

GIFDL: Generated Image Fluctuation Distortion Learning for Enhancing Steganographic Security

Xiangkun Wang, Kejiang Chen, Yuang Qi, Ruiheng Liu, Weiming Zhang, Nenghai Yu

Abstract—Minimum distortion steganography is currently the mainstream method for modification-based steganography. A key issue in this method is how to define steganographic distortion. With the rapid development of deep learning technology, the definition of distortion has evolved from manual design to deep learning design. Concurrently, rapid advancements in image generation have made generated images viable as cover media. However, existing distortion methods based on machine learning do not fully leverage the advantages of generated cover media, resulting in suboptimal security performance. To address this issue, we propose GIFDL (Generated Image Fluctuation Distortion Learning), a steganographic distortion learning method based on the fluctuations in generated images. Inspired by the idea of natural steganography, we take a series of highly similar fluctuation images as the input to the steganographic distortion generator and introduce a new GAN training strategy to disguise stego images as fluctuation images. Experimental results demonstrate that GIFDL, compared with state-of-the-art GAN-based distortion learning methods, exhibits superior resistance to steganalysis, increasing the detection error rates by an average of 3.30% across three steganalyzers.

Index Terms—Generative adversarial networks (GANs), steganography, generative model.

I. INTRODUCTION

IMAGE steganography is a technique for hiding information, aiming to embed secret messages within images so that they are not easily detected [20]. Steganalysis, a countermeasure to steganography, seeks to identify the presence of hidden messages in images [21]. Early steganography methods, known as LSB steganography [22], hide secret messages in the least significant bits of pixels. These methods do not consider the different risks of performing steganographic modifications in different regions of the image, making them susceptible to detection by statistical steganalysis techniques. Content-adaptive steganography is then proposed, which adjusts the location and capacity of hidden messages based on the texture complexity of the image. The most typical achievement of this domain is minimum distortion steganography [23], which separates steganography into two processes: defining steganographic distortion and steganographic encoding, offering higher security than LSB steganography. Steganographic coding schemes [14], [?], and [45] are approximately optimal in minimizing total distortion, thus research focuses on how to better define steganographic distortion. Methods such as

This work was supported in part by the Natural Science Foundation of China under Grant U2336206, 62472398, U2436601 and 62402469

All the authors are with School of Cyber Science and Technology, University of Science and Technology of China and Anhui Province Key Laboratory of Digital Security, Hefei 230026, China.

Corresponding author: Kejiang Chen (Email:chenkj@ustc.edu.cn)

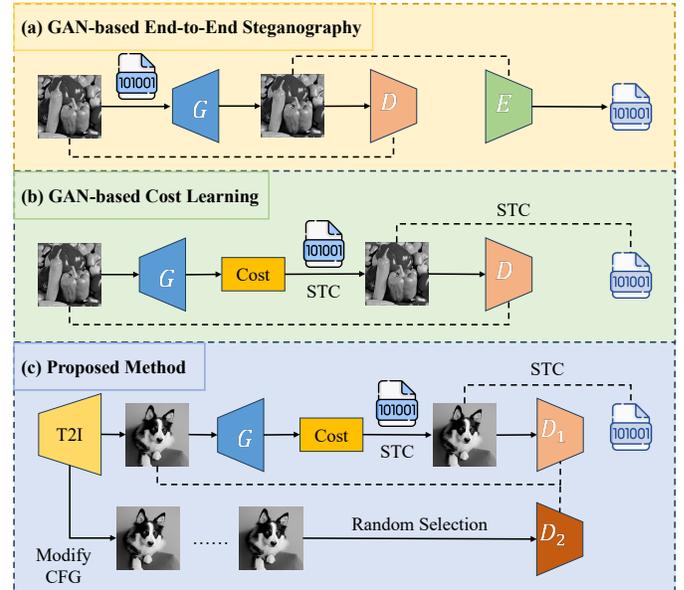


Fig. 1. Comparison of existing GAN-based steganography methods with the proposed method.

WOW [1], UNIWARD [2], HILL [3], MiPOD [4], UERD [5], MS [44], and IP [49] have been proposed to improve steganographic security by rationally defining distortion.

With the development of deep learning, traditional steganographic methods are not secure enough against deep learning-based steganalysis. Moreover, the excellent performance of generative adversarial networks (GANs) in image processing has inspired new steganography methods. GAN-based image steganography methods can be categorized into two types: GAN-based end-to-end steganography and GAN-based cost learning. As shown in Fig. 1(a), GAN-based end-to-end steganography does not employ the minimum distortion steganography framework. Instead, it explores the direct generation of stego images from cover images via the generator. For instance, SteganoGAN [13] embeds secret messages within the feature maps of cover images, achieving a payload of up to 4 bpp (bits per pixel). ABDH [35] introduces attention mechanisms in GANs to improve the quality and robustness of stego images. Building on SteganoGAN and ABDH, ChatGAN [9] further enhances image quality using XuNet [26] as the discriminator. However, these end-to-end methods suffer from significant security weaknesses against steganalysis. Deep learning-based steganalyzers such as SRNet [40] can accurately detect stego images via a small number of training

images.

Unlike GAN-based end-to-end steganography, GAN-based cost learning methods adhere to the minimum distortion steganography framework. They use GANs to learn steganographic distortions, convert them into steganographic costs, and combine steganographic encoding to embed and extract secret messages, as shown in Fig. 1(b). ASDL-GAN [6] is the first to apply GANs to learn steganographic distortion, using XuNet as the discriminator, but its security is weaker than that of manually designed distortion methods [3]. UT-GAN [7] improved upon ASDL-GAN by adopting a U-Net architecture in the generator and adding a double-tanh function, achieving better security than manual distortion definitions. GMAN [8] highlights that the security of GAN-based steganography is limited by discriminator performance, and a multi-discriminator training strategy is proposed to enhance security further. GACL [51] enhances the cover image to highlight its contour information, which in turn improves steganographic security. Despite these advancements, GAN-based distortion learning methods still face security challenges at high payloads, and designing more secure steganographic methods remains a crucial objective for researchers.

The development of deep learning has provided us with not only better cost learning methods but also new types of cover media. According to the Everypixel Journal¹, text-to-image generation models have created over 15 billion images during just a year and a half, equivalent to the number of photos taken by humans over 149 years. The widespread popularity of generated images makes them ideal steganographic covers. Researchers have proposed several steganographic methods based on generated images. For example, IDEAS [16] synthesizes stego images by generating image structural features from secret messages. GSN [17] uses secret messages as latent variables to guide the generator in image synthesis. StegaDDPM [18] generates stego images by sampling noise according to secret messages in the final step of the denoising diffusion probabilistic model. These methods require specially designed generator architectures or access to the internal parameters of the generation model to control the image generation process. However, popular image generation models are often considered black-box, with users having no access to internal information. Recently, Zhang et al. [47] proposed volatility distortion (VC) based on the fluctuation of black-box generative models, which can be used to improve the security of existing steganographic distortion learning methods by approximating the image pixel distribution using a manual method. However, the hand-designed method may make it difficult to comprehensively and accurately model the fluctuation distribution of an image, and further improvements are possible.

To address this issue, we propose a new GAN-based steganographic cost learning method, named GIFDL (Generated Image Fluctuation Distortion Learning), as shown in Fig. 1(c). We observe that when the input parameters of an image generation model are slightly modified, it outputs images that are highly similar to the original, which we call

fluctuation images. Inspired by the idea of natural steganography [46], the goal of GIFDL is to disguise stego images as fluctuation images. To this end, we take a series of fluctuation images as inputs to the generator, select one of them as the cover image, and randomly select one of the remaining fluctuation images, called “flu”. We pursue the stego image indistinguishable from fluctuation images, so GIFDL employs two discriminators, one for distinguishing between stego and cover, and the other for distinguishing between stego and “flu”, with the two discriminators updating their parameters alternately. Considering that the cover is also a member of the fluctuation images, the joint function of the two discriminators can ensure that the stego image is close to fluctuation images in terms of pixel distribution.

With the above design, the stego images resemble the fluctuation images, enabling the generator to learn the fluctuation distribution of the generated images, which allows the generator to modify pixels that deviate from the distribution at a higher cost. Under the minimum distortion steganography framework, we ultimately obtain stego images that are close to the fluctuation images in distribution. Unlike other GAN-based steganography methods, GIFDL focuses on the characteristics of black-box generated images and leverages their fluctuations to achieve steganographic distortion learning. Our contributions can be summarized as follows:

- **A novel steganographic distortion learning method based on the fluctuations of generated images:** Based on the observation of the fluctuation characteristics of the generated images, we take the fluctuation images as the input of our network and use a GAN to learn the distribution of fluctuation images, which in turn disguises the stego images as fluctuation images.
- **A new discriminator training strategy that considers the inherent differences between cover and fluctuation images:** We use a steganalysis network to distinguish between cover and stego images, while another steganalysis network is used to distinguish between fluctuation and stego images. By updating these two discriminators alternately, GIFDL can better learn the characteristics of fluctuation images, achieving better camouflage.
- **Extensive experiments to validate the effectiveness of GIFDL:** Extensive experiments have shown that GIFDL achieves considerable security against three mainstream steganalyzers. Compared with state-of-the-art methods, the detection error rate of steganalysis is improved by 3.30% on average.

II. RELATED WORK

In this section, we first introduce GAN-based cost learning methods, then we introduce volatility cost, followed by an introduction to text-to-image generation models.

A. GAN-based cost learning

The GAN consists of two adversarial sub-networks: the generator, which aims to create samples that resemble real data, and the discriminator, which aims to distinguish between

¹<https://journal.everyapixel.com/ai-image-statistics>

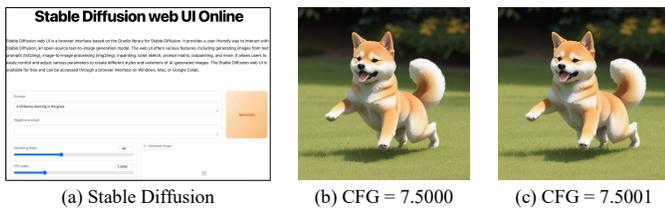


Fig. 2. (a) Users can access the black-box Stable Diffusion at [Stable Diffusion web UI Online](#) to use Stable Diffusion. (b) Images generated with CFG = 7.5000. (c) Images generated with CFG = 7.5001, where the Classifier-Free Guidance (CFG) scale is an input parameter in Stable Diffusion that controls how closely a prompt should be followed.

real data and fake samples generated by the generator. GAN-based cost learning methods are built on the minimum distortion steganography framework, which uses a GAN to generate steganographic costs and employs steganographic encoding to embed and extract secret messages. GAN-based cost learning methods leverage the adversarial nature of GANs and use deep-learning steganalyzers as discriminators. During training, these methods learn the implicit features of steganalysis algorithms within the discriminator and feed this information back to the generator to produce a modification probability map, thus obtaining steganographic cost. Notable methods include ASDL-GAN [6], UT-GAN [7], GMAN [8] and GACL [51].

1) *ASDL-GAN*: ASDL-GAN is the first to propose using the adversarial nature of GANs to learn steganographic cost, employing the steganalysis network Xu-Net as the discriminator. The generator learns the modification probabilities of the cover image, which are then mapped to modifications through a ternary embedding simulator (TES). The loss function of the generator is directly related to the undetectability by the adversarial steganalyzer and achieved backpropagation through TES. However, the security performance was not satisfactory. Experimental results revealed that ASDL-GAN was less effective against steganalysis compared with manually designed methods such as HILL.

2) *UT-GAN*: UT-GAN builds upon ASDL-GAN by employing a generator based on the U-Net architecture. Additionally, it introduces a dual-tanh activation function that does not require training, replacing the TES in ASDL-GAN. This improvement means that steganographic performance is no longer constrained by the pre-training of TES, and that the training time of GAN is reduced. Experimental results demonstrate that UT-GAN outperforms ASDL-GAN and the manual method HILL in resisting steganalysis.

3) *GMAN*: GMAN addresses the limitations of ASDL-GAN and UT-GAN, which use a single, relatively weak steganalyzer, Xu-Net, as the GAN discriminator. A weak discriminator can limit the generator's performance, thereby constraining the overall security of the steganographic method. GMAN introduces multiple steganalyzers as discriminators. During training, GMAN adaptively updates the parameters of the discriminators based on their performance. Experimental results indicate that GMAN achieves the highest security performance among the methods tested.

4) *GACL*: GACL enhances the cover image using Laplace operators to highlight the edges and contours of the image

to obtain an enhanced image. GACL then learns the steganographic distortion by feeding the cover image together with the enhanced image into a two-stream U-Net network.

These methods focus on optimizing the design of the network structure of the GAN itself and pay less attention to steganographic cover.

B. Volatility Cost

On the basis of the fluctuation of generated images, Zhang et al. [47] first proposed the idea of disguising the steganographic modification as the inherent volatility of the generated model, which improves the steganographic security and can be used in the black-box scenarios of the generated model. They approximated the pixel distribution of fluctuation images as a Gaussian distribution and estimated the parameters of the Gaussian distribution through a series of fluctuation images, which were then integrated to obtain the occurrence probability of each pixel. Furthermore, Volatility Cost (VC) was introduced based on the estimated distribution, with the probability of generated pixel occurrences as the steganographic modification probability, thus translating into volatility cost. Since VC calculates distortion only in terms of image volatility, for comprehensive consideration, Since VC calculates distortion solely in terms of image volatility, for a more comprehensive approach, it was combined with existing distortion definitions, improving security by an average of 4.64%. However, the distortion definition method of VC is based on some approximation assumptions, which inevitably have a bias in estimating the pixel distribution of fluctuation images and cannot comprehensively model the fluctuation characteristics of the generated images. Considering the limitations of VC, we propose to use deep learning methods to learn the fluctuation characteristics of generated images.

C. Text-to-image Generative Model

Text-to-image (T2I) generation models take text descriptions as inputs and produce high-quality, realistic images that correspond to the given descriptions. To increase image quality, numerous methods have been developed. For instance, Text-conditional GAN [28] was the first to implement an end-to-end differentiable architecture from the character level to the pixel level. StackGAN [29] utilizes multiple stacked generators, AttnGAN [30] incorporates attention mechanisms, and ControGAN [31] employs a word-level discriminator and perceptual loss to control image generation. Recently, Denoising Diffusion Probability Models (DDPMs) have emerged as a new paradigm in image generation due to their outstanding performance. DDPM-based models such as Imagen [34], Stable Diffusion [12], and DALL-E 3 [41] can generate images that are close to real images while also having artistic qualities. According to Everyapixel Journal, in 2024, people created an average of 34 million images per day, which are widely shared on social networks.

III. PROPOSED METHOD

A. Motivation

Research in the field of image steganography has consistently shown that introducing steganographic modifications

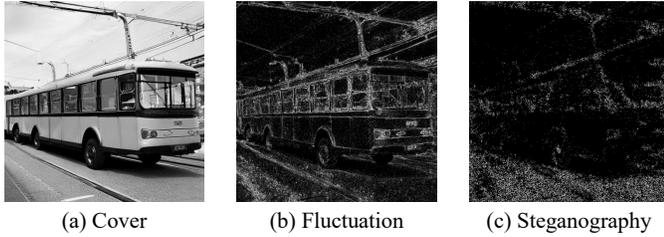


Fig. 3. (a) Cover image, (b) average pixel difference between the 10 fluctuation images and the cover image, (c) pixel difference between the stego image and the cover image, where the stego image is generated by GMAN. For clearer observation, the brightness of (b) and (c) is multiplied by 50.

in areas with complex textures is less likely to be detected. Based on this insight, researchers have proposed heuristic steganographic distortion methods such as WOW [1], S-UNIWARD [2], HILL [3], MiPOD [4], and UERD [5]. With the advancement of deep learning, researchers have also developed GAN-based methods to learn steganographic distortion definitions, including ASDL-GAN [6], UT-GAN [7], GMAN [8] and GACL [51]. In these methods, the GAN’s discriminator is composed of steganalysis networks, and the generator learns the definition of steganographic distortion based on the adversarial loss from the discriminator. This means that the learning of steganographic distortion is driven solely by the adversarial interaction with the steganalysis network.

However, as demonstrated in GMAN, current methods typically use the relatively weak steganalysis network XuNet as the discriminator. Stronger steganalysis networks such as YedNet can lead to gradient vanishing and training failure. These factors limit the performance of the generator. By introducing a new discriminator training strategy, GMAN employs multiple steganalyzers as discriminators to increase the security of stego images. Nonetheless, experimental results revealed that using three steganalyzers was less effective than using two, indicating that steganographic security does not necessarily improve with an increasing number of steganalyzers and can even decrease. Therefore, we pose the following question: In scenarios where improving the discriminator’s performance does not enhance steganographic security, how can GAN-based distortion learning methods further improve security?

We have observed that generated images exhibit certain fluctuation characteristics. For example, in Stable Diffusion, the input parameter “Classifier-Free Guidance”(CFG) scale controls the similarity between the text and the image. This value is typically set to 7 to balance similarity and image quality. When the CFG value is slightly changed, Stable Diffusion outputs two nearly identical images. As shown in Fig. 2 (b) and (c), when the CFG changes from 7.5000 to 7.5001, the differences between the two generated images are imperceptible to the human eye. To further explore the distribution properties of fluctuation images, we used Stable Diffusion to generate 11 images with CFG values of 7.5000, 7.5010, 7.5020, ..., and so on. We selected one of these images as the cover image and the remaining 10 as fluctuation images, using GMAN to generate the stego image corresponding to the

cover image. Fig. 3 (b) shows the average pixel differences between the cover image and the 10 fluctuation images, and Fig. 3 (c) shows the pixel difference between the cover and the stego image. For better visualization, the brightness in Fig. 3 (b) and (c) has been increased by 50 times. As depicted in Fig. 3, the pixel differences between fluctuation images and the cover image are concentrated in areas with complex textures, which closely aligns with the distribution of modifications introduced by steganography.

Based on these observations, we propose that the fluctuation characteristics of generated images can provide distribution information about the images, which we term the fluctuation distribution. **Inspired by the idea of natural steganography [46], we aim to disguise stego images as fluctuation images, i.e., modifications of steganography, disguised as fluctuation differences.** Therefore, we propose a steganographic cost learning method based on the fluctuations of generated images, named GIFDL. By using fluctuation images to provide side information, we ensure that stego images are not only indistinguishable from cover images but also difficult to distinguish from fluctuation images, which makes the distribution of stego images similar to that of fluctuation images, disguising the stego images as those produced under slight input parameter fluctuations.

B. Framework

As shown in Fig. 4, the overall framework of the proposed GIFDL consists of three components. In GIFDL, the U-Net of the Generator, as well as the two discriminators are trainable, while the T2I Model and the Simulator are fixed.

1) *Generative Model*: The generative model includes a T2I Model, for which we use the pre-trained Stable Diffusion as the T2I Model to generate the cover image C from the input text. We represent C as $C = (c_{i,j})^{H \times W}$ ($1 \leq i \leq H, 1 \leq j \leq W$), a grayscale image with height H and width W . By slightly altering the input parameter “CFG scale”, we obtain a series of fluctuation images F_1, F_2, \dots, F_N , where $F_k = (F_{i,j}^k)^{H \times W}$, $k = 1, 2, \dots, N$. In this paper, we set $N = 10$.

The T2I model is used solely for generating cover images and the corresponding fluctuation images; it does not participate in the training process. During training, the cover image is input into the generator to produce the stego image. In each batch, one image is randomly selected from the fluctuation images, denoted as $F_{ran} = (F_{i,j}^{ran})^{H \times W}$. F_{ran} is input into the discriminator to provide side information on the fluctuation distribution of the generated images.

We observed that fluctuation images F_k sometimes show significant differences from the cover image C , as shown in Fig. 5. When the difference between the cover image and the fluctuation image is too large, constraining the stego image to be similar to both at the same time becomes impossible. These large differences may provide incorrect information to the generator. Therefore, we introduce a parameter τ to limit the distance between the cover image and the fluctuation images:

$$MSE(C, F_k) \leq \tau, \quad (1)$$

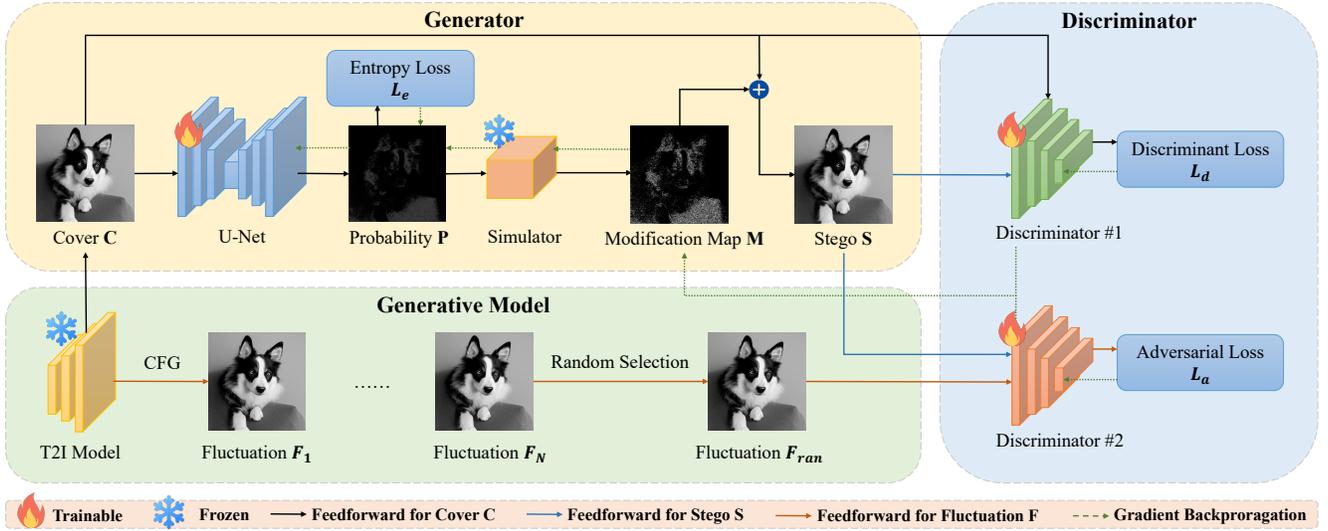


Fig. 4. The overall framework of the proposed GIFDL consists of three components: the Generative Model, the Generator, and the Discriminator. In the Generative Model, we use a T2I model to generate cover image C and corresponding fluctuation images $F_i, i = 1, 2, \dots$. From this series of fluctuation images, a random fluctuation image F_{ran} is selected to participate in the subsequent training of the generator and discriminator. In the Generator, we employ a U-Net architecture to generate the modification probability map P . Using a simulation embedder, we obtain the modification map M , which is further used to produce the stego image S . In the Discriminator, Discriminator #1 is used to distinguish between the cover image C and the stego image S , while Discriminator #2 distinguishes between the fluctuation image F_{ran} and the stego image S . It is important to note that F_{ran} is randomly selected in each training epoch and is not fixed.

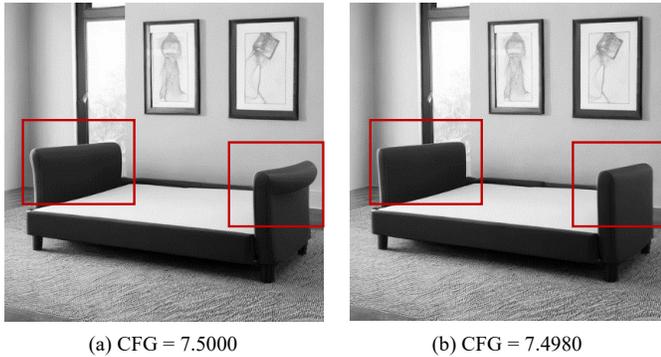


Fig. 5. (a) Image generated with $CFG = 7.5000$. (b) Image generated with $CFG = 7.4980$. There are significant differences between (a) and (b).

where for two images X and Y , the Mean Squared Error (MSE) is defined as follows:

$$MSE = \frac{1}{N} \sum_i \sum_j (X(i, j) - Y(i, j))^2, \quad (2)$$

where N is the total number of pixels in the image, and $X(i, j)$ and $Y(i, j)$ are the pixel values of images X and Y at position (i, j) , respectively. A smaller MSE indicates that the two images are more similar, meaning that their differences are smaller.

During training, if F_k and C satisfy the above MSE condition, the corresponding sample will participate in the subsequent training process; otherwise, F_k will be regenerated with a different “CFG” scale.

2) *Generator*: The generator takes the cover image as input and outputs the stego image, which can be divided into two parts:

(1) **Generate the modification probability map**: This process converts the input cover image $C = (c_{i,j})^{H \times W}$ into an embedding probability $P = (p_{i,j})^{H \times W}$, where $p_{i,j}$ represents the probability that the pixel value $c_{i,j}$ will be modified by ± 1 due to message embedding at pixel location (i, j) . In this work, we constrain the probability of each pixel value $c_{i,j}$ being modified by $+1$ (i.e., $p_{i,j}^{+1}$) to be equal to the probability of being modified by -1 (i.e., $p_{i,j}^{-1}$), as shown below:

$$p_{i,j}^{+1} = p_{i,j}^{-1} = \frac{p_{i,j}}{2}. \quad (3)$$

Following the setup in UT-GAN [7], we utilize a U-Net-based architecture for this image-to-image task (i.e., from C to P). The U-Net consists of 15 blocks and a deconvolution layer. The first 8 blocks perform down-sampling, and the remaining 7 perform up-sampling. The output of the i -th block ($i = 1, 2, \dots, 7$) is concatenated with the output of the $(16 - i)$ -th block and passed into the $(17 - i)$ -th block. Each block contains two convolutional layers, followed by batch normalization and LeakyReLU activation.

(2) **Simulated embedding**: This step simulates STC embedding to generate the stego image based on the probabilities P obtained in the first step. First, random noise $R = (r_{i,j})^{H \times W}$ is generated, where $r_{i,j}$ is independently and identically distributed (i.i.d.) over a uniform interval $[0, 1]$. Comparing $p_{i,j}$ and $r_{i,j}$, the modification map $M = (m_{i,j})^{H \times W}$ in the simulated embedding is defined as follows:

$$m_{i,j} = \begin{cases} -1, & \text{if } r_{i,j} < p_{i,j}^{-1}, \\ +1, & \text{if } r_{i,j} > 1 - p_{i,j}^{+1}, \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

Since the above piecewise function is non-differentiable, we follow the setup in UT-GAN [7] and use the Double-

Tanh function as an embedding simulator during training to obtain the modification map $M = (m_{i,j})^{H \times W}$. The double-tanh function serves as a differentiable approximation of the discrete piecewise function, which facilitates the backpropagation process in the GAN. The double-tanh function is defined as:

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

where, $m_{i,j}$ in the interval $[-1, 1]$ represents the simulated modification amount, and the parameter γ controls the precision of the simulation. We set $\gamma = 60$ as in UT-GAN. Finally, we obtain the stego image as $S = C + M$.

Once the training is complete, the embedding costs ρ for the input image C can be calculated as follows:

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

Based on the above embedding costs, we then use STC, rather than the embedding simulator, to perform the actual message embedding, resulting in the final stego image S while minimizing the total embedding cost.

3) *Discriminator*: The discriminator consists of two distinct steganalysis networks, namely Discriminator #1 and Discriminator #2. Discriminator #1 differentiates between cover images and stego images, while Discriminator #2 distinguishes between fluctuation images F_{ran} and stego images. Experimental observations indicate that steganalysis networks generally find it easier to distinguish between fluctuation images F_{ran} and stego images than between cover images and stego images. Consequently, we empirically select the more powerful steganalysis network, Yed-Net [19], as Discriminator #1, and the less powerful Xu-Net [26] as Discriminator #2. We name this strategy of using two different discriminators to perform two different categorization tasks as ‘‘assignment’’. We will explore the design rationale of ‘‘assignment’’ in Section IV-C4 and Section IV-C5.

To balance the performance of the two discriminators given their different classification tasks, we apply the training strategy from GMAN [8], which updates the parameters of the weaker discriminator in each iteration, preventing the vanishing gradient problem during training. The specific training process of the discriminators can be divided into the following two steps:

(1) **Compute cross-entropy**: For each steganalysis network D_i within the discriminators, the output of D_i is the softmax layer’s classification result, represented as a two-dimensional normalized vector. The classification performance of the steganalysis network D_i is evaluated by binary cross-entropy E_i , which is defined as follows:

$$E_1(C, S) = -z_0 \log(D_1(C)) - z_1 \log(D_1(S)), \quad (7)$$

$$E_2(F_{ran}, S) = -z_0 \log(D_2(F_{ran})) - z_1 \log(D_2(S)), \quad (8)$$

Algorithm 1 Training Steps of GIFDL

Input: cover image C , fluctuation image F_i , $i \in \{1, \dots, N\}$, number of epochs N_e , learning rate η .

Output: $\theta_g, \theta_{d_1}, \theta_{d_2}$.

- 1: Initialize the generator G , and the discriminator D_1, D_2
 - 2: **for** $epoch$ in $\{1, \dots, N_e\}$ **do**
 - 3: $P \leftarrow G(C; \theta_g)$
 - 4: Compute M by double-tanh function
 - 5: $S \leftarrow C + M$
 - 6: Randomly select $F_{ran} \in \{F_1, \dots, F_N\}$
 - 7: Compute $E_1(C, S)$ and $E_2(F_{ran}, S)$ by Eq. 7 and Eq. 8.
 - 8: **if** $E_1 \leq E_2$ **then**
 - 9: Update D_2 by $\theta_{d_2} \leftarrow \theta_{d_2} - \eta \nabla_{\theta_{d_2}} E_2$
 - 10: **else**
 - 11: Update D_1 by $\theta_{d_1} \leftarrow \theta_{d_1} - \eta \nabla_{\theta_{d_1}} E_1$
 - 12: **end if**
 - 13: Compute $l_a = E_1 + \lambda E_2$
 - 14: Compute l_e by Eq. 13
 - 15: $l_G \leftarrow -\alpha l_a + \beta l_e$
 - 16: Update G by $\theta_g \leftarrow \theta_g - \eta \nabla_{\theta_g} l_G$
 - 17: Learning rate η decay
 - 18: **end for**
 - 19: **return** $\theta_g, \theta_{d_1}, \theta_{d_2}$.
-

where, $D_1(C), D_1(S)$ denote the classification scores of discriminator #1 for cover C and stego S respectively, and $D_2(S), D_2(F_{ran})$ denote the classification scores of discriminator #2 for stego S and fluctuation F_{ran} respectively. The vectors $z_0 = (1, 0)^T$ and $z_1 = (0, 1)^T$ are the true labels.

(2) **Update Discriminators and Generator**: A larger binary cross-entropy indicates a weaker steganalysis network. Suppose in a particular iteration that $E_2 \leq E_1$, which suggests that Discriminator #1 is weaker than Discriminator #2. Note that the classification performance of the steganalysis networks may vary across iterations. To enhance discriminative performance, we only update the parameters of the weaker steganalysis network (i.e., Discriminator #1) in each iteration, keeping the stronger network (i.e., Discriminator #2) unchanged. This method enables the steganalysis networks to improve gradually and progressively. To further enhance the generator’s performance, both discriminators are used to guide its updates in each iteration.

C. Loss Function

Below are the detailed descriptions of the loss functions in GIFDL, including the discriminator loss l_{D_1}, l_{D_2} and the generator loss l_G :

(1) **Discriminator loss l_{D_1} and l_{D_2}** : The loss functions of Discriminator #1 and Discriminator #2 are defined as l_{D_1} and l_{D_2} respectively:

$$l_{D_1} = -z_0 \log(D_1(C)) - z_1 \log(D_1(S)), \quad (9)$$

$$l_{D_2} = -z_0 \log(D_2(F_{ran})) - z_1 \log(D_2(S)). \quad (10)$$

As described in Section III-B3, in each iteration, only the parameters of the relatively weakly performing discriminator

are updated. Thus, in each iteration, if $l_{D_1} > l_{D_2}$, the parameters of Discriminator #1 will be updated using the loss function l_{D_1} , and the parameters of Discriminator #2 will be kept unchanged. Otherwise, the parameters of Discriminator #2 will be updated using the loss function l_{D_2} , and the parameters of Discriminator #1 will be kept unchanged.

(2) **Generator loss l_G** : The generator’s goal is to embed messages into cover images imperceptibly, making the resulting stego images difficult for the discriminator’s steganalysis networks to detect. Accordingly, the generator loss l_G is defined as:

$$l_G = -\alpha \cdot l_a + \beta \cdot l_e, \quad (11)$$

where l_G consists of two components. The first term l_a represents the adversarial loss against the discriminator, and the second term, l_e , is the entropy loss, which ensures effective payload embedding. The weights of these terms, α and β , are set as in UT-GAN [7], with $\alpha = 1$ and $\beta = 10^{-7}$. Specifically, l_a and l_e are defined as follows:

$$\begin{aligned} l_a &= E_1(C, S) + \lambda E_2(F_{ran}, S) \\ &= (-z_0 \log(D_1(C)) - z_1 \log(D_1(S))) + \\ &\quad \lambda \cdot (-z_0 \log(D_2(F_{ran})) - z_1 \log(D_2(S))), \end{aligned} \quad (12)$$

$$l_e = - \left(\sum_{\forall i,j} \sum_{\forall m} \log_2(p_{i,j}^{(m)}) - (H \times W \times q) \right)^2, \quad (13)$$

where, H and W represent the height and width of the input image, respectively, with $1 \leq i \leq H$, $1 \leq j \leq W$. The variable m denotes the embedding modification, where $m \in \{-1, 0, +1\}$, and q represents the embedding payload. The parameter λ is used to weigh the adversarial losses from two discriminators.

During the training phase, these loss functions l_D and l_G are utilized to compute gradients and update the model parameters for both the discriminator and generator in the proposed GIFDL model. The training steps of the model are outlined in Algorithm 1.

D. The Applicable Scope of GIFDL

The design of GIFDL depends on the fluctuation characteristics of generated images. To the best of our knowledge, we have observed that text-to-image diffusion models exhibit fluctuation characteristics in the generated images when the value of the input parameter “CFG scale” changes. The CFG scale is used in mainstream text-to-image diffusion models, such as Stable Diffusion and Midjourney. Therefore, the applicable scope of GIFDL can be defined as text-to-image diffusion models with the CFG scale.

IV. EXPERIMENTS

A. Experimental Setup

Since GIFDL uses generated images as steganographic covers, we employ the widely popular image generation model Stable Diffusion v1.4 to test GIFDL’s effectiveness. Stable Diffusion can generate images that match the input text (called the “prompt”), where the “CFG scale” parameter controls the

degree of alignment between the generated image and the input text, and the “seed” parameter determines the specific content of the generated image. We use 1,000 categories from the ImageNet [39] as prompts for Stable Diffusion, assigning each category 10 different random seeds. Additionally, the “CFG scale” parameter is set to 7.5000. As a result, each category contains 10 images with varying content, and the resulting dataset consists of 10,000 images, each of size 512×512 , saved in PGM format. We name this dataset IN_{train} .

In section III-A, we observe fluctuations in the generation model, where Stable Diffusion, given the same prompt and seed, could produce two highly similar images by slightly adjusting the “CFG scale” value. The pixel differences between the two images are mainly concentrated in the texture areas. Therefore, we keep the prompt and seed constant and only alter the “CFG scale” of IN_{train} , with values set to 7.4950, 7.4960, 7.4970, 7.4980, 7.4990, 7.5010, 7.5020, 7.5030, 7.5040, and 7.5050. This results in 10 fluctuation datasets, consistent in content with IN_{train} , which we refer to as IN_{flu} . In subsequent experiments, we train the baseline model using the IN_{train} dataset and train the proposed GIFDL using both IN_{train} and IN_{flu} datasets.

Additionally, we generate another dataset, named IN_{test} , using 10 different random seeds not included in IN_{train} , including 10,000 images. All experiments are conducted on four NVIDIA GEFORCE RTX 2080 Ti GPU cards.

B. Evaluation Metrics

1) *Resistance to Steganalysis*: The security performance of steganographic methods is usually evaluated using the detection error rate of steganalyzers. The detection error rate P_E is defined as:

$$P_E = \frac{1}{2} (P_{FA} + P_{MD}), \quad (14)$$

where P_{FA} and P_{MD} represent the false-alarm (FA) rate and the missed detection (MD) rate of steganalyzers, respectively.

C. Experimental Results and Analysis

1) *The optimal value of the parameter τ* : In the design of GIFDL, the adversarial loss l_a of the Generator consists of two parts that constrain the generator in two ways: (1) the stego image should be indistinguishable from the cover image, and (2) the stego image should be indistinguishable from the randomly chosen fluctuation image. To balance the impact of two constraints on steganographic security, we introduce a parameter λ in section III-C to regulate the contribution of $E_2(F_{ran}, S)$ to the total generator loss l_a . To determine the optimal value for λ , we conduct the following experiment:

We set the value of λ to 1, 2, 4, and 8, respectively. After training GIFDL, we use 10,000 grayscale images from the IN_{test} dataset as the cover images for testing the security of GIFDL. Specifically, we generate the corresponding steganographic cost through GIFDL and embed the secret message via STC, with the payload set to 0.4 bpp. We obtain 10,000 stego images via the above operations. We then divide the 10,000 cover-stego image pairs into three subsets of sizes 4,000,

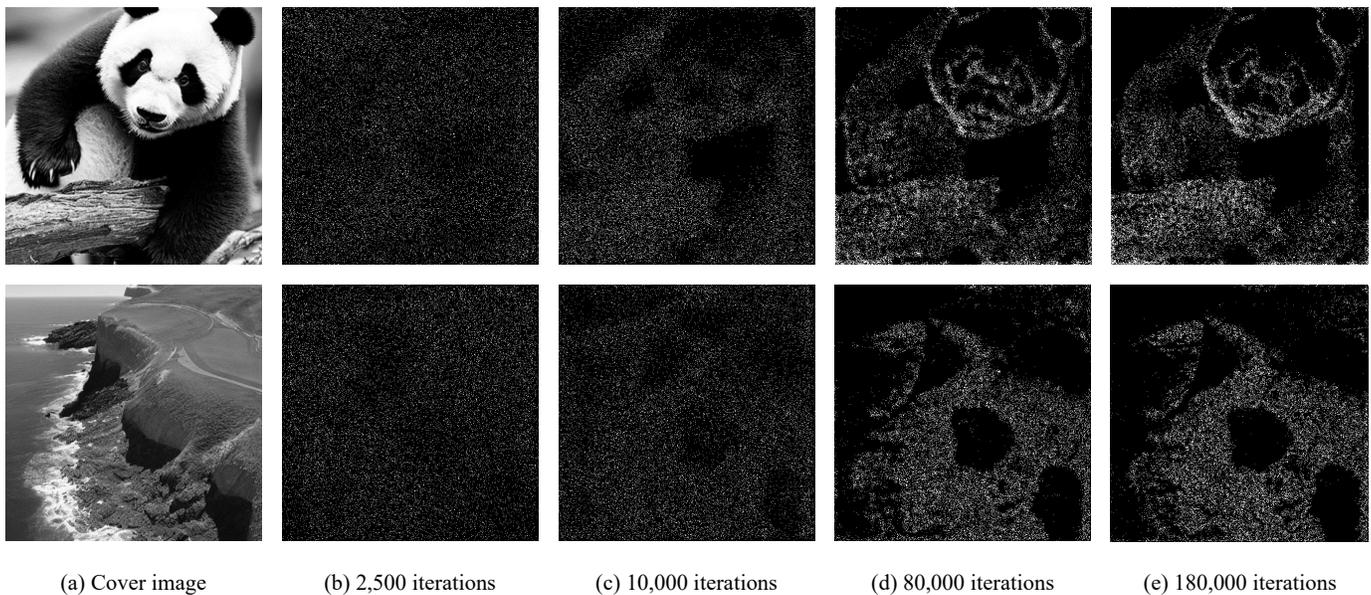


Fig. 6. Cover images (generated by Stable Diffusion) and modification maps with different number of iterations.

TABLE I
DETECTION ERROR RATE(%) OF SRM, COVNET, AND LWENET UNDER DIFFERENT λ . A HIGHER RATE INDICATES HIGHER SECURITY.

Steganalyzer	Parameter	0.1 bpp	0.2 bpp	0.3 bpp	0.4 bpp	Average
SRM	$\lambda = 1$	46.74	44.09	38.53	33.99	40.84
	$\lambda = 2$	46.49	42.68	39.43	33.35	40.49
	$\lambda = 4$	45.83	42.08	38.66	33.80	40.09
	$\lambda = 8$	46.09	42.72	38.92	34.42	40.54
CovNet	$\lambda = 1$	45.07	41.69	35.31	29.40	37.87
	$\lambda = 2$	44.25	40.01	36.96	29.66	37.72
	$\lambda = 4$	43.41	40.87	34.20	30.34	37.21
	$\lambda = 8$	40.44	39.46	36.56	31.82	37.07
LWENet	$\lambda = 1$	43.35	42.36	36.49	30.52	38.18
	$\lambda = 2$	42.58	40.80	38.57	30.45	38.10
	$\lambda = 4$	43.48	40.95	34.48	29.97	37.22
	$\lambda = 8$	40.17	38.70	36.99	30.77	36.66

1,000, and 5,000, which are used as the training, validation, and test datasets, respectively.

Finally, we train and test three steganalyzers (SRM [43], CovNet [15], and LWENet [11]) using the above datasets, and record the detection error rate (P_E) on the test dataset. The experimental results, shown in Table I, indicate that when $\lambda = 1$, GIFDL exhibits the best resistance against the three steganalyzers, achieving optimal security performance. Therefore, in all subsequent experiments, we set the parameter λ to 1 for GIFDL. Additionally, we observe that a larger λ results in higher security as the payload increases. This phenomenon can be explained by the fact that a larger λ gives greater weight to the role of fluctuation images, and the fluctuation distribution can provide more modification regions for steganographic modifications, which are also relatively secure, i.e., a larger

λ offers more optional regions for steganography. When the payload increases, the security is increased compared with the lower λ because there are more “secure” locations to modify.

As shown in Fig. 6, as the number of training iterations of GIFDL increases, the steganographic modifications shift from random modifications to concentrate on regions with complex image textures. This indicates that the training is effective and that the performance of the generator gradually improves.

2) *Security Performance*: In this section, we compare the security performance of GIFDL with that of other steganographic cost learning methods such as SUNIWARD [2], HILL [3], GMAN [8], and GACL [51]. Among them, the distortion definitions of SUNIWARD and HILL are heuristic and do not require pretraining. First, we train GMAN and GACL using the IN_{train} dataset, and the proposed GIFDL

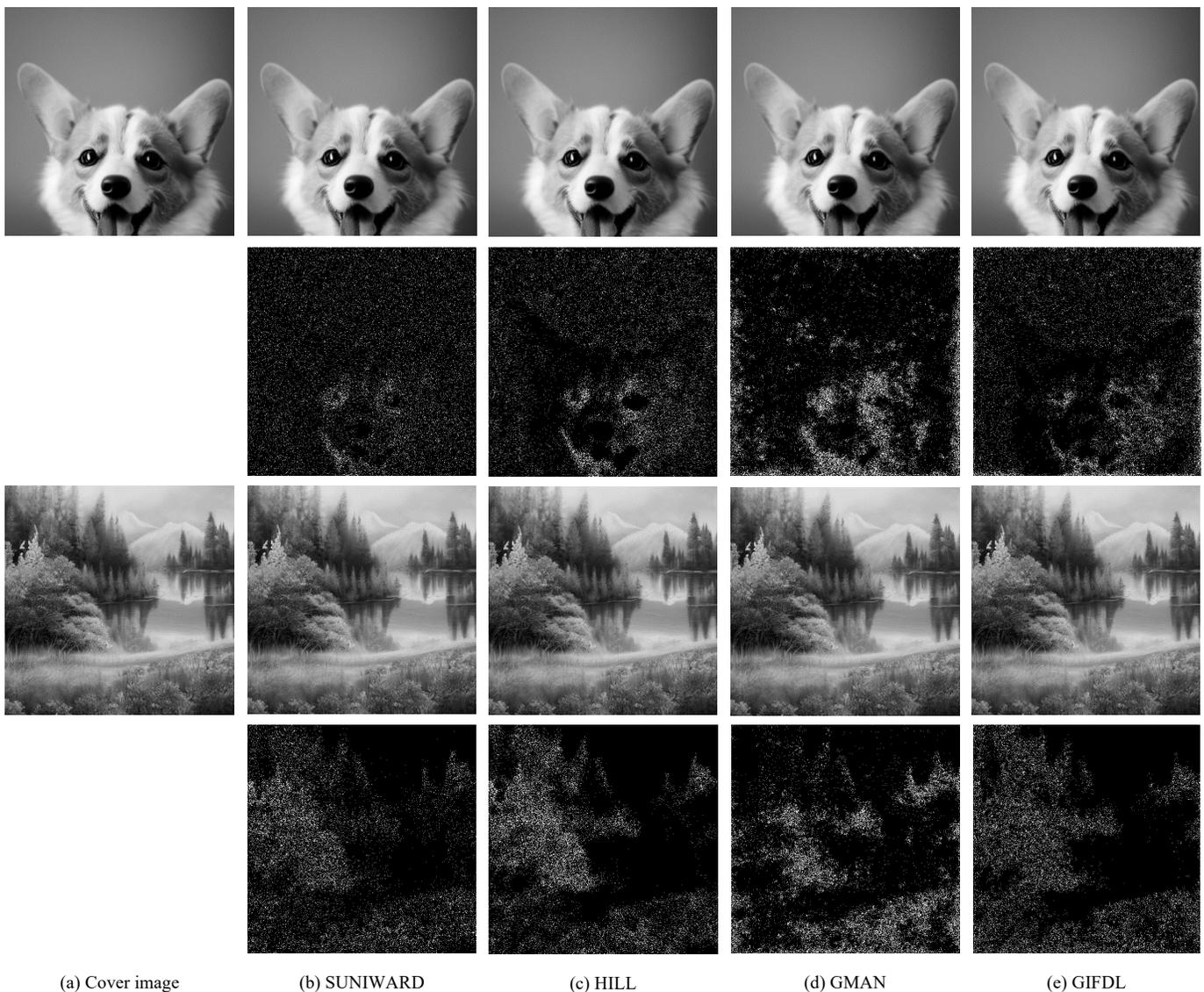


Fig. 7. Cover images, stego images (first row, third row), and corresponding modification maps (second row, fourth row).

using the IN_{train} and IN_{flu} datasets. Next, we use the IN_{test} dataset as the cover images, and use SUNIWARD, HILL, GMAN, GACL, and GIFDL, respectively, with payloads of 0.1 bpp, 0.2 bpp, 0.3 bpp, 0.4 bpp respectively to obtain the corresponding stego images. The stego images and modification maps of the cover images with the four steganography methods are shown in Fig. 7. Unlike other steganography methods, the steganographic modifications of GIFDL are not only concentrated in the complex regions of the image texture but also distributed in the regions of fluctuation, which is very similar to Fig. 3(b). We use the above cover-stego pairs to train three steganalyzers, SRM, CovNet, and LWENet, respectively, where the cover-stego pairs are divided into three datasets of sizes 4000, 1000, and 5000, which are used for steganalyzer training, validation, and testing, respectively. The detection error rates (P_E) of the steganalyzers are shown in Table. II, where it can be observed that all four steganographic methods are more easily detected by the steganalysis as the payload

increases. Moreover, P_E for GIFDL is significantly lower than that of the comparison methods, and the average P_E is 3.30% higher than that of the suboptimal GMAN. Therefore, GIFDL has the highest security in terms of resistance to steganalysis.

3) *Security performance on real datasets*: In this section, we explore the generalization performance of GIFDL on different datasets. We use the dataset IN_{train} to train GIFDL and the comparison methods. Then we test them on the dataset DiffusionDB [48] and JourneyDB [52]. DiffusionDB is the first large-scale text-to-image prompt dataset. It contains 14 million images generated by Stable Diffusion using prompts and hyperparameters specified by real users. We randomly select 10,000 images of size 512×512 from the DiffusionDB dataset to form our test dataset, which is named *DDB*. Meanwhile, We obtain 10,000 Midjourney-generated images from the JourneyDB dataset. These images have a resolution of 1024×1024 and are then resized to 512×512 using the *imresize* function with the bicubic interpolation

TABLE II
DETECTION ERROR RATE(%) OF SRM, COVNET, AND LWENET UNDER DIFFERENT PAYLOAD. A HIGHER RATE INDICATES HIGHER SECURITY.

Steganalyzer	Method	0.1 bpp	0.2 bpp	0.3 bpp	0.4 bpp	Average
SRM	S-UNIWARD	42.20	38.28	32.66	27.29	35.11
	HILL	43.70	37.15	33.24	27.72	35.45
	GACL	44.16	38.62	35.57	33.68	38.01
	GMAN	45.43	41.57	34.16	30.78	37.99
	GIFDL	46.74	44.09	38.53	33.99	40.84
CovNet	S-UNIWARD	38.65	33.06	25.56	18.87	29.04
	HILL	39.09	32.70	27.23	20.73	29.94
	GACL	39.95	34.10	33.89	26.95	33.72
	GMAN	41.59	33.64	30.69	27.10	33.26
	GIFDL	45.07	41.69	35.31	29.40	37.87
LWENet	S-UNIWARD	36.20	30.24	27.15	21.25	28.71
	HILL	39.56	31.31	29.18	23.12	30.79
	GACL	40.57	36.94	28.64	22.44	32.15
	GMAN	42.43	39.64	33.66	27.25	35.75
	GIFDL	43.35	42.36	36.49	30.52	38.18



Fig. 8. Differences between datasets *IN*, *DDB* and *MDB*. (a) The prompts used in *IN* come from class labels of ImageNet, and the images in *IN* are usually single objects with simpler textures. (b) The prompts used in *DDB* and *MDB* are from real users, and the images in *DDB* are more artistic and have more complex textures.

method and the default parameter settings in MATLAB. We name the above dataset the Midjourney Database (*MDB*). Fig. 8 shows the main differences between the training set *IN* and the test set *DDB*, *MDB*. The images in *DDB* and *MDB* usually along with longer and more complex prompts, and the models used are usually stable diffusion v1.5, v2.0, and Midjourney. We use the *DDB* and *MDB* datasets as cover images and use GIFDL and comparison methods to obtain the corresponding stego images, respectively. Cover-stego pairs are partitioned into three datasets of sizes 4000, 1000, and 5000, which are used for training, validation, and testing of the steganalyzers, respectively. The experimental results are shown in Table III, where it can be observed that GIFDL maintains the resistance to steganalysis performance on the real dataset and possesses good generalizability to the stable diffusion family of models. Meanwhile, GIFDL maintains a certain generalization performance on the image generation model Midjourney. Although the performance of GIFDL shows some degradation on Midjourney, it is still significantly better than

the comparison methods.

TABLE III
DETECTION ERROR RATES(%) OF SRM, COVNET, AND LWENET ON DIFFERENT DATASETS. A HIGHER RATE INDICATES HIGHER SECURITY.

Dataset	Method	SRM	CovNet	LWENet
<i>IN</i>	SUNIWARD	27.29	18.87	21.25
	HILL	27.72	20.73	23.12
	GACL	33.68	26.95	22.44
	GMAN	30.78	27.10	27.25
	GIFDL	33.99	29.40	30.52
<i>DDB</i>	SUNIWARD	27.86	23.77	23.46
	HILL	31.10	27.11	26.56
	GACL	33.44	28.58	24.06
	GMAN	30.31	27.51	28.97
	GIFDL	35.85	31.60	33.24
<i>MDB</i>	SUNIWARD	23.10	19.45	16.21
	HILL	26.56	22.40	16.63
	GACL	24.19	18.25	15.27
	GMAN	23.79	19.31	17.99
	GIFDL	26.34	25.99	24.99

4) *Ablation study*: In this section, we explore the role of each component of GIFDL. For this purpose, we design several variants of GIFDL in Table IV. Among them, Variant #1 refers to GMAN, “Fluctuation image” denotes the use of fluctuation images during training, and “Threshold” denotes the use of threshold τ to filter the fluctuation images. “Discriminator strategy” indicates that the assigned discriminator update strategy is used, i.e., two different steganalyzers are used as two discriminators to discriminate cover-stego pairs and

TABLE IV
SEVERAL VARIANTS OF GIFDL.

Variant	Fluctuation image	Threshold	Discriminator strategy
Variant #1	×	×	×
Variant #2	✓	×	×
Variant #3	✓	✓	×
Variant #4	✓	✓	✓

fluctuation-stego pairs, respectively. If this item is unchecked, it means that two identical steganalyzers are used as two discriminators to discriminate cover-stego pairs and fluctuation-stego pairs, respectively. We use the dataset IN_{train} to train the above variants and test the performance of these variants against steganalysis on the dataset IN_{test} , and the experimental results are shown in Table V.

Based on the experimental results of Variant #1 and Variant #2, using fluctuation images without a threshold to constrain their differences from cover images during training does not significantly improve the security of steganographic images. This can be explained from two perspectives. First, the majority of fluctuation images and cover images are visually indistinguishable, so their differences are concentrated in regions with complex textures. These images are involved in the training process, which helps to improve steganographic security. Second, without a threshold restriction, some fluctuation images with substantial differences from the cover image (e.g., Fig. 5) are included in the training. When these highly different images are involved in training, GIFDL evaluates the content change regions as suitable areas for steganographic modification. However, in reality, these regions are very smooth and not suitable for steganographic modification. These highly different images neutralize the gains brought by learning distortion from similar fluctuation images, which is why the experimental results of Variant #1 and Variant #2 are similar. When we introduce a threshold into the framework (i.e., Variant #3), these differentiated images are excluded, and the training set only includes fluctuation images that we consider ideal. Naturally, the performance is improved as a result. The experimental results for Variant #3 and Variant #4 demonstrate that the steganographic security can be effectively improved by using the assigned discriminator update strategy, which suggests that it is reasonable to use two discriminators that differ in performance. In fact, by observing the change in the loss of the discriminator in Variant #3, it can be found that the loss of the discriminator used to discriminate the fluctuation-stego pairs always reduces to 0 very quickly, which results in the generator not being able to learn effective knowledge during the adversarial process, i.e., the gradient disappears. In conclusion, the use of “Fluctuation image” can provide additional information for steganography, which can improve the steganographic security under the “Threshold” constraint. Moreover, the allocation-based discriminator update strategy can balance the contributions of cover images and fluctuation images to further improve steganographic security.

TABLE V
DETECTION ERROR RATES(%) OF SRM, COVNET, AND LWENET UNDER DIFFERENT VARIANTS. A HIGHER RATE INDICATES HIGHER SECURITY.

Variant	SRM	CovNet	LWENet
Variant #1	30.78	27.10	27.25
Variant #2	31.81	26.80	27.45
Variant #3	32.56	29.03	29.71
Variant #4	33.99	29.40	30.52

5) *An alternative discriminator training strategy*: In section III-B, our design of the discriminator is such that two different discriminators are applied to different discrimination tasks. That is, Yed-Net is used to discriminate between cover and stego, and XuNet is used to discriminate between flu and stego. It has been shown in section IV-C4 that the use of two discriminators with different performances better balances the contributions of the cover and the fluctuation images, compared with the use of the same discriminator. However, there is naturally a more mundane design where two discriminators are applied to the same discrimination task. This is done by Yed-Net and XuNet, which are used to discriminate not only cover and stego but also flu and stego. Thus, we redefine the discriminator loss:

$$l_D = \max\{E_i(F_{ran}, S) + \lambda' E_i(C, S), i = 1, 2\}, \quad (15)$$

$$l_a = \min\{E_i(F_{ran}, S) + \lambda' E_i(C, S), i = 1, 2\}, \quad (16)$$

where we use a different training strategy: the two discriminators have the same classification task, i.e., discriminating both cover-stego pairs and fluctuation-stego pairs, using the parameter λ' to weigh the contributions of both. In each epoch, we update the parameters of the discriminator with the higher loss, leaving the other discriminator unchanged. At the same time, the adversarial loss l_a of the steganography generator is the smaller loss. We call this experimental setup GIFDL*.

We use the GIFDL and GIFDL* models with λ' values of 1, 2, 4, and 8 to generate 10,000 stego images respectively, resulting in five stego datasets. For each stego dataset, we divide the 10,000 stego images into training, validation, and test datasets, with sizes of 4000, 1000, and 5000, respectively. Subsequently, we use these datasets to train three state-of-the-art steganalysis networks, SRM, CovNet, and LWENet. The experimental results are presented in Table VI. This mundane design method does not achieve better security performance, which can be explained by the fact that the simultaneous execution of two tasks by each discriminator leads to a degradation of its performance under both tasks. In contrast, our proposed method of assigning different discriminators to each of the two tasks is able to maximize the performance of the discriminators, and thus the performance of the generator, while taking into account the differences in the cover and the fluctuation images.

6) *Combined with Volatility Cost*: In this section, we combine GIFDL and GMAN with the volatility cost (VC) proposed

TABLE VI

DETECTION ERROR RATES(%) OF SRM, COVNET, AND LWENET UNDER DIFFERENT STRATEGIES. A HIGHER RATE INDICATES HIGHER SECURITY.

Method	SRM	CovNet	LWENet
GIFDL	33.99	29.40	30.52
GIFDL* with $\lambda' = 1$	31.96	27.55	26.66
GIFDL* with $\lambda' = 2$	32.10	26.21	27.24
GIFDL* with $\lambda' = 4$	32.85	28.60	28.93
GIFDL* with $\lambda' = 8$	31.91	27.95	27.31

in [47] as a way to explore the effectiveness of GIFDL in capturing the fluctuations of generated images via deep learning networks. To this end, we generate steganographic costs on the dataset IN_{test} via GMAN and GIFDL, respectively, and compute the volatility costs of the dataset IN_{test} in the manner presented in [47]. The original steganographic cost is denoted as ρ^o , the volatility cost as ρ^v , and the steganographic cost after combining is denoted as ρ^c . The above process can be expressed as follows:

$$\begin{aligned} \rho^c(+1) &= \beta \cdot \rho^v(+1) + (1 - \beta) \cdot (\alpha \cdot \rho^o(+1)), \\ \rho^c(-1) &= \beta \cdot \rho^v(-1) + (1 - \beta) \cdot (\alpha \cdot \rho^o(-1)), \end{aligned} \quad (17)$$

where $\rho^c(+1)$ and $\rho^c(-1)$ represent the costs associated with +1 modification and -1 modification, β is the hyperparameter that determines the proportion of volatility cost, and α is the scaling factor that equalizes the mean of ρ^o with the volatility cost ρ^v . α is calculated as follows:

$$\alpha = \frac{\sum_{i,j} ([\rho_{ij}^v \neq \text{wetcost}] \cdot \rho_{ij}^v) / \sum_{i,j} [\rho_{ij}^v \neq \text{wetcost}]}{\sum_{i,j} ([\rho_{ij}^o \neq \text{wetcost}] \cdot \rho_{ij}^o) / \sum_{i,j} [\rho_{ij}^o \neq \text{wetcost}]}, \quad (18)$$

where α represents the ratio between the average value of the volatility cost and the original cost, the Iverson bracket $[Q]$ is defined to be 1 if the logical expression Q is true and 0 otherwise, and “wetcost” represents the cost tending towards infinity, which is excluded to avoid substantial impacts on the mean. It has been experimentally demonstrated in [47] that optimal performance is achieved with $\beta = 0.15$, so we take $\beta = 0.15$.

We combine the volatility cost with the steganographic cost of GMAN, named GMAN+VC, and similarly, combine the volatility cost with the steganographic cost of GIFDL, named GIFDL+VC. We utilize the above costs to embed the secret message via STC [14] to obtain the corresponding stego image. The detection error rates of the three steganalyzers are shown in Table VII. Compared with GMAN, the steganalysis error rate of GMAN+VC is increased by 8.77% on average. Also, GMAN+VC improves by 5.85% on average compared to GIFDL. In addition, GIFDL+VC improves 9.16% on average compared to GIFDL.

In VC proposed by Zhang et al. [47], the pixel differences between fluctuation images are modeled as Gaussian distributions. However, there is a difference between the actual distribution and the standard Gaussian distribution. In GIFDL,

we directly learn the pixel distribution of fluctuation images from training data through a deep learning network, rather than simply assuming it to be a Gaussian distribution. Due to the differences in method design, new knowledge can possibly be learned, and experimental results have also verified that GIFDL and VC can be complementary.

TABLE VII

DETECTION ERROR RATES(%) OF SRM, COVNET, AND LWENET UNDER DIFFERENT COMBINATIONS. A HIGHER RATE INDICATES HIGHER SECURITY.

Method	SRM	CovNet	LWENet	Average
GMAN	30.78	27.10	27.25	28.38
GMAN + VC	36.09	38.86	36.51	37.15
GIFDL	33.99	29.40	30.52	31.30
GIFDL + VC	41.27	40.64	39.46	40.46

V. CONCLUSION

To further enhance steganographic security, we propose a steganographic distortion learning method based on the fluctuations of generated images. Specifically, we observe that generated images exhibit a fluctuation distribution, and we use this as side information to guide the generator in learning steganographic distortion. To avoid the problem of gradient vanishing during training and to fully utilize the performance of the discriminators, we introduce a training strategy called “assignment”, which assigns different tasks to the two discriminators and balances their performance by updating the parameters alternately. Experimental results indicate that compared with GMAN, GIFDL can further improve steganographic security and has some generalizability, maintaining performance on new datasets.

In our experiments, we observed that the differences between fluctuation images and cover images are not entirely concentrated in areas with complex textures; differences often exist in the background regions as well. This is due to the randomness introduced by the noise process in DDPMs. Future research will focus on exploring whether these random differences could potentially reduce steganographic security. Additionally, we will investigate how to better utilize the fluctuation characteristics of generated images, considering a more refined estimation of their distribution. Overall, we propose a new method to enhance steganographic security using generated images and explore the direction of steganography via black-box generative models.

REFERENCES

- [1] V. Holub and J. Fridrich, “Designing steganographic distortion using directional filters,” in *2012 IEEE International workshop on information forensics and security (WIFS)*. IEEE, 2012, pp. 234–239.
- [2] 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.
- [3] B. Li, M. Wang, J. Huang, and X. Li, “A new cost function for spatial image steganography,” in *2014 IEEE International conference on image processing (ICIP)*. IEEE, 2014, pp. 4206–4210.

- [4] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 221–234, 2015.
- [5] L. Guo, J. Ni, W. Su, C. Tang, and Y.-Q. Shi, "Using statistical image model for jpeg steganography: Uniform embedding revisited," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2669–2680, 2015.
- [6] 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.
- [7] J. Yang, D. Ruan, J. Huang, X. Kang, and Y.-Q. Shi, "An embedding cost learning framework using gan," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 839–851, 2019.
- [8] D. Huang, W. Luo, M. Liu, W. Tang, and J. Huang, "Steganography embedding cost learning with generative multi-adversarial network," *IEEE Transactions on Information Forensics and Security*, 2023.
- [9] J. Tan, X. Liao, J. Liu, Y. Cao, and H. Jiang, "Channel attention image steganography with generative adversarial networks," *IEEE transactions on network science and engineering*, vol. 9, no. 2, pp. 888–903, 2021.
- [10] L. Meng, X. Jiang, Z. Zhang, Z. Li, and T. Sun, "A robust coverless image steganography based on an end-to-end hash generation model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, pp. 3542–3558, 2023. [Online]. Available: <https://api.semanticscholar.org/CorpusID:255251897>
- [11] S. Weng, M. Chen, L. Yu, and S. Sun, "Lightweight and effective deep image steganalysis network," *IEEE Signal Processing Letters*, vol. 29, pp. 1888–1892, 2022.
- [12] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.
- [13] K. A. Zhang, A. Cuesta-Infante, L. Xu, and K. Veeramachaneni, "Steganogan: High capacity image steganography with gans," *arXiv preprint arXiv:1901.03892*, 2019.
- [14] T. Filler, J. Judas, and J. Fridrich, "Minimizing additive distortion in steganography using syndrome-trellis codes," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 920–935, 2011.
- [15] X. Deng, B. Chen, W. Luo, and D. Luo, "Fast and effective global covariance pooling network for image steganalysis," in *Proceedings of the ACM workshop on information hiding and multimedia security*, 2019, pp. 230–234.
- [16] X. Liu, Z. Ma, J. Ma, J. Zhang, G. Schaefer, and H. Fang, "Image disentanglement autoencoder for steganography without embedding," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 2303–2312.
- [17] P. Wei, S. Li, X. Zhang, G. Luo, Z. Qian, and Q. Zhou, "Generative steganography network," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 1621–1629.
- [18] Y. Peng, D. Hu, Y. Wang, K. Chen, G. Pei, and W. Zhang, "Stegadp-m: Generative image steganography based on denoising diffusion probabilistic model," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 7143–7151.
- [19] M. Yedroudj, F. Comby, and M. Chaumont, "Yedroudj-net: An efficient cnn for spatial steganalysis," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 2092–2096.
- [20] R. J. Anderson and F. A. Petitcolas, "On the limits of steganography," *IEEE Journal on selected areas in communications*, vol. 16, no. 4, pp. 474–481, 1998.
- [21] J. Fridrich and M. Goljan, "Practical steganalysis of digital images: state of the art," *security and Watermarking of Multimedia Contents IV*, vol. 4675, pp. 1–13, 2002.
- [22] J. Mielikainen, "Lsb matching revisited," *IEEE signal processing letters*, vol. 13, no. 5, pp. 285–287, 2006.
- [23] T. Filler and J. Fridrich, "Gibbs construction in steganography," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 705–720, 2010.
- [24] Q. Yao, W. Zhang, and N. Yu, "Optimality of polar codes in additive steganography under constant distortion profile," in *2022 14th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2022, pp. 404–408.
- [25] K. Chen, H. Zhou, H. Zhao, D. Chen, W. Zhang, and N. Yu, "Distribution-preserving steganography based on text-to-speech generative models," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3343–3356, 2021.
- [26] G. Xu, H.-Z. Wu, and Y.-Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, 2016.
- [27] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.
- [28] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *International conference on machine learning*. PMLR, 2016, pp. 1060–1069.
- [29] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5907–5915.
- [30] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1316–1324.
- [31] B. Li, X. Qi, T. Lukasiewicz, and P. Torr, "Controllable text-to-image generation," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [32] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-shot text-to-image generation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8821–8831.
- [33] OpenAI, "Gpt-4 technical report," in *Proceedings of the Conference on Artificial Intelligence*, 2023. [Online]. Available: <https://api.semanticscholar.org/CorpusID:257532815>
- [34] K. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding," *Advances in Neural Information Processing Systems*, vol. 35, pp. 36 479–36 494, 2022.
- [35] C. Yu, "Attention based data hiding with generative adversarial networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 1120–1128.
- [36] X. Weng, Y. Li, L. Chi, and Y. Mu, "High-capacity convolutional video steganography with temporal residual modeling," in *Proceedings of the 2019 on international conference on multimedia retrieval*, 2019, pp. 87–95.
- [37] Z. J. Wang, E. Montoya, D. Munechika, H. Yang, B. Hoover, and D. H. Chau, "DiffusionDB: A large-scale prompt gallery dataset for text-to-image generative models," *arXiv:2210.14896 [cs]*, 2022. [Online]. Available: <https://arxiv.org/abs/2210.14896>
- [38] Z. Guan, J. Jing, X. Deng, M. Xu, L. Jiang, Z. Zhang, and Y. Li, "Deepmih: Deep invertible network for multiple image hiding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, pp. 372–390, 2022. [Online]. Available: <https://api.semanticscholar.org/CorpusID:245854988>
- [39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.
- [40] M. Boroumand, M. Chen, and J. Fridrich, "Deep residual network for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1181–1193, 2018.
- [41] J. Betker, G. Goh, L. Jing, T. Brooks, J. Wang, L. Li, L. Ouyang, J. Zhuang, J. Lee, Y. Guo *et al.*, "Improving image generation with better captions," 2023.
- [42] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.
- [43] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Transactions on information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012.
- [44] K. Chen, H. Zhou, W. Zhou, W. Zhang, and N. Yu, "Defining cost functions for adaptive jpeg steganography at the microscale," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1052–1066, 2018.
- [45] Q. Yao, W. Zhang, K. Chen, and N. Yu, "Ldgm codes based near-optimal coding for adaptive steganography," *IEEE Transactions on Communications*, 2023.
- [46] P. Bas, "Steganography via cover-source switching," in *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2016, pp. 1–6.
- [47] J. Zhang, K. Chen, W. Li, W. Zhang, and N. Yu, "Steganography with generated images: Leveraging volatility to enhance security," *IEEE Transactions on Dependable and Secure Computing*, 2023.

- [48] Z. J. Wang, E. Montoya, D. Munechika, H. Yang, B. Hoover, and D. H. Chau, "Diffusiondb: A large-scale prompt gallery dataset for text-to-image generative models," *arXiv preprint arXiv:2210.14896*, 2022.
- [49] Y. Chen, H. Wang, W. Li, and W. Li, "A steganography immunoprocessing framework against cnn-based and handcrafted steganalysis," *IEEE Transactions on Information Forensics and Security*, 2024.
- [50] M. Zhu, H. Chen, Q. Yan, X. Huang, G. Lin, W. Li, Z. Tu, H. Hu, J. Hu, and Y. Wang, "Genimage: A million-scale benchmark for detecting ai-generated image," 2023.
- [51] D. Wang, G. Yang, J. Chen, and X. Ding, "Gan-based adaptive cost learning for enhanced image steganography security," *Expert Systems with Applications*, vol. 249, p. 123471, 2024.
- [52] K. Sun, J. Pan, Y. Ge, H. Li, H. Duan, X. Wu, R. Zhang, A. Zhou, Z. Qin, Y. Wang *et al.*, "Journeydb: A benchmark for generative image understanding," *Advances in neural information processing systems*, vol. 36, pp. 49 659–49 678, 2023.