CNN-BASED ADVERSARIAL EMBEDDING FOR IMAGE STEGANOGRAPHY

J.SUMANTH ROYAL¹, ZIAVR RAHIMAN SHAIK²
¹PG Student, Dept of ECE, GLBC, Kadapa, AP, India.
²Assistant Professor, Dept of ECE, GLBC, Kadapa, AP, India.

ABSTRACT- Steganographic schemes are commonly designed in a way to preserve image statistics or steganalytic features. Since most of the state-of-the-art steganalytic methods employ a machine learning (ML) based classifier, it is reasonable to consider countering steganalysis by trying to fool the ML classifiers. However, simply applying perturbations on stego images as adversarial examples may lead to the failure of data extraction and introduce unexpected artefacts detectable by other classifiers. In this paper, we present a steganographic scheme with a novel operation called adversarial embedding (ADV-EMB), which achieves the goal of hiding a stego message while at the same time fooling a convolutional neural network (CNN) based steganalyzer. The proposed method works under the conventional framework of distortion minimization. In particular, ADV-EMB adjusts the costs of image elements modifications according to the gradients back propagated from the target CNN steganalyzer. Therefore, modification direction has a higher probability to be the same as the inverse sign of the gradient. In this way, the so called adversarial stego images are generated. Experiments demonstrate that the proposed steganographic scheme achieves better security performance against the target adversary-unaware steganalyzer by increasing its missed detection rate. In addition, it deteriorates the performance of other adversary-aware steganalyzers, opening the way to a new class of modern steganographic schemes capable to overcome powerful CNN-based steganalysis.

I. INTRODUCTION

Steganography is the technique of concealing particulars in data of any digital format in such a way that these hidden parts are known only by sender and receiver of the data [1]. The opposite procedure, steganalysis, can detect the existence of steganographic methods on a digital object [1]. Nowadays, the variety of steganographic algorithms, combine with the explosive growth of tools made for steganography reasons, have initiated a hunting game of stego objects that can be found in almost everywhere. However, according to that growth, steganalysis is still in its infancies, and always one step behind steganography but it is expected to be developed equally in the nearest future. In the "real world" steganography, someone can just connect to a website and download files that only appeared to be legitimate HTML or JPEG files. However, it is not always bona fide as it has become effortless to copy and distribute illegal digital information [2], [3] which is easily created by changing file’s elements in a visually imperceptible way. As the human visual system (HVS) is unable to detect small changes on a pixel level of an image file, in numerous cases, the process of embedding secret information into an innocent-looking carrier requires only 0.01 bits per pixel rate. In that case, even the most descent steganalyzers might face difficulties in detecting whether a carrier file contains hidden information. Furthermore, it is important to consider the media type for selecting a steganalysis method. Due to the expansive use of images to cover information, this paper will focus on detecting images with covered data. JPEG is one of the most common file types used as a carrier because of the number of JPEG images uploaded online every day, increases their ability to pass under the radar as an unsuspected material. Also, all digital cameras, all mobile phones capture directly to JPEG format, so it is not a problem to find such an image. However, JPEG format appears with one extra difficulty while analyzing it. JPEG is a lossy compression scheme and the steganalysis has to occur on a data-point, instead of on a pixel-level. Due to the variety of steganographic methods, it becomes effortless to hide a piece of information on a media without being caught. A steganalysis expert often has to have some information about the steganographic algorithm in order to detect whether there is additional information on the examined piece of media. As a consequence, developing steganalysers, which are updated and independent of any steganographic method, has become essential. These factors allow the steganalytic tool to resolve if a media (i.e. an image) is a stego, without any previous clue about the hidden content or the embedding method. Moreover, steganalysis should be trustworthy for a variety of steganographic techniques in order to take suitable countermeasures.
II. EXISTING SYSTEM

2.1 Introduction
Broadly, deep learning is defined as a class of machine learning algorithms that use multi-layered neural networks to extract higher-level features from raw data. These methods use large amounts of training data to extract complicated, feature-rich data for either generative or classification tasks. In this thesis, we use a number of deep learning approaches to improve our steganalyzers.

2.1.1 Generative Adversarial Network
A generative adversarial network (GAN) is a deep learning architecture comprised of two neural networks: a generator and a discriminator. The generator learns to generate plausible data, which act as negative training examples for the discriminator. When training begins, the generator produces obviously fake data, and the discriminator quickly learns to tell that it is fake, penalizing the generator for producing implausible results. Over time, the generator gets better at generating realistic looking data while the discriminator learns to better distinguish the generator's fake data from real data. Figure 2-1 shows an example GAN system in which the generator is learning to produce fake handwritten Arabic numerals and the discriminator is tasked with determining whether the images were produced by a human or the generator. GAN systems are increasingly used in steganography since they are excellent at generating hard-to-detect steganographic images.

Figure 2-1: Generative adversarial networks (GANs) are deep learning architectures that are composed of a generator and a discriminator. The generator learns how to transform random noise into a target distribution, while the discriminator attempts to identify if the generated images are fake or real. This adversarial setup allows the generator to effectively model target distributions.

2.1.2 Convolutional Neural Network
A convolutional neural network (CNN) is a deep learning architecture capable of taking an input image, assigning importance (learnable weights and biases) to various aspects/objects in the image, and then using these signals for image classification. Figure 2-2 shows an example CNN, which is composed of several specialized layers. CNNs are used extensively in steganalysis for their ability to learn important features relevant to detecting steganographic images. Since we use CNNs extensively in this thesis, we now describe several of the specialized layers shown in Figure 2-2.

Figure 2-2: Convolutional neural networks (CNNs) are deep learning architectures that use convolutional layers and sub-sampling layers to extract and classify meaningful signals from image data.

**Convolution Layer**

Figure 2-3: Convolutional layers extract spatially relevant data by convolving weights with input data. Convolutions are defined by their kernel size, stride, and padding. These parameters affect the types of spatial features the convolution can extract.

The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), each of which has a small receptive field but extends through the full depth of the input volume. Figure 2-3 shows an example convolution, showcasing how convolutions extract spatially-relevant data.

**Pooling Layer**

Figure 2-4: Pooling layers down-sample input data by applying a function on subregions. For example, max pooling applies a max function to extract the strongest sub-region signals. Pooling is effective at reducing noise and extracting larger features.
The pooling layer is a form of non-linear down-sampling. There are several nonlinear functions for implementing pooling, among which max pooling is the most common. It partitions the input image into a set of non-overlapping rectangles and outputs the maximum for each such sub-region. Figure 2-4 shows how max pooling extracts the maximal signal from its receptive field.

III. THE PROPOSED ADV-EMB STEGANOGRAPHIC SCHEME

In this section, we will propose a novel steganographic scheme, called ADV-EMB, to counter a target steganalyzer. First, we will outline the basic idea of the proposed scheme. Then we will discuss the two most important operations in the proposed scheme, i.e., adversarial embedding and minimizing the amount of adjustable elements, in detail. Finally, we will give a practical implementation of ADV-EMB.

A. Basic Idea

In the proposed scheme, the image elements are randomly divided into two groups, i.e., a common group containing common elements, and an adjustable group containing adjustable elements. Data embedding is performed in two phases. In the first phase, a portion of the stego message is embedded into the common group by using a conventional baseline steganographic scheme. In the second phase, the remaining part of the stego message is embedded into the adjustable group by using the proposed adversarial embedding scheme. Adjustible elements are modified in such a way that a target steganalyzer would output a wrong class label. We use a well-performed deep learning based steganalyzer, i.e., Xu’s CNN, as the target steganalyzer, since the gradient values of its loss function with respect to the input can be used to guide the modification of adjustable elements. Other steganalyzers possessing such a property may also be used. In order to prevent over-adaptation to the target steganalyzer and enhance the security performance against other advanced steganalyzers, the number of adjustable elements is minimized, resulting in a minimization problem with constraints. In adversarial ML, an attack with full knowledge of a ML classifier is called a white-box attack. When the model, parameters, and training data of the target classifier are not known, the attack is referred to as a black-box attack. In our case, we adopt a white-box assumption in designing the steganographic scheme, however, we also test the new scheme in black-box scenarios against feature-based and CNN-based steganalyzers other than the targeted one.

B. Adversarial Embedding

Denote y as the ground truth label of X. In steganalysis, we have y \in \{0; 1\}, where 0 indicates a cover and 1 indicates a stego. Let \( L(X; y; \_C;S) \) be the loss function of a steganalyzer \( \_C;S \). For example, for a deep neural network steganalyzer, the binary decision could be given as

\[
\phi_{C,S}(X) = \begin{cases}
0, & \text{if } F(X) < 0.5, \\
1, & \text{if } F(X) \geq 0.5,
\end{cases}
\]

where \( F(X) \in [0; 1] \) is the network output indicating the probability that X is a stego. The corresponding loss function may be designed in a form of cross entropy as

\[
L(X,y;\phi_{C,S}) = -y \log(F(X)) - (1-y) \log(1-F(X)).
\]

In [33]–[35], adversarial examples are generated to fool ML models by updating input elements \( x_{i,j} \) according to the gradient of the loss function with respect to the input (abbreviated as gradient if it is not specified otherwise), i.e., \( \Delta x_{i,j} L(X, \hat{y}; \_C;S) \), by using a target label \( \hat{y} \). However, it is impossible to directly apply these methods for securing steganography. In fact, modifying the elements of a stego image may lead to the failure of data extraction thus contradicting the aim of steganography. This motivates us to design an embedding method with two objectives of equal importance: performing adversarial operation to combat steganalyzer \( \_C;S \) and performing data embedding to carry information. To this end, we propose a method that we will call adversarial embedding to generate adversarial stego images under the framework of steganographic distortion minimization. In, it is observed that when a perturbation signal associated with a target label is added to the input, the updated input, called adversarial example, is usually misclassified into the target class by the ML classifier. The perturbation signal can be designed in various ways, including using the gradient of the loss function with respect to the input. Since adding a perturbation with the inverse sign of the gradient has an adversarial effect, the objective of the proposed adversarial embedding is to modify image elements in such a way that the sign of the modification tends to be in accordance with the inverse sign of the gradient. To achieve such an objective with a high probability, together with data embedding, we operate under the distortion minimization framework by making the embedding costs bear the following properties:

\[
\begin{align*}
\rho_{i,j}^+ &< \rho_{i,j}^- & \text{if } -\nabla x_{i,j} L(X, \hat{y}; \phi_{C,S}) > 0, \\
\rho_{i,j}^- & = \rho_{i,j}^+ & \text{if } -\nabla x_{i,j} L(X, \hat{y}; \phi_{C,S}) = 0, \\
\rho_{i,j}^+ & > \rho_{i,j}^- & \text{if } -\nabla x_{i,j} L(X, \hat{y}; \phi_{C,S}) < 0.
\end{align*}
\]

Such costs yield asymmetric probabilities of increasing and decreasing the element \( x_{i,j} \), if the gradient is not zero. In this way, data can be
embedded into the image elements, and the direction of the modification has the effect of inducing the steganalyzer _C;S to decide for the target label _y = 0. Please note that the adversarial embedding may lead to higher modification rates due to the asymmetric embedding costs.

C. Minimum Amount of Adjustable Elements

With adversarial embedding, the adversarial stego images may effectively evade steganalysis. However, since the costs of increasing and decreasing are asymmetric, it increases the number of changed image elements. The reason is that the maximum entropy can only be obtained when the image element has an equal probability of increasing and decreasing. With the payload constraint, asymmetric costs lead to a higher change rate when compared to symmetric costs. Although a higher change rate may not necessarily lead to a worse security performance, we would still like to minimize it by reducing the frequency of adversarial embedding. This is due to three facts. First, it is sufficient to fool the ML classifier by using only a part of the elements to perform the adversarial operation. In fact, it is even unnecessary to perform adversarial embedding to those stego images which are generated by conventional steganographic schemes but are already incorrectly classified by the target steganalyzer. Second, if all elements are used for adversarial embedding, the generated adversarial stego images may be overly adapted to the target steganalyzer and may possibly become more detectable by other advanced steganalyzers. We may minimize the amount of elements for adversarial embedding to prevent introducing other detectable artefacts that can be exploited by an adversary-aware steganalyzer. Third, when the change rate is minimized, the image quality should be preserved better. We propose to divide image elements into two groups i.e., a common group containing common elements for conventional steganographic embedding, and an adjustable group containing adjustable elements for adversarial embedding. The objective is that the amount of adjustable elements should be minimized while the target steganalyzer should output a wrong class label. Mathematically speaking, the problem is formulated as

\[
\min \beta, \quad \text{s.t.} \quad \phi_{C,S}(Z) = 0 \quad \text{and} \quad \psi(Z) = k,
\]

where \( \beta \in [0; 1] \) denotes the ratio of the amounts of adjustable elements to all image elements, and \( \phi \) and \( k \) have the same definition as in Eq.(5). It is obvious that there is no explicit solution to such a problem. To solve it efficiently, the target steganalyzer is employed to numerically search for “just enough” amount of adjustable elements to satisfy the constraints in (11). The details will be described in the next subsection.

D. A Practical Implementation of ADV-EMB

In this part, we present a practical ADV-EMB steganographic scheme. Since JPEG images are widely used and pervasive on the Internet, we use them as cover. We use Xu-CNN [26] as the target steganalyzer and J-UNIWARD [12] as the baseline steganographic scheme for conventional data embedding. The target steganalyzer is a CNN model composed of a fixed DCT filtering layer and 20 learnable convolutional layers. To the best of our knowledge, it achieves the best performance in detecting JPEG image steganography. In this paper, we use JPEG cover images and stego images generated by J-UNIWARD to train the target steganalyzer. However, other image formats, conventional embedding schemes or steganalyzers, may also be applicable, as indicated in Section III-A. The detailed steps of the proposed scheme are described as follows, and Fig. 1 illustrates an example.

![Fig. 1. Illustration of the process of the proposed ADV-EMB scheme.](image-url)
UNIWARD) to compute the initial embedding costs, i.e., \( \alpha_{ij} = 3 \) for the DCT coefficients. Initialize the parameter \( \gamma = 0 \).

2) Divide the elements in \( C \) into two disjoint groups, i.e., a common group containing \( l_1 = \lfloor H^- W^- (1 - \gamma) \rfloor \) common elements, and an adjustable group containing \( l_2 = H^- W^- (1 - \gamma) \) adjustable elements. In Fig. 1, common group and adjustable group are labeled as blue and red boxes respectively. The positions of these two kinds of elements can be fixed in advance or randomized with the details to be discussed later.

3) Embed \( k_1 = [k \cdot (1 - \gamma)] \) bits into the common group using the initial embedding costs computed in Step 1 by applying a distortion minimization coding scheme, such as STC (syndrome-trellis codes) [45]. The resulting image is denoted as \( Z_c \). In Fig. 1, the modified coefficients in common group are highlighted with blue strides.

4) Compute the gradients \( 5z_i;L(Z_c; \gamma; _C;S) \) of the steganalyzer using the target label \( \gamma = 0 \). Update the embedding costs for the adjustable elements by

\[
\begin{align*}
\rho_i,j^{(i+1)} &= \begin{cases} 
\rho_i,j^{(i)} / \alpha_i & \text{if } - \nabla z_i,j L(Z_c,0;\phi) > 0, \\
\rho_i,j^{(i)} & \text{if } - \nabla z_i,j L(Z_c,0;\phi) = 0, \\
\rho_i,j^{(i)} / \alpha_i & \text{if } - \nabla z_i,j L(Z_c,0;\phi) < 0,
\end{cases} \\
\rho_i,j^{(i+1)} &= \begin{cases} 
\rho_i,j^{(i)} / \alpha_i & \text{if } - \nabla z_i,j L(Z_c,0;\phi) > 0, \\
\rho_i,j^{(i)} & \text{if } - \nabla z_i,j L(Z_c,0;\phi) = 0, \\
\rho_i,j^{(i)} / \alpha_i & \text{if } - \nabla z_i,j L(Z_c,0;\phi) < 0,
\end{cases}
\end{align*}
\]

where \( \gamma \) is a scaling factor larger than 1 to ensure that equations (12) and (13) necessarily fulfill equation (10). \( \gamma \) is set to 2 in this work. Embed \( k_2 = k - k_1 \) bits into the adjustable elements by using the updated embedding costs computed from (12) and (13) and the same coding scheme used for the common group. The resultant image is \( Z_c \). Figure 1 shows that the costs of the elements in the adjustable group are either doubled or halved, depending on the signs of the corresponding gradients. After data embedding, the modified coefficients in adjustable group are highlighted with red strides.

5) Take \( Z \) as the input of the steganalyzer \( _C;S \). If \( _C;S(Z) = 0 \), which means the adversarial stego \( Z \) can fool the steganalyzer with a minimum value of \( \gamma \), use \( Z \) as the output and terminate the embedding process. Otherwise, the amount of adjustable elements may not be enough. In this case, update \( \gamma \) by \( \gamma + \gamma \), and repeat Step 2 to Step 5 until \( \gamma = 1 \). We use \( \gamma = 0:1 \) in this work. If \( \gamma = 1 \) and \( _C;S(Z) = 1 \), which corresponds to the failure case of adversarial embedding, we just use a conventional steganographic scheme for embedding and output a conventional stego image. Since the same coding scheme, such as STC, is used both in the adjustable group and the common group, the message receiver neither needs to be informed about the value of \( \gamma \), nor needs to know which image elements belong to the adjustable group or the common group. Data is extracted in the same way as the baseline steganographic scheme. As we know, in most existing steganographic schemes, an embedding order of image elements is generated by scrambling the indexes of image elements, where the scrambling operation is determined by a secret key shared between the sender and the receiver. The secret key can be fixed for different images, or changed as a session key. In the ADV-EMB implementation, the positions of the common elements and that of adjustable elements can be determined as follows. First, generate an embedding order in the same way as the baseline steganographic scheme. Then, the common group is formed by the first \( l_1 = \lfloor H^- W^- (1 - \gamma) \rfloor \) elements according to the embedding order. Finally, the adjustable group is formed by the remaining elements. In other words, the positions of adjustable elements can be fixed or randomized for different images, depending on whether the embedding order is fixed or randomized. We recommend randomization for enhancing security.

CONCLUSION

In this paper, we proposed a novel approach to look at the steganographic problem; namely, we proposed to embed the stego message while simultaneously taking into account the necessity of countering an advanced CNN-based steganalyzer. Such an aim is achieved by introducing a new adversarial embedding method, which takes both data embedding and adversarial operation into account. A practical steganographic scheme, ADV-EMB, which generates adversarial stego images with minimum amount of adjustable elements, has been illustrated to counter a deep learning based target steganalyzer. The extensive experiments we have carried out permitted us to reach the following conclusions: 1) When the target steganalyzer is accessible by the steganographer but the steganalist is unaware of the adversary operation, a high missed detection rate can be achieved by ADV-EMB to counter the target steganalyzer. 2) When the steganalist is aware of the adversarial embedding, and uses adversarial stego images to re-train the steganalyzer, the proposed ADV-EMB leads to a higher detection error rate compared to the state-of-the-art baseline steganographic scheme, for both target and non-target steganalysts. 3) When both the steganographer and the steganalist iteratively adjust their strategies according to the updated knowledge about the other side, adversarial stego images still have an advantage over their conventional counterparts. Our approach to
adversarial embedding shows a promising way to enhance steganographic security, still there are several unsolved issues to consider. To start with, the proposed ADVEMB scheme uses only the signs of the gradients. It worths investigating whether the amplitudes of the gradients can also be helpful. Besides, it is worth studying on whether universal perturbations are feasible in obtaining adversarial stego images. Furthermore, for a complete characterization of the interplay between the steganographer and the steganalyst, it would be interesting to resort to a game-theoretic formulation of the problem.

REFERENCES