Exploiting Adversarial Embeddings for Better Steganography

Solène Bernard, Tomáš Pevný, Patrick Bas, John Klein

Exploiting Adversarial Embeddings for Better Steganography

Solène Bernard
Univ. Lille, CNRS, Centrale Lille, UMR 9189, CRIStAL
Lille, France
solene.bernard@centrale.centralelille.fr

Tomáš Pevný
FEL CTU Prague
Prague, Czech Republic
pevnytom@fel.cvut.cz

Patrick Bas
Univ. Lille, CNRS, Centrale Lille, UMR 9189, CRIStAL
Lille, France
patrick.bas@centralelille.fr

John Klein
Univ. Lille, CNRS, Centrale Lille, UMR 9189, CRIStAL
Lille, France
john.klein@univ-lille.fr

ABSTRACT

This work proposes a protocol to iteratively build a distortion function for adaptive steganography while increasing its practical security after each iteration. It relies on prior art on targeted attacks and iterative design of steganalysis schemes. It combines targeted attacks on a given detector with a min max strategy, which dynamically selects the most difficult stego content associated with the best classifier at each iteration. We theoretically prove the convergence, which is confirmed by the practical results. Applied on J-Uniward this new protocol increases P_{err} from 7% to 20% estimated by Xu-Net, and from 10% to 23% for a non-targeted steganalysis by a linear classifier with GFR features.

CCS CONCEPTS

• **Security and privacy** → *Domain-specific security and privacy architectures*;

KEYWORDS

steganography, adversarial attacks, security, game-theory

ACM Reference Format:
Solène Bernard, Tomáš Pevný, Patrick Bas, and John Klein. 2019. Exploiting Adversarial Embeddings for Better Steganography. In *ACM Information Hiding and Multimedia Security Workshop (IH&MMSec '19), July 3–5, 2019, TROYES, France.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3335203.3335737

1 INTRODUCTION

Most of steganographic methods are based on the distortion minimization principle, first demonstrated in Hugo algorithm [13] and later used in Uniward [11], MiPod [17], etc. Each pixel in the image is assigned to an additive cost related to its value and its neighborhood. During embedding, the Steganographer changes the content such that the detectability (the total cost induced by all changes) is minimized subject to the message being communicated. This is typically done using Syndrome Trellis Codes (STC) [7], which are very efficient for a class of additive distortion functions.

Because the Steganographer has to face a Detector (its adversary) in order to benchmark the practical security of the embedding scheme, steganography is adversarial by design. In the literature however, this adversarial constraint is only considered by a specific generation of embedding schemes which usually offer high practical security. We can distinguish two classes of adversarial schemes.

(1) The first class explicitly considers an adversary (the detector) to design the embedding scheme which attacks the detector. For example contribution [21] proposes to optimize basic cost functions with respect to the output of SVM classifiers or Maximum Mean Discrepancy for a given database. However the pioneering scheme belonging to this class is ASO [12] (see also section 2.1) which modifies specifically the embedding costs for each image with respect to outputs of a set of classifiers. Recently ADV-EMB [19] (see also section 2.2) modifies embedding costs computed by J-Uniward to attack a deep learning classifier. Other embedding schemes based on Generative Adversarial Networks (GANs) [20] iteratively generate a cost function in such a way that the associated embedding is less and less detectable by the adversary. In this case, the steganographer and the classifier are both convolutional networks and a global cost function, considering both the payload constraint and the constraint on practical detectability, is minimized.
(2) The second class of embedding schemes implicitly considers the cost function as a proxy of the steganalyzer output, and the embedding is tailored in order to minimize the statistical discrepancy between Cover and Stego contents. This class is associated with the concept of model-based embedding which was first proposed by Sallee [16] and had been used by different schemes such as Gibbs construction [6], Dynamic programming based Syndrome Trellis Codes (Dynamic STC) [14], modeling image residuals in MiPod [17] or mimicking the sensor noise [2].

The proposed work shares common ideas with designing targeted attacks on steganalysis schemes as proposed by ASO or ADV-EMB, but also uses iterative embedding procedure as GANs do. Its main originality relies in the fact that it combines adversarial embedding and game-theory in order to design a cost function (and consequently an embedding scheme) which is more and more secure w.r.t. an increasing set of classifiers, but also as presented in section 4, w.r.t. other steganalysis schemes.

More specifically, this work investigates if a set of Detectors $\mathcal{F}$ can be used to assign costs for changing values of image components (e.g. pixels or DCT coefficients). The proposed scheme embeds

IH&MMSec '19, July 3–5, 2019, TROYES, France

ACM ISBN 978-1-4503-6821-6/19/06…$15.00
https://doi.org/10.1145/3335203.3335737

the message such that the stego image is the least detectable image by all $f \in \mathcal{F}$. Since the set of Detector $\mathcal{F}$ should be theoretically infinite, an iterative scheme associated to a min max strategy is proposed to make $\mathcal{F}$ finite and small.

Notations

In the following, letters in bold are used to represent vectors. The corresponding non bold letters are used for vector elements. The caligraphic letters are used for sets. Cover and stego contents are respectively denoted as $\mathbf{x} = (x_i)^{H\times W}$ and $\mathbf{y} = (y_i)^{H\times W}$ where H and W are the height and width of the image. We use $\mathbf{z} = (z_i)^{H\times W}$ to denote the proposed adversarial stego contents. Note that $\mathbf{z}$ is a special type of $\mathbf{y}$. The corresponding sets are denoted as $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ respectively. $\omega \in \{0, 1\}$ will denote the class of a content $\mathbf{x}$ which is cover ($\omega = 0$) or stego ($\omega = 1$). N is the number of contents in the data base.

2 ADAPTING COST FUNCTIONS AGAINST DETECTORS: PREVIOUS WORKS

The most successful contemporary steganographic schemes are based on the *distortion minimization principle*, where costs are assigned to a change of each element of cover object. During embedding, a cover object is modified to communicate a message while minimizing the total distortion measured by the sum of costs of changing individual elements, specifically

$$D(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{H\times W} \rho_i^+ \delta_1(y_i - x_i) + \rho_i^- \delta_{-1}(y_i - x_i). \tag{1}$$

There is no single strategy as to how to assign costs ρ_i, which gives rise to different steganographic schemes. Below, we review two schemes adjusting costs ρ_i to evade directly a particular detector f. While both schemes have a very different background, they share a similar strategy: to adjust cost according to the gradient of f with respect to embedding changes.

2.1 Adaptive Steganography by Oracle

ASO [12] derives embedding costs from an ensemble of Fisher Linear Discriminants (FLD). In the notation of this paper, the set of all possible FLDs corresponds to $\mathcal{F}$ and the set of FLDs extracted from a trained ensemble corresponds to $\hat{\mathcal{F}}^k$, where it is assumed that all $f \in \mathcal{F}$ are already normalized according to [12, equations (9) and (10)]. ASO then defines embedding cost for changing a pixel x_i as

$$\rho_i^+ = \frac{1}{|\hat{\mathcal{F}}^k|} \sum_{f\in\hat{\mathcal{F}}^k} (f(\mathbf{x}_i^+) - f(\mathbf{x})),$$
$$\rho_i^- = \frac{1}{|\hat{\mathcal{F}}^k|} \sum_{f\in\hat{\mathcal{F}}^k} (f(\mathbf{x}_i^-) - f(\mathbf{x})),$$

where $\mathbf{x}_i^+$ and $\mathbf{x}_i^-$ denote the version of an image $\mathbf{x}$ with pixel (i) increased or decreased by one. Note that $f(\mathbf{x}_i^+) - f(\mathbf{x})$ and $f(\mathbf{x}_i^-) - f(\mathbf{x})$ are numerical estimates of the partial derivative $\frac{\partial f}{\partial x_i}$. The assignment of costs therefore assumes that each classifier $f \in \hat{\mathcal{F}}^k$ contributes equally to the detectability of the image. In contrast, the proposed protocol detailed in Section 3 assumes the worst case, where the detector always picks his best classifier for a given content.

2.2 ADV-EMB steganography

ADV-EMB method [19] modifies costs ρ_i^+ and ρ_i^- for increasing and decreasing the (i)-pixel obtained by some prior function to evade detection by classifier f as

$$q_i^+ = \begin{cases} \rho_i^+/\alpha & \text{if } -\frac{\partial f}{\partial x_i} > 0, \\ \rho_i^+ & \text{if } -\frac{\partial f}{\partial x_i} = 0, \\ \rho_i^+\alpha & \text{if } -\frac{\partial f}{\partial x_i} < 0, \end{cases} \tag{2}$$

where $\frac{\partial f}{\partial x_i}$ is the partial derivative of f with respect to the value of the (i)-pixel at its current value x_i and α is a parameter that authors recommend to set to 2. Costs q_i^- are adjusted in a similar way with reversed inequalities.

The proposed attack first calculates cost of pixel changes using the standard J-Uniward. Then, all DCT coefficients are divided into two disjoint groups: a common group $\mathcal{L}_c$ containing $(1 - \beta)$ fraction of DCT coefficients and an adjustable group $\mathcal{L}_a$ containing the remaining β fraction of DCT coefficients. To embed a message m of length ℓ, the algorithm first embeds $\ell(1 - \beta)$ bits into the common group using the initial embedding costs ρ_i. Costs in the adjustable group are modified according to Equation (2), and the rest of the message is embedded into adjustable DCT coefficients $\mathcal{L}_a$, producing the final stego content carrying the whole message.

The same work also suggests an iterative scheme, where the classifier is retrained on a mixture of stego images obtained by attacking the classifier trained in the previous iteration in the aim of training a more robust detector. In contrast, the goal of this paper is to create a more secure steganographic algorithm by using min max strategy.

3 EXPLOITING ADVERSARIAL ATTACKS

Under uniform class distribution, the error $P_{\text{err}}(f|P_{\mathcal{X}}, P_{\mathcal{Y}})$ of a classifier[1] $f : \mathbb{X} \mapsto \mathbb{R}$ on cover and stego contents with distribution $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$ is

$$\begin{aligned} P_{\text{err}}(f|P_{\mathcal{X}}, P_{\mathcal{Y}}) &= P\left(\text{sign}\left(f\right) \neq \omega\right) \\ &= \frac{1}{2}\mathbb{E}_{P_{\mathcal{X}}}[\mathbb{I}\left\{f > 0\right\}] + \frac{1}{2}\mathbb{E}_{P_{\mathcal{Y}}}[\mathbb{I}\left\{f < 0\right\}], \end{aligned} \tag{3}$$

where $\mathbb{I}$ denotes the indicator function. Should the steganographer be maximally undetectable with respect to the class of detectors $\mathcal{F}$, the steganographer should choose the embedding function h_{emb} maximizing the error of the best classifier the detector can possess, i.e.

$$\max_{h_{\text{emb}}} \min_{f\in\mathcal{F}} P_{\text{err}}(f|P_{\mathcal{X}}, P_{\mathcal{Y}}(h_{\text{emb}})), \tag{4}$$

where $P_{\mathcal{Y}}(h_{\text{emb}})$ denotes the probability distribution of stego images created by embedding function h_{emb} from cover images with distribution $P_{\mathcal{X}}$. This is of course not trivial, since $\mathcal{F}$ can contain

[1] Without loss of generality it is assumed the output of a detector f to be positive / negative if content $\mathbf{x}$ is classified as stego / cover. Also, for improved readability, we call f a classifier while it is the discriminant function which must be compared to threshold zero to obtain a content class label.

virtually any function f. In practice $\mathcal{F}$ consists of a set of fixed functions parameterized by $\theta \in \Theta$. These functions are for example steganographic features coupled by a machine learning based classifier, where θ are parameters of the classifier. Alternatively, they can be convolutional neural networks in which case θ corresponds to weights and other parameters of neural networks.

Assuming the set of functions $\mathcal{F}$ to be finite, in order to find a solution of Equation (4), we can use methods [6, 14, 19] to embed message m into content x to create a stego content y by, minimizing the detectability of the most sensitive detector $f \in \mathcal{F}$ instead of maximizing P_{err}, i.e

$$\min_{\mathbf{y}} \max_{f \in \mathcal{F}} f(\mathbf{y}). \tag{5}$$

In (5) it is assumed that the output of the detector f is calibrated, i.e. for example it outputs the probability that the content will be classified as stego. This calibration can be achieved either by passing the output through a logistic function, or by using empirical probability distribution functions.

Equation (5) converts the problem of distortion function design to the problem of finding a set of functions $\mathcal{F}$, that would be sufficiently rich to detect all types of steganographic distortions and small enough such that the minimization will be possible. In the rest of this section, an iterative protocol to construct such a set $\mathcal{F}$ of classifiers for a fixed or a small set of architecture(s) of neural networks is presented.

3.1 MinMax distortion function

The protocol to build a small but representative set of classifiers relies on the ability of the steganographer to use any function $f : \mathcal{X} \to \mathbb{R}$ to derive a distortion function for embedding [6, 14, 19]. It further assumes that the steganograher possesses a reasonably large set of cover contents $\mathcal{X}$ and some steganographic algorithm h_{emb} to initiate the protocol.

The protocol starts by creating a set of stego-contents $\mathcal{Y}^0$ using the initial steganographic algorithm h_{emb}. Then, a classifier f^0 is trained to classify contents from $\mathcal{X}$ and $\mathcal{Y}^0$ and is added to the set of available classifiers $\hat{\mathcal{F}}_0 = \{f^0\}$. To simplify notations, we also define $\mathcal{Z}^0 = \mathcal{Y}^0$.

In the next iteration, the steganographer creates a set of stego contents $\mathcal{Z}^1$ by attacking f^0. Then from $\mathcal{Z}^0$ and $\mathcal{Z}^1$ the steganographer generates a new set of stego contents, $\mathcal{Y}^1$, by always selecting *the most difficult* version with respect to the set of available classifiers $\hat{\mathcal{F}}^0$ (which at the moment contains only f^0) as

$$\mathcal{Y}^1 = \left\{ \mathbf{z} \,\middle|\, \mathbf{z} = \underset{\mathbf{z} \in \{\mathbf{z}_i^0, \mathbf{z}_i^1\}}{\arg\min} \max_{f \in \hat{\mathcal{F}}^0} f(\mathbf{z}) \,, 1 \le i \le n \right\}, \tag{6}$$

where $\mathbf{z}_i^{\cdot}$ denotes a stego content from $\mathcal{Z}^{\cdot}$ created from a cover content $\mathbf{x}_i \in \mathcal{X}$. We recall here that the output of a detector f to be respectively positive / negative if content $\mathbf{x}$ is respectively classified as stego / cover. It is why the steganographer aims at minimizing the output of f. The steganographer then continues by creating a new classifier f^1 classifying $\mathcal{X}$ and $\mathcal{Y}^1$ and adding it to $\hat{\mathcal{F}}^0$. His new set of classifiers is $\hat{\mathcal{F}}^1 = \hat{\mathcal{F}}^0 \cup \{f^1\}$.

At the k^{th} iteration of the protocol, the steganographer creates a set of contents $\mathcal{Z}^k$ attacking the classifier trained in the previous iteration f^{k-1}. Contents $\mathcal{Z}^0, \mathcal{Z}^1, \ldots, \mathcal{Z}^k$ are then used to create a

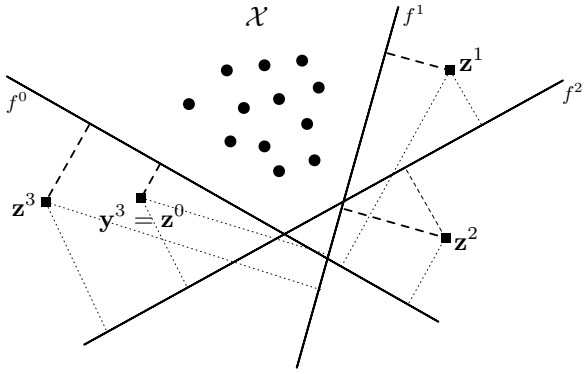

Figure 1: Illustration of a creation of a stego image $\mathbf{y}^3$ at iteration $k = 3$ by the min max strategy using $\{\mathbf{z}^0, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3\}$. We assume linear classifiers and the function $f^i(\mathbf{z}^j)$ is the algebric distance w.r.t. the separation plane. Dashed lines are for positive distances, dotted lines for negative ones, and bold lines are $\max_i f^i(\mathbf{z}^j)$. Here the min max strategy outputs $\mathbf{y}^3 = \mathbf{z}^0$. Note that the adversarial embedding fails for $\mathbf{z}^2$ here.

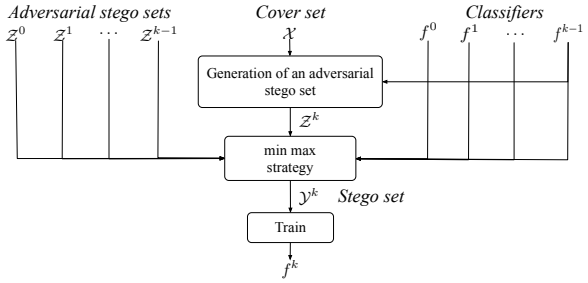

Figure 2: Diagram of the protocol at iteration k.

new set of stego contents $\mathcal{Y}^k$ by selecting those that are maximally undetectable by any detector $f \in \hat{\mathcal{F}}^{k-1}$, *i.e.*

$$\mathcal{Y}^k = \left\{ \mathbf{z} \,\middle|\, \mathbf{z} = \underset{\mathbf{z} \in \{\mathbf{z}_i^0, \mathbf{z}_i^1, \ldots, \mathbf{z}_i^k\}}{\arg\min} \max_{f \in \hat{\mathcal{F}}^{k-1}} f(\mathbf{z}) \,, 1 \le i \le n \right\}. \tag{7}$$

Contents $\mathcal{Y}^k$ are then used to train a new classifier f^k between covers $\mathcal{X}$ and $\mathcal{Y}^k$, which is added to the set, i.e. $\hat{\mathcal{F}}^k = \hat{\mathcal{F}}^{k-1} \cup \{f^k\}$. The creation of $\mathcal{Y}^k$ by using the min max protocol is illustrated in Figure 1. The whole protocol is illustrated in Figure 2.

The following theorem proves that the above protocol converges under mild conditions on $\mathcal{F}$, albeit it does not prove the solution to be optimal. Consequently, this protocol avoids pathological behavior like periodicity.

Theorem 3.1. *Let $\mathcal{F} = \{f : \mathbb{X} \to \mathbb{R}\}$ be a set of functions and let $\hat{\mathcal{F}}^1, \hat{\mathcal{F}}^2, \ldots, \hat{\mathcal{F}}^k, \ldots$ be a sequence of subsets such that $\hat{\mathcal{F}}^1 \subset \hat{\mathcal{F}}^2 \subset \ldots \subset \hat{\mathcal{F}}^k \subset \ldots \subset \mathcal{F}$. Furthermore let all functions $f \in \mathcal{F}$ be bounded by some constant c, i.e. $(\exists c \in \mathbb{R})(\forall f \in \mathcal{F})(\forall \mathbf{x} \in \mathbb{X})(f(\mathbf{x}) \le c)$.*

Then the limit $\hat{f}(\mathbf{x}) = \lim_{k \to \infty} \max_{f \in \hat{\mathcal{F}}^k} f(\mathbf{x})$ exists.

Proof. Let define the function $f^k_{\max}(\mathbf{x}) = \max_{f\in\hat{\mathcal{F}}^k} f(\mathbf{x})$. Then for every $\mathbf{x} \in \mathbb{X}$, the sequence $f^0_{\max}(\mathbf{x}), f^1_{\max}(\mathbf{x}), \ldots, f^k_{\max}(\mathbf{x}), \ldots$ is nondecreasing and because of the boundedness assumption $\forall f \in \mathcal{F}$, $f(\mathbf{x}) \le c$, the sequence is bounded by c as well. The monotone convergence theorem then states that the sequence $f^k_{\max}(\mathbf{x})$ converges to some value, which is denoted by $\hat{f}(\mathbf{x})$, which proves the theorem. □

The above theorem implies that, when k is large, the maximization w.r.t. $f \in \mathcal{F}^{k-1}$ is replaced by $\hat{f}$ (or a function ϵ-close to $\hat{f}$).

The protocol defines detectability $\hat{f}(x)$ as a limit

$$\hat{f}(\mathbf{x}) = \lim_{k\to\infty} \max_{f\in\hat{\mathcal{F}}^k} f(\mathbf{x}).$$

Note that the security of the resulting steganographic algorithm depends on two factors: (i) the set of all possible detectors $\mathcal{F}$; (ii) the quality attack on the classifier $f \in \mathcal{F}$. Thus improving any of them should improve the quality of the scheme.

Theorem 3.1 assumes functions $f \in \mathcal{F}$ to be bounded. This condition can be trivially ensured for any function based on machine learning classifiers, as they are already bounded (e.g. Neural Networks), or they can be trivially bounded by applying some scaling or passing their output through a bounded and monotonous functions like tanh .

4 EXPERIMENTS

To implement the protocol introduced in the previous section, we need (i) a suitably general set of classifiers $\mathcal{F}$, and (ii) an embedding algorithm to embed some message into a cover object while avoiding being detected by $f \in \mathcal{F}$. The rest of this section describes the different ingredients of our protocol, which are later used in this experimental section.

4.1 Choice of classifiers $\mathcal{F}$

Similarly to [9], we choose to attack Convolutional Neural Networks (CNN) since the classification function is differentiable and consequently it is easy and fast, using GPUs, to evaluate $\frac{\partial f}{\partial x}$. CNNs are not detailed here, as their inner functionality is not important for the paper and for the method. A reader interested in details is referred to [8] for general introduction and to [15, 23, 24] for their uses in steganography.

For the purpose of this work, it is sufficient to view neural networks as an efficient procedure selecting f from a large class of functions $\mathcal{F}$ such that f minimizes the empirical estimate of the mis-classification error (3):

$$\hat{P}_{\text{err}}(f;\mathcal{X},\mathcal{Y}) = \frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}} \mathbb{I}\{f(\mathbf{x}) > 0\} + \frac{1}{|\mathcal{Y}|}\sum_{\mathbf{x}\in\mathcal{Y}} \mathbb{I}\{f(\mathbf{x}) < 0\}\,. \tag{8}$$

An important property of CNNs is their differentiability, which means that a gradient $\frac{\partial f}{\partial x}$ with respect to their inputs exists for almost every x and for every $f \in \mathcal{F}$.

The set of classifiers $\mathcal{F}$ is equal to all convolutional neural networks with an architecture known as Xu-Net [22]. We chose this classifier for its very good performance in JPEG steganalysis, and because its training requires less memory and is faster than deeper CNNs such as SRNet [4]. With shortcut connections, the depth of Xu-Net following the shortest path is only 5, whereas it is 8 for SRNet.

4.2 Experimental details

We detail bellow the implementations that are used to run the presented protocol, they concern the embedding scheme, the classifier/steganalyzer which is attacked, the attack, the overall strategy to generate stego contents and the different steganalysis schemes used to evaluate the presented embedding.

Embedding: The embedding algorithm serving to initialize the protocol and to calculate default costs for changing elements is J-Uniward [11]. The experiments use the JPEG version of the popular BossBase database [3] of size 512×512 in grayscale format and compressed with Quality Factor 75. All images are embedded using an embedding rate of 0.4 bits per non-zero AC DCT coefficient (bpnzac) at each iteration of the protocol.

Classification/Steganalysis: The proposed implementation of Xu-Net uses the TensorFlow [1] library[2]. In each iteration of the protocol, a new steganalyzer f^k is trained by classifying cover contents $\mathcal{X}$ and stego contents $\mathcal{Y}^k$ given by (7). This classifier is trained starting with randomly initialized weights (zero mean Gaussian with standard deviation 0.01) using 2×4000 Cover and Stego contents for training and using remaining 2×6000 to estimate error rates. 290 epochs are used for training using ADAM optimization algorithm with initial learning rate 0.001 decreased after each 5000 steps to 0.9 times the current value. Remaining parameters of Adam are kept to default setting. The size of mini-batch is 64 (32 cover-stego pairs) and the training uses full-size images of 512×512 pixels. The configuration achieving the best training accuracy is used as the result of training. The experiments were run on an Nvidia GPU Quadro P6000 (24 GB of memory). Training XU-Net takes approximately 30 hours at each iteration k, and the generation of an adversarial data-base 5 hours multi-threaded on 36 cores.

Attack: The ADV-EMB attack described in Section 2.2 is implemented in order to adjust costs of changing DCT coefficients according to (2). Note that in order to compute the partial derivative $\frac{\partial f}{\partial x_i}$ with respect of the i^{th}-DCT coefficient, and because XU-Net uses a spatial image without rounding as input, IDCT is treated in an additional layer placed as first layer. The partial derivative is consequently handled by automatic differentiation using the function `tf.gradient()` from the TensorFlow library, and derivating with respect to the image coded in the JPEG domain.

Note that it is possible that embedding using ADV-EMB fails for some images, which means that even by modifying all costs ρ_i^+, ρ_i^- of changing DCT coefficients of a cover $\mathbf{x}$ (e.g. $\beta = 1$, $\mathcal{L}_c = \varnothing$) w.r.t the Equation (2), the adversarial stego is classified as stego by the classifier. As suggested in [19], in this case, the stego content which is kept is $\mathbf{z}^0$, i.e. the stego obtained by J-Uniward embedding with $\beta = 0$.

ADV-EMB generates at each iteration a set of adversarial stego $\mathcal{Z}^k$ and a set of stego contents $\mathcal{Y}^k$ is created w.r.t the min max strategy defined in Equation (7). In order to compute this set, one

[2]The codes for the experiments will be made available after publication.

must compute all the values $f(\mathbf{z})$ for $f \in \hat{\mathcal{F}}^{k-1}$ and $\mathbf{z} \in \mathcal{Z}^0 \cup \mathcal{Z}^1 \cup \cdots \cup \mathcal{Z}^k$. Because f is real-valued, it can be fairly assumed that, for any cover content $\mathbf{x}_i$, there is only one $\mathbf{z}$ in $\{\mathbf{z}_i^0, \mathbf{z}_i^1, \ldots, \mathbf{z}_i^k\}$ minimizing $\max_{f \in \hat{\mathcal{F}}^{k-1}} f(\mathbf{z})$.

Other embedding strategies. In the paper [19] are suggested two different iterative strategies, which are called in the following "Random strategy" and "Last iteration strategy" and whose protocols are illustrated in Figure 3. The random strategy differs from the min max strategy by the attack of the steganographer, who chooses at iteration k, for each cover, randomly and uniformly the embedding scheme from $\mathcal{Z}^0, \mathcal{Z}^1, \ldots$ or $\mathcal{Z}^k$. So the steganalyzer trains the classifier f^k between covers $\mathcal{X}$ and $\mathcal{Y}^k_{\text{random}} = \mathcal{Z}^0 \cup \mathcal{Z}^1 \cup \cdots \cup \mathcal{Z}^k$. The last iteration strategy derives from the random strategy : $\mathcal{Z}^k$ is still computed in order to fool f^{k-1}_{random}, but the steganographer sends the attack $\mathcal{Y}^k_{\text{last it}} = \mathcal{Z}^k$. So $f^k_{\text{last it}}$ is trained between $\mathcal{X}$ and $\mathcal{Y}^k_{\text{last it}}$. In Table 1, the results of this two strategies are compared to the min max strategy for the first six iterations.

Evaluation. In order to benchmark the practical security related to this new adversarial embedding scheme w.r.t. steganalysis schemes which are different from the target adversary (here XU-Net), we also compute DCTR [10] and GFR [18] feature sets. The training set and the testing set here were constitued of pairs of 5000 Cover and Stego images. The regularized linear classifier [5] was used to compute P_{err}, defined as the minimal total classification error probability under equal priors, $P_{\text{err}} = \min_{\text{Pr}_{\text{FA}}} \frac{1}{2}(\text{Pr}_{\text{FA}} + \text{Pr}_{\text{MD}})$, with Pr_{FA} and Pr_{MD} standing for the false-alarm and missed detection empirical probabilities. We also train SRNet [4] only for the first and last iteration, because of its computational cost. The size of images is 512×512, size of mini-batch is 16 (8 cover-stego pairs) and the training lasts 290 epochs.

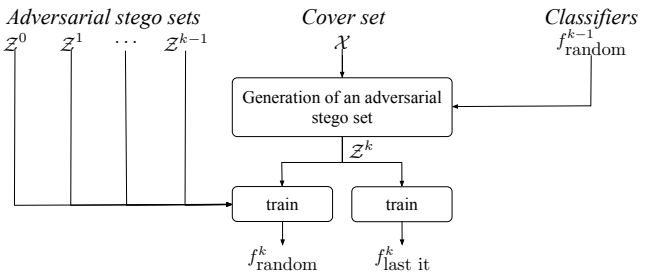

Figure 3: Diagram of two other strategies ("random" and "last iteration") suggested in [19] at iteration k.

4.3 Experimental results

We now evaluate the empirical security of the proposed embedding scheme w.r.t. different steganalysis schemes. We also evaluate the impact of the iteration parameter k and we compare the proposed min max strategy w.r.t. other ones.

Figure 4 shows the evolution of error P_{err} w.r.t parameter k for different *matched classifiers* f^k where the term "matched classifiers" means that the classifier is trained to classify cover images

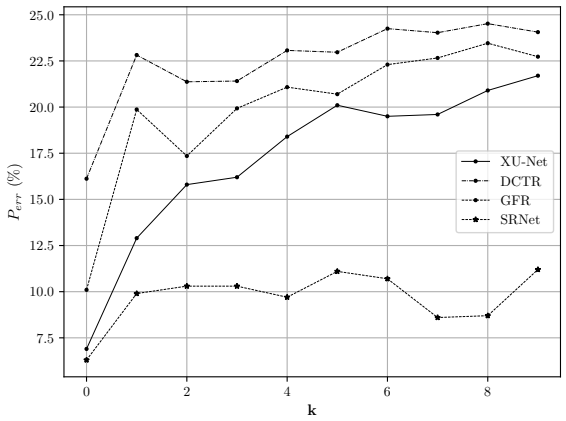

Figure 4: P_{err} of matched classifiers (this means that each classifier is trained and evaluated on $\mathcal{X}$ and $\mathcal{Y}^k$) for each iteration k using different classifiers. The algorithm is optimized with respect to XU-net. Classifiers DCTR and GFR are based on the combination of steganographic features (DCTR [10] and GFR [18]) and regularized Fisher Linear Discriminant classifier [5].

$\mathcal{X}$ and stego images $\mathcal{Y}^k$ communicated by the steganographer at the k^{th}-iteration of the protocol. Note also that contents in $\mathcal{Y}^k$ are selected such that they are the least detectable by the previous trained classifier $\{f^0, f^1, \ldots, f^{k-1}\}$, which means that they are not optimized with respect to the current classifier f^k used to benchmark our embedding scheme and which is the best classifier the detector can have since it is optimized with respect to the current strategy of the steganalyst. This means that this evaluation scenario obeys Kerckhoffs' principle as the detector knows the strategy of the steganographer.

This figure shows also that the error of the matched classifier steadily increases for the different steganalysis methods, which means that the resulting steganographic scheme becomes on average more and more secure after each iteration, even if after $k = 6$ a plateau is reached. The error rate at iteration 0 corresponds to the error after J-Uniward steganalysis. Evaluating the quality of the algorithm by the error P_{err} facing the XU-Net classifier, the security has improved from 6.9% to 21.7% after nine iterations of the algorithm, which is substantial.

We see that P_{err} is globaly increasing but sometimes it decreases (see iteration $k = 2$). It may come from the not certainty to reach optimization when training a new classifier, and because each classifier is randomly initialized.

If we now compare the practical security benchmarked by XU-Net with other steganalysis schemes, we notice that the evolution for other schemes is similar with increasing and converging undetectability w.r.t. the iteration number. The improvement proposed by this protocol is consequently not only relevant for the targeted steganalyzer, but for a broad class of steganalysis schemes.

In Table 1 embedding strategy (7) is compared with the "random" and "last-iteration" strategies that are suggested in reference [19]. We can see that the min max strategy offers higher practical security until iteration $k = 4$, with a gain of more than 3% w.r.t. the "Random" strategy at iteration 4 and 5% w.r.t. the "Last iteration" strategy at iteration 2. The bad behavior of the "Last iteration" strategy can be explained by the fact that the embedding targets a classifier different from the one used to benchmark the scheme.[3]

k	0	1	2	3	4	5	6
min max	6.9	12.9	15.8	16.2	18.4	20.1	19.5
Random	6.9	12.1	12.4	16.2	15.3	15.4	16.2
Last iteration	6.9	12.1	10.5	16.3	–	16.2	17.3

Table 1: P_{err} (in %) given by different strategies for the first six iterations: the min max **strategy (each classifier is trained and evaluated on $\mathcal{X}$ and $\mathcal{Y}^k$), the "random" strategy given in [19] (each classifier is trained and evaluated on $\mathcal{X}$ and $\mathcal{Z}^0 \cup \mathcal{Z}^1 \cup \ldots \cup \mathcal{Z}^k$) and finally the "last iteration" strategy where each classifier is trained and evaluated on $\mathcal{X}$ and $\mathcal{Z}^k$.**

5 CONCLUSION AND PERSPECTIVES

This paper proposes a steganographic scheme exploiting adversarial embeddings and based on a min max strategy. This protocol can be used to improve classical additive cost-based steganographic schemes such as J-Uniward, but it can practically be applied on any schemes that can locally adjust the embedding costs. Our tests assess both the benefits of (i) iterating in order to generate more secure stego contents and (ii) using a min max strategy instead of randomly selecting stego contents from previous iterations or selecting stego contents only from the last iteration.

Note also that contrary to other adversarial embedding strategies based on tailored sampling of the costs such as GANs [20], this scheme relies on a targeted attack to the classifier, which enables to speed up the convergence and improves its efficiency.

Future works will be devoted to design better strategies to adjust costs, and will also consider other strategies to attack the classifier.

ACKNOWLEDGMENTS

This work has been funded in part by the French National Research Agency (ANR-18-ASTR-0009), ALASKA project: https://alaska.utt.fr, and by the French ANR DEFALS program (ANR-16-DEFA-0003). The authors acknowledge the support of the OP VVV project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" and a support by Czech Ministry of Education 19-29680L.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*. 265–283.

[2] Patrick Bas. 2016. Steganography via Cover-Source Switching. In *WIFS*. Abu Dabi, United Arab Emirates. https://hal.archives-ouvertes.fr/hal-01386190

[3] Patrick Bas, Tomáš Filler, and Tomáš Pevný. 2011. "Break Our Steganographic System": The Ins and Outs of Organizing BOSS. In *International Workshop on Information Hiding*, Vol. 6958, LNCS. Springer Berlin Heidelberg, 59–70.

[4] Mehdi Boroumand, Mo Chen, and Jessica Fridrich. 2019. Deep residual network for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security* (2019).

[5] Rémi Cogranne, Vahid Sedighi, Jessica Fridrich, and Tomáš Pevný. 2015. Is ensemble classifier needed for steganalysis in high-dimensional feature spaces?. In *IEEE International Workshop on Information Forensics and Security (WIFS)*. 1–6.

[6] Tomáš Filler and Jessica Fridrich. 2010. Gibbs construction in steganography. *IEEE Transactions on Information Forensics and Security* 5, 4 (2010), 705–720.

[7] Tomáš Filler, Jan Judas, and Jessica Fridrich. 2010. Minimizing embedding impact in steganography using trellis-coded quantization. In *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 754105–754105.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

[9] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[10] Vojtěch Holub and Jessica Fridrich. 2015. Low-complexity features for JPEG steganalysis using undecimated DCT. *IEEE Transactions on Information Forensics and Security* 10, 2 (2015), 219–228.

[11] Vojtěch Holub, Jessica Fridrich, and Tomáš Denemark. 2014. Universal distortion function for steganography in an arbitrary domain. *EURASIP Journal on Information Security* 2014, 1 (2014), 1.

[12] Sarra Kouider, Marc Chaumont, and William Puech. 2013. Adaptive steganography by oracle (ASO). In *IEEE International Conference on Multimedia and Expo (ICME)*. 1–6.

[13] Tomáš Pevný, Tomáš Filler, and Patrick Bas. 2010. Using high-dimensional image models to perform highly undetectable steganography. In *International Workshop on Information Hiding*, Vol. 6387, LNCS. Springer Berlin Heidelberg, 161–177.

[14] Tomáš Pevný and Andrew D Ker. 2018. Exploring Non-Additive Distortion in Steganography. In *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*. ACM, 109–114.

[15] Lionel Pibre, Jérôme Pasquet, Dino Ienco, and Marc Chaumont. 2016. Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover sourcemismatch. *Electronic Imaging* 2016, 8 (2016), 1–11.

[16] Phil Sallee. 2005. Model-based methods for steganography and steganalysis. *International Journal of Image and graphics* 5, 01 (2005), 167–189.

[17] Vahid Sedighi, Rémi Cogranne, and Jessica Fridrich. 2016. Content-Adaptive Steganography by Minimizing Statistical Detectability. *IEEE Transactions on Information Forensics and Security* 11, 2 (2016), 221–234.

[18] Xiaofeng Song, Fenlin Liu, Chunfang Yang, Xiangyang Luo, and Yi Zhang. 2015. Steganalysis of adaptive JPEG steganography using 2D Gabor filters. In *Proceedings of the 3rd ACM workshop on information hiding and multimedia security*. ACM, 15–23.

[19] Weixuan Tang, Bin Li, Shunquan Tan, Mauro Barni, and Jiwu Huang. 2019. CNN-based Adversarial Embedding for Image Steganography. *IEEE Transactions on Information Forensics and Security* (2019).

[20] Weixuan Tang, Shunquan Tan, Bin Li, and Jiwu Huang. 2017. Automatic steganographic distortion learning using a generative adversarial network. *IEEE Signal Processing Letters* 24, 10 (2017), 1547–1551.

[21] Jessica Fridrich Tomáš Filler. 2011. Design of adaptive steganographic schemes for digital images. , 7880 - 7880 - 14 pages.

[22] Guanshuo Xu. 2017. Deep convolutional neural network to detect J-UNIWARD. In *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*. ACM, 67–73.

[23] G. Xu, H. Wu, and Y. Shi. 2016. Structural Design of Convolutional Neural Networks for Steganalysis. *IEEE Signal Processing Letters* 23, 5 (May 2016), 708–712.

[24] J. Ye, J. Ni, and Y. Yi. 2017. Deep Learning Hierarchical Representations for Image Steganalysis. *IEEE Transactions on Information Forensics and Security* 12, 11 (Nov 2017), 2545–2557.

[3] The final version of the paper will present results after more iterations, this was not possible for this submission due to computational time.