leftarrow W_G + \gamma_G \nabla_G L_{G\_C}^{\text{SSS-GAN}} \tag{5}$$

where W_{X} represents the weights of neural network X, and $\nabla_{\text{X}} L^{\text{SSS-GAN}}$ represents the gradient update value of the X network.

3.4 Training strategy for the extractor

Generator G, discriminator D, and auxiliary classifier C

are obtained in the first phase. There is no need to design a new classifier to identify the stego image because we obtain classifier C in the process of adversarial training. This classifier can extract the semantic labels of most real images and stego images. We attempt to optimize the classifier on the stego image dataset to further improve the extraction accuracy for the stego image.

We first use the trained G to combine the random noises and the random labels to generate the stego image dataset and connect D and C as a network $\hat{C}$ (connect the trained D and C structures and keep the both networks' parameters). Then, we perform further training on the stego image dataset, as shown in Fig. 1. In this process, we optimize the extractor $\hat{C}$ by as much training as possible to improve its accuracy. The optimization goal for training $\hat{C}$ is to minimize the deviation between the input label c and recovery label $\hat{c}$ resolved by extractor $\hat{C}$. The loss function of the extractor model is defined as the cross-entropy loss, which is shown in Eq. (6), where $H(\cdot)$ represents the cross-entropy loss function,

$$L(\hat{C}) = H(c, \hat{c}) = H(c, \hat{C}(G(z, c))) \tag{6}$$

When the training loss is sufficiently small, we stop training for $\hat{C}$ and use $\hat{C}$ to recover secret data from the stego images generated by G.

3.5 Secret communication algorithms

After the SSS-GAN model is trained, the sender and receiver follow the algorithms below for secret communication.

Information hiding algorithm. The sender uses the semantic label and the random noise as the driver, generates the image through the G of the trained SSS-GAN model, and then transmits the image. The detailed algorithm is shown in Algorithm 1.

Information extraction algorithm. After the receiver acquires the image, the image semantic label is identified by $\hat{C}$, and then, the secret message is extracted through reverse mapping. The detailed algorithm is shown in Algorithm 2.

Algorithm 1 Stego image generation

Input: secret information S; random noise z_j
Output: *stego*

train the model; obtain G and $\hat{C}$ networks;
n = ceil(length(S)/m); # divide secret information S into n segments with length of m-bit;
Map each information segment to one kind of semantic labels c_j;
for j = 1 to n **do**
    x_j = Contact(z_j, c_j); # connect z_j and c_j as a local variable x_j;
    $stego_j = G(x_j)$; # put x_j into G network;
    insert $stego_j$ into *stego*;
end for
return *stego*

Algorithm 2 Secret information extraction

Input: *stego*
Output: S

Obtain $\hat{C}$ network;
n = *length*(*stego*);
for j = 1 to n **do** # loop will iterate for all stego images
    $\hat{c}_j = \hat{C}(stego_j)$; # put $stego_j$ into $\hat{C}$ network
    recover secret data $\hat{s}_j$ from $\hat{c}_j$ according to the reverse mapping rules;
    insert $\hat{s}_j$ into S;
end for
return S

4 Experiments

In this section, we introduce our experimental details and results. The experiments consist of two parts. First, the SSS-GAN model is trained on three different datasets to verify the feasibility of our method. Then, we evaluate our model on three axes: security, capacity, and robustness.

4.1 Training of the SSS-GAN

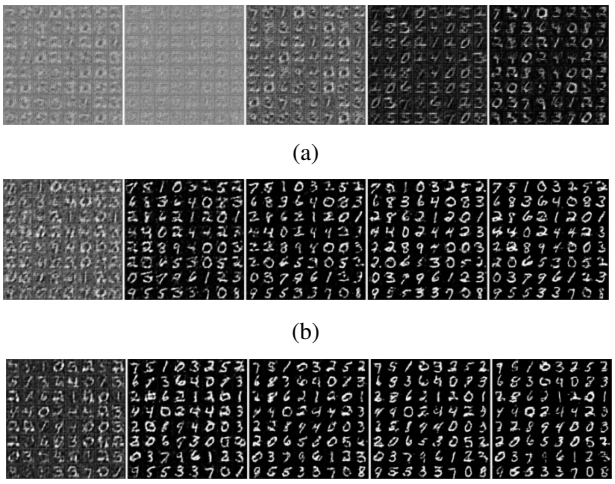
We train the SSS-GAN model on three datasets: the MNIST dataset[27] contains 60 000 handwritten digits of grayscale images, the image size is 28×28, and it is divided into 10 categories according to different numbers; the CIFAR-10 dataset[28] contains 50 000 color images, the image size is 32×32, and it is divided into 10 categories; the CIFAR-100 dataset[28] contains 50 000 color pictures, the picture size is also 32×32, and it is divided into 100 categories. All experiments are performed with TensorFlow on a workstation with a GTX 1080ti GPU card and 32 GB memory.

4.1.1 Experiments on MNIST

In this experiment, the feasibility of our method was tested through experimental results, and its reliability was analyzed on MNIST. In our experiment, we trained the model with 11 000 training steps for approximately 2 h. To illustrate the impact of α on model training, we trained the model under three different α values to

determine the effects on model training. The batch size (the minibatch size of the training sample) was set to 64; the dimension of the random noise z was set to 100 (z is sampled from a uniform distribution $(-1, 1)$). For training, we used the Adam optimizer. The learning rate was 2×10^{-4} for G and 16×10^{-4} for D and C (based on previous experimental experience). The weights of each layer were initialized with the Xavier method[30]. For each training step of the model, the two loss function values L_{D_C} and L_{G_C} are counted as shown in Fig. 3. Every 100 steps, 64 labels are inputted into G, and the quality of the images generated by G is viewed, as shown in Fig. 4. The accuracy of the auxiliary classifier C was also counted, as shown in Fig. 5. The experimental results are as follows.

As shown in Fig. 3a, when the value of α was set to 0.9, the model training tended to converge, but the model ignored the quality of the generated image, such

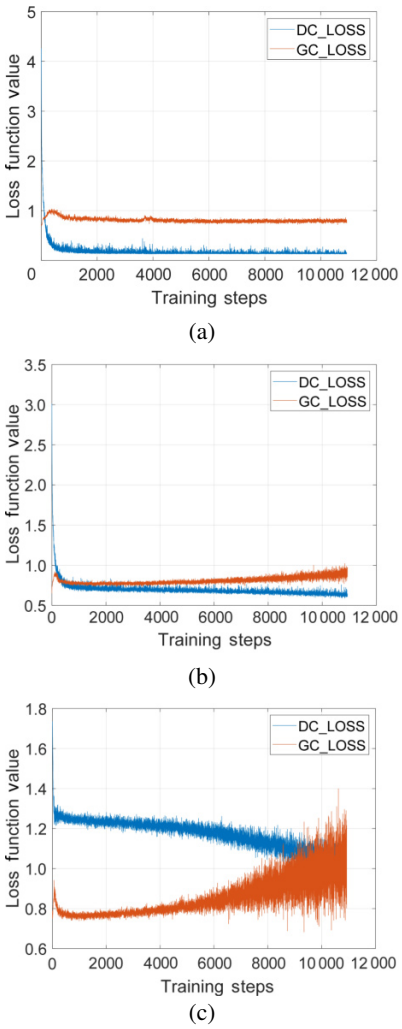


(a)

(b)

(c)

Fig. 4 Generated images during the first 500 training steps of the model under different α values: (a) $\alpha = 0.9$; (b) $\alpha = 0.5$; and (c) $\alpha = 0.1$.

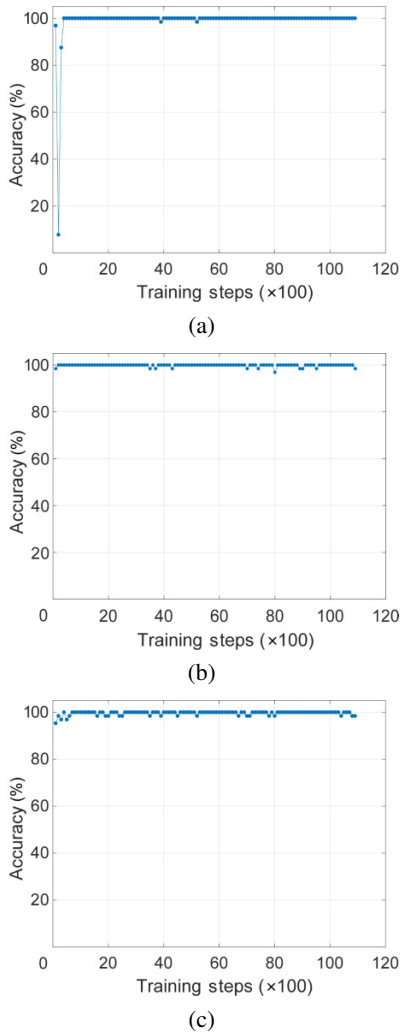


(a)

(b)

(c)

Fig. 3 SSS-GAN model training under different α values: (a) diagram $\alpha = 0.9$, (b) diagram $\alpha = 0.5$, and (c) diagram $\alpha = 0.1$.



(a)

(b)

(c)

Fig. 5 Prediction accuracy curve of auxiliary classifier during the training of the model under different α values: (a) $\alpha = 0.9$; (b) $\alpha = 0.5$; and (c) $\alpha = 0.1$.

as Fig. 4a.

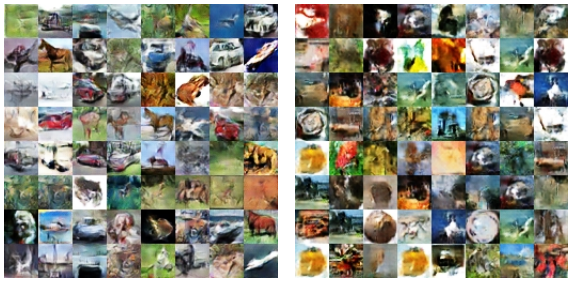
In Fig. 4, we can see that our model can generate images with high visual quality. During the training process, the message extractor quickly reaches a high recovery accuracy and remains stable, as shown in Fig. 5.

From the experimental results, SSS-GAN converged quickly on the MNIST dataset and could generate perfect images. In the experiment, G generated a batch of stego images (64 sheets) in less than 0.02 s. The message extractor could quickly reach high accuracy and remain stable, as shown in Fig. 5. The hyperparameter α can affect and balance the model training.

4.1.2 Experiments on CIFAR

In this experiment, we retrained SSS-GAN on the CIFAR-10 and CIFAR-100 training sets. Compared with the MNIST dataset, the CIFAR-10 dataset is a complex color natural image dataset, which is more complex for model training. The experiment mainly verifies the quality of the generated image and the accuracy of message extraction after training the SSS-GAN model on the complex image dataset. The model structure was unchanged, and some settings were adjusted as shown in Tables 2–4. According to the previous experiments, we used the Adam optimizer with a learning rate of 2×10^{-4} for G and 8×10^{-4} for D and C. Other settings were the same as the experiment on MNIST. We trained the SSS-GAN model on the CIFAR-10 and CIFAR-100 datasets for 15 625 training steps over approximately 6.5 hours, observing the quality of the G-generated images and the accuracy of the C, as shown in Fig. 6.

According to experimental results, the model maintained good characteristics on complex datasets. It can be seen from the curve in Fig. 7 that when the model was trained on the CIFAR-10 (CIFAR-100) dataset with 9000 steps, G could output natural images, and the accuracy of the C network was also relatively stable above 98.5%. In the experiment, G generated a batch of stego images (64 sheets) in less than 0.029 s.



(a) (b)

Fig. 6 Generated images from SSS-GAN after training 9000 steps on the CIFAR dataset: (a) samples generated by G (trained on CIFAR-10); and (b) samples generated by G (trained on CIFAR-100).

4.1.3 Further training of the extractor

We performed further training for the extractor $\hat{C}$ on the three image datasets. First, we generated a stego image dataset with G combined with 50 000 random labels (generated using random.randint in NumPy). Then, we trained $\hat{C}$ for 10 epochs (50 000 training steps per epoch) with Eq. (6) on this generated image dataset. For training, we used the Adam optimizer with a learning rate of 2×10^{-4}.

When the training was completed, we used a seed to generate 1000 random noises, inputted these noises and individual semantic labels into G to generate images of each semantic category, and then we used our trained $\hat{C}$ to perform message extraction on each group of images. We repeat the above experiment five times on various image datasets, and the results are presented in Table 5.

As shown in Table 5, the accuracy of the extractor $\hat{C}$ was stable at 100% for most of the groups, and very few

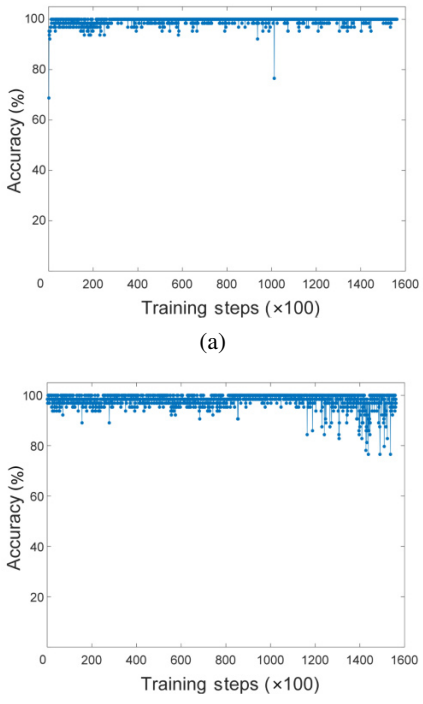


Fig. 7 Prediction accuracy curve of the auxiliary classifier during the training: (a) prediction accuracy curve of C on CIFAR-10; and (b) prediction accuracy curve of C on CIFAR-100.

Table 5 Extractor accuracy on different image datasets. (%)

Dataset	Experiment				
	1	2	3	4	5
MNIST	100	100	100	99.9	100
CIFAR-10	100	100	99.9	100	100
CIFAR-100	100	100	100	99.9	100

groups had an error rate of only 0.1%. Figure 8 shows an example of an error detection graph. It can be seen from the figure that the quality of individual samples is not good, even though people cannot distinguish them.

The errors can be attributed to the insufficient image generation capability of the G network. The weak network structure hinders the G network generation capability, resulting in image distortion generated by some individual noise. This improves with the optimization of the G network structure. In practical applications, if the sender can test the random seed that generates 100% recovery accuracy at a fixed length of the image sequence in advance and use these seeds to generate stego images, then the reliable transmission of messages in secret communication can be guaranteed.

The above experimental results show that the SSS-GAN generated images with good visual effects on all three datasets. Due to the introduction of category constraint information, our model training efficiency was higher and the information extractor performed perfectly, better than the method of Hu et al.[22]

4.2 Security analysis

Goodfellow et al.[15] proved the convergence of the GAN model in theory. Theoretically, when the GAN model converges, the generator's distribution P_g is equal to the distribution of real data P_{data}. Then,

$$D_{\text{JSD}}\left(P_{\text{data}}, P_g\right) = 0 \tag{7}$$

Furthermore, $D_{\text{JSD}}(\cdot)$ is the Jensen-Shannon divergence between two distributions and is always nonnegative and zero only when they are equal. When we use the generator to directly obtain the stego image, ideally, the distribution of stego images is indistinguishable from the real data distribution. In theory, absolute security is achieved. However, in practice, the distribution of generated stego images cannot be equal to the real data distribution. In this case, we define ε-security in the case of JS divergence between the stego distribution and the real data distribution as follows:

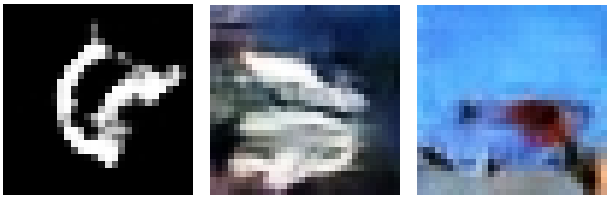
$$D_{\text{JSD}}\left(P_{\text{stego}}, P_{\text{data}}\right) < \varepsilon \tag{8}$$

where $\varepsilon > 0$. In reality, the data distribution of stego images P_{stego} is infinitely close to that of the training data P_{data} because of the limited training samples and the ability of the generator and discriminator.

Since we cannot obtain the real data distribution, the above security has only theoretical guiding significance. We still use the traditional steganalysis method to evaluate the security of steganalysis. Our method can effectively resist the detection of steganalysis algorithms based on machine learning. To prove that, experiments were conducted on the detection of stego images generated by our method via two state-of-the-art image steganalysis algorithms[8,31]. To the best of our knowledge, there are no open-source implementations of recent methods for GAN-based cover synthesis methods. As a baseline, we conducted the experiment using the same setting as the work of Hu et al.[22] We generated 5000 stego images via the G network (trained on the CIFAR-100 image dataset) of the SSS-GAN model as samples and detected them with two different steganalysis algorithms: the Color Rich Model (CRM) steganalytic features set[31] and the model in Ref. [8]. The CRM is an extension of the spatial rich model for steganalysis of color images, while the model in Ref. [8] is a CNN-based steganalysis algorithm in the spatial domain.

In this experiment, we considered two different scenarios for steganalysis algorithm training. In the first scenario, the training dataset of the steganalysis methods[8,31] consists of the cover images (5000 images from the CIFAR-100 image dataset) and the stego images (5000 images generated using embedding-based algorithms). In the second scenario, the training sets consist of the cover images (5000 images from the CIFAR-100 image dataset) and stego images (5000 images generated by the G network of our model). Compared with the experimental result in Ref. [22], the probability of being identified as a stego image is shown in Table 6.

As seen in Table 6, in the first scenario, the two



(a) (b) (c)

Fig. 8 Three error detection samples on three datasets: (a) MNIST; (b) CIFAR-10; and (c) CIFAR-100.

Table 6 Probability of identifying the stego image under detection by different algorithms.

Method	Experimental scenario	Detection probability	
		CRM	Model in Ref. [8]
Method in Ref. [22]	First scenario	0.008	0.470
	Second scenario	0.560	0.980
Ours	First scenario	0.003	0.380
	Second scenario	0.440	0.900

steganalysis algorithms had a low rate of probability of identification for both our method and the method in Ref. [22]. In the second scenario, when the generated images from our model were directly used as a training set to train the classifiers in Refs. [8, 31], good detection ability was achieved. However, it is very difficult for an adversary to acquire our image training set and G network at the same time to generate a training set for the steganalysis algorithms. The training convergence of GAN often requires human judgment. Therefore, a great deal of uncertainty exists in the model training steps. Furthermore, we can keep the training set of our method secret, thus ensuring security in terms of resisting detection by steganalysis algorithms. Therefore, we have confirmed that our approach is secure.

4.3 Capacity analysis

Our method is equivalent to directly mapping the m-bit information (we set $m = 6$ in this paper) into a small picture. Therefore, the relative hidden capacity of each picture, which is related to the number of bits contained in each stego image, can be expressed as

$$\textit{Relative capacity} = \frac{\text{Absolute capacity}}{\text{Size of stego image}} \tag{9}$$

We simply compare the steganographic capacity with some main SWM methods. The comparison results are presented in Table 7, where the second column is the absolute steganographic capacity (steganographic capacity per image), the third column is the size of the stego image, and the last column is the relative steganography capacity (bytes per pixel).

Table 7 Steganography capacities of SWM methods.

Method	Absolute capacity (bytes/image)	Image size	Relative capacity (bytes/pixel)
Method in Ref. [10]	1.125	512×512	4.77×10^{-6}
Method in Ref. [11]	3.720	$\geqslant$512×512	1.42×10^{-5}
Method in Ref. [12]	2.250	512×512	8.58×10^{-6}
Method in Ref. [32]	25∼100	480×640	$8.14\times10^{-5}\sim 3.26\times10^{-4}$
Method in Ref. [22]	37.500	64×64	9.16×10^{-3}
Ours	$\geqslant$0.750	32×32	$\geqslant 7.3\times10^{-4}$

Currently, the relative capacity of our method is over 7.3×10^{-4} bytes/pixel, which is lower than that of the method in Ref. [22], but better than that of other methods. This is because the capacity of our method depends on the number of semantic tags included in one stego image. In the future, with the number of semantic tags used in our method increasing in the stego image, the capacity of our method will significantly increase.

4.4 Robustness analysis

Since messages actually exist in the semantic information of images, the dense images using the SSS-GAN model have the robustness to resist image attacks. We tested the robustness of our method to common image attacks. We considered applying four typical image attacks. These attack conditions are listed as follows.

C_1: Brightness changes by changing the intensity of image pixels by 1.1 and 1.5 times.

C_2: Gaussian noise addition (variance 0.01).

C_3: Salt noise addition (density 0.05).

C_4: JPEG compression with varying quality facts (q.f. 50, q.f. 70 and q.f. 90).

We used the G network to generate three groups of stego images on the CIFA-100 dataset (5000 per group) and applied the four typical methods to attack each group of images. Then, we used $\hat{C}$ to test the accuracy of the message extraction after the attack. A group of results is presented in Table 8.

From the experimental results, our method is robust to all four attacks, especially JPEG compression and brightness changes. The model has no errors at a JPEG compression factor of 90, and brightness changes at an intensity of 1.1 times because our method relies on the recognition of image semantic labels by neural networks. The neural networks have good fault tolerance. When inputting fuzzy or incomplete

Table 8 Extraction accuracy of extractor $\hat{C}$ for the attacked stego images.

Test group	Accuracy (%)						
	C_1		C_2	C_3	C_4		
	1.1 times	1.5 times			q.f. 50	q.f. 70	q.f. 90
Group 1	100	98.00	98.99	99.96	98.52	99.89	100
Group 2	100	98.99	99.67	99.61	99.59	100.00	100
Group 3	100	98.00	99.67	98.52	99.85	99.89	100
Average	100	98.33	99.51	99.30	99.42	99.82	100

information, a suboptimal approximate solution can be given to achieve correct identification of incomplete input information.

5 Conclusion

This paper proposes a novel SCS method using a deep-generative model. The SSS-GAN model can directly generate stego images through the auxiliary semantic information. Due to the introduction of category constraint information, the model can be quickly trained to converge on different image datasets and obtain an extractor that can accurately extract messages. In other words, our method constructs the map relationship between secret information and image semantic information. Moreover, since messages actually exist in the semantic information of images, the dense images using the SSS-GAN model have the robustness to resist image attack. However, due to the limitation of the image dataset and the generative model performance, some generated images are not sufficiently natural. At present, we have to admit that the method seems to require complex calculations by neural networks when generating stego images. Furthermore, our model uses only the category of image semantic information for image generation, and the steganographic capacity is not high, which decreases the meaning of these types of steganography. In future work, we will attempt to design a new generative model to enhance the quality of the generated image and use the multiple semantic features of the image to improve the steganographic capacity of the method.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (NSFC) (Nos. 61872384 and 61672090)

References

[1] T. Pevný, T. Filler, and P. Bas, Using high-dimensional image models to perform highly undetectable steganography, in *Proc. International Workshop on Information Hiding*, Calgary, Canada, 2010, pp. 161–177.

[2] V. Holub and J. Fridrich, Designing steganographic distortion using directional filters, in *Proc. IEEE International Workshop on Information Forensics and Security*, Tenerife, Spain, 2012, pp. 234–239.

[3] V. Holub, J. Fridrich, and T. Denemark, Universal distortion function for steganography in an arbitrary domain, *EURASIP Journal on Information Security*, vol. 2014, pp. 1–13, 2014.

[4] K. Niu, X. Yang, and Y. Zhang, A novel video reversible data hiding algorithm using motion vector for H.264/AVC, *Tsinghua Science and Technology*, vol 22, no. 5, pp. 489–498, 2017.

[5] Y. Zhang, M. Zhang, X. Yang, D. Guo, and L. Liu, Novel video steganography algorithm based on secret sharing and error-correcting code for H.264/AVC, *Tsinghua Science and Technology*, vol 22, no. 2, pp. 198–209, 2017.

[6] J. Zeng, S. Tan, B. Li, and J. Huang, Large-scale JPEG steganalysis using hybrid deep-learning framework, *IEEE Transactions on Information Forensics and Security*, vol 13, no. 5, pp. 1200–1214, 2016.

[7] G. Xu, Deep convolutional neural network to detect J-UNIWARD, in *Proc. The 5th ACM Workshop on Information Hiding and Multimedia Security*, Philadelphia, PA, USA, 2017, pp. 67–73.

[8] J. Ye, J. Ni, and Y. Yi, Deep learning hierarchical representations for image steganalysis, *IEEE Transactions on Information Forensics and Security*, vol 12, no. 11, pp. 2545–2557, 2017.

[9] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge, UK: Cambridge University Press, 2010.

[10] Z. Zhou, H. Sun, R. Harit, X. Chen, and X. Sun, Coverless image steganography without embedding, in *Proc. International Conference on Cloud Computing and Security*, Nanjing, China, 2015, pp. 123–132.

[11] Z. Zhou, Y. Cao, and X. Sun, Coverless information hiding based on bag-of-words model of image, (in Chinese), *Journal of Applied Sciences*, vol. 34, no. 5, pp. 527–536, 2016.

[12] S. Zheng, L. Wang, B. Ling, and D. Hu, Coverless information hiding based on robust image hashing, in *Proc. International Conference on Intelligent Computing*, Liverpool, UK, 2017, pp. 536–547.

[13] J. Xu, X. Mao, X. Jin, A. Jaffer, S. Lu, L. Li, and M. Toyoura, Hidden message in a deformation-based texture, *The Visual Computer*, vol. 31, no. 12, pp. 1653–1669, 2015.

[14] K. Wu and C. Wang, Steganography using reversible texture synthesis, *IEEE Trans. Image Process.,* vol. 24, no. 1, pp. 130–139, 2015.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, in *Proc. Neural Information Processing Systems 2014*, Montreal, Canada, 2014, pp. 2672–2680.

[16] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., Photo-realistic single image super-resolution using a generative adversarial network, in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 4681–4690.

[17] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, Generative adversarial text to image synthesis, https://arxiv.org/abs/1605.05396, 2016.

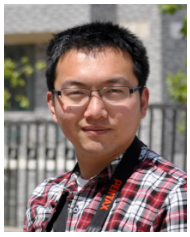
[18] D. Volkhonskiy, I. Nazarov, B. Borisenko, and E. Burnaev, Steganographic generative adversarial networks, https://arxiv.org/abs/1703.05502, 2017.

[19] W. Tang, S. Tan, B. Li, and J. Huang, Automatic

steganographic distortion learning using a generative adversarial network, *IEEE Signal Processing Letters,* vol. 24, no. 10, pp. 1547–1551, 2017.
[20] J. Yang, K. Liu, X. Kang, E. Wong, and Y. shi, Spatial Image Steganography based on generative adversarial network, https://arxiv.org/abs/1804.07939v1, 2018.
[21] J. Hayes and G. Danezis, Generating steganographic images via adversarial training, in *Proc. Neural Information Processing Systems 2017*, Long Beach, CA, USA, 2017, pp. 2672–2680.
[22] D. Hu, L. Wang, W. Jiang, S. Zheng, and B. Li, A novel image steganography method via deep convolutional generative adversarial networks, *IEEE Access*, vol. 6, pp. 38303–38314, 2018.
[23] J. Liu, T. Zhou, Z. Zhang, Y. Ke, Y. Lei, M. Zhang, and X. Yang, Digital cardan grille: A modern approach for information hiding, https://arxiv.org/abs/1803.09219, 2018.
[24] T. Filler, J. Judas, and J. Fridrich, Minimizing embedding impact in steganography using trellis-coded quantization, in *Proc. Media Forensics and Security II*, San Jose, CA, USA, 2010, p. 754105.
[25] T. Fang, M. Jaggi, and K. Argyraki, Generating steganographic text with LSTMs, https://arxiv.org/abs/1705.10742v1, 2017.
[26] Z. Yang, X. Du, Y. Tan, Y. Huang, and Y. Zhang, Aag-stega: Automatic audio generation-based steganography, https://arxiv.org/abs/1809.03463, 2018.
[27] The MNIST database of handwritten digits, http://yann.lecun.com/exdb/mnist/, 2019.
[28] The CIFAR-10 and CIFAR-100 dataset, http://www.cs.toronto.edu/ kriz/CIFAR.html, 2019.
[29] A. Odena, C. Olah, and J. Shlens, Conditional image synthesis with auxiliary classifier GANs, in *Proc. the 34-th International Conference on Machine Learning*, Sydney, Australia, 2017, pp. 2642–2651.
[30] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *Proc. the Thirteenth International Conference on Artificial Intelligence and Statistics*, Sardinia, Italy, 2010, pp. 249–256.
[31] M. Goljan, J. Fridrich, and R. Cogranne, Rich model for steganalysis of color images, in *Proc. 2014 IEEE International Workshop on Information Forensics and Security (WIFS)*, Atlanta, GA, USA, 2014, pp. 185–190.
[32] H. Otori and S. Kuriyama, Texture synthesis for mobile data communications, *IEEE Computer Graphics and Applications*, vol. 29, no. 6, pp. 74–81, 2009.

Zhuo Zhang received the MS degree from National University of Defense Technology, China, in 2010. He is currently a PhD candidate at Rocket Force University of Engineering, Xi'an, China. He concurrently is a lecturer at Engineering University of PAP, Xi'an, China. His research interests include deep learning and information hiding.

Guangyuan Fu received the MS degree from Southwest Jiaotong University, China, in 1993, and the PhD degree from The Second Artillery Engineering College, China, in 2004. He is currently a professor and a PhD advisor at Rocket Force University of Engineering, Xi'an, China. His research interests include computer vision and pattern recognition.



Jia Liu received the PhD degree from Shanghai Jiao Tong University, China, in 2012. He works as an associate professor at the Key Laboratory of Network and Information Security of PAP, Xi'an, China. His research interests include machine learning, intelligent information processing, and information hiding.

Rongrong Ni received the BS degree and the PhD degree from Beijing Jiaotong University (BJTU), Beijing, China, in 1998 and 2005, respectively. Since 2005, she has been the faculty of the School of Computer and Information Technology, the Institute of Information Science, BJTU, where she is a professor since 2013. She is currently a visiting scholar at the University of British Columbia, Canada. Her current research interests include image processing, data hiding and digital forensics, pattern recognition, and computer vision.

Xiaoyuan Yang received the MS degree from Xidian University in 1991. He is currently a professor and a PhD advisor at the Key Laboratory of Network and Information Security of PAP, Xi'an, China. His research interests include cryptography and information security.