When Provably Secure Steganography Meets Generative Models

Kejiang Chen, Hang Zhou, Hanqing Zhao, Dongdong Chen, Weiming Zhang, Nenghai Yu

***Abstract*—Steganography is the art and science of hiding secret messages in public communication so that the presence of the secret messages cannot be detected. There are two provably secure steganographic frameworks, one is black-box sampling based and the other is compression based. The former requires a perfect sampler which yields data following the same distribution, and the latter needs explicit distributions of generative objects. However, these two conditions are too strict even unrealistic in the traditional data environment, because it is hard to model the explicit distribution of natural image. With the development of deep learning, generative models bring new vitality to provably secure steganography, which can serve as the black-box sampler or provide the explicit distribution of generative media. Motivated by this, this paper proposes two types of provably secure stegosystems with generative models. Specifically, we first design block-box sampling based provably secure stegosystem for broad generative models without explicit distribution, such as GAN, VAE, and flow-based generative models, where the generative network can serve as the perfect sampler. For compression based stegosystem, we leverage the generative models with explicit distribution such as autoregressive models instead, where the adaptive arithmetic coding plays the role of the perfect compressor, decompressing the encrypted message bits into generative media, and the receiver can compress the generative media into the encrypted message bits. To show the effectiveness of our method, we take DFC-VAE, Glow, WaveNet as instances of generative models and demonstrate the perfectly secure performance of these stegosystems with the state-of-the-art steganalysis methods.**

***Index Terms*—Steganography, Generative Model, Arithmetic Coding, Provable Security**

I. Introduction

STEGANOGRAPHY is the art and science of communicating in such a way that the presence of a message cannot be detected. It has been be applied in various applications, such as anonymous communication and covert communication, by hiding message into media, such as text [1], audio [2], image [3], [4], video [5], etc. Cachin [6] firstly formalized an information-theoretic model for steganography in 1998, where a relative entropy function is employed as a basic measure of steganographic security for the concealment system. The security of a steganographic system can be quantified in terms of the relative entropy $D(P_c \parallel P_s)$ between the distributions of *cover* P_c and *stego* P_s, which yields bound on the detection capability of any adversary. From another perspective, Hopper *et al.* [7] formalized perfectly secure system based on computational-complexity, which is immigrated from cryptography. The perfect security is defined as a polynomial-time distinguisher that cannot distinguish the *cover* and *stego*.

With the definition of steganographic security, provably secure steganography has developed a lot in the past two decades and can be divided into two frameworks: compression based and black-box sampling based. Anderson *et al.* [8] observed that *cover* can be compressed to generate a secret message, and for embedding message, we can decompress it into *stego*. Le *et al.* [9] constructed a provably secure steganography called $\mathcal{P}$-code based on arithmetic coding and it assumes that the sender and receiver know the distribution of *cover* exactly. Sallee [10] designed a compression based stegosystem for JPEG images that assumes the Alternating Current (AC) coefficients in JPEG images following Generalized Cauthy distribution, and the receiver can estimate the distribution as well.

As for black-box sampling based stegosystem, Cachin [6] proposed rejection-sampling to generate *stego* that looks like *cover*. In detail, the stegosystem samples from the *cover* distribution until it selects a document whose hash value equals to message XOR k, where k is a session key both parties derive from the shared secret key. Hopper [7] improved Cachin's method and generalized it to be applicable to any distribution, which assumes it has sufficient entropy and can be sampled perfectly based on prior history. Von Ahn *et al.* [11] created public-key provably secure stegosystem and chosen-stegotext attacks (SS-CSA). Lysyanskaya *et al.* [12] analyzed the problem of imperfect sample by weakening assumption, where the *cover* distribution is modeled as a stateful Markov process. Zhu *et al.* [13] provided a more general construction of secure steganography with one-way permutation and unbiased sampler.

Nevertheless, these two types of steosystems are both not effective and not feasible in the realistic traditional data environment. For compression based systems, they need to know the exact distribution of *cover*, which is too strict. Because the complexity and dimensionality of *covers* formed by digital media objects, such as natural audios, images and videos, prevent us from modeling a complete distribution P_c of *cover*. As for black-box sampling based system, the difficulty is that the perfect sampler is hard to obtain, and the capacity of the existing schemes is rather low.

With the development of deep learning, generative models has obtained significant success in recent years and bring new vitality to provably secure steganography. By approaching the distribution of training datasets, they can generate diverse data samples in the test stage. Prominent models include variational

This work was supported in part by the Natural Science Foundation of China under Grant U1636201, 61572452.

All the authors are with CAS Key Laboratory of Electro-magnetic Space Information, University of Science and Technology of China, Hefei 230026, China.

Corresponding author: Weiming Zhang (Email: zhangwm@ustc.edu.cn)

auto-encoders (VAE) [14], [15], which maximize a variational lower bound on the data log likelihood, generative adversarial networks (GAN) [16], which employs an adversarial framework to train a generative model that mimics the true transition model, and auto-regressive models [17]–[19] and normalizing flows [20] which train with maximum likelihood (ML), avoiding approximations by choosing a tractable parameterization of probability density.

For VAE, GAN, and flow-based generative models, they can generate vivid objects from latent variables, which follow a prior distribution, e.g. normal distribution. For auto-regressive models, they can give explicit distribution of generative objects instead. These properties solve the difficulties in the traditional data environment, showing a new direction of designing provably secure steganography. Note that, though generative models have been utilized for steganography in some recent works [21]–[23], most of them do not focus on provably secure steganography.

In this paper, we design provably secure stegosystems on generative models, which owns the advantages of efficiency, practicality and high capacity. As for VAE, GAN and flow-based models, since they cannot provide implicit distribution of generative media but can be seen as perfect black-box samplers, so they can be leveraged to design black-box-sample based stegosystems. In detail, the encrypted message following uniform distribution is mapped into the latent vector which follows normal distribution first, and then the latent vector is fed to the generative component, yielding the generative data. To extract messages, another neural network whose structure is similar to the discriminator in GAN or encoder in VAE should be trained. For flowed-based generative models, thanks to their inherent reversible structures, they can directly recover message from generated data. If the parameters of generative models are kept unchanged, feeding the latent vector following the specified distribution (e.g., normal distribution), the distribution of the generated data will be same thus guarantee the perfect security. In the experiment part, DFC-VAE and Glow are taken as the realizations for building the stegosystem, and steganalytic analysis is further used to verify the final secure performance.

When it comes to auto-regressive models, since they can express tractable distribution of generative data, which meets the requirement of compression based stegosystem. Therefore, compression based stegosystem are adopted for these models. In this paper, we use one of most representative generative model (i.e., WaveNet) as an example to construct the system. This model is originally designed to synthesize high quality audios, and the semantic of generated audios is very robust. Specifically, to build the system, we combine arithmetic coding into the generation process. In the sender end, it decompresses the encrypted message into audio samples following the given distribution predicted by WaveNet. Then the receiver can reproduce the same probability distribution as that in the sender end with the same generative model. Finally, by feeding the distribution and the *stego*, the message can be extracted by the encoder of arithmetic coding. In the following part, we will provide the theoretical analysis of the stegosystem using arithmetic coding to prove the perfect security.

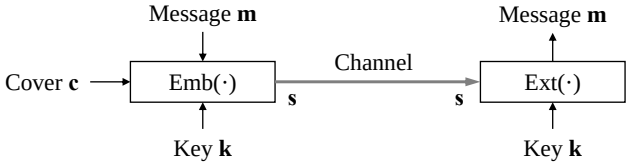


Fig. 1. A diagram of steganographic communication.

To sum up, the main contributions of this work are threefold:

- We introduce VAEs, GANs and flow-based generative models for black-box sampling based provably secure steosystems.
- Based on auto-regressive generative model WaveNet, we design the compression based provably secure stegosystem. Benefit from the robustness of the semantics of audio, the receiver can resume the generative process without additional information. Theoretical analysis is also given to prove the perfect security.
- Extensive experiments demonstrate the perfect security proposed stegosystems by defending against the state-of-the-art steganalytic methods. Most of the proposed stegosystems show perfect secure performance.

The rest of this paper is organized as follows. We review the related work in Section II, and the proposed provably secure stegosystem based on different kinds of generative models are elaborated in Section III and Section IV respectively. The experiments are presented in Section V. Conclusion and future work are given in Section VI.

II. Related Work

A. The Prisoners' Problem

In order to improve the readability of this paper, the prisoners' problem [24] formulated by Simmons is first introduced, which is a simple simulation on the background of steganography - Alice and Bob are imprisoned in separate cells and want to hatch an escape plan. They are allowed to communicate but their communication is monitored by warden Eve. If Eve finds out that the prisoners are secretly exchanging the messages, she will cut the communication channel and throw them into solitary confinement.

B. The Diagram of Steganography

According to the prisoners' problem, the diagram of the steganography is depicted in Fig. 1. A steganographic scheme can be regarded as a pair of embedding and extraction functions Emb() and Ext() for Alice and Bob, respectively [25].

$$\mathrm{Emb}(\mathbf{c}, \mathbf{k}, \mathbf{m}) = \mathbf{s}, \tag{1}$$

$$\mathrm{Ext}(\mathbf{s}) = \mathbf{m}, \tag{2}$$

where $\mathbf{c}, \mathbf{k}, \mathbf{m}, \mathbf{s}$ are *cover* object, secret key, message and *stego* object, respectively. Eve judges the object $\mathbf{s}$ is innocent or not by all the possible knowledge except secret key according to Kerckhoffs' principle [26].

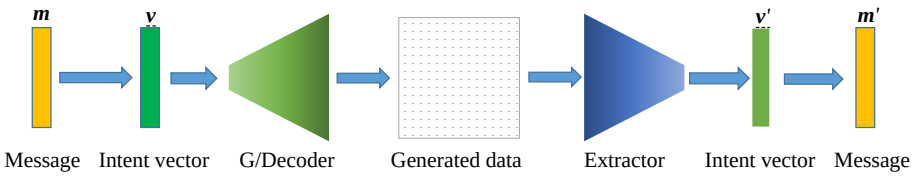


Fig. 2. The pipeline of black-box sample based provably secure steganography using VAE or GAN.

C. The Theoretical Definition of Steganographic Security

The theoretical definition of steganographic security is given by Cathin [6] with the information theory. Assuming the *cover* is drawn from C with probability distribution P_c and the steganographic method will generate *stego* object which has the distribution P_s. The distance of two distributions can be measured using relative entropy:

$$D(P_c \parallel P_s) = \sum_{x \in C} P_c(x) \log \frac{P_c(x)}{P_s(x)}. \tag{3}$$

When $D(P_c \parallel P_s) = 0$, the stegosystem is perfectly secure.

D. Existing Provably Secure Stegosystem

The black-box sampling based provably secure stegosystem can be briefly described in Algorithm 1. Given a mapping function $f_k(\cdot) : \mathcal{K} \times C \to \mathcal{R}$ with the secret key k and $\mathcal{R} = \{0,1\}^e$, the message embedding in the stegosystem is based on rejection sampling algorithm $Sample_f^C(k,b)$. The algorithm can sample *cover* following the given distribution C by oracle O^C, such that an e-bit symbol b will be embedded in it. The algorithm randomly chooses a sample s until $f_k(s) = b$, where $\leftarrow_R$ denotes the random choosing of oracle O. Nevertheless, the perfect samplers for multimedia are hard to obtain in the traditional data environment, and the capacity of existing schemes, such as adopting document, network protocol [27] as the carrier, is rather low, e.g. one document carries 1 bit message. In addition, there seems no black-box sampling based perfectly secure steganography for multimedia.

Algorithm 1 $Sample_f^C(k,b)$

Require: a key k, a value $b \in \{0,1\}^e$

1: **repeat**
2: $\quad s \leftarrow_R O^C$
3: **until** $f_k(s) = b$
4: Return s

Based on arithmetic coding, Le *et al.* [9] proposed $\mathcal{P}$-code for provably secure steganography, which requires both sides know the distribution of *cover* exactly. Sallee [10] designed the compression-based stegosystem for JPEG images by assuming Alternating Current (AC) coefficients follow Generalized Cathin distribution. However, the complexity and dimensionality of covers formed by digital media objects, such as natural audios, images and videos, will prevent us from determining a complete distribution P_c of cover, which implies Sallee's method cannot achieve perfectly secure.

In summary, compression-based schemes need to know the exact distribution of cover, which is too difficult to capture the distribution of digital media objects. As for black-box-sample based system, the perfect sampler are hard to obtain, and the capacity of the existing schemes is rather low. To this end, we introduce the generative models into provably secure steganography which can serve as the perfect sampler, or supply the explicit probability distribution of *cover* object, so that the provably secure stegosystem can be practical and effective.

E. Generative Models

Generative model describes how media are generated, in terms of a probabilistic model. The generation process can be seen as random sampling from the probability distribution learned in the stage of training, as shown in Fig. 3. The generative models can be divided into two categories, implicit density probability distribution and explicit density probability distribution. VAEs, GANs and flow-based models belong to the first category, and auto-regressive models attribute to the second category. The former meets the requirement of the black-box sampling based stegosystem, while the latter is able to be adopted to develop compression based stegosystem. In the subsequent sections, we will design provably secure stegosystem on different generative models.

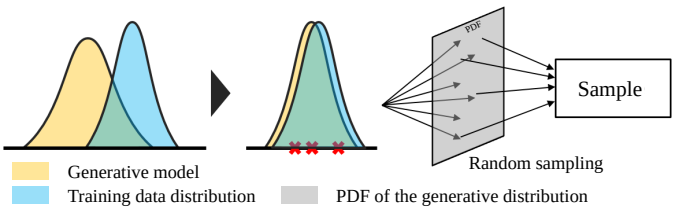


Fig. 3. The pipeline of sample generation using generative model.

III. Provably Secure Steganography on Generative Model Without Explicit Probability Distribution

VAEs, GANs and flow-based models do not own explicit probability distribution, however, the generative components in their models can serve as the perfect black-box sampler, which generate media with the same distribution. The black-box sampling based provably secure stegosystem can be built on them. Details will be expanded below.

A. Provably Secure Stegosystem Using VAE / GAN

VAE includes *encoder* and *decoder*. To generate a sample from the model, the VAE first draws a sample $\mathbf{z}$ from the prior distribution $p_\theta(\mathbf{z})$. The sample $\mathbf{z}$ is used as input to a differentiable generator network $g(\mathbf{z})$. Finally, $\mathbf{x}$ is sampled from a distribution $p_\theta(\mathbf{x}|g(\mathbf{z})) = p_\theta(\mathbf{x}|\mathbf{z})$. During the training, the approximate inference network, *i.e.*, encoder network $q_\phi(\mathbf{z}|\mathbf{x})$ is used to obtain $\mathbf{z}$ and $p_\theta(\mathbf{x}|\mathbf{z})$ is viewed as a decoder network.

GAN is one of the most popular generative deep models. The core of a GAN is to play a min-max game between a discriminator D and a generator G, *i.e.*, adversarial training. The discriminator D tries to differentiate if a sample is from real-world or generated by the generator while the generator G tries to generate samples that can fool the discriminator. The generator takes a noise $\mathbf{z}$ sampled from a prior distribution $p(\mathbf{z})$ as input, and maps the noise to the data space as $G_{\theta_g}(\mathbf{z})$. Typical choices of the prior $p(\mathbf{z})$ can be uniform distribution or Gaussian distribution. $D_{\theta_d}(\mathbf{x})$ is the classifying network that outputs a single scalar which represents the probability that $\mathbf{x}$ comes from the real-world data rather than generated data.

Once the parameters of the generative components in VAE/GAN are determined, the generative data will follow the same distribution, which meets the requirement of the perfect sampler. As a result, they can be adopted to design black-box sampling based stegosystem. The pipeline of the stegosystem building on VAE or GAN is presented in Fig. 2.

First, we map the message into latent vectors, and then feed the latent vector $\mathbf{z}$ to a pretrained decoder/generator, which will yield generated *stego* $\mathbf{y}$. Generally, the message is encrypted first, so the encrypted message follows uniform distribution. In VAE or GAN, the latent vectors are constrained to follow normal distribution. Then the mapping module is used to map uniform distribution to normal distribution. Here, we define payload p as the information that each dimension of latent vector carrying. Given p bits of message, we can map it into a random noise using mapping function $\mathcal{M}(m, p)$:

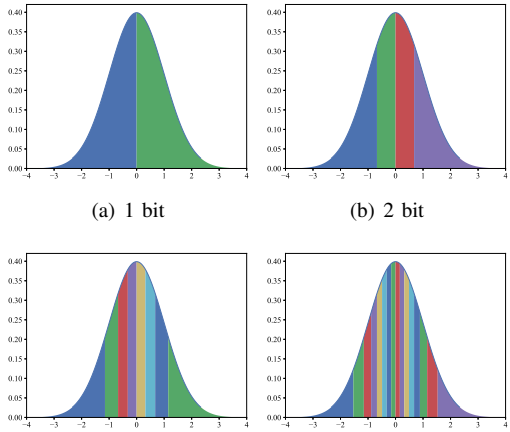
$$z = \mathcal{M}(m,p) = RS\left(Norm.ppf\left(\frac{m}{2^{p-1}}\right), Norm.ppf\left(\frac{m+1}{2^{p-1}}\right)\right), \tag{4}$$

where the $RS(x, y)$ is a reject sampling function that will keep randomly sampling a value z from $(-\infty, \infty)$ until z dropping into the interval (x, y). m is the secret message in p-ary form transferred from p bits binary message, and $Norm.ppf(\cdot)$ is the percent point function (inverse of Cumulative Distribution Function (CDF)) of normal distribution, which can be used to keep the CDF is average divided. Fig. 4 gives the example of the interval division of $p = 1, 2, 3, 4$, respectively. The bigger the interval is, the easier extracting message is. After the division is determined, reject sampling is utilized to mapping the binary stream into the corresponding interval. Details of the mapping process are given in Algorithm 2. The module of mapping reconstructed latent vectors $\mathbf{z}'$ to message $\mathbf{m}'$ is the reverse process of Algorithm 2.

An extra extractor E will be trained to extract the message from the stego, denoted by $\mathbf{z}' = E(\mathbf{y})$ and the extractor E will be sent to receiver before covert communication. The structure of extractor can be similar to the encoder/discriminator with a slight adjustment in the output layer, e.g. setting the dimension of output of the fully connected layer as the dimension of latent vector. The objective function $\mathcal{L}_E$ of the extractor can be set as the mean square error between the original latent vector $\mathbf{z}$ and the reconstructed latent vector $\mathbf{z}'$:

$$\mathcal{L}_E = \|\mathbf{z}' - \mathbf{z}\|_2. \tag{5}$$

Receivers use the extractor network to translate the generated *stego* into reconstructed latent vectors $\mathbf{z}'$ and map $\mathbf{z}'$ into reconstructed message $\mathbf{m}'$ using Demodulator, the inverse process of Modulator.



(a) 1 bit (b) 2 bit

(c) 3 bit (d) 4 bit

Fig. 4. The examples of division in the module of mapping.

Algorithm 2 Mapping function $\mathcal{M}$

Require: Payload p, message $\mathbf{m}$ of length n
Ensure: latent vectors $\mathbf{v}$

```
Divide the interval (-∞,∞) into 2^p subintervals according to the CDF of Uniform distribution.
Make every p bits message as one group and compose n/p message groups G.
for each item i in G do
    repeat
        Sample s from Uniform distribution.
    until s drops into the corresponding interval of item i.
    Append s to latent vectors z
end for
```

Since the GAN and VAE share the similar framework, in implementation, we only take Deep Feature Consistent Variational Autoencoder (DFC-VAE) [28] as a representative. The structure and the initiation of Extractor are set the same as the encoder in DFC-VAE, the training of Extractor is independent of Encoder and Decoder. In other words, the VAE can be pretrained to generate images.

B. Provably Secure Stegosystem Using Flow-based Generative Models

There exists a type of generative model owning the property of reversibility, which is also called flow-based generative

model, such as i-ResNet [29], Glow [20]. In most flow-based generative models, the generative process is defined as:

$$\mathbf{z} \sim p_{\theta}(\mathbf{z}), \tag{6}$$

$$\mathbf{x} = \mathbf{g}_{\theta}(\mathbf{z}), \tag{7}$$

where $\mathbf{z}$ is the latent variables and p_{θ} has a tractable density, such as a spherical multivariate Gaussian distribution: $p_{\theta}(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, \mathbf{I})$. $\mathbf{g}_{\theta}$ is invertible, such that given a datapoint $\mathbf{x}$, latent-variable inference is done by $\mathbf{z} = \mathbf{f}_{\theta}(\mathbf{x}) = \mathbf{g}_{\theta}^{-1}(\mathbf{x})$. The flow-based generative models have achieved considerable visual quality in field of generating images. Since flow-based generative models are reversible, it is very convenient to convert them into steganographic frameworks. The pipeline of stegosystem using flow-based generative model can be expressed as:

$$\mathbf{m} \xrightarrow{\mathcal{M}} \mathbf{z} \xrightarrow{\mathbf{g}_{\theta}} \mathbf{x} \xrightarrow{\mathbf{g}_{\theta}^{-1}} \mathbf{z}' \xrightarrow{\mathcal{M}^{-1}} \mathbf{m}'. \tag{8}$$

The message $\mathbf{m}$ is first mapped into latent vector $\mathbf{z}$, then $\mathbf{z}$ is fed into generative function $\mathbf{g}_{\theta}$ to obtain the generated *stego* $\mathbf{x}$. The receiver reconstructs the latent variable $\mathbf{z}'$ using $\mathbf{g}_{\theta}^{-1}$, and uses the reverse process of mapping to reconstruct message $\mathbf{m}'$.

Reversibility means the reconstructed latent vector $\mathbf{z}'$ is same as the original latent vector $\mathbf{z}$. Always the mapping function $\mathcal{M}$ is invertible as well. Consequently, the message can be losslessly extracted if the communication channel is noise-free. Unlike the aforementioned framework, this work does not require an extra extractor for it is reversible, which brings great convenience to us. Furthermore, since the process is reversible, the payload will not affect the accuracy of message extraction. Consequently, the capacity of this framework can reach as big as the file size of image. In our default implementation, Glow is selected as the generative model.

C. Discussion of the Security

After the training phase, the DFC-VAE and Glow can generate images from latent vector with normal distribution. For innocent users, a random latent vector following normal distribution will be fed to the generative component for generating images. Since the generative component are kept unchanged after training, if the steganographer maps the message into latent vector with normal distribution, the distribution of the generated images will be the same as that generated by innocent users. Consequently, the perfect security can be achieved. It is worth mentioning that the steganographer pursues the distribution of their *stego* images are identical to that of the ordinary generated images rather than natural images.

IV. Provably Secure Steganography on Generative Models With Explicit Probability Distribution

The compression based stegosystem can be seen as generative steganography, which decompresses the message into *stego*, and can be seen as the dual problem of source coding. Given a perfect compressor, the media can be compressed into bits with the max entropy (following uniform distribution in other words.). The inverse process is decompress uniform bits into media, which coincidentally is similar to the message embedding process, since the message is always encrypted first. Then the perfect security depends on two aspects: perfect compressor and knowing the distribution of the media. There exist many kinds of source coding, such as Huffman codes, Lempel-Ziv codes and Arithmetic coding, asymptotically achieving the theoretical bound. As a result, the tricky problem is that the distribution of natural signal is hard to capture. Luckily, the auto-regressive models with explicit density probability distribution can overcome the problem.

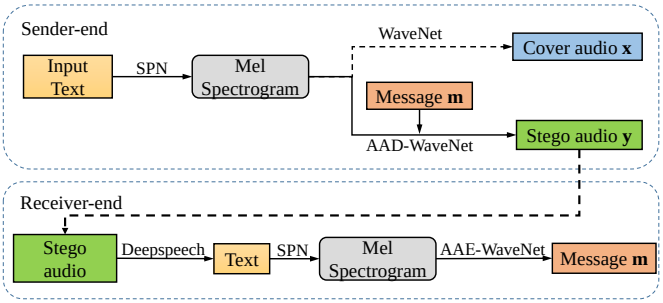


Fig. 5. The diagram of generative steganography using *WaveNet* and Arithmetic Coding.

Autoregressive generative models over high-dimensional data $\mathbf{x} = (x_1, \ldots, x_n)$ factor the joint distribution as a product of conditionals:

$$p(\mathbf{x}) = p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1}). \tag{9}$$

Optionally, the model can condition on additional global information $\mathbf{h}$ (audio semantic), in which case the conditionals become $p(x_i | x_{1:i-1}, \mathbf{h})$. With the explicit probability distribution of the *cover* object, the source coding can be combined into the generative process for decompressing the message into generated data. Given generated data, if the receivers can repeat the generated process to obtain the probability distribution, they can compress the generated data into message.

A. Provably Secure Stegosystem Using WaveNet

In the previous work [30], provably secure steganography on image synthesis using autogressive models was presented and shown impressive performance. To extend this work, we design provably secure steganography scheme on audio synthesis system with *WaveNet*. Audio synthesis has natural advantage in this field. The quality of synthesis speech is close to human speech [19], and can be easily recognized by speech-to-text system. Furthermore, the system is based on the conditional autogressive models. With different input words, the system can generate the corresponding audio, which means the diversity and the certainty of this system are better than those of the previous work [30].

The whole diagram of the stegosystem is shown in Fig. 5. At the sender-end, given the input text, the mel-spectrogram can be predicted by the Spectrogram Prediction Network

(SPN). As for ordinary waveform generation, the sample is randomly chosen according to the distribution $p(x_t)$, where the distribution is yielded by the *WaveNet* vocoder. In detail, the probability of the second sample $p(x_2 \mid x_1, \mathbf{h})$ is produced by the WaveNet with the mel-spectrogram. As for ordinary waveform generation, the value of the second sample x_2 is randomly chosen according to $p(x_2 \mid x_1, \mathbf{h})$. Given x_1, x_2, the probability distribution of x_3 is predicted by the network as $p(x_3 \mid x_1, x_2, \mathbf{h})$, similar process will be repeated until all samples are generated.

When it comes to steganography, the message embedding process is integrated with the waveform generation in WaveNet vocoder. Since the generation is a sequential generation, adaptive arithmetic decoding (AAD) is utilized to embed message. Given $p(y_2 \mid y_1, \mathbf{h})$ (same as $p(x_2 \mid x_1, \mathbf{h})$), and message $\mathbf{m}$, a part of message can be embedded and obtained y_1 using AAD. Then the probability will be updated as $p(y_3 \mid y_2, y_1, \mathbf{h})$. Analogously, the process will be repeated until all samples are generated. The whole process of message embedding is denoted as:

$$\mathbf{y} = AAD\left(p\left(\mathbf{y} \mid \mathbf{h}\right), \mathbf{m}\right). \tag{10}$$

where $p(\mathbf{y} \mid \mathbf{h})$ represents the probability distribution of $\mathbf{y}$. The process is also named AAD-WaveNet, labeled in Fig. 5. To point out, the message length should be shorter than the entropy of $p(\mathbf{y} \mid \mathbf{h})$. For simplicity, the message length L will be set as different multiples of a constant value, so that L is not required to send to the receiver. If the real message does not reach the length, just padding 0 to message before encryption. Before the covert communication, the sender should send SPN and WaveNet vocoder to the receiver, or use public models and just tell the receiver where to download.

The receiver utilizes speech-to-text tool such as DeepSpeech [31], to recognize speech into text. It is worth mentioning that, the recognition is not perfect, so the sender must verify the generated audio can be correctly recognized before communication. Since owning the same model, $\mathbf{y}$ and message length L, the probability distribution $p(\mathbf{y})$ of every step will be identical to that generated in the sender-end, so the message can be extracted using adaptive arithmetic encoding (AAE), denoted by:

$$\mathbf{m} = AAE\left(p\left(\mathbf{y} \mid \mathbf{h}\right), \mathbf{y}\right). \tag{11}$$

The details of message embedding and extraction using *AAD* and *AAE* will be elaborated below.

B. Message Embedding and Extraction

Given the distribution P_c of generative media, the process of embedding message corresponds to source decoding, and extraction corresponds to source encoding. $\mathcal{A} = \{a_1, a_2, ..., a_m\}$ is the alphabet for a generative *cover* in a certain order with the probability $\mathcal{P} = \{P(a_1), P(a_2), ..., P(a_m)\}$. The cumulative probability of a symbol can be defined as

$$F\left(a_i\right) = \sum_{k=1}^{i} P\left(a_k\right). \tag{12}$$

Owning these notations, we start to introduce the process of message embedding and extraction.

1) Message embedding: Here, adaptive arithmetic decoding (AAD) is selected as the source coding. Given the encrypted message $\mathbf{m} = [m_1 m_2 m_3 ... m_L]$, it can be interpreted as a fraction q in the range $[0, 1)$ by prepending "0." to it:

$$m_1 m_2 m_3 ... m_L \rightarrow q = 0.m_1 m_2 m_3 ... m_L = \sum_{i=1}^{L} m_i \cdot 2^{-i}. \tag{13}$$

Following the adaptive arithmetic decoding algorithm, we start from the interval $[0, 1)$ and subdivide it into the subinterval $[l, l+h)$ according to the probability $\mathcal{P}$ of the symbols in $\mathcal{A}$, and then append the symbol a_j corresponding to the subinterval in which the dyadic fraction q lies into the *stego* $\mathbf{y}$:

$$\mathbf{y} = \mathbf{y} :: a_j, \tag{14}$$

where :: represents appending the subsequent symbol into the previous vector. Regularly, the probability $\mathcal{P}$ of symbols will be updated. Then calculate the subinterval $[l, h)$ according to the updated probability $\mathcal{P}$ by

$$h_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_j), \tag{15}$$

$$l_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_{j-1}), \tag{16}$$

where h_k and l_k are the bound of subinterval in the k-th step. Repeat the process until the fraction q satisfies the constraint:

$$\begin{cases} q + (0.5)^L \notin [l_k, h_k) \\ q - (0.5)^L \notin [l_k, h_k) \end{cases} \tag{17}$$

This constraint guarantees that the dyadic fraction q is the unique fraction of length L in the interval $[l_k, h_k)$, such that the message can be extracted correctly. The message length L and the probabilities $\mathcal{P}$ of symbol are shared with the receiver.

To further clarify the scheme of message embedding using arithmetic decoding, we provide a pseudo-code that describes the implementation of message embedding by adaptive arithmetic decoding in Appendix.

2) Message extraction: Correspondingly, the message extraction refers to adaptive arithmetic encoding (AAE). On the receiver-end, the interval $[l, h)$ starts from $[0, 1)$, and will be subdivided into subintervals of length proportional to the probabilities of the symbols. if k-th element y_j corresponds to the symbol a_j, update the subinterval as follows:

$$h_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_j), \tag{18}$$

$$l_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_{j-1}), \tag{19}$$

Repeat the process until the number of steps reaches the length of $\mathbf{y}$. Finally, find the fraction $q = \sum_{i=1}^{n} m_i 2^{-i}$ satisfying $l_n \leq q \leq h_n$, where $m_i \in \{0, 1\}$ is the message bit and n is the length of message. Analogously, the pseudo code of message extraction is presented in Appendix.

C. A Simple Example of Message Embedding and Extraction Using AAD and AAE

Fig. 6 presents an example of embedding message using *AAD*. Given the encrypted message $\mathbf{m} = (m_j)^L$, and the cover distribution $P_c = (p_i)^n$, we will implement embedding

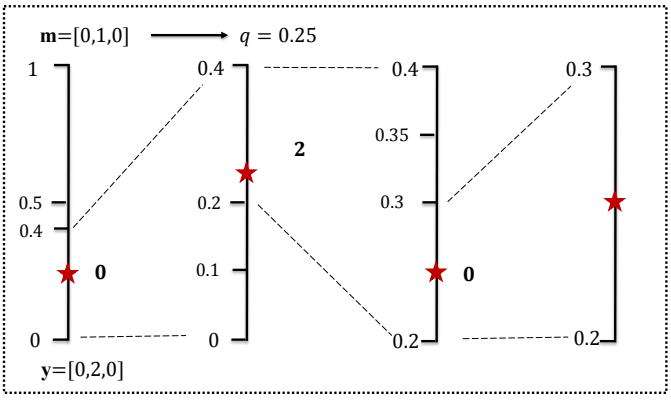


Fig. 6. The process of embedding message using adaptive arithmetic decoding (AAD).

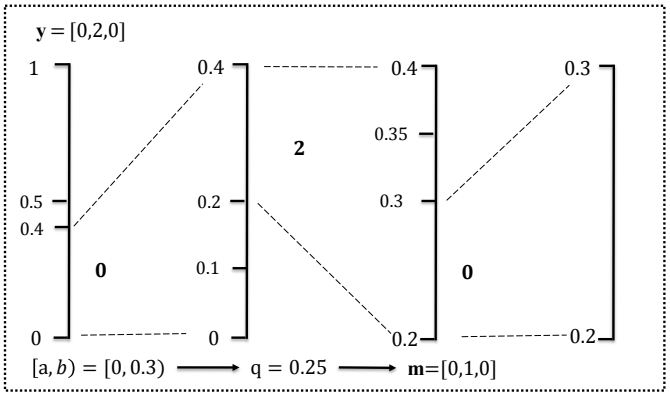


Fig. 7. The process of message extraction using adaptive arithmetic encoding (AAE).

by $\mathbf{y} = AAD(\mathbf{m}, \mathbf{P}_c)$. Suppose we get $\mathbf{m} = [0, 1, 0]$ and $P_c = [p_1, p_2, p_3]$ with each p_j as follows:

$$\begin{aligned} p_1 &= (0.40, 0.10, 0.50) \\ p_2 &= (0.25, 0.25, 0.50) \\ p_3 &= (0.50, 0.25, 0.25) \end{aligned} \tag{20}$$

Transfer the message $\mathbf{m} = [0, 1, 0]$ into fraction $q = 0.25$. The initial interval is set as [0, 1). Given $p_1 = (0.40, 0.10, 0.50)$, the three candidate intervals (for y_i to be 0, 1 and 2 respectively) is as follows:

$$I_0 = [0, 0.4), I_1 = [0.4, 0.5), I_2 = [0.5, 1.0). \tag{21}$$

As $q \in I_0$, Alice gets $y_1 = 0$ and updates the interval as $[0, 0.4)$. Then given $p_2 = (0.25, 0.25, 0.50)$, the subintervals can be obtained as:

$$I_0 = [0, 0.1), I_1 = [0.1, 0.2), I_2 = [0.2, 0.4). \tag{22}$$

As $q \in I_2$, Alice gets $y_2 = 2$ and updates the interval as $[0.2, 0.4)$. Thereafter, the new subintervals are divided according to $p_3 = (0.50, 0.25, 0.25)$:

$$I_0 = [02, 0.3), I_1 = [0.3, 0.35), I_2 = [0.35, 0.4). \tag{23}$$

On the receiver-end, we use adaptive arithmetic encoding $\mathbf{m} = AAE(\mathbf{y}, \mathbf{P})$, as shown in Fig. 7. The parameters of the generative model is shared with the receiver, so the receiver can obtain the cover's distribution P_c. Given $\mathbf{y}$, message length L, and the initial interval $[0, 1)$, Bob first divide the interval according to P_1 in Equation (20). Due to $y_1 = 0$, we update the interval as [0,0.4). Given $p_2 = (0.25, 0.25, 0.50)$ and $y_2 = 2$, we update the interval as [0.2,0.3). Given $p_3 = (0.50, 0.25, 0.25)$ and $y_3 = 0$. we get the final interval as [0.2,0.3). The final interval contains only a fraction whose binary form is L = 3 bits, denoted by $q = 0.25$. So we restore the original message $m = (010)$.

D. Proof of Asymptotically Perfect Security

The secure proof of using Arithmetic coding is discussed in the subsection. The arithmetic code is prefix free, and by taking the binary representation of q and truncating it to $l(c) = \lceil \log \frac{1}{P(c)} \rceil + 1$ bits [32], we obtain a uniquely decodable code. When it comes to encoding the entire sequence $\mathbf{c}$, the number of bits $l(\mathbf{c})$ required to represent $F(\mathbf{c})$ with enough accuracy such that the code for different values of $\mathbf{c}$ are distinct is

$$l(\mathbf{c}) = \lceil \log \frac{1}{P(\mathbf{c})} \rceil + 1. \tag{24}$$

Remember that $l(\mathbf{c})$ is the number of bits required to encode the entire sequence $\mathbf{c}$. Therefore, the average length of an arithmetic code for a sequence of length n is given by

$$\begin{aligned} l_{A^n} &= \sum P(\mathbf{c}) l(\mathbf{c}) \\ &= \sum P(\mathbf{c}) \left[l(c) = \lceil \log \frac{1}{P(c)} \rceil + 1 \right] \\ &< \sum P(\mathbf{c}) \left[l(c) = \log \frac{1}{P(c)} + 1 + 1 \right] \\ &= -\sum P(\mathbf{c}) \log P(\mathbf{c}) + 2 \sum P(\mathbf{c}) \\ &= H(C^n) + 2. \end{aligned} \tag{25}$$

Given that the average length is always greater than the entropy, the bounds on l_{A^n} are

$$H(C^n) \leq l_{A^n} < H(C^n) + 2. \tag{26}$$

The length per symbol l_A, or rate of the arithmetic code is $\frac{l_A(n)}{n}$. Therefore, the bounds on l_A are

$$\frac{H(C^n)}{n} \leq l_A < \frac{H(C^n)}{n} + \frac{2}{n}. \tag{27}$$

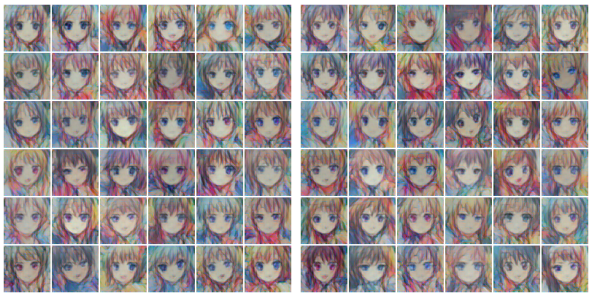
Also we know that the entropy of the sequence is nothing but the length of the sequence times the average entropy of every symbol [33]:

$$H(C^n) = nH(C). \tag{28}$$

Therefore,

$$H(C) \leq l_A < H(C) + \frac{2}{n}. \tag{29}$$

In our framework, P_s^n is the real distribution of n samples generated by the process of message embedding using AAD, and P_c is the target distribution which we are desired to approximate. According to [34, Theorem 5.4.3], using the wrong distribution P_s^n for encoding when the true distribution is P_c incurs a penalty of $D(P_c \parallel P_s^n)$. In other words, the



(a) cover (b) stego

Fig. 8. The *cover* images and *stego* images generated by DFC-VAE stegosystem.

increase in expected description length due to the approximate distribution P_s^n rather than the true distribution P_c is the relative entropy $D(P_c \parallel P_s^n)$. Directly extended from Eq. (29), $D(P_c \parallel P_s^n)$ has upper bound:

$$D(P_c \parallel P_s^n) < \frac{2}{n}, \tag{30}$$

and if $n \to \infty$, then

$$D(P_c \parallel P_s^n) \to 0. \tag{31}$$

By increasing the length of the sequence, the relative entropy between P_c and P_s^n turns to be 0, meaning that the proposed steganographic scheme can asymptotically achieve perfect security with sufficient elements using arithmetic coding.

V. Experiments

In this section, experimental results and analysis are presented to demonstrate the feasibility and effectiveness of the proposed schemes.

A. DFC-VAE Stegosystem

Anime dataset which includes 50,000 color cartoon images is selected as the dataset with cropping and scaling the aligned images to 64×64 pixels like. We train the encoder and decoder in DFC-VAE[1] with a batch size of 64 for 150 epochs over the training dataset and use Adam optimizer with an initial learning rate of 0.0005, which is decreased by a factor of 0.5 for the following epochs. Thereafter, map the random message into the latent vector $\mathbf{z}$ and feed $\mathbf{z}$ to the decoder to generate *stego* image $\mathbf{y}$. With the pair latent vectors and *stego* images, the extractor can be trained with the same setting as encoder.

Quantitative Results and Visualization Fig.8 shows *cover* samples and *stego* samples obtained by feeding random latent vector and message-driven latent vector to DFC-VAE, respectively. The state-of-the-art image steganalysis methods XuNet [35] and SRNet [36] with a slight modification (modify input channel from 1 to 3 for color images) are used to verify the secure performance of proposed method. Testing error $P_E = \frac{1}{2}(P_{FA} + P_{MD})$ is used to quantize the security performance,

[1]The source code can be downloaded at https://github.com/svenrdz/DFC-VAE.

TABLE I
The testing error of DFC-VAE stegosystem versus different payloads.

Steganalysis	Payload (bpp)			
	1	2	3	4
XuNet [35]	0.4952	0.4968	0.4954	0.4944
SRNet [36]	0.4936	0.4942	0.4986	0.4949

(a) cover (b) stego

Fig. 9. The *cover* images and *stego* images generated by Glow stegosystem.

where P_{FA} and P_{MD} are the false-alarm probability and the missed-detection probability respectively. 10,000 *cover* images and 10,000 *stego* images for different payloads (1,2,3,4 bpp(bit per pixel)) are generated for classifying.

Table I shows P_E of DFC-VAE stegosystem, which are all close to 0.5, meaning that the steganalyzer nearly randomly judge that image is *cover* or *stego*. Namely, the DFC-VAE stegosystem is nearly perfect secure. The performance of steganalyzer varies from different datasets, to verify whether the steganalysis methods are powerful, classifying different distribution of this kind of images are added as control experiment. The testing error rates of detecting images generated by unbalance message ($N_0 : N_1 = 1 : 2$) are 0.15 and 0.10 by XuNet and SRNet, respectively, which shows validity of steganalyzers. Accuracy of the secret message recovering has also been explored, and the results are listed in Table III, It can be seen that the larger payload is, the lower accuracy is. large payload means small interval, results in bad fault tolerance.

B. Glow Stegosystem

Anime dataset also serves as the dataset, while due to the limit computing resource, the images are resized to 32×32. The architecture of Glow has a depth of flow K, and number of levels L. Here, Glow model was trained with $K = 10$, $L = 3$, a batch size of 64 for 300 epochs over the training dataset and use Adam optimizer with an initial learning rate of 0.0001.

Quantitative Results and Visualization Fig.9 shows *cover* samples and *stego* samples obtained by Glow stegosystem. Similar steganalytic experiments are carried out and the results

TABLE II
The testing error of Glow stegosystem versus different payloads.

Steganalysis	Payload (bpp)			
	1	2	3	4
XuNet [35]	0.4968	0.4945	0.4975	0.4986
SRNet [36]	0.4976	0.4954	0.4964	0.4990

TABLE III
THE MESSAGE EXTRACTION ACCURACY OF DFC-VAE STEGOSYSTEM AND GLOW STEGOSYSTEM.

Extraction Accuracy	Payload (bpp)			
	1	2	3	4
DFC-VAE stegosystem	97.3%	92.3%	82.87%	66.89%
Glow stegosystem	99.3%	99.1%	99.2%	99.2%

are shown in TableII. The Glow stegosystem are nearly perfect safe, and the accuracy of message recovery approaches 100%, which show superior to DFC-VAE stegosystem. The image quality is not as good as that in [20], because the limited computing resources require us to choose a simple neural network structure. As shown in Table III, the accuracy rates of the secret message recovering are nearly 100%, and are independent of the message length, meaning that the capacity of Glow stegosystem is large.

C. WaveNet Stegosystem

We randomly collect 1,000 short text sentences and transfer them into mel-spectrograms using the SPN[2] in Tacotron-2 [37]. Then WaveNet vocoder is used for audio waveform generation[3]. The WaveNet vocoder is trained on *CMU ARCTIC* dataset [38] with 100,000 steps. All the audio clips are stored in the uncompressed WAV format. The audio length ranges from 0.5s to 3s, and the sample rate is 16kHz.

1) Steganalysis Features: The state-of-the-art steganalysis features D-MC [39], the combined feature of Time-Markov and Mel-Frequency (abbreviated as CTM) [40] are selected to evaluate the secure performance. The detectors are trained as binary classifiers implemented using the FLD ensemble with default settings [41]. A separate classifier is trained for each embedding algorithm and payloads. The ensemble by default minimizes the total classification error probability under equal priors:

$$P_{\mathrm{E}} = \min_{P_{\mathrm{FA}}} \frac{1}{2}(P_{\mathrm{FA}} + P_{\mathrm{MD}}), \tag{32}$$

The ultimate security is qualified by average error rate $\overline{P}_{\mathrm{E}}$ averaged over ten 500/500 database splits, and larger $\overline{P}_{\mathrm{E}}$ means stronger security.

We also realize other steganographic methods to show the effectiveness of the selected steganalysis features. LSB matching [42] and AACbased [43] algorithms are chosen, where the former is the conventional and the latter is content-adaptive. LSB matching means that if the message bit does not match the LSB of the *cover* element, then one is randomly either added or subtracted from the value of the *cover* pixel element. Otherwise, there is no need for modification. AACbased algorithm is simulated at its payload-distortion bound. The distortion of AACbased is defined as the reciprocal of the difference between the original audio and the reconstructed audio through compression and decompression by advanced audio coding.

[2]The architecture of spectrogram prediction network can be downloaded at https://github.com/Rayhane-mamah/Tacotron-2.

[3]The architecture of WaveNet vocoder can be downloaded at https://github.com/r9y9/wavenet_vocoder

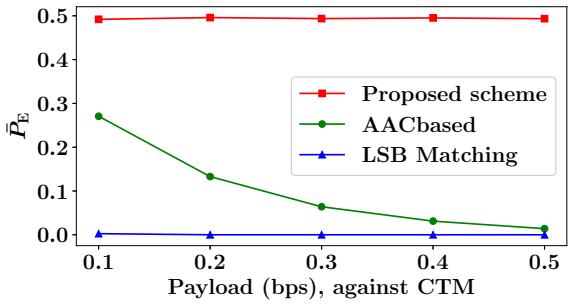


Fig. 10. The average detection error rate $\overline{P}_{\mathrm{E}}$ as a function of payload in bits per sample (bps) for steganographic algorithm payloads ranging from 0.1-0.5 bps against CTM.

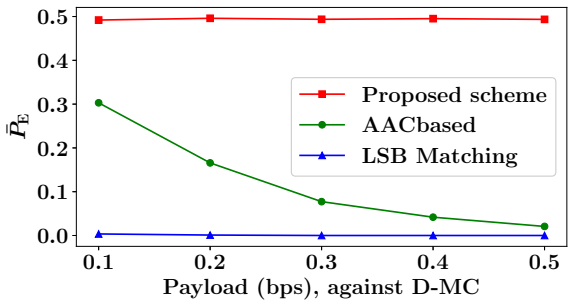


Fig. 11. The average detection error rate $\overline{P}_{\mathrm{E}}$ as a function of payload in bits per sample (bps) for steganographic algorithm payloads ranging from 0.1-0.5 bps against D-MC.

Fig.10 and Fig.11 show the average detection error rate $\overline{P}_{\mathrm{E}}$ as a function of payload in bps for steganographic algorithm payloads ranging from 0.1-0.5 bps against CTM and D-MC. It can be observed in Fig. 10 and Fig. 11 that the $\overline{P}_{\mathrm{E}}$ of AACbased decreases with the increment of payload and turns to be nearly 0%, and that of LSB matching is always nearly 0%, showing that the steganalysis is effective with respect to the generated audio. Accordingly, $\overline{P}_{\mathrm{E}}$ of the proposed scheme is nearly 50%, which means the proposed scheme is nearly perfectly secure. The experimental results verify the security performance as proved in Section IV-D.

VI. CONCLUSIONS

In this paper, we first review the provably secure steganography, which has two main frameworks: black-box sampling based stegosystem and compression based stegosystem, and conclude the limitation of current work: lack of efficient perfect sampler and cannot model the explicit distribution of natural media. However, the fast development of generative models brings great opportunity to the provably secure steganography.

Based on generative models without explicit distribution (GAN, VAE, flow-based models), we design block-box sampling based stegosystems. As for GAN/VAE, the generator in GAN and decoder in VAE serve as the perfect sampler, respectively. A message mapping module transfers the message into the latent vector, and then the latent vector will be fed to the perfect sampler to produce the *stego*. An extra extractor network should be trained to extract message. As

for flow-based generative model, the generative network can serve as both message embedder and extractor, resulting from its reversible structure. It can be easily found that the flow-based generative models show great convenience in designing stegosystem, for there is no need for an extra extractor.

For generative models with explicit distribution (autoregressive models), we design compression based stegosystems. Given the distribution of generative media, combining with source coding, we can decompress the message into generative media. The receivers can compress the generative media to extract message.

Take DFC-VAE, Glow, WaveNet as instances, the asymptotically perfectly secure performance is verified by the state-of-the-art steganalysis methods. Additionally, the theoretical proof is given for WaveNet stegosystem using Arithmetic coding.

In our future work, we will explore other effective source encoding schemes and try to transfer them to generative steganographic encoding. Furthermore, other generative media, such as text and video, will be developed under the proposed framework.

Acknowledgment

The authors would like to thank Prof. Weiqi Luo from Sun Yat-sen University for providing us the source codes of audio steganalysis. The authors also would like to thank Ryuichi Yamamoto for his valuable advice.

References

[1] Z. L. Yang, X. Q. Guo, Z. M. Chen, Y. F. Huang, and Y. J. Zhang, "Rnn-stega: Linguistic steganography based on recurrent neural networks," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1280–1295, 2018.

[2] K. Chen, H. Zhou, W. Li, K. Yang, W. Zhang, and N. Yu, "Derivative-based steganographic distortion and its non-additive extensions for audio," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[3] W. Su, J. Ni, X. Li, and Y.-Q. Shi, "A new distortion function design for jpeg steganography using the generalized uniform embedding strategy," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 12, pp. 3545–3549, 2018.

[4] Y. Yeung, W. Lu, Y. Xue, J. Huang, and Y.-Q. Shi, "Secure binary image steganography with distortion measurement based on prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[5] Y. Wang, Y. Cao, X. Zhao, Z. Xu, and M. Zhu, "Maintaining rate-distortion optimization for ipm-based video steganography by constructing isolated channels in hevc," in *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*. ACM, 2018, pp. 97–107.

[6] C. Cachin, "An information-theoretic model for steganography," in *International Workshop on Information Hiding*. Springer, 1998, pp. 306–318.

[7] N. J. Hopper, J. Langford, and L. Von Ahn, "Provably secure steganography," in *Annual International Cryptology Conference*. Springer, 2002, pp. 77–92.

[8] 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.

[9] T. Van Le, "Efficient provably secure public key steganography." *IACR Cryptology ePrint Archive*, vol. 2003, p. 156, 2003.

[10] P. Sallee, "Model-based steganography," in *International workshop on digital watermarking*. Springer, 2003, pp. 154–167.

[11] L. Von Ahn and N. J. Hopper, "Public-key steganography," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 323–341.

[12] A. Lysyanskaya and M. Meyerovich, "Provably secure steganography with imperfect sampling," in *International Workshop on Public Key Cryptography*. Springer, 2006, pp. 123–139.

[13] Y. Zhu, M. Yu, H. Hu, G.-J. Ahn, and H. Zhao, "Efficient construction of provably secure steganography under ordinary covert channels," *Science China Information Sciences*, vol. 55, no. 7, pp. 1639–1649, 2012.

[14] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[15] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," *arXiv preprint arXiv:1401.4082*, 2014.

[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[17] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "Made: Masked autoencoder for distribution estimation," in *International Conference on Machine Learning*, 2015, pp. 881–889.

[18] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves *et al.*, "Conditional image generation with pixelcnn decoders," in *Advances in Neural Information Processing Systems*, 2016, pp. 4790–4798.

[19] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[20] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 215–10 224.

[21] J. Hayes and G. Danezis, "Generating steganographic images via adversarial training," in *Advances in Neural Information Processing Systems*, 2017, pp. 1954–1963.

[22] D. Hu, L. Wang, W. Jiang, S. Zheng, and B. Li, "A novel image steganography method via deep convolutional generative adversarial networks," *IEEE Access*, vol. 6, pp. 38 303–38 314, 2018.

[23] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 657–672.

[24] G. J. Simmons, "The prisoners' problem and the subliminal channel," in *Advances in Cryptology*. Springer, 1984, pp. 51–67.

[25] J. Fridrich, *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009.

[26] A. Kerckhoffs, "A. kerckhoffs, la cryptographie militaire, journal des sciences militaires ix, 38 (1883)." *Journal des sciences militaires*, vol. 9, p. 38, 1883.

[27] N. Hopper, L. von Ahn, and J. Langford, "Provably secure steganography," *IEEE Transactions on Computers*, vol. 58, no. 5, pp. 662–676, 2009.

[28] X. Hou, L. Shen, K. Sun, and G. Qiu, "Deep feature consistent variational autoencoder," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 1133–1141.

[29] J.-H. Jacobsen, A. Smeulders, and E. Oyallon, "i-revnet: Deep invertible networks," in *ICLR 2018-International Conference on Learning Representations*, 2018.

[30] K. Yang, K. Chen, W. Zhang, and N. Yu, "Provably secure generative steganography based on autoregressive model," in *International Workshop on Digital Watermarking*. Springer, 2018, pp. 55–68.

[31] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[32] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017.

[33] A. Said, "Introduction to arithmetic coding-theory and practice," *Hewlett Packard Laboratories Report*, pp. 1057–7149, 2004.

[34] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[35] 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.

[36] 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, 2019.

[37] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," *arXiv preprint arXiv:1712.05884*, 2017.

[38] J. Kominek and A. W. Black, "The cmu arctic speech databases," in *Fifth ISCA workshop on speech synthesis*, 2004.

[39] Q. Liu, A. H. Sung, and M. Qiao, "Derivative-based audio steganalysis," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 7, no. 3, p. 18, 2011.

[40] W. Luo, H. Li, Q. Yan, R. Yang, and J. Huang, "Improved audio steganalytic feature and its applications in audio forensics," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 2, p. 43, 2018.

[41] J. Kodovskỳ, J. J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media." *IEEE Trans. Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, 2012.

[42] J. Mielikainen, "Lsb matching revisited," *IEEE signal processing letters*, vol. 13, no. 5, pp. 285–287, 2006.

[43] W. Luo, Y. Zhang, and H. Li, "Adaptive audio steganography based on advanced audio coding and syndrome-trellis coding," in *International Workshop on Digital Watermarking*. Springer, 2017, pp. 177–186.

Appendix

Algorithm 3 and Algorithm 4 give the pseudo codes of message embedding and extraction, respectively.

Algorithm 3 Message embedding using AAD

Require: The random message $\mathbf{m}$, the probability distribution $\mathcal{P} = \{P(a_1), P(a_2), ..., P(a_n)\}$, and the cumulative probability $F(a_i) = \sum_{k=1}^{i} P(a_k)$

Ensure: The stego sequence $\mathbf{s}$.

convert the random message bits $m_1m_2m_3...m_n$ into a fraction q using

$$m_1m_2m_3...m_L \rightarrow q = 0.m_1m_2m_3...m_L = \sum_{i=1}^{L} m_i \cdot 2^{-i}. \quad (33)$$

$h_0 = 1, l_0 = 0, k = 0$

while $q + 0.5^k > h_k$ & $q - 0.5^k \leq l_k$ **do**

  $k = k + 1$

  subdivide the interval $[l_{k-1}, h_{k-1})$ into subintervals of length proportional to the probability $\mathcal{P}$ of the symbols in cover in the predefined order. The probability $\mathcal{P}$ will be updated when generating next audio sample.

  take the symbol a_j corresponding to the subinterval in which q lies.

  $\mathbf{s} = \mathbf{s} :: a_j$

  $h_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_j)$.

  $l_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_{j-1})$.

end while

$output = \mathbf{s}$

Algorithm 4 Message extraction using AAE

Require: The stego sequence $\mathbf{s}$, the probability distribution $\mathcal{P}$, the message length n and the CDF F.

Ensure: The message $\mathbf{m}$.

$h_0 = 1$

$l_0 = 0$

k = 0

while $h_k \leq 2^n$ **do**

  $k = k + 1$

  subdivide the interval $[l_{k-1}, l_{k-1} + h_{k-1})$ into subintervals of length proportional to the probabilities $\mathcal{P}$ of the symbols in cover (in the predefined order). The probability $\mathcal{P}$ will be updated when generating next audio sample.

  $h_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_j)$.

  $l_k = l_{k-1} + (h_{k-1} - l_{k-1}) * F(a_{j-1})$.

end while

find the fraction $q = \sum_{i=1}^{n} m_i 2^{-i}$ satisfying $l_n \leq q \leq h_n$, where $m_i \in \{0, 1\}$ is the message bit.

$output = \mathbf{m} = [m_1, m_2, m_3, ...m_n]$