Steganography Embedding Cost Learning With Generative Multi-Adversarial Network

Dongxia Huang, Weiqi Luo, *Senior Member, IEEE*, Minglin Liu, Weixuan Tang, and Jiwu Huang, *Fellow, IEEE*

***Abstract*—Since the generative adversarial network (GAN) was proposed by Ian Goodfellow et al. in 2014, it has been widely used in various fields. However, there are only a few works related to image steganography so far. Existing GAN-based steganographic methods mainly focus on the design of generator, and just assign a relatively poorer steganalyzer in discriminator, which inevitably limits the performances of their models. In this paper, we propose a novel Steganographic method based on Generative Multi-Adversarial Network (Steg-GMAN) to enhance steganography security. Specifically, we first employ multiple steganalyzers rather than a single steganalyzer like existing methods to enhance the performance of discriminator. Furthermore, in order to balance the capabilities of the generator and the discriminator during training stage, we propose an adaptive way to update the parameters of the proposed GAN model according to the discriminant ability of different steganalyzers. In each iteration, we just update the poorest one among all steganalyzers in discriminator, while update the generator with the gradients derived from the strongest one. In this way, the performance of generator and discriminator can be gradually improved, so as to avoid training failure caused by gradient vanishing. Extensive comparative results show that the proposed method can achieve state-of-the-art results compared with the traditional steganography and the modern GAN-based steganographic methods. In addition, a large number of ablation experiments verify the rationality of the proposed model.**

***Index Terms*—Steganography, generative adversarial network, steganalysis.**

I. Introduction

IMAGE steganography aims at hiding a secret message within a digital image in an imperceptible manner. Recently, most steganographic methods have been designed under the distortion minimization framework [1]. Within this framework, the design of the embedding cost is a critical aspect. Embedding cost refers to the level of distortion or disruption caused by concealing a secret message within an image's embedding unit, such as a pixel or a DCT coefficient. Typically, regions with complex image textures exhibit relatively low embedding costs, whereas regions with simple and flat textures have higher embedding costs. These varying embedding costs significantly impact the security performance of image steganography. Once the embedding cost is defined, the syndrome trellis code (STC) [2] is then utilized to embed the secret message and minimize the overall sum of embedding costs. Until now, many handcrafted steganographic methods have been proposed, such as HUGO [3], WOW [4], HILL [5], S-UNIWARD [6], MiPOD [7] and CMD [8]. For instance, WOW defines the spatial additive steganographic distortion with several directional high-pass filters. HILL designs a high-pass filter and two low-pass filters to make the distortion distribution more concentrated in texture areas. The distortion function of UNIWARD is defined as the sum of the relative changes of the wavelet coefficients obtained by three directional filters. CMD assumes that embedding modifications in textured regions are locally toward the same direction and dynamically adjusts the embedding cost to reduce the risk of detection by steganalysis.

On the other hand, image steganalysis tries to identify those stego images with secret message. Some traditional steganalytic methods, such as SPAM [9] and SRM [10], usually transfer the input image into different residual domains, and then extract high-dimensional statistical features to capture the embedding modifications introduced by some steganographic methods, finally employ ensemble classifiers [11] for binary classification. Recently, some steganalytic methods based on convolutional neural networks (CNNs) [12] have been proposed, such as Xu-Net [13], Yedroudj-Net [14], SRNet [15] and CovNet [16], and they have achieved great improvements compared with traditional methods.

Steganography and steganalysis are opposed and reinforcing mutually. To enhance the security performance, some deep learning based steganographic methods have been proposed recently. These methods can be divided into two categories, that is, methods based on adversarial attack [17], [18], [19], [20], [21], [22] and methods based on GAN [23], [24], [25]. Inspired by fast gradient sign method [26], some methods based on adversarial attack try to adjust the original embedding costs according to the gradients derived from some pre-trained CNN steganalyzers, such as ADV-EMB [17], JS-IAE [18],

Manuscript received 20 January 2023; revised 21 June 2023; accepted 13 September 2023. Date of publication 27 September 2023; date of current version 16 November 2023. This work was supported in part by NSFC under Grant 61972430 and Grant U19B2022. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Pedro Comesana. *(Corresponding author: Weiqi Luo.)*

Dongxia Huang and Weiqi Luo are with the Guangdong Key Laboratory of Information Security Technology and the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China (e-mail: huangdx23@mail2.sysu.edu.cn; luoweiqi@mail.sysu.edu.cn).

Minglin Liu is with the School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou 450001, China (e-mail: liuminglin@zzu.edu.cn).

Weixuan Tang is with the Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou 510006, China (e-mail: tweix@gzhu.edu.cn).

Jiwu Huang is with the Guangdong Key Laboratory of Intelligent Information Processing and the Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China (e-mail: jwhuang@szu.edu.cn).

Digital Object Identifier 10.1109/TIFS.2023.3318939

MAE [19], SGS [20], SPS-ENH [21], USGS [22], min max strategy [27] and Backpack [28], [29]. One example is the Backpack algorithm [28], [29], which iteratively attacks target steganalyzers and optimize the embedding cost. In Backpack algorithm, the min max strategy [27] is combined to find the optimal stego images during iterations.

Note that the aforementioned methods based on adversarial attacks can be considered as improved strategies, as they rely on existing steganographic methods for calculating initial embedding costs and pre-trained CNN-based steganalyzers. On the other hand, GAN-based steganographic methods have the ability to automatically generate highly secure stego images from scratch by iteratively training the generator and the discriminator. For instance, ASDL-GAN [23] is the first method to introduce GAN model into image steganography to generate embedding probability. However, the performance of ASDL-GAN is worse than the traditional steganography HILL. Based on ASDL-GAN, UT-GAN [24] improves the generator, embedding simulator and discriminator of ASDL-GAN, which exceeds the performance of the traditional steganographic methods. SPAR-RL [25] is an enhancement strategy based on existing GAN steganographic methods. It introduces modules of GAN-based steganographic methods into reinforcement learning framework, and improves the performance of existing methods. Additionally, another class of GAN-based steganography is demonstrated in [30]. It comprises three subnetworks for generating stego images, recovering secret messages from stego images, and discriminating between cover and stego images, respectively. The primary objective of [30] is to evaluate the correlation between information extraction errors and steganographic security, which significantly differs from the cost learning methods.

Note that existing steganographic methods based on GAN mainly focus on the design of generator, especially for the network for generating the probability map and the embedding simulator. In addition, they usually assign a relatively poorer steganalyzer in discriminator, and do not consider the relationship between the generator and the discriminator during training stage. In this paper, we propose a new GAN-based steganographic method to enhance the security performance. Unlike related methods which mainly focus on the design of generator, we first employ multiple steganalyzers rather than a single steganalyzer in discriminator, and then propose an adaptive way to update parameters of the generator and discriminator in order to balance the capabilities of the generator and the discriminator. In this way, richer gradient information can be introduced into the generator to improve the security performance of the model. Meanwhile, the performance of the generator and discriminator can be gradually improved, thus avoiding training failure due to gradient vanishing. Overall, the contributions of this paper are summarized as follows:

1) We analyze the limitation of existing steganographic methods based on GAN, and point out that the single and the poor steganalyzer employed in the existing discriminators and the update stratgies would limit the security performances in existing methods;
2) We propose a novel steganographic method based on GAN to enhance steganography security. The main idea of our method is to introduce multiple steganalyzers into the discriminator, and update the parameters of the GAN model adaptively according to the performance of the steganalyzers during training stage;
3) We provide extensive comparative results to show that the proposed method can outperform existing steganographic methods, and achieve state-of-the-art results. In addition, we provide many ablation experiments to verify the rationality of the proposed model.

The rest of the paper is organized as follows. Section II gives a brief overview of related works on generative adversarial network and GAN-based steganographic methods. Section III describes the proposed method in detail. Section IV shows comparative results and analysis. Finally, the concluding remarks of this paper and future work are given in Section V.

II. Related Work

In this section, we firstly describe the framework of the generative adversarial network (GAN), and then show some typical image steganographic methods based on GAN.

A. Generative Adversarial Network

Generative adversarial network [31] is first proposed by Ian Goodfellow et al. in 2014, which is a model for generating tasks through the minimax game between two players. On the one hand, one player (i.e., the generator) tries to generate fake samples similar to the real data. Specifically, the generator $G(z;\theta_g)$ is a differentiable function represented by a multi-layer perceptron with the parameter θ_g. The input of the generator $G(z;\theta_g)$ is the noise z which follows a simple distribution $p_z(z)$, while the output of the generator is a fake sample which is close to the real data distribution $p_{data}(x)$. On the other hand, the other player (i.e., the discriminator) tries to distinguish between the fake samples generated by the generator and real data. Specifically, the discriminator $D(x;\theta_d)$ is another multi-layer perceptron with the parameter θ_d and outputs a scalar in the range of $[0, 1]$ representing the probability that the input x comes from the real data. The model finally reaches the Nash equilibrium state through mutual confrontation and learning between the generator and the discriminator. In a word, D and G play the minimax game with the value function $V(D, G)$ as the following equation:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{1}$$

where $\mathbb{E}$ denotes the mathematical expectation about the distribution of the real data x and the noise z. x denotes a real sample that follows the data distribution $p_{data}(x)$ and the label of x is 1, $G(z)$ denotes a fake sample generated by the noise z through the generator G, and the label of $G(z)$ is 0. In the minimax game, we train D to maximize $[\log D(x) + \log(1-D(G(z)))]$ to increase the probability of assigning the correct label to both real samples and fake samples, and simultaneously train G to minimize $\log(1-D(G(z)))$ to make the generated fake samples as close to the real data distribution.

Recently, GAN model is widely used in various applications, especially in image synthesis [32], [33], [34], [35].

B. Steganographic Methods Based on GAN

Steganography aims to embed secret message into cover images in an imperceptible manner, while the steganalysis tries to identify those resulting stego images after data embedding with steganography from the cover images. Steganography and steganalysis confront and facilitate each other, like the generator and the discriminator in GAN. Thus, several steganographic methods employ the GAN architecture to enhance steganography security. In the following, we will give a brief descriptions about three typical methods - ASDL-GAN, UT-GAN and SPAR-RL, separately.

1) ASDL-GAN: ASDL-GAN first introduces the GAN into image steganography for embedding cost learning. In ASDL-GAN, the generator contains a Ker-Bohme (KV) high-pass filter [36] followed by 25 groups of convolutional layers, and the discriminator is Xu-Net. The KV high-pass filter kernel is defined as follows:

$$K_{kv} = \begin{pmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{pmatrix} \tag{2}$$

ASDL-GAN employs a pre-trained Ternary Embedding Simulator (TES) subnetwork as the embedding simulator in order to map the probability to the modification and aid in the back-propagation process within GANs. The loss function of generator is directly related to the undetectability of the steganalyzer in the discriminator, while the loss function of discriminator is defined as the binary cross entropy of classification. Based on the experiments in [23], however, the security performance of ASDL-GAN is far from satisfactory, and it is relatively poorer than the traditional steganographic method - HILL.

2) UT-GAN: UT-GAN utilizes a U-Net [37] network in generator, and employs the Xu-Net with 6 high-pass filters in discriminator. In addition, it designs a differentiable Double-Tanh function without pre-training to replace TES subnetwork as the embedding simulator, which achieves much shorter training time. The loss functions in UT-GAN are defined in the same way as ASDL-GAN. Experimental results show that UT-GAN can outperform ASDL-GAN and HILL [24], [25].

3) SPAR-RL: SPAR-RL (Steganographic Pixel-wise Actions and Rewards with Reinforcement Learning) is an enhancement strategy for existing GAN-based steganographic methods. In SPAR-RL, the generator network in GAN-based methods is regarded as the policy network and the discriminator network as the environment network. The policy network formulates a policy to create stego images that are meant to deceive the environment network. The environment network attempts to distinguish between cover image and generated stego images, thereby judging the effectiveness of the policy network's strategies. In this method, a discrete stair function is utilized rather than continuous embedding simulator like related methods. In addition, the loss function of the policy network is computed by the pixel-level reward assigned to each embedded unit from the environment network. Experiments show that SPAR-RL can improve the security performance of existing methods (i.e., ASDL-GAN and UT-GAN).

Please note that all existing GAN-based steganographic methods just employ a single steganalyzer (i.e., Xu-Net) in discriminator, which inevitably limits the performance of generator. Furthermore, they mainly focus on the design of generator (i.e., the network for generating probability map and the embedding simulator), but not on the balance between generator and discriminator during training stage, which will significantly affect the performance of the GAN model.

III. Proposed Method

As illustrated in Fig. 1, the network architecture of the proposed method includes two important modules, that is, the generator and the discriminator, which will be described in the two following subsections. Finally, we will show the design of loss functions in the proposed model.

A. Design of Generator

The generator aims to generate the corresponding stego image according to an input cover image and the secret message. To this end, the generator includes two steps:

1) Step #1: Generating Probability Map: This step tries to transform an input cover image $\mathbf{X} = (x_{i,j})^{\mathrm{H}\times\mathrm{W}}$ of size $\mathrm{H}\times\mathrm{W}$ into the embedding probability $\mathbf{P} = (p_{i,j})^{\mathrm{H}\times\mathrm{W}}$, whose element $p_{i,j}$ denotes the probability that the pixel value $x_{i,j}$ is modified by ± 1 due to message embedding with the index (i, j). In the proposed method, we limit that the probability of +1 (i.e., p^{+1}) is equal to that of -1 (i.e., p^{-1}) for each pixel value $x_{i,j}$. It can be expressed as

$$p^{+1}_{i,j} = p^{-1}_{i,j} = p_{i,j}/2 \tag{3}$$

As illustrated in Fig. 2, we employ a U-Net based architecture to complete this image-to-image (i.e., $\mathbf{X}$ to $\mathbf{P}$) task. It contains 15 blocks and a deconvolution layer. The first 8 blocks downsample the image, and the last 7 blocks upsample the image. The output of the i-th block ($i = 1, 2, \ldots, 7$) and the output of the $(16 - i)$-th block will be concatenated as the input to the $(17 - i)$-th block. As shown in Fig. 2(b), each block includes two convolution layers followed by batch normalization and LeakyReLU. Note that the strides are 1 and 2 for the first and the second convolution separately.

2) Step #2: Simulating Message Embedding: This step tries to simulate the STC to obtain the stego image from the resulting probability map $\mathbf{P}$ in Step #1 by the STC simulator. To this end, we firstly generate a random noise $\mathbf{R} = (r_{i,j})^{\mathrm{H}\times\mathrm{W}}$, where $r_{i,j}$ is independent and identically distributed (i.i.d.) for all i and j and each element is drawn from a uniform distribution on the interval $[0, 1]$. By comparing the $p_{i,j}$ and $r_{i,j}$, the modification map $\mathbf{M} = (m_{i,j})^{\mathrm{H}\times\mathrm{W}}$ in the STC

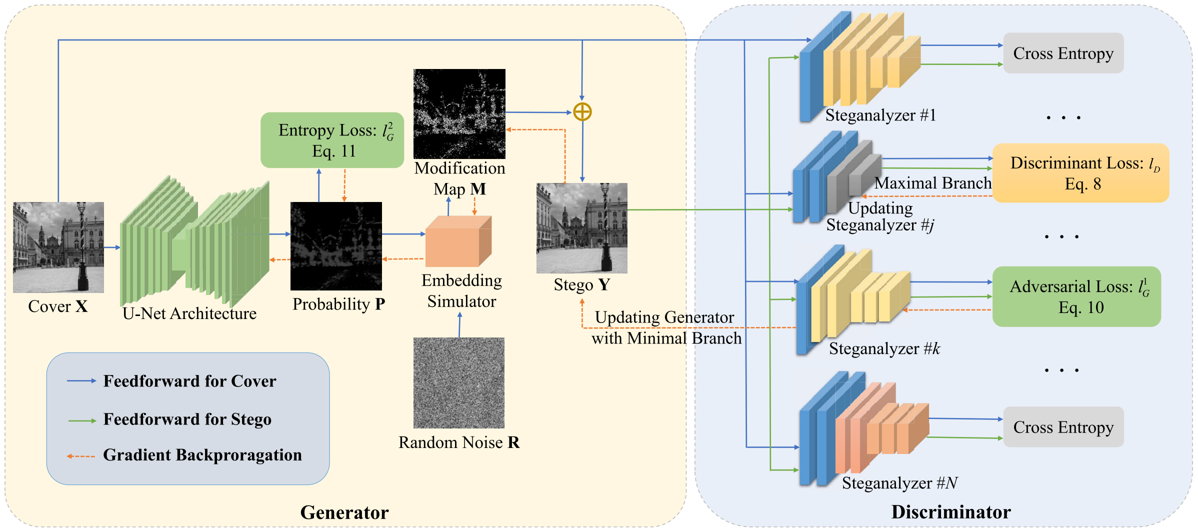
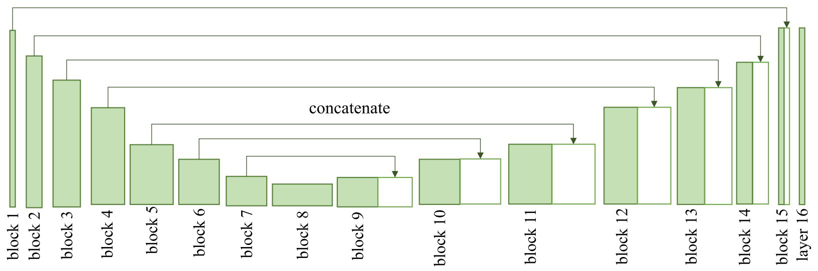
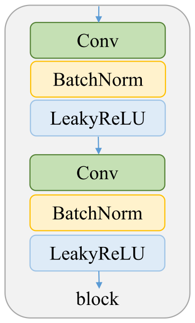


Fig. 1. The network architecture of the proposed method.



(a) U-Net based network architecture.



(b) The proposed block.

Fig. 2. The proposed U-Net based network architecture to produce the probability map.

simulator is defined as follows:

$$m_{i,j} = \begin{cases} -1 & r_{i,j} < p_{i,j}^{-1} \\ +1 & r_{i,j} > 1 - p_{i,j}^{+1} \\ 0 & otherwise \end{cases} \tag{4}$$

Since the above stair function is not differentiable, we employ the Double-Tanh function [24] as the embedding simulator to get the modification map $\mathbf{M} = (m_{i,j})^{\mathrm{H}\times\mathrm{W}}$ during training stage. It substitutes the discrete stair function with a differentiable approximation, enable the back-propagation process in GANs. The Double-Tanh function is defined as follows:

$$m_{i,j} = \frac{1}{2}\tanh(\lambda(p_{i,j}^{+1} - r_{i,j})) - \frac{1}{2}\tanh(\lambda(p_{i,j}^{-1} - (1 - r_{i,j}))) \tag{5}$$

where the $m_{i,j}$ in the interval $[-1, +1]$ denotes the modification for simulation, the parameter λ controls the accuracy of the simulation, we set $\lambda = 60$ as it did in [24]. Finally, we get the stego image by $\mathbf{Y} = \mathbf{X} + \mathbf{M}$.

Note that when the training is completed, the embedding cost ρ of the input image $\mathbf{X}$ can be calculated as follows [7]:

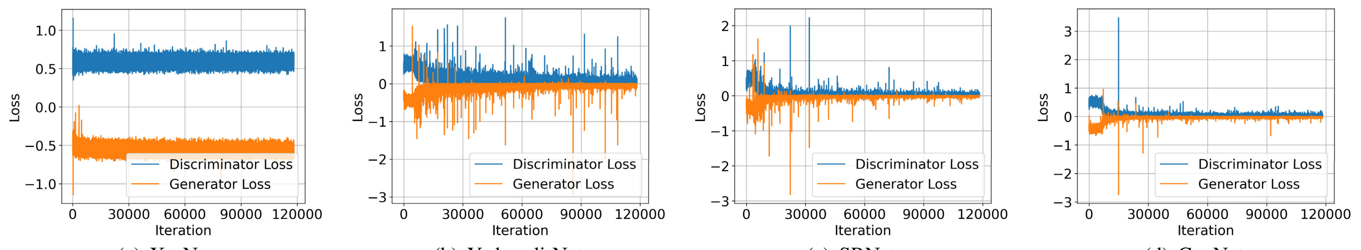
$$\begin{cases} \rho_{i,j}^{+1} = \ln(\frac{1}{p_{i,j}^{+1}} - 2) \\ \rho_{i,j}^{-1} = \ln(\frac{1}{p_{i,j}^{-1}} - 2) \\ \rho_{i,j}^{0} = 0 \end{cases} \tag{6}$$

Based on the resulting embedding cost, we switch to using STC[1] (rather than the embedding simulator) to complete the message embedding and get the actual stego image $\mathbf{Y}$, aiming to minimize the overall sum of embedding costs.

B. Design of Discriminator

The discriminator tries to determine whether an input image is a cover image $\mathbf{X}$ or a stego image $\mathbf{Y}$ created by the generator, and provides helpful information to update the network parameters both in generator and discriminator. Note that existing GAN-based steganographic methods [23], [24], [25] only utilize a single steganalyzer (i.e., Xu-Net) in discriminator, which

[1]Available at: http://dde.binghamton.edu/download/syndrome/



(a) Xu-Net (b) Yedroudj-Net (c) SRNet (d) CovNet

Fig. 3. The discriminator loss v.s. generator loss with increasing the training iterations using four different steganalyers in UT-GAN separately. Note that the latter three steganalyzers are relatively stronger than the first one (i.e., Xu-Net). In our experiments, the total number of iterations is 119, 952.

will limit the the final performances of the existing models. The main reason is that based on recent literatures [15], [38] in image steganalysis, Xu-Net is relatively poorer than some modern steganalyzers. The poorer performance of discriminator will inevitably restrict the performance of generator since the inaccurate decision bound of the discriminator will provide limited feedback for the generator as described in [39] and [40]. In addition, some steganographic methods [19], [22] based on adversarial attack also show that the steganography security can be further improved when a strong steganalyzer is served as the targeted steganalyzer. Thus, a direct and simple idea is utilizing a modern steganalyzer to enhance the performance of discriminator. For this reason, we replace the discriminator (i.e., Xu-Net) in the UT-GAN with three advanced steganalyzers (i.e., Yedroudj-Net, SRNet and CovNet) individually, while maintaining the core framework of UT-GAN intact. Fig. 3 shows the discriminator loss and the generator loss as the number of training iterations increases using three mentioned steganalyzers separately in the discriminator. From Fig. 3, we observe that the loss of generator and discriminator can maintain dynamic balance for Xu-Net (see Fig. 3-(a)) during training stage. In this case, we finally obtain a relatively better generator. For three other stronger steganalyzers (see Fig. 3-(b), Fig. 3-(c), and Fig. 3-(d)), however, the discriminator loss converges to 0 rapidly, meaning that the discriminator can identify those generated stego images from cover images accurately. In this case, the gradient of the discriminator will vanish, and thus it can not provide any effective information for updating the generator.

Based on above experiments and analysis, it can be concluded that we should introduce stronger steganalyzers to enhance the performance of discriminator, and simultaneously design a new training strategy to keep the dynamic balance between the discriminator and generator. To this end, we first utilize multiple steganalyzers in the discriminator as shown in Fig. 1. Then we employ an adaptive way to update the parameters in the proposed GAN model in order to avoid the gradient vanishing due to the rapid convergence of the strong discriminators as illustrated in Fig. 3-(b,c,d). Specifically, the proposed discriminator includes the three following steps during training stage.

1) Step #1: Calculating Cross Entropy: For each steganalyzer D_i in discriminator, the input is an image, either the cover image $\mathbf{X}$ or the corresponding stego $\mathbf{Y}$ generated by the generator. The output of D_i (i.e., $\mathrm{D}_i(\mathbf{X})$ or $\mathrm{D}_i(\mathbf{Y})$) is the discriminant result of the softmax layer in steganalytic network for the input image, represented as a 2-D normalized vector. The discriminant performance of the steganalyzer D_i is measured by the binary cross entropy E_i, defined as follows:

$$E_i = -z_0 \log(\mathrm{D}_i(\mathbf{X})) - z_1 \log(\mathrm{D}_i(\mathbf{Y})) \tag{7}$$

where the $\mathrm{D}_i(\mathbf{X})$ and $\mathrm{D}_i(\mathbf{Y})$ denote the i-th discriminant results of the cover image and the stego image respectively, $i = 1, 2, \ldots N$; The vectors $z_0 = (1 \quad 0)^T$ and $z_1 = (0 \quad 1)^T$ are the ground-truth labels for $\mathbf{X}$ and $\mathbf{Y}$. In the proposed model, the discriminant performance of the i-th steganalyzer D_i is judged by the binary cross entropy E_i. The larger the binary cross entropy, the weaker the steganalyzer. Assume that the j-th steganalyzer is the weakest one, while k-th steganalyzer is the strongest one in some iteration. Note that the discriminant performance of a steganalyzer may change for different iterations.

2) Step #2: Updating Discriminator: To enhance the discriminant performance of the discriminator, we just update the weakest steganalyzer (i.e., the j-th steganalyzer) in discriminator in each iteration, while preserve other $N-1$ stronger steganalyzers unchanged. In this way, the steganalyzers can be enhanced in a slow and gradual manner.

3) Step #3: Updating Generator: To enhance the generation performance of the generator, we use the strongest steganalyzer (i.e., the k-th steganalyzer) to guide generator update in each iteration. The main reason is that the the strongest steganalyzer can provide relatively more effective feedback to the generator.

Based on our comparative experiments in Section IV, we employ two steganalyzers (i.e., Xu-Net and Yedroudj-Net) within the discriminator of the proposed GAN model in this paper, and achieve a good trade-off between the steganography security and training efficiency.

C. Design of Loss Function

There are two loss functions in the proposed model, that is, the discriminator loss l_D and the generator loss l_G, which will be described separately below.

1) Discriminator Loss: The steganalyzer in discriminator aims to determine whether an image is a cover or a stego. As described previously, we employ multiple steganalyzers in the proposed discriminator, and update the weakest one in each iteration. Therefore, the discriminator loss l_D is defined as follows:

$$l_D = \max_{i \in \{1,2,\ldots N\}} \{-z_0 \log(\mathrm{D}_i(\mathbf{X})) - z_1 \log(\mathrm{D}_i(\mathbf{Y}))\} \tag{8}$$

Algorithm 1 Training Steps of Steg-GMAN

```
Input: cover image $\mathbf{X}$, random noise $\mathbf{R}$, $epochs$, generator
    loss parameter $\alpha$ and $\beta$, number of discriminators $N$, label
    of cover $z_0$, label of stego $z_1$, learning rate $\eta$
Output: generator parameter $\theta_g$
 1: Initialize the generator and the discriminator
 2: for $epoch$ in $\{1, \ldots, epochs\}$ do
 3:     $\mathbf{P} \leftarrow G(\mathbf{X}; \theta_g)$
 4:     Compute $\mathbf{M}$ by Double-Tanh function
 5:     $\mathbf{Y} \leftarrow \mathbf{X} + \mathbf{M}$
 6:     for $i$ in $\{1, \ldots, N\}$ do
 7:         $E_i \leftarrow -z_0 \log(\mathrm{D}_i(\mathbf{X})) - z_1 \log(\mathrm{D}_i(\mathbf{Y}))$
 8:     end for
 9:     $l_D \leftarrow \max_{i \in \{1,2,\ldots N\}}\{E_i\}$
10:     Update Discriminator by $\theta_d \leftarrow \theta_d - \eta \nabla_{\theta_d} l_D$
11:     $l_G^1 \leftarrow \min_{i \in \{1,2,\ldots N\}}\{E_i\}$
12:     Compute $l_G^2$ by Eq. 11
13:     $l_G \leftarrow -\alpha \cdot l_G^1 + \beta \cdot l_G^2$
14:     Update Generator by $\theta_g \leftarrow \theta_g - \eta \nabla_{\theta_g} l_G$
15:     Learning rate $\eta$ decay
16: end for
17: return $\theta_g$
```

2) Generator Loss: The generator aims to embed some message into a cover image in an imperceptible manner so that the resulting stego image is difficult to be detected by the steganalyzers in discriminator. Thus, the generator loss l_G is defined as follows:

$$l_G = -\alpha \cdot l_G^1 + \beta \cdot l_G^2 \tag{9}$$

Note that there are two items in l_G. The first item l_G^1 denotes the adversarial loss against the strongest steganalyzer in discriminator rather than the weakest steganalyzer in each iteration, while the second item l_G^2 is the entropy loss and used to assure the embedding payload. The weights of the two items are α and β, we set $\alpha = 1$ and $\beta = 10^{-7}$ as it did in [24]. Specifically, l_G^1 and l_G^2 are defined as follows:

$$l_G^1 = \min_{i \in \{1,2,\ldots N\}} \{-z_0 \log(\mathrm{D}_i(\mathbf{X})) - z_1 \log(\mathrm{D}_i(\mathbf{Y}))\} \tag{10}$$

$$l_G^2 = (-(\sum_{\forall(i,j)} \sum_{\forall m} p_{i,j}^{(m)} \cdot \log_2(p_{i,j}^{(m)})) - (\mathrm{H} \times \mathrm{W} \times q))^2 \tag{11}$$

where H and W denote the height and width of the input image respectively, $1 \leq i \leq H, 1 \leq j \leq W$. m denotes the embedding modifications, $m \in \{-1, 0, +1\}$. q denotes the embedding payload.

During training stage, the loss function l_D and l_G described above are utilized in the proposed model for computing the gradient and updating the model parameters of the discriminator and the generator respectively. The training steps of the proposed model are illustrated in Algorithm 1.

IV. Experimental Results

In the GAN training stage, we employ 40,000 images from SZUBase [23] to train the GAN model as it did in [24]. The image size of the SZUBase is 512×512, and then resized to 256×256 by *imresize* function with the bicubic interpolation method as well as the default parameter settings in MATLAB, as it did in [24] and [25]. In the test stage, we feed the 10,000 images of size 256×256 from BOSSBase [41] into the generator of the resulting GAN model previously, and get the corresponding embedding probability $\mathbf{P}$ of each image separately. Finally, we calculate the embedding costs of each image based on the Eq. 6. With the STC, we achieve 10,000 stego images. As it did in [24], we randomly select 4,000 cover-stego image pairs for training steganalyzers, 1,000 image pairs for validation, and the rest 5,000 image pairs for test. In our proposed method, we use the U-Net architecture as the generator as illustrated in Fig. 2. Two steganalyzers, i.e., Xu-Net and Yedroudj-Net, are included in the discriminator. The update strategy is that update the relatively weaker steganalyzer in discriminator and update the generator using the gradients from relatively stronger steganalyzer in each training iteration. In the experiment, the batch-size is set to 24, the epoch is set to 72, the learning rate is set to initial 0.0001 and decays to 0.4 times every 20 epochs. We have provided the source codes of our model on line[2] so that readers can repeat our results easily.

Four steganographic methods are used for comparative study, including one traditional steganography HILL, and three steganographic methods based on GAN (i.e., ASDL-GAN, UT-GAN and SPAR-RL based on UT-GAN). To ensure a fair comparison, all methods were evaluated on the same image dataset partition, following the approach used in previous studies [24], [25]. We retained the training parameters as defined in the original publications of those methods. For each steganographic method, five embedding rates from 0.1 bpp to 0.5 bpp are considered. Note that for all GAN-based steganographic methods, we firstly train a GAN model on 0.4 bpp, and then we use the resulting model to generate stego images with different embedding rates. After generating the stego images by our method, we employ five different steganalyzers, including one traditional method (i.e., SRM) and four modern CNN-based methods (i.e., Xu-Net, Yedroudj-Net, SRNet and CovNet), to evaluate the security of our method and other steganographic methods.

A. Comparative Study on Different Settings in Generator

In generator, we design a U-Net based network to produce the probability map of the input image, as illustrated in Fig. 2. Under the same architecture (i.e., Fig. 2-(a)), in this section, four variants of the proposed block (i.e., Fig. 2-(b)) are included for comparative study.

- **Variant #1**: As illustrated in Fig. 4-(a), the block includes a convolution layer, a batch normalization layer and a LeakyReLU function. Note that this block is employed in UT-GAN.
- **Variant #2**: As illustrated in Fig. 4-(b), the block includes 2 convolution layers, 2 batch normalization layers, 2 LeakyReLU functions and a residual connection [42]. Note that the input of the residual part is the feature map

[2]Available at: https://github.com/HuangDX56/Steganographic-method-based-on-GMAN

TABLE I

DETECTION ERROR RATE (%) OF THE PROPOSED METHOD WITH DIFFERENT NETWORKS FOR GENERATING PROBABILITY MAP IN GENERATOR (0.4 BPP). THOSE VALUES WITH AN ASTERISK DENOTE THE BEST RESULTS IN THE CORRESPONDING CASES

Generator	SRM	Xu-Net	Yedroudj-Net	SRNet	CovNet
Variant #1	28.71	31.84	25.99	20.88	21.72
Variant #2	30.31	35.03	30.37	25.10	24.81
Variant #3	30.56	34.51	30.24	24.41	24.03
Variant #4	29.17	30.67	27.09	22.34	21.93
Steg-GMAN	**30.62***	**35.29***	**30.85***	**26.15***	**25.38***

TABLE II

DETECTION ERROR RATE (%) OF THE PROPOSED MODEL WITH DIFFERENT COMBINATIONS OF STEGANALYZERS IN DISCRIMINATOR (0.4 BPP). THOSE VALUES WITH AN ASTERISK DENOTE THE BEST RESULTS IN THE CORRESPONDING CASES

Combination of Steganalyzers	Configuration	SRM	Xu-Net	Yedroudj-Net	SRNet	CovNet
Single Steganalyzer	Xu-Net	28.22	34.24	24.74	20.06	18.49
	Yedroudj-Net	12.34	9.12	6.34	4.17	4.87
	SRNet	5.74	4.37	3.94	2.95	3.24
	CovNet	5.16	4.23	3.57	2.06	2.68
Double Steganalyzers	Xu-Net & SRNet	27.24	28.53	26.24	26.77	17.84
	Xu-Net & CovNet	29.15	33.14	27.64	24.93	24.05
	Steg-GMAN (Xu-Net & Yedroudj-Net)	**30.62***	**35.29***	**30.85***	26.15	25.38
Triple Steganalyzers	Xu-Net & Yedroudj-Net & SRNet	30.30	35.15	30.31	**28.41***	**25.78***
	Xu-Net &Yedroudj-Net & CovNet	28.49	31.94	26.35	24.22	22.34
	Xu-Net & CovNet & SRNet	27.96	31.43	26.90	23.41	23.07

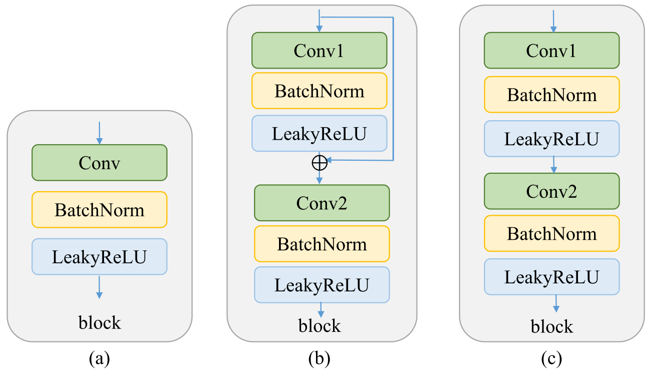


Fig. 4. Three different block structures used in the proposed U-Net structure for comparative study.

from the previous block and the corresponding down-sampling block with concatenation when up-sampling (i.e. for those blocks $9-15$ in Fig. 2-(a)).

- **Variant #3**: The block uses the same architecture as Variant #2. The main difference is that the input of the residual part is the feature map from the previous block without concatenation when up-sampling (i.e. for those blocks $9-15$ in Fig. 2-(a)). Hence, the number of output channels in the first convolution layer $Conv1$ is half of those in Variant #2 when up-sampling.
- **Variant #4**: As illustrated in Fig. 4-(c), the block just removes the residual connection structure from Variant #2, and employs the same convolution layer parameters in Variant #2.

The comparative results are shown in Table I. From Table I, we observe that the probability maps generated by different variants would significantly affect the model performances. First of all, the proposed network always achieves the highest steganography security in all cases. Compared with the original network used in UT-GAN (i.e., Variant #1), the proposed network can achieve over 5.20% and 3.66% improvements for the two modern steganalyzers (i.e., SRNet CovNet) separately.

B. Comparative Study on Different Settings in Discriminator

In discriminator, we employ two steganalyzers, that is Xu-Net and Yedroudj-Net in the proposed model. In this section, several different combinations of the steganalyzers in discriminator are included for comparative study, including using four single steganalyzer, three double steganalyzers, and three triple steganalyzers. The comparative results are shown in Table II. From Table II, we obtain three following observations:

- For single steganalyzer, Xu-Net can achieve the highest steganography security compared with three other steganalyzers (i.e., Yedroudj-Net, SRNet, and CovNet). Based on previous literatures [14], [15], [16], however, Xu-Net is relatively poorer for detecting steganography among the four steganalyzers. Thus, it is better to assign weaker steganalyzers rather than stronger steganalyzers in the proposed model. Otherwise, the steganography security will significantly drop. Taking CovNet for instance, the corresponding detection error rates would drop below 5.2% for all cases;

TABLE III

THE PROPORTION (%) OF TWO UPDATED STEGANALYZERS IN DISCRIMINATOR, THE PROPORTION (%) OF TWO STEGANALYZERS USED FOR UPDATING GENERATION, AND THE DETECTION ERROR RATE (%) WITH DIFFERENT UPDATING STRATEGIES (0.4 BPP). IN THIS TABLE, G AND D DENOTE THE GENERATOR AND DISCRIMINATOR SEPARATELY

Update Strategy	Proportion of Iterations in D		Proportion of Iterations in G		Security Performance				
	Xu-Net	Yedroudj-Net	Xu-Net	Yedroudj-Net	SRM	Xu-Net	Yedroudj-Net	SRNet	CovNet
Strategy #1	100.00	100.00	99.48	0.52	28.49	28.53	23.21	17.53	16.78
Strategy #2	100.00	100.00	0.53	99.47	24.76	20.67	16.87	15.04	13.70
Strategy #3	7.09	92.91	92.91	7.09	5.53	3.67	3.26	2.76	2.38
Strategy #4	5.14	94.86	5.14	94.86	18.57	18.65	15.64	14.76	13.62
Strategy #5	97.91	2.09	97.91	2.09	28.44	**35.87***	26.87	20.57	20.72
Steg-GMAN	91.22	8.78	8.78	91.22	**30.62***	35.29	**30.85***	**26.15***	**25.38***

TABLE IV

DETECTION ERROR RATE (%) OF THE PROPOSED MODEL WITH DIFFERENT GMAN INTEGRATION STRATEGY (0.4 BPP). THOSE VALUES WITH AN ASTERISK DENOTE THE BEST RESULTS IN THE CORRESPONDING CASES

Update Strategy	SRM	Xu-Net	Yedroudj-Net	SRNet	CovNet
GMAN*	26.59	26.41	21.45	18.06	15.17
GMAN-max	28.49	28.53	23.21	17.53	16.78
GMAN-1	29.71	**35.65***	29.11	23.75	22.92
GMAN-0.7	21.37	20.89	13.18	12.84	11.19
GMAN-0.5	23.29	21.67	19.54	13.08	11.40
GMAN-0	28.65	34.03	25.80	21.18	19.84
Steg-GMAN	**30.62***	35.29	**30.85***	**26.15***	**25.38***

- Besides of Xu-Net, if another steganalyzer is added in the proposed Discriminator, the resulting steganography security is not always improved. Taking Xu-Net & SRNet for instance, the detection error rate will drop around 1%, 6% and 1% evaluated by SRM, Xu-Net and CovNet separately. When Xu-Net & Yedroudj-Net is used, however, we always achieve the best results compared with using a single Xu-Net. The improvements are significant in some cases. For example, the detection error rate will increase from 20.06% to 26.15% evaluated by SRNet.
- Similarly, the steganography security is not always improved when triple steganalyzers are employed. Meanwhile, the training time will be increased based on our experiments, especially when SRNet is used.

Based on above observations, we conclude that different combinations of steganalyzers will significantly affect the proposed model performance. Considering the performance and efficiency of the model, we employ Xu-Net & Yedroudj-Net in the Discriminator in our experiments.

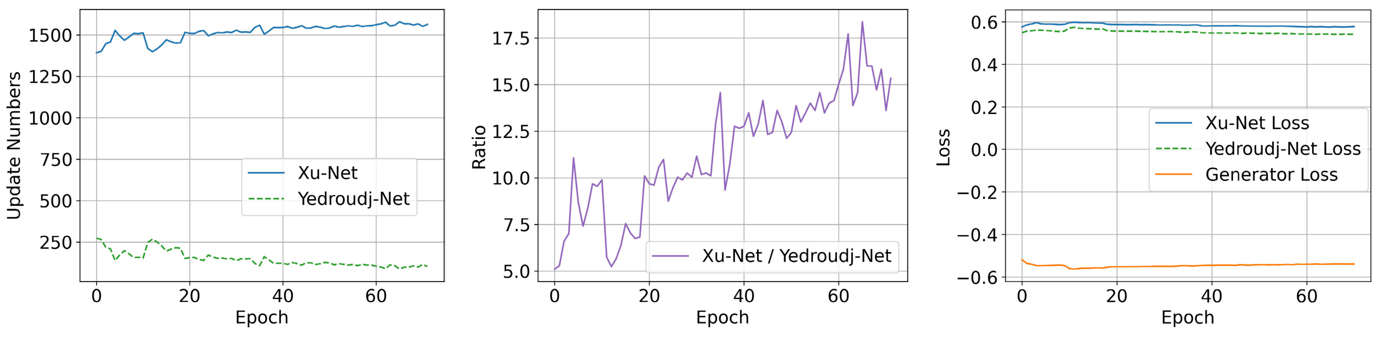
C. Comparative Study on Different Update Strategies

Since two steganalyzers are employed in the proposed discriminator, the iteration strategy for updating the generator and discriminator would affect the model performances significantly. As described in section III-B, the proposed method updates the relatively weaker steganalyzer (i.e., with larger cross entropy) in discriminator, and updates the generator using the gradients from relatively stronger steganalyzer (i.e., with smaller cross entropy) in each training iteration. In this section, we would evaluate the model performances with various update strategies.

In this experiment, we compare the proposed model with five following update strategies in each iteration.

- **Strategy #1**: Update both stronger and weaker steganalyzers in discriminator, and update generator with the weaker steganalyzer.
- **Strategy #2**: Update both stronger and weaker steganalyzers in discriminator, and update generator with the stronger steganalyzer. Note that since both steganalyzers are updated for strategy #1 and strategy #2, the iteration proportions in discriminator in Table IV are 100%.
- **Strategy #3**: Update the stronger steganalyzer in discriminator, and update generator with the weaker steganalyzer.
- **Strategy #4**: Update the stronger steganalyzer in discriminator, and update generator with the stronger steganalyzer.
- **Strategy #5**: Update the weaker steganalyzer in discriminator, and update generator with the weaker steganalyzer.

The comparative results are shown in Table III. From Table III, we can obtain two observations: 1) During training stage, Yedroudj-Net is stronger than Xu-Net in most cases (over 91.20%). Especially when both steganalyzers are updated in each iteration (i.e., Strategy #1 and Strategy #2), over 99.45% stronger steganalyzers are Yedroudj-Net; 2) Different update strategies would significantly affect the security performances of the proposed model. The security performance of Strategy #3 is quite poor. The main reason is that the stronger steganalyzer is always updated in each iteration,



(a) Update numbers of two steganalyzers (b) The ratio of two update numbers (c) Generator loss v.s. two steganalyzer losses

Fig. 5. The update numbers and ratios of two steganalyzers in the proposed discrminator with increasing the training epochs; and the generator loss v.s. two steganalyzer losses with increasing the training epochs.

while the weaker steganalyzer can not be updated enough times so that it can not effectively update the generator. For the proposed strategy, however, it can obtain the best security performances in most cases, especially evaluated on the modern steganalyzers. Taking CovNet for instance, the proposed method can achieve around 5% improvement compared with the best one (i.e., Strategy #5) among the five compared strategies.

In previous experiment, only one steganalyzer (stronger or weaker) in discriminator is used for updating the generator. As it did in GMAN [43] for generating images, in this experiment, we firstly update the two steganalyzers in discriminator, and then calculate the adversarial loss $l_G^{1'}$ by integrating the cross entropy of the two steganalyzers as follows:

$$l_G^{1'} = \frac{e^{\lambda E_1}}{e^{\lambda E_1} + e^{\lambda E_2}} \cdot E_1 + \frac{e^{\lambda E_2}}{e^{\lambda E_1} + e^{\lambda E_2}} \cdot E_2 \qquad (12)$$

where E_i denotes the cross entropy of the i-th steganalyzer based on the Eq. 7, $i = 1, 2$.

Note that the parameter λ is used to control the weights of E_i in the strategy GMAN-λ. Table IV shows some comparative results with several typical parameter λ. Note that GMAN* indicates that λ is updated automatically. GMAN-max indicates that $\lambda = +\infty$. $\lambda = 1$ denotes a $softmax$ operation, while $\lambda = 0$ denotes an average operation. From Table IV, we observe that the proposed strategy outperforms the other strategies except for Xu-Net. Compared with the best one (i.e., GMAN-1) among six strategies, we obtain 0.91%, -0.36%, 1.74%, 2.40% and 2.46% improvements evaluated by SRM, Xu-Net, Yedroudj-Net, SRNet and CovNet separately. Note that Eq. 12 is not actually part of our proposed method, but rather serves as a comparative approach to integrate multiple steganalyzers. We include the weighted sum strategy in order to demonstrate a different perspective on the integration of steganalyzers. In addition, we hope that this part of the experiment will inspire readers to consider and explore various possibilities for integrating steganalyzers.

D. Analysis of Iterative Process of the Proposed Model

In each iteration, the proposed model just updates the weaker steganalyzer in the discriminator, and updates the generator with the stronger steganalyzer. Fig. 5-(a) shows the average numbers of updating for the two steganalyzers with increasing the epoch (each epoch includes 1,666 iterations), and Fig. 5-(b) shows the corresponding ratios of two update numbers in every epoch. From Fig. 5-(a) and Fig.5-(b), we observe that Xu-Net is relatively poorer than Yedroudj-Net in most iterations, and the ration of two update numbers (i.e., Xu-Net / Yedroudj-Net) tends to increase from 5 to 17 with increasing the epoch. Fig.5-(c) shows the generator loss v.s. two steganalyzer losses with increasing training epochs. From Fig.5-(c), we observe that the proposed model can balance the capabilities of the generator and the discriminator during the training stage. Thus, the capability of generator can be improved gradually with increasing the training epochs. Fig. 6 shows two cover images and the modification maps at different iterations. From Fig. 6, we observe that with the increase of the iteration numbers of the model, the modifications will change from random to the those texture and edge regions within the image. This demonstrates that the training is effective and the performance of the generator improves gradually.

E. Comparison With Related Steganographic Methods

In this section, we compare the proposed method with four other steganographic methods evaluated on five steganalyzers. For a fair comparison, we maintained the models with the exact parameters as defined in their respective publications. Note that all the previous methods based on GAN employed a single steganalyzer (i.e., Xu-Net) as the discriminator. In our method, we ultimately selected two steganalyzers (i.e., Xu-Net & Yedroudj-Net) in the discriminator based on previous experiments in Section IV-B. The comparative results are shown in Table V. From Table V, we can obtain three following observations:

- On average, the proposed method can outperform the other steganographic methods for all traditional and CNN-based steganalyzers, and achieve state-of-the-art results in terms of steganography security;
- Compared with the current best steganography based on GAN (i.e., SPAR-RL), the proposed method has made great improvements, especially for CNN-based steganalyzers (i.e., SRNet & CovNet). Taking SRNet for instance, the proposed method achieves 2.77% average improvement. This is a significant improvement in image steganography;

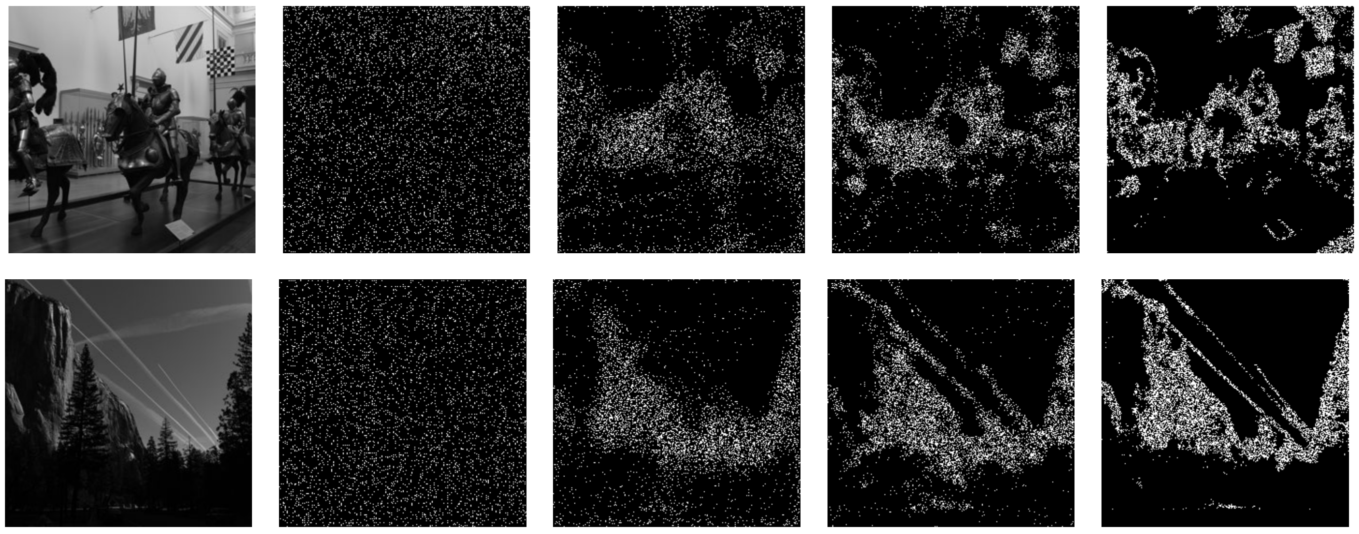


Fig. 6. Cover images (6744.pgm and 9012.pgm in BOSSBase) and the corresponding modification maps at different iterations.

TABLE V
DETECTION ERROR RATE (%) EVALUATED ON FIVE STEGANALYZERS ON BOSSBASE. THOSE VALUES WITH AN ASTERISK DENOTE THE BEST RESULTS IN THE CORRESPONDING CASES

Steganalyzer	Steganographic method	0.1 bpp	0.2 bpp	0.3 bpp	0.4 bpp	0.5 bpp	Average
SRM	HILL [5]	43.14	37.02	31.46	26.92	22.04	32.12
	ASDL-GAN [23]	43.22	34.46	29.58	24.50	19.54	30.26
	UT-GAN [24]	44.35	38.21	32.38	28.64	24.38	33.59
	SPAR-RL [25]	44.94	39.15	33.29	30.34	25.33	34.61
	Steg-GMAN	**45.87***	**40.34***	**34.69***	**30.62***	**26.15***	**35.53***
Xu-Net	HILL [5]	43.02	37.90	31.86	27.97	23.80	32.91
	ASDL-GAN [23]	42.09	39.55	32.86	28.75	24.77	33.60
	UT-GAN [24]	45.33	41.24	36.95	33.61	28.14	37.05
	SPAR-RL [25]	46.27	42.07	38.57	**35.53***	**29.73***	38.43
	Steg-GMAN	**46.65***	**42.62***	**39.85***	35.29	29.44	**38.77***
Yedroudj-Net	HILL [5]	42.97	31.33	24.42	19.54	15.62	26.78
	ASDL-GAN [23]	43.43	33.46	28.17	23.12	17.85	29.21
	UT-GAN [24]	46.90	37.21	32.03	27.14	23.18	33.29
	SPAR-RL [25]	47.72	39.52	32.89	28.59	23.77	34.50
	Steg-GMAN	**49.91***	**41.57***	**35.21***	**30.85***	**24.15***	**36.34***
SRNet	HILL [5]	37.02	27.21	21.85	17.33	14.80	23.64
	ASDL-GAN [23]	40.03	24.48	19.25	16.23	11.87	22.37
	UT-GAN [24]	40.99	30.82	26.31	21.92	16.45	27.30
	SPAR-RL [25]	42.00	32.52	27.23	22.81	17.67	28.45
	Steg-GMAN	**42.99***	**35.57***	**30.77***	**26.15***	**20.60***	**31.22***
CovNet	HILL [5]	35.33	26.28	20.91	16.10	13.52	22.43
	ASDL-GAN [23]	33.78	22.34	17.86	13.74	10.64	19.67
	UT-GAN [24]	40.03	31.73	25.29	21.97	16.29	27.06
	SPAR-RL [25]	41.38	32.53	26.38	23.21	17.11	28.12
	Steg-GMAN	**42.20***	**34.75***	**28.42***	**25.38***	**19.31***	**30.01***

- Except for ASDL-GAN, all steganographic methods based on GAN outperform the traditional steganography HILL significantly. Taking CovNet for instance, the average improvements are 4.65%, 5.69% and 7.58% separately for the UT-GAN, SPAR-RL and the proposed method, which means that GAN is

TABLE VI
DETECTION ERROR RATE (%) EVALUATED ON FIVE STEGANALYZERS ON BOWS2. THOSE VALUES WITH AN ASTERISK DENOTE THE BEST RESULTS IN THE CORRESPONDING CASES

Steganalyzer	Steganographic method	0.1 bpp	0.2 bpp	0.3 bpp	0.4 bpp	0.5 bpp	Average
SRM	HILL [5]	45.73	40.84	35.51	30.03	25.60	35.54
	ASDL-GAN [23]	45.74	39.76	31.47	25.99	20.46	32.68
	UT-GAN [24]	46.82	43.46	38.34	33.07	27.16	37.77
	SPAR-RL [25]	48.05	44.75	39.12	34.12	28.84	38.98
	Steg-GMAN	**48.47***	**45.35***	**40.42***	**34.71***	**29.23***	**39.64***
Xu-Net	HILL [5]	45.96	42.12	36.68	31.61	26.90	36.65
	ASDL-GAN [23]	46.36	42.44	36.76	31.91	25.50	36.60
	UT-GAN [24]	46.92	43.90	40.87	36.71	30.65	39.81
	SPAR-RL [25]	47.64	44.75	42.31	37.12	31.64	40.69
	Steg-GMAN	**48.10***	**45.36***	**42.79***	**37.43***	**31.88***	**41.11***
Yedroudj-Net	HILL [5]	49.90	35.53	27.55	21.50	17.52	30.40
	ASDL-GAN [23]	49.92	36.98	29.97	23.32	18.25	31.69
	UT-GAN [24]	49.60	43.68	35.92	30.82	24.85	36.97
	SPAR-RL [25]	49.86	44.58	36.64	31.44	26.34	37.77
	Steg-GMAN	**49.94***	**45.69***	**38.62***	**33.87***	**27.85***	**39.19***
SRNet	HILL [5]	49.80	30.69	26.42	20.53	16.80	28.85
	ASDL-GAN [23]	49.97	31.57	22.40	18.54	14.51	27.40
	UT-GAN [24]	49.98	38.65	31.89	27.76	22.75	34.21
	SPAR-RL [25]	50.00	39.27	33.15	29.28	23.64	35.07
	Steg-GMAN	**50.03***	**42.96***	**36.25***	**33.69***	**26.39***	**37.86***
CovNet	HILL [5]	39.89	30.76	24.58	19.34	15.15	25.94
	ASDL-GAN [23]	38.30	26.71	19.09	14.74	11.54	22.08
	UT-GAN [24]	45.30	36.53	31.65	26.20	21.10	32.16
	SPAR-RL [25]	46.32	37.42	32.17	27.48	22.75	33.23
	Steg-GMAN	**47.16***	**40.45***	**34.79***	**31.86***	**25.20***	**35.89***

very promising for enhancing the steganography security.

F. Security Evaluation on Other Image Databases

In this section, we compare the proposed method with related methods on two other image databases - BOWS2 [44] and ALASKA [45]. For BOWS2, we divide all images (i.e., 10,000 images) from BOWS2 into three parts: 4,000 images for training steganalyzers, 1,000 images for validation, and the rest 5,000 images for test. The comparative results evaluated on the test dataset are shown in Table VI. From Table VI, we observe that the proposed method still outperforms four other steganographic methods in all cases. The security improvement is significant for modern CNN-based steganalyzers. Taking SRNet and CovNet for instance, the proposed method achieves over 2% average improvement compared with the current best steganographic method based on GAN - SPAR-RL, and achieves over 9% average improvement compared with the traditional steganographic method - HILL.

For ALASKA, we randomly select 40,000 images from ALASKA, and then divide them into three parts: 15,000 images for training steganalyzers, 5,000 images for validation, and the rest 20,000 images for test. The comparative results evaluated on the test dataset are shown in Table VII. Compared with the results in Table VI and Table V, we find that the those images from ALASKA are relatively difficult to be detected in most cases since the this database is from more than 40 cameras (included smartphones, tablets, low-end cameras to high-end full frame DLSR) and processed in a realistic and highly heterogeneous way, which can faithfully reflect the high diversity of media "in the real world" [45]. However, the comparative results in Table VII still demonstrate that the proposed method can obtain the current best steganography security in almost all cases.

G. Comparison With Backpack Algorithm

In this section, we compare the proposed method with the Backpack [29] in two different scenarios: cover source match and cover source mismatch. Additionally, we will also compare the training time of both methods. It is important to note that the cover source mismatch [46] problem arises when the image used for steganography training and the image used for steganography testing are from different sources.

1) Scenario #1: Cover Source Match: In this experiment, the Backpack algorithm utilized images from the BOSSBase dataset during both the training and testing phases. Following the experimental setup described in [29], we used HILL as the initial embedding cost (referred to as HILL+Backpack)

TABLE VII
DETECTION ERROR RATE (%) EVALUATED ON FIVE STEGANALYZERS ON ALASKA. THOSE VALUES WITH AN ASTERISK DENOTE THE BEST RESULTS IN THE CORRESPONDING CASES

Steganalyzer	Steganographic method	0.1 bpp	0.2 bpp	0.3 bpp	0.4 bpp	0.5 bpp	Average
SRM	HILL [5]	45.87	41.84	36.26	33.98	29.53	37.50
	ASDL-GAN [23]	43.32	39.08	35.67	33.05	30.14	36.25
	UT-GAN [24]	46.24	42.76	37.93	34.45	30.91	38.46
	SPAR-RL [25]	46.71	43.03	37.16	34.97	31.62	38.70
	Steg-GMAN	**46.93***	**43.23***	**38.65***	**35.74***	**31.96***	**39.30***
Xu-Net	HILL [5]	47.41	45.07	42.79	39.02	36.79	42.22
	ASDL-GAN [23]	42.95	40.16	37.12	33.25	31.41	36.98
	UT-GAN [24]	48.93	46.28	43.49	39.81	37.17	43.14
	SPAR-RL [25]	48.66	46.93	43.74	**40.81***	38.15	43.66
	Steg-GMAN	**49.61***	**47.03***	**44.81***	40.38	**38.57***	**44.08***
Yedroudj-Net	HILL [5]	44.32	36.57	33.54	29.82	26.94	34.24
	ASDL-GAN [23]	40.81	36.01	32.54	29.26	27.74	33.27
	UT-GAN [24]	43.94	39.05	35.19	32.94	30.33	36.29
	SPAR-RL [25]	47.73	37.56	34.24	32.97	28.89	36.28
	Steg-GMAN	**49.25***	**39.81***	**37.18***	**33.87***	**30.72***	**38.17***
SRNet	HILL [5]	48.07	39.23	34.74	32.62	28.46	36.62
	ASDL-GAN [23]	48.15	32.25	28.96	27.56	26.58	32.70
	UT-GAN [24]	48.96	39.85	37.32	33.41	30.32	37.97
	SPAR-RL [25]	49.28	37.47	35.62	33.39	30.11	37.17
	Steg-GMAN	**49.72***	**43.82***	**39.57***	**36.93***	**32.73***	**40.55***
CovNet	HILL [5]	35.96	32.07	30.15	28.65	26.38	30.64
	ASDL-GAN [23]	32.44	26.94	23.92	22.92	20.93	25.43
	UT-GAN [24]	36.32	33.08	31.03	29.91	27.56	31.58
	SPAR-RL [25]	**38.08***	33.38	29.86	28.51	27.82	31.53
	Steg-GMAN	37.72	**34.14***	**32.52***	**30.84***	**28.24***	**32.69***

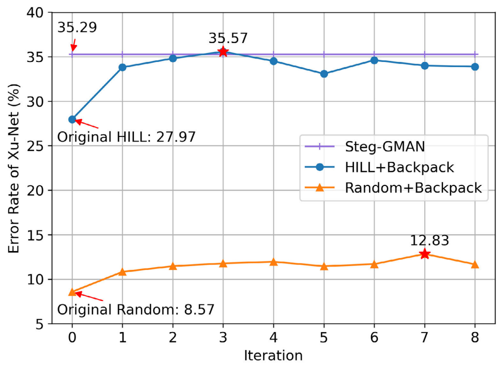


Fig. 7. Detection errors (%) on the BOSSbase dataset against Xu-Net with an increasing number of iterations. The values marked with an asterisk indicate the best results during the iterative process.

and performed 8 iterations using Xu-Net as the target steganalytic network. Additionally, to ensure fairness in our comparison, since our method learns the embedding cost from scratch, we also used random cost as the initial embedding costs and conducted similar iterative training (referred to as Random+Backpack). The comparative results are presented in Fig. 7. From Fig. 7, we can observe two observations:

1) Firstly, the security of Random+Backpack is significantly weaker than HILL+Backpack, indicating that Backpack algorithm is an improved scheme that is highly sensitive to the choice of initial cost. When the initial cost is poor, its security is greatly compromised.
2) Regarding HILL+Backpack, its security performance does not monotonically increase with the number of iterations, which can also be seen in [29]. In this experiment, the highest error rate of 35.57% was achieved in the third iteration, which is only slightly better than our method by 0.28%. However, the results obtained in other iterations were consistently worse than our method.

In addition to the target steganalytic network Xu-Net, we have also provided the detection results of other steganalytic methods on the BOSSBase dataset, as shown in the Table VIII. From Table VIII, we can observe that apart from the target steganalytic network Xu-Net, the HILL+Backpack method achieves slightly better performance compared to our method. However, under the other four steganalysis methods, our method consistently achieves the best performance. Furthermore, the improvement in performance offered by our method is significant. For example, when considering Yedroudj-Net, SRNet, and CovNet, our method outperforms HILL+Backpack by a margin of 7.97%.

TABLE VIII
DETECTION ERRORS (%) EVALUATED ON BOSSBASE USING FIVE STEGANALYTIC METHODS. THE EMBEDDING PAYLOAD IS 0.4 BPP. THE VALUES MARKED WITH AN ASTERISK INDICATE THE BEST PERFORMANCE IN EACH CASE. B DENOTES THE BEST ITERATION, AND L DENOTES THE LAST ITERATION

Method	SRM	Xu-Net	Yedroudj-Net	SRNet	CovNet
HILL+ Backpack(B)	26.02	**35.57***	21.45	18.18	17.08
HILL+ Backpack(L)	26.98	33.89	20.31	16.69	16.47
Steg-GMAN	**30.62***	35.29	**30.85***	**26.15***	**25.38***

TABLE IX
DETECTION ERRORS (%) EVALUATED ON BOW2 USING FIVE STEGANALYTIC METHODS. THE EMBEDDING PAYLOAD IS 0.4 BPP. THE VALUES MARKED WITH AN ASTERISK INDICATE THE BEST PERFORMANCE IN EACH CASE. B DENOTES THE BEST ITERATION, AND L DENOTES THE LAST ITERATION

Method	SRM	Xu-Net	Yedroudj-Net	SRNet	CovNet
HILL+ Backpack(B)	30.09	34.41	22.86	21.54	20.14
HILL+ Backpack(L)	29.23	33.24	22.60	21.25	19.27
Steg-GMAN	**34.71***	**37.43***	**33.87***	**33.69***	**31.86***

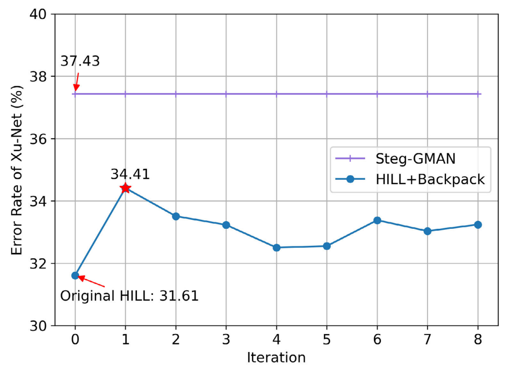


Fig. 8. Detection errors (%) on the BOWS2 dataset against Xu-Net with an increasing number of iterations. The values marked with an asterisk indicate the best results during the iterative process.

2) Scenario #2: Cover Source Mismatch : Since the proposed method is designed to handle the cover source mismatch scenario, while the experimental procedure of Backpack involves solely working on a single image database, during the training and testing phases. To ensure a fair comparison, in this experiment, we applied the HILL+Backpack method, which was pre-trained using the BOSSBase dataset, to generate and select stegos on the BOWS2 dataset, and subsequently subjected the obtained stegos to security testing. Similarly, we provide the detection results against the Xu-Net steganalyzer for each of the 8 iterations, as well as the detection results using other steganalysis methods, as shown in Fig. 8 and Table IX. From the above comparative results, it is evident that the proposed method consistently outperforms the HILL+Backpack method in all cases. This observation supports the conclusion that the Backpack algorithm, as mentioned in their paper [29], is not effective in addressing the cover source mismatch problem.

3) Training Time : We would like to highlight that the training process of Backpack is highly time-consuming. In our experiments, we utilized an Intel Xeon Silver 4210R CPU operating at a clock speed of 2.40GHz, coupled with an NVIDIA GeForce RTX 3080Ti GPU. The training duration for Backpack encompassed approximately 10 days (i.e., 240 hours) to execute the attack and generate $10,000$ stego images. This was achieved using HILL initialization, an image batch size of 25, and 8 iterations. Notably, the training duration is even longer when the random initial cost is used. Since it is difficult to deceive the target steganalyzer by the generated stego with random initialization cost, more number of attacks are required. Conversely, our method completed the training and generation of $10,000$ stego images in less than 8 hours.

V. CONCLUSION

The existing related methods based on GAN mainly focus on the design of generator, but ignore the design of discriminator. In this paper, we propose a novel GAN-based steganographic framework to enhance the security performance. We first introduce multiple steganalyzers to enhance the performance of discriminator. Furthermore, we propose an adaptive way to update the parameters in GAN so that it can balance the capabilities of the generator and the discriminator dynamically. Extensive comparative results show that the proposed methods can achieve state-of-the-art results compared with the modern steganographic methods on three image databases (i.e., BOSSBase, BOWS2 and ALASKA). In addition, extensive ablation experiments show the rationality of the proposed model.

There are several important issues worth further study. For instance, the proposed method employs a relatively deeper U-Net architecture (see Fig. 2) in generator. In future, some compact network architectures will be considered, since that local feature within an image seems more important for calculating embedding cost based on previous studies [5], [6]. In addition, multiple networks for producing the probability map in generator will be considered to enhance the diversity of stego, and to achieve a better dynamic balance between generator and discriminator. Like related methods, furthermore, the proposed method just generates symmetric embedding costs. Asymmetric costs learning will be considered to further improve the steganography security.

REFERENCES

[1] J. Fridrich and T. Filler, “Practical methods for minimizing embedding impact in steganography,” *Proc. SPIE*, vol. 6505, Feb. 2007, Art. no. 650502.
[2] T. Filler, J. Judas, and J. Fridrich, “Minimizing additive distortion in steganography using syndrome-trellis codes,” *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 920–935, Sep. 2011.
[3] T. Pevný, T. Filler, and P. Bas, “Using high-dimensional image models to perform highly undetectable steganography,” in *Proc. Int. Workshop Inf. Hiding*, 2010, pp. 161–177.
[4] V. Holub and J. Fridrich, “Designing steganographic distortion using directional filters,” in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2012, pp. 234–239.
[5] B. Li, M. Wang, J. Huang, and X. Li, “A new cost function for spatial image steganography,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 4206–4210.

[6] V. Holub, J. Fridrich, and T. Denemark, “Universal distortion function for steganography in an arbitrary domain,” *EURASIP J. Inf. Secur.*, vol. 2014, no. 1, p. 1, Dec. 2014.

[7] V. Sedighi, R. Cogranne, and J. Fridrich, “Content-adaptive steganography by minimizing statistical detectability,” *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 2, pp. 221–234, Feb. 2016.

[8] B. Li, M. Wang, X. Li, S. Tan, and J. Huang, “A strategy of clustering modification directions in spatial image steganography,” *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 1905–1917, Sep. 2015.

[9] T. Pevny, P. Bas, and J. Fridrich, “Steganalysis by subtractive pixel adjacency matrix,” *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 2, pp. 215–224, Jun. 2010.

[10] J. Fridrich and J. Kodovsky, “Rich models for steganalysis of digital images,” *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 868–882, Jun. 2012.

[11] J. Kodovsky, J. Fridrich, and V. Holub, “Ensemble classifiers for steganalysis of digital media,” *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 432–444, Apr. 2012.

[12] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Mar. 1998.

[13] G. Xu, H.-Z. Wu, and Y.-Q. Shi, “Structural design of convolutional neural networks for steganalysis,” *IEEE Signal Process. Lett.*, vol. 23, no. 5, pp. 708–712, May 2016.

[14] M. Yedroudj, F. Comby, and M. Chaumont, “Yedroudj-Net: An efficient CNN for spatial steganalysis,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 2092–2096.

[15] M. Boroumand, M. Chen, and J. Fridrich, “Deep residual network for steganalysis of digital images,” *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 5, pp. 1181–1193, May 2019.

[16] X. Deng, B. Chen, W. Luo, and D. Luo, “Fast and effective global covariance pooling network for image steganalysis,” in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jul. 2019, pp. 230–234.

[17] W. Tang, B. Li, S. Tan, M. Barni, and J. Huang, “CNN-based adversarial embedding for image steganography,” *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 8, pp. 2074–2087, Aug. 2019.

[18] H. Mo, T. Song, B. Chen, W. Luo, and J. Huang, “Enhancing JPEG steganography using iterative adversarial examples,” in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, Jan. 2019, pp. 1–6.

[19] M. Liu, W. Luo, P. Zheng, and J. Huang, “A new adversarial embedding method for enhancing image steganography,” *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4621–4634, 2021.

[20] T. Song, M. Liu, W. Luo, and P. Zheng, “Enhancing image steganography via stego generation and selection,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 2695–2699.

[21] C. Qin, W. Zhang, X. Dong, H. Zha, and N. Yu, “Adversarial steganography based on sparse cover enhancement,” *J. Vis. Commun. Image Represent.*, vol. 80, Oct. 2021, Art. no. 103325.

[22] M. Liu, T. Song, W. Luo, P. Zheng, and J. Huang, “Adversarial steganography embedding via stego generation and selection,” *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 3, pp. 2375–2389, May 2022.

[23] W. Tang, S. Tan, B. Li, and J. Huang, “Automatic steganographic distortion learning using a generative adversarial network,” *IEEE Signal Process. Lett.*, vol. 24, no. 10, pp. 1547–1551, Oct. 2017.

[24] J. Yang, D. Ruan, J. Huang, X. Kang, and Y.-Q. Shi, “An embedding cost learning framework using GAN,” *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 839–851, 2020.

[25] W. Tang, B. Li, M. Barni, J. Li, and J. Huang, “An automatic cost learning framework for image steganography using deep reinforcement learning,” *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 952–967, 2021.

[26] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–11.

[27] S. Bernard, P. Bas, J. Klein, and T. Pevny, “Explicit optimization of min max steganographic game,” *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 812–823, 2021.

[28] S. Bernard, P. Bas, T. Pevný, and J. Klein, “Optimizing additive approximations of non-additive distortion functions,” in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jun. 2021, pp. 105–112.

[29] S. Bernard, P. Bas, J. Klein, and T. Pevný, “Backpack: A backpropagable adversarial embedding scheme,” *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 3539–3554, 2022.

[30] M. Yedroudj, F. Comby, and M. Chaumont, “Steganography using a 3-player game,” *J. Vis. Commun. Image Represent.*, vol. 72, Oct. 2020, Art. no. 102910.

[31] I. J. Goodfellow et al., “Generative adversarial nets,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[32] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” in *Proc. Int. Conf. Learn. Represent.*, 2016.

[33] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2242–2251.

[34] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, “StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8789–8797.

[35] W. Xu, C. Long, R. Wang, and G. Wang, “DRB-GAN: A dynamic ResBlock generative adversarial network for artistic style transfer,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 6363–6372.

[36] J. Kodovsky, J. Fridrich, and V. Holub, “On dangers of overtraining steganography to incomplete cover model,” in *Proc. 13th ACM Multimedia Workshop Multimedia Secur.*, C. Heitzenrater, S. Craver, and J. Dittmann, Eds., Sep. 2011, pp. 69–76.

[37] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, 2015, pp. 234–241.

[38] R. Zhang, F. Zhu, J. Liu, and G. Liu, “Depth-wise separable convolutions and multi-level pooling for an efficient spatial CNN-based steganalysis,” *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1138–1150, 2020.

[39] Y. Jin, Y. Wang, M. Long, J. Wang, P. S. Yu, and J. Sun, “A multi-player minimax game for generative adversarial networks,” in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2020, pp. 1–6.

[40] Y.-L. Wu, H.-H. Shuai, Z.-R. Tam, and H.-Y. Chiu, “Gradient normalization for generative adversarial networks,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 6353–6362.

[41] P. Bas, T. Filler, and T. Pevny, “‘Break our steganographic system’: The ins and outs of organizing BOSS,” in *Proc. Int. Workshop Inf. Hiding*, 2011, pp. 59–70.

[42] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[43] I. Durugkar, I. Gemp, and S. Mahadevan, “Generative multi-adversarial networks,” in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[44] P. Bas and T. Furon. (2007). *BOWS-2*. [Online]. Available: http://bows2.ec-lille.fr

[45] R. Cogranne, Q. Giboulot, and P. Bas, “The ALASKA steganalysis challenge: A first step towards steganalysis,” in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jul. 2019, pp. 125–137.

[46] J. Kodovský, V. Sedighi, and J. Fridrich, “Study of cover source mismatch in steganalysis and ways to mitigate its impact,” *Proc. SPIE*, vol. 9028, pp. 90280J-1–90280J-12, Feb. 2014.

Dongxia Huang received the B.E. degree from the School of Computer Science and Engineering, Central South University, Changsha, China, in 2021. She is currently pursuing the M.S. degree with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. Her research interests include digital image steganography, steganalysis, digital image forensics, and adversarial attacks.

Weiqi Luo (Senior Member, IEEE) received the Ph.D. degree from Sun Yat-sen University, Guangzhou, China, in 2008. He is currently a Professor with the School of Computer Science and Engineering, Sun Yat-sen University, where he is also a Researcher with the Guangdong Key Laboratory of Information Security Technology. His current research interests include digital multimedia forensics, steganography, and steganalysis.

Minglin Liu received the B.S. degree from Shandong University, Jinan, China, in 2013, the M.S. degree from Zhengzhou University, Zhengzhou, China, in 2017, and the Ph.D. degree from Sun Yat-sen University, Guangzhou, China, in 2022. He is currently a Lecturer with the School of Cyber Science and Engineering, Zhengzhou University. His current research interests include steganography, steganalysis and watermarking, adversarial attacks, and defense.

Weixuan Tang received the B.E. and Ph.D. degrees from the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China, in 2014 and 2019, respectively. He is currently an Associate Professor with Guangzhou University, Guangzhou. His research interests include digital image steganography, digital image forensics, and adversarial examples.

Jiwu Huang (Fellow, IEEE) received the B.S. degree from Xidian University, Xi'an, China, in 1982, the M.S. degree from Tsinghua University, Beijing, China, in 1987, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Science, Beijing, in 1998. He is currently a Professor with the College of Information Engineering, Shenzhen University, Shenzhen, China. Before joining Shenzhen University, he has been with the School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China, since 2000. His current research interests include multimedia forensics and security. He serves as a member for the IEEE CASS Multimedia Systems and Applications Technical Committee and the IEEE SPS Information Forensics and Security Technical Committee. He is an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY.