Deep Learning in steganography and steganalysis from 2015 to 2018

Marc Chaumont

Draft version n1 (April 2019). This book chapter is a draft version that will be published as a book by ELSEVIER Inc. This chapter will appear in 2020 in the book titled: "*Digital Media Steganography: Principles, Algorithms, Advances*". A revised version should be available before the end of 2019.

CHAPTER

Deep Learning in steganography and steganalysis from 2015 to 2018

1

Marc CHAUMONT[a,*]

[*] *Montpellier University, LIRMM (UMR5506) / CNRS, Nîmes University, France, LIRMM/ICAR, 161, rue Ada, 34392 Montpellier, France*

[a] *Corresponding:* marc.chaumont@lirmm.fr

ABSTRACT

For almost 10 years, the detection of a message hidden in an image has been mainly carried out by the computation of a Rich Model (RM), followed by a classification by an Ensemble Classifier (EC). In 2015, the first study using a convolutional neural network (CNN) obtained the first results of steganalysis by Deep Learning approaching the results of two-step approaches (EC + RM). Therefore, over the 2015-2018 period, numerous publications have shown that it is possible to obtain better performances notably in spatial steganalysis, in JPEG steganalysis, in Selection-Channel-Aware steganalysis, in quantitative steganalysis. This chapter deals with deep learning in steganalysis from the point of view of the existing, by presenting the different neural networks that have been evaluated with a methodology specific to the discipline of steganalysis, and this during the period 2015-2018. The chapter is not intended to repeat the basic concepts of machine learning or deep learning. We will thus give in a generic way the structure of a deep neural network, we will present the networks proposed in the literature for the different scenarios of steganalysis, and finally, we will discuss steganography by GAN.

Keywords: Steganography, steganalysis, Deep Learning, GAN

Neural networks have been studied since the 1950s. Initially, they were proposed to model the behavior of the brain. In computer science, especially in artificial intelligence, they have been used for 30 years for learning purposes. Ten or so years ago, neural networks were considered to have a too long learning time and to be less effective than classifiers such as SVMs or random forests.

With recent advances in the field of neuron networks [7], thanks to the computing power provided by graphics cards (GPUs), and finally thanks to the profusion of data, deep learning approaches have been proposed as a natural extension of neural networks. Since 2012, these deep networks have deeply marked the fields of signal processing and artificial intelligence, because their performances make it possible to surpass state-of-the-art, but also to solve problems that scientists did not manage to solve [55].

In steganalysis, for 8 years, the detection of a hidden message in an image was mainly carried out by calculating a rich model (RM) [25] followed by a classification by a classifier (EC) [48]. In 2015, the first study using a convolutional neural network (CNN) obtained first results of deep-learning steganalysis approaching the results of two-step approaches (EC + RM [1]) [74]. Therefore, over the period 2015 - 2018, many publications have shown that it is possible to obtain better performance in spatial steganalysis, JPEG steganalysis, side-informed steganalysis, quantitative steganalysis, etc.

In the Section 1.1 we present the structure of a deep neural network generically. This Section is centred on the existing in steganalysis and should be supplemented by reading on artificial learning and in particular on the gradient descent, and the stochastic gradient descent. In Section 1.2 we will get to the different steps of the convolution module. In the Section 1.3 we will tackle the complexity and learning times. In the Section 1.4 we will give links between Deep Learning and past approaches. In the Section 1.5 we will come back to the different networks that were proposed during the period 2015-2018 for different scenarios of steganalysis. Finally, in the Section 1.6 we will discuss steganography by GAN which sets up a game between two networks in the manner of the precursor algorithm ASO [52].

1.1 THE BUILDING BLOCKS OF A DEEP NEURONAL NETWORK

In the following sub-sections, we recall the major concepts of a Convolutional Neural Network (CNN). More especially, we will recall the basic building

[1] We will note EC + RM in order to indicate two-step approaches based on the calculation of a rich model (RM) then the use of an ensemble classifier (EC).

blocks of a network based on the Yedroudj-Net[2] network that was published in 2018[3] [101], and which takes up the ideas present in Alex-Net [53], as well as ideas present in networks developed for steganalysis including the very first network of Qian *et al.* [74], and networks of Xu-Net [95], and Ye-Net [99].

1.1.1 GLOBAL VIEW OF A CONVOLUTIONAL NEURAL NETWORK

Before describing the structure of a neural network as well as its elementary bricks, it is useful to remember that a neural network belongs to the machine-learning family. In the case of supervised learning, which is the case that most concerns us, it is necessary to have a database of images, with, for each image, its label, that is to say, its class.

Deep Learning networks are large neural networks that can directly take raw input data. In image processing, the network is directly powered by the pixels forming the image. A deep learning network thus learns, in a joint way, both the compact intrinsic characteristics of the image (we speak of *feature map* or of *latent space*) and at the same time the separation boundary allowing the classification (we also talk of *separator plans*).

The learning protocol is similar to the classical machine learning methods. Each image is given as input to the network. Each pixel value is transmitted to one or more neurons. The network consists of a given number of *blocks*. A block consists of neurons that take real input values, perform calculations, and then transmit the actual calculated values to the next block. A neural network can, therefore, be represented by an oriented graph where each node represents a computing unit. The learning is then done by supplying the network with examples composed of an image and its label, and the network modifies the parameters of these calculation units (it learns) thanks to the mechanism of back-propagation.

The Convolutional Neuronal Networks used for the steganalysis are mainly built in three parts, which we will call *module*: the pre-processing module, the convolution module, and the classification module. As an illustration, the figure 1.1 schematizes the network proposed by Yedroudj *et al.* in 2018 [101]. The network processes grayscale images of 256×256.

[2] GitHub link on Yedroudj-Net: `https://github.com/yedmed/steganalysis_with_CNN_Yedroudj-Net`.

[3] A second version of Yedroudj-Net should be available in 2019.

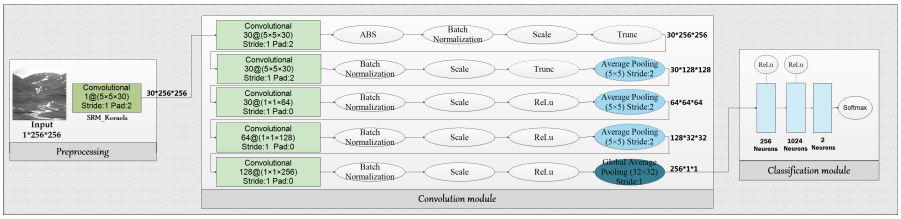

Figure 1.1 Yedroudj-Net network [101]

1.1.2 THE PRE-PROCESSING MODULE

$$F^{(0)} = \frac{1}{12}\begin{pmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{pmatrix} \tag{1.1}$$

We can observe in Figure 1.1 that in the *pre-processing module*, the image is filtered by 30 high-pass filters. The use of one or more high-pass filters as pre-processing is present in the majority of networks used for steganalysis during the period 2015-2018. An example of a kernel of a high-pass filter – the square S5a filter [25] – is given in Equation 1.1. This preliminary filtering step allows the network to converge faster and is probably needed to get good performance when the learning base is too small [100] (only 4 000 pairs cover/stego images of size 256×256). The filtered images are then transmitted to the first convolution block of the network. Note that the recent SRNet [9] network does not use any fixed pre-filters but learn the filters. It requires thus a more important database (more than 15 000 pairs cover/stego images of size 256×256), and strong know-how for its initialization. Note that there is a debate in the community if one should use fixed filters, or initialize the filters with pre-chosen values and then continue the learning, or learn filters with random initialization. At the beginning of 2019, in practice (real-world situation [44]), the best choice is probably in relation to the size of the learning database (which is not necessary BOSS [4] or BOWS2 [5]), and the possibility to use or not a transfer learning.

1.1.3 THE CONVOLUTION MODULE

Within the *convolution module*, we find several macroscopic computation units that we will call *blocks*. A *block* is composed of calculation units that take real input values, perform calculations, and return real values, which are supplied to the next block. Specifically, a *block* takes a set of *feature maps* (= a set of images) as input and returns a set of *feature maps* as output (= a set of

images). Inside a block, there are a number of operations including the following four operations: the *convolution* (see Section 1.2.1), the *activation* (see Section 1.2.2), the *pooling* (see Section 1.2.3), and finally the *normalization* (see Section 1.2.4).

Note that the concept of neuron, as defined in the literature, before the emergence of convolutional networks, is still present, but it does no longer exist as a data structure in the neural network libraries. In convolution modules, we must imagine a neuron as a computing unit which, for a position in the *feature map* taken by the convolution kernel during the convolution operation, performs the weighted sum between the kernel and the group of considered pixels. The concept of neuron corresponds to the scalar product between the input data (the pixels) and data specific to the neuron (the weight of the convolution kernel), followed by the application of a function of $\mathbb{R}$ in $\mathbb{R}$ called the activation function. Then, by extension, we can consider that pooling and normalization are operations specific to neurons.

Thus, the notion of *block* corresponds conceptually to a " layer " of neurons. Note that in deep learning libraries, we call *layer* any elementary operation such as convolution, activation, pooling, normalization, etc. To remove any ambiguity, for the convolution module we will talk about *block*, and *operations*, and we will avoid using the term *layer*.

Without counting the pre-processing block, the *Yedroudj-Net* network [101] has a convolution module made of 5 convolution blocks, like the networks of Qian *et al.* [74] and Xu *et al.* [95], the *Ye-Net* network [99] has a convolution module made of 8 convolution blocks, and SRNet network [9] has a convolution module made of 11 convolution blocks.

1.1.4 THE CLASSIFICATION MODULE

The last block of the convolution module (see the previous Section) is connected to the *classification module* which is usually a *fully connected* neural network composed of one to three blocks. This *classification module* is often a traditional neural network where each neuron is fully connected to the previous *block* of neurons and to the next *block* of neurons.

The fully connected blocks often end with a softmax function which normalize the outputs delivered by the network between $[0, 1]$, such that the sum of outputs equals one. The outputs are named imprecisely "probability". We will keep this denomination. So, in the usual binary steganalysis scenario, the network delivers two values as output: one giving the probability of classifying into the first class (e.g. the cover class), and the other giving the probability of classifying into the second class (e.g. the stego class). The classification decision is then obtained by returning the class with the highest probability.

Note that in front of this *classification module*, we can find a *particular* pooling operation such as a *global average pooling*, a *Spatial Pyramid Pooling (SPP)* [32], a *statistical moments extractor* [91], etc. Such a pooling operation

returns a fixed-size vector of values, that is, a feature map of fixed dimensions. The block just next to this pooling operation is then always connected to a vector of fixed size. This block has thus a fixed input number of parameters. It is thus possible to present to the network images of any size without having to modify the topology of the network. For example, this property is available in the Yedroudj-Net [101] network, the Zhu-Net [107] network, or the Tsang et al. network [91].

Also note that [91] is the only paper, at the writing date of this chapter, end 2018, which has seriously considered the viability of an invariant network to the dimension of the input images. The problem remains open. The solution proposed in [91] is a variant of the concept of average pooling. For the moment, there is not enough studies on the subject to determine what is the correct topology of the network, how to learn the network, how much the number of embedded bits influences the learning, if we should take into account the *square root law* for learning at a fixed security-level or any payload size, etc.

1.2 THE DIFFERENT STEPS OF THE CONVOLUTION MODULE

In Section 1.1.3, we indicated that a block within the convolution module contained a variable number among the following four operations: the *convolution* (see Section 1.2.1), the *activation* (see Section 1.2.2), the *pooling* (see Section1.2.3), and finally the *normalization* (see Section 1.2.4). Let's now explain in more detail each step (convolution, activation, pooling, and normalization) within a *block*.

1.2.1 CONVOLUTION

The first treatment within a *block* is often to apply the convolutions on the input *feature maps*.

Note that for the pre-processing *block*, see Figure 1.1, there is only one input image. A convolution is therefore done between the input image and a filter. In the Yedroudj-Net network, there are 30 high-pass filters extracted from SRM filters [25]. In old networks, there is only one pre-processing filter [74, 72, 95].

Except for the pre-processing *block*, in the other *blocks*, once the convolution has been applied, we apply activation steps (see Section 1.2.2), a pooling (see Section 1.2.3), and a normalization (see Section 1.2.4). We then obtain a new image named *feature map*.

Formally, let $I^{(0)}$ be the input image of the pre-processing *block*. Let $F_k^{(l)}$ be the k^{th} ($k \in \{1, ..., K^{(l)}\}$) filter of the *block* of number $l = \{1, ..., L\}$, with L the number of *blocks*, and with $K^{(l)}$ the number of filters of the l^{th} *block*.

The convolution within the pre-processing *block* with the k^{th} filter results in a filtered image, denoted $\tilde{I}_k^{(1)}$, such that:

$$\tilde{I}_k^{(1)} = I^{(0)} \star F_k^{(1)}. \tag{1.2}$$

From the first *block of the convolution module* to the last *block* of convolution (see Figure 1.1), the convolution is less conventional because there is $K^{(l-1)}$ *feature maps* ($K^{(l-1)}$ images) as input, denoted $I_k^{(l-1)}$ with $k = \{1, ..., K^{(l-1)}\}$.

The "convolution" that will lead to the k^{th} filtered image, $\tilde{I}_k^{(l)}$, resulting from the convolution *block* numbered l, is actually the sum of $K^{(l-1)}$ convolutions, such as:

$$\tilde{I}_k^{(l)} = \sum_{i=1}^{i=K^{(l-1)}} I_i^{(l-1)} \star F_{k,i}^{(l)}, \tag{1.3}$$

with $\{F_{k,i}^{(l)}\}_{i=1}^{i=K^{(l-1)}}$ a set of $K^{(l-1)}$ filters for a given k value.

This operation is quite unusual since each *feature map* is obtained by a *sum* of $K^{(l-1)}$ convolutions with a different filter kernel for each convolution. This operation can be seen as a spatial convolution plus a sum on the channels-axis [4].

This joined operation can be replaced by a separate operation called *SeparableConv* or *Depthwise Separable Convolutions* [16], which allows to integrate a non-linear operations (an activation function) such as a ReLU, between the spatial convolution and the convolution on the "depth" axis (for the "depth" axis we use a 1×1 filter). Thus, the *Depthwise Separable Convolution* can roughly be resumed as a weighted sum of convolution which is a more descriptive operation than just a sum of convolution (see Equation 1.3).

If we replace the operation described in the equation 1.3, by a *Depthwise Separable Convolutions* operation integrated within an *Inception* module (the Inception allows mainly to use filters of variable sizes), one gets a performance improvement [16]. In steganalysis, this has been observed in the article [107], when modifying the first two layers of the convolution module of the Figure 1.1.

As a reminder, in this document, we name a *convolution block* the set of operation made of one convolution (or many convolutions performed in parallel in the case of an Inception, and/or two convolutions in the case of a Depthwise Separable convolution), a few activation functions, a pooling, and a normalization. These steps can be formally expressed in a simplified way (case without Inception or Depthwise Separable Convolution) in recursive form by

[4] The channels axis is also referred as the "feature maps"-axis, or the "depth"-axis.

linking a *feature map* at the input of a block and the *feature map* at the output of this block:

$$I_k^{(l)} = norm\left(pool\left(f\left(b_k^{(l)} + \sum_{i=1}^{i=K^{(l-1)}} I_i^{(l-1)} \star F_{k,i}^{(l)}\right)\right)\right), \qquad (1.4)$$

with $b_k^{(l)} \in \mathbb{R}$ the scalar standing for the convolution bias, $f()$ the activation function applied pixel by pixel on the filtered image, $pool()$, the *pooling* function that is applied to a local neighborhood, and finally a normalization function.

Note that the kernels of the filters (also called weights) and the bias must be learned and are therefore modified during the back-propagation phase.

1.2.2 ACTIVATION

Once each convolution of a *convolution block* has been applied, an *activation* function, $f()$ (see Eq. 1.4), is applied on each value of the filtered image, $\tilde{I}_k^{(l)}$ (Eq. 1.2 and Eq. 1.3). This function is called the activation function with reference to the notion of binary activation found in the very first work on neuron networks. The activation function can for example be an absolute value function $f(x) = |x|$, a sinusoidal function $f(x) = sinus(x)$, a Gaussian function as in [74] $f(x) = \frac{e^{-x^2}}{\sigma^2}$, a ReLU (for *Rectified Linear Unit*): $f(x) = max(0, x)$, etc.

These functions break the linearity resulting from linear filtering performed during convolutions. Non-linearity is a mandatory property that is also exploited in *two-steps machine-learning approaches*, such as in the ensemble classifier [48] during the weak-classifiers thresholding, or through the final majority vote, or in the Rich Models with the Min-Max features [25]. The chosen activation function must be differentiable to perform the back-propagation.

The most often retained solution for the selection of an activation function is those whose derivative requires little calculation to be evaluated. Besides, functions that have low slope regions, such as the hyperbolic tangent, are also avoided, since this type of function can cause the value of the back-propagated gradient to be canceled during the back-propagation (the phenomenon of *vanishing gradient*), and thus make learning impossible. Therefore, in many networks, one very often finds the ReLU activation function, or one of its variants. For example, in the Yedroudj-Net network (see figure 1.1) we find the absolute value function, the parameterized Hard Tanh function (Trunc function), and the ReLU function. In the SRNet network [9] we only find the ReLU function.

1.2.3 POOLING

The pooling operation is to calculate the *average* or the *maximum* in a local neighborhood. In the field of classification of objects in images, the maximum pooling guarantees a local invariance in translation when recomputing the features. That said, in most steganalysis networks, it is preferred to use average pooling to preserve stego noise which is very low power.

Moreover, pooling is often coupled to a down-sampling operation (when the *stride* is greater than 1) to reduce the size (i.e., the height and width) of the resulting *feature map* compared to feature maps from the previous block. For example, in Yedroudj-Net (see figure 1.1), blocks 2, 3, and 4, reduce by a four-factor the size of the input feature maps. We can consider the pooling operation, accompanied by a stride greater than 1, as a conventional sub-sampling with preliminary low-pass filtering. This is useful for reducing the memory occupancy in the GPU. This step can also be perceived as denoising, and from the point of view of the signal processing, it induces a loss of information. It is probably better not to sub-sample in the first blocks as it was initially highlighted in [72], set up in Xu-Net [95], Ye-Net [99], Yedroudj-Net [101], and evaluated again in SRNet [9].

1.2.4 NORMALIZATION

In the first proposed networks in steganalysis, over the period 2014 − *beginning of* 2016 (Tan and Li [88], Qian *et al.* [74], Pibre *and al.* [72]), if there was a normalization, it remained local to the spatial neighborhood, with *Local Constrast Normalization*, or inter-feature, with the *Local Response Normalization*.

A big improvement occurred with the arrived of the *batch normalisation*. The *batch normalization* (BN) was proposed in 2015 [43], and was then widely adopted. This normalization is present in most of the new networks for steganalysis. The BN [43] (see Eq. 1.5) consists of normalizing the distribution of each feature of a feature map so that the average is zero and the variance is unitary, and possibly, if necessary allows re-scaling and re-translation of the distribution.

Given a random variable X whose realization is a value $x \in \mathbb{R}$ of the feature map, the BN of this value x is:

$$BN(x, \gamma, \beta) = \beta + \gamma \frac{x - E[X]}{\sqrt{Var[X] + \epsilon}}, \tag{1.5}$$

with $E[X]$ the expectation, $Var[X]$ the variance, and γ and β two scalars representing a re-scaling and an re-translation. The expectation $E[X]$ and the variance $Var[X]$ are updated at each batch, while γ and β are learned by back-propagation. In practice, the BN makes the learning less sensitive to the initialization of parameters [43], allows to use a higher learning rate which

speeds up the learning speed, and improves the accuracy of the classification [15].

In Yedroudj-Net, the terms γ and β are treated by an independent layer called *Scale Layer* (See Figure 1.1), in the same way as in ResNet [33]. The increment in performance is very minor.

1.3 MEMORY / TIME COMPLEXITY AND EFFICIENCY

Learning a network can be considered as the optimization of a function with many unknown parameters, thanks to the use of well-thought stochastic gradient descent. In the same way as for traditional neural networks, the CNN networks used for steganalysis have a large number of parameters to learn. As an example, without taking into account the Batch Normalization and Scale parameters, the Xu-Net [95] network described in the paper [101] has a number of parameters of the order of 50 000. In comparison, the network Yedroudj-Net [101], has a number of unknown parameters of the order of 500 000.

In practice, using a previous-generation GPU (Nvidia TitanX) on an Intel Core i7-5930K at 3.50 GHz $\times$ 12 with 32 GB of RAM, it takes less than a day to learn the Yedroudj-Net network on 4,000 pairs of 256×256 cover / stego images of the "BOSS [4]", three days on 14,000 pairs of 256×256 cover / stego images of " BOSS + BOWS2 [5]", and more than seven days on the 112,000 pairs of 256×256 cover / stego images of "BOSS + BOWS2 + a virtual database augmentation [100]". These long learning times are because the databases are large and have to be browsed repeatedly so that the back-propagation process makes converge the network.

Due to the large number of parameters to be learned, neural networks need a database containing a large number of examples to be in *the power-law region* [34] allowing comparisons between different networks. In addition, the examples within the learning database must be sufficiently diversified to obtain a good generalization of the network. For CNN steganalysis, with current networks (in 2018), the number of examples needed to reach a region of *good performance* (that is, as good as using a Rich Model[25] and an Ensemble Classifier [48]), in the case where there is no cover-source mismatch, is most likely in the order of 10,000 images (5000 covers and 5000 stegos) when the size is 256×256 [100]. However, the number of examples is still insufficient [100] in the sense that performance can be increased simply by increasing the number of examples. The so-called *irreducible error region* [34] probably requires more than a million images [105]; It should therefore at least 100 times more images. In addition, the images used have very small dimensions, and it would be necessary to be able to work with larger images. It is therefore

evident that in the future it will be essential to find one or more solutions to reach the region of *irreducible error*. This can be done with huge bases, and several weeks or months of apprenticeships, or by better networks, or with solutions to be invented.

Note that of course, there are tips to increase performance and it may be possible to reach faster the *irreducible error* region. We can use the transfer learning [73] and/or the curriculum learning [99] to start learning from a network that has already learned. We can use a set of CNNs [97], or a network made of sub-networks [58], which can save a few percents on accuracy. One can virtually increase the database [53], but this does not solve the problem of increasing the learning time. We can add images of a database that is similar to the test database, as it is done for example when BOSS and BOWS2 are used for learning, in the case where the test is done on BOSS [99], [100]. It is not evident that in practice we can have access to a data-base similar to the tested one. We can determine the device(s) and perform a similar development [100] to the images of the test database to increase the learning base. Once again, this approach is difficult to implement and costly in time. Note that a general rule shared by people playing with Kaggle competitions is that the main practical rules to win are [54] [5]: (i) to use an ensemble of modern networks (ResNet, DenseNet, etc.) that have learned for example on ImageNet, and then use transfer learning, (ii) to do data-augmentation, (iii) to eventually collect data to increase the database size.

1.4 LINK BETWEEN DEEP-LEARNING AND PAST APPROACHES

In previous Sections, we explained that deep-learning learning consisted of minimizing a function with many unknown parameters with a technique similar to gradient descent. In this subsection, we establish links with previous research on the subject in the steganography/steganalysis community. This sub-section tries to make the link with some past research of the domain and is an attempt to demystify deep learning.

Convolution is an essential part of CNN networks. Learning filters kernels (weights) is done by minimizing the classification error using the back-propagation procedure. It is, therefore, a simple optimization of the filter kernels. Such a strategy can be found as early as 2012 in a two-step approach

[5] The authors of [54] finished second at the Kaggle competition for *IEEE's Signal Processing Society - Camera Model Identification - Identify from which camera an image was taken.* `https://www.kaggle.com/c/sp-society-camera-model-identification`. `https://towardsdatascience.com/forensic-deep-learning-kaggle-camera-model-identification-challenge-f6a3892561bd`.

using a Rich Model and an Ensemble Classifier in the article [35]. The kernel values used to calculate the feature vector are obtained by optimization via the simplex algorithm. In this article, the goal is to minimize the probability of error of classification given by an Ensemble Classifier in the same way as with a CNN. CNNs share the same goal of building custom kernels that are well suited to steganalysis.

Looking at the first *block* of convolution just after the of pre-processing *block* (Ye-Net [99], Yedroudj-Net [101], ReST-Net [58], etc.), the convolutions act as a multi-band filtering performed on the residuals obtained from the pre-processing block (see Figure 1.1). For this first block, the network analyzes the signal residue in different frequency bands. In the past, when computing Rich Models [25], some approaches have applied a similar idea thanks to the use of a filters bank. Some approaches make a spatio-frequency decomposition via the use of Gabor filters (GFR Rich Models) [86], [93], some use Discrete Cosinus filters (DCTR Rich Models) [37], some use Steerable Gaussian filters [2], some make a projection on random carriers (PSRM Rich Models) [36], etc. For all these Rich Models, the result of these filtering is then used to calculate a histogram (co-occurrence matrix) which is then used as a vector of features. The first convolution block of the CNNs for steganalysis thus share similarities with the spatio-frequency decomposition of some Rich Models.

From the convolution blocks that start to down-sampling the feature maps, there is a summation of the results of several different convolutions. This amounts to accumulating signs of the presence of a signal (the stego noise) by observing clues in several bands. We do not find such a principle in the past. The only way to accumulate evidence was based on the computation of a histogram [25, 36] but this approach is different from what is done in CNNs. Note that in the article [81], the authors explore how to incorporate the histogram computation mechanism into a CNN network, but the results are not encouraging. Thus, starting from the second block, the mechanism involved to create a latent space separating the two classes, i.e. to obtain a feature vector per image, which makes it possible to distinguish the covers from the stegos, is different from that used in the Rich Models. Similarly, some past techniques such as non-uniform quantization [68], features selection [13], dimension reduction [70], are not directly visible within a CNN networks.

A brick present in most convolution blocks is the normalization of the feature maps. Normalization has often been used in steganalysis, for example in [52], [17], [10], etc. Within a CNN, normalization is performed among other things to obtain comparable output values in each feature maps.

The activation function introduces a non-linearity in the signal and thus makes it possible to have many convolution blocks. This non-linearity is found for example in the past in the Ensemble Classifier through the majority vote [48], or in Rich Models with the Min or Max operations [25].

The structure of a CNN network and the bricks that improve the performance of a network are now better understood in practice. As we saw above,

there is in a CNN, some part that are similar to propositions made in steganalysis in the past. Some bricks of a CNN are also explained by the fact that they are guided by computational constraints (uses of simple differentiable activation function like ReLU), or for facilitates the convergence (non-linearity allows convergence, activation function should not be too flat or steep, in order to avoid vanishing gradient or rapid variation, the shortcuts allows to avoid vanishing gradient during the back-propagation, and thus allows to create deeper networks, the batch normalization, the initialization such as Xavier, the optimization such Adam, etc). Note that some of the ingredients of a CNNs also comes from the theory of optimization of differentiable functions.

Although it is easy to use in practice a network, and to have some intuition about its behavior, it still lacks theoretical justification. For example, what should be the number of layers, and in particular the number of parameters according to a problem? In the coming years, there is no doubt that the building of a CNN network adapted for steganalysis could go through an automatic adjustment of its topology, in the spirit of works on AutoML and Progressive Neural Architecture Search (PNAS) [62], [71]. That said the theory must also try to explain what is happening inside the network. One can notably look at the work of Stéphane Mallat [65] for an attempt to explain a CNN from a signal processing point of view. Machine learning theorists will also better explain what happens in a network and why this mathematical construction work so well.

To conclude on this discussion on the links between two-step learning approaches and deep learning approaches, CNN networks as well as two-steps (Rich Models + Ensemble Classifier) approaches are not able to cope with the cover-source mismatch [12, 26]. This is a defect used by detractors [6] of neural network approaches in domains such as object recognition [3]. CNNs learn a distribution, but if it differs in test phase, then the network cannot detect it. Maybe the ultimate track is for the network to "understand" that the test database is not distributed as the learning database?

1.5 THE DIFFERENT NETWORKS USED OVER THE 2015-2018 PERIOD

A chronology of the main CNNs proposed for steganography and steganalysis from 2015 to 2018 is given in Figure 1.2. The first attempt to use Deep Learning methods for steganalysis dates back to the end of 2014 [88] with auto-encoders. At the beginning of 2015, Qian *et al.* [74] proposed to use Convolutional Neural Networks. One year later Pibre *et al.* [72] proposed to

[6] See Gary Marcus' web-press article `https://medium.com/@GaryMarcus/the-deepest-problem-with-deep-learning-91c5991f5695`.

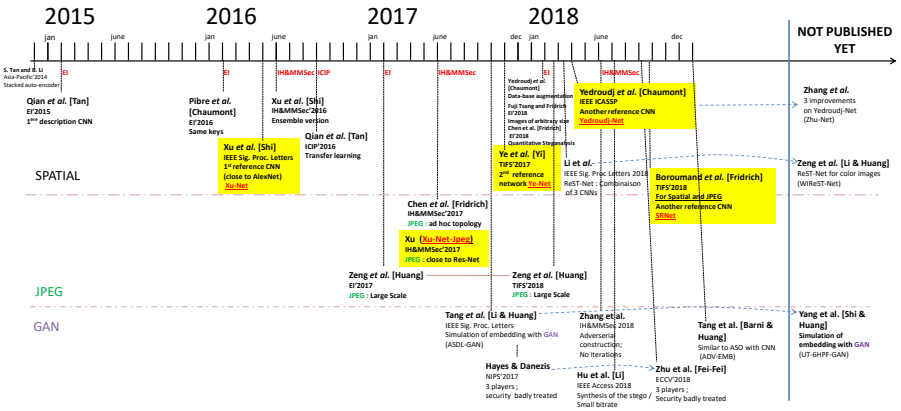

Figure 1.2 Chronology from 2015 to 2018.

pursue the study.

In 2016, the first results, close to those of the state-of-the-art, were obtained with an ensemble of CNNs [97]. The **Xu-Net** [7] [95] CNN is used as a *base learner* of an ensemble of CNNs.

Other networks were proposed in 2017, this time for JPEG steganalysis. In [106] [105], authors proposed a pre-processing inspired by the Rich Models, and the use of a big learning database. The results were close to those of the state-of-the-art. In [15], the network is built with a *phase-split* inspired by the JPEG compression process. An ensemble of CNNs was required to obtain results that were slightly better than those of the state-of-the-art. In **Xu-Net-Jpeg** [96], a CNN inspired by ResNet [33] with the *shortcut connection* trick, and 20 layers also improved the results in term of accuracy. Note that in 2018 the ResDet [42] propose a variant of **Xu-Net-Jpeg** [96] with close results.

These results were highly encouraging, but regarding the gain obtained in other image processing tasks using Deep Learning methods [55], the steganalysis results were not "10% better" compared to the classical approaches that use an Ensemble Classifier [48] with a Rich Model [25], [94] or a Rich Model with a Selection-Channel Awareness [20], [21], [18]. In 2017, the main trends to improve CNN results were using an ensemble of CNNs, modifying the topology by mimicking the Rich Models extraction process or using ResNet. In most of the cases, the design or the experimental effort was very high for a very small improvement of the performance in comparison to the network such

[7] In this chapter, we reference *Xu-Net* a CNN similar to the one given in [95], and not to the ensemble version [97].

as AlexNet [53], VGG16 [85], GoogleNet [87], ResNet [33], etc, that inspired those researches.

By the end of 2017, and in 2018, the studies have strongly concentrated on spatial steganalysis. Ye-Net [99], Yedroudj-Net [100, 101], ReST-Net [58], SRNet [9] have been published respectively in November 2017, January 2018, May 2018, and May 2019 (with a online version in September 2018). All those networks clearly surpass the "old" two-steps machine learning paradigm that was using an Ensemble Classifier [48] and a Rich Models [25]. Most of those networks can learn with not too big database (i.e. around 15 000 pairs cover/stego of 8-bits-coded images of 256×256 size from BOSS+BOWS2).

In 2018, the best network were Yedroudj-Net [101], ReST-Net [58], and SRNet [9]. Yedroudj-Net is a small network that can learn on a very small database and can be effective even without using the tricks known to improve the performances such as transfer learning [73] or virtual augmentation of the database [99], etc. This network is a good candidate when working on GANs. This network is better than Ye-Net [99], and can be improved to face the other recent networks [107]. ReST-Net [58] is a huge network made of three sub-networks which are using various pre-processing filters bank. SRNet [9] is a network that can be adapted to spatial or Jpeg steganalysis. It requires trick such as virtual augmentation and transfer learning, and thus a bigger database compared to Yedroudj-Net. Those three networks are described in Section 1.5.1.

To resume, from 2015-2017, publications were in spatial steganalysis, from 2017 to 2018, the publications were mainly on JPEG steganalysis. In 2018, publications were again mainly in spatial steganalysis. Finally, from the end of 2017, the first publications using GANs appeared. In Section 1.6 we present the new propositions using GAN and give classification per family.

In the subsection below, we report the most successful networks until the end of 2018, for various scenarios. In Section 1.5.1, we describe the *not-informed* scenario, in Section 1.5.2 we discuss the scenario known as *Side Channel Informed* (SCA), in Sections 1.5.3 we deal with the JPEG steganalysis *not-informed* and *Side Channel Informed* scenarios. In Section 1.5.4 we discuss very briefly the cover-source mismatch although for the moment the proposals using a CNN do not exist.

We will not tackle the scenario of CNN invariant to the size of the images because it is not yet mature enough. This scenario is briefly discussed in the Section 1.1.4, and the papers of Yedroudj-Net [101], Zhu-Net [107], or Tsang *et al.* [91], give first solutions.

We will not approach the scenario of quantitative steganalysis per CNN, which consists in estimating the embedded payload size. This scenario was very well treated in the paper [14] and serves as a new state-of-the-art. The approach surpasses the previous state-of-the-art approach [49] [103] that relied on a Rich Models, an Ensemble of trees, and an efficient normalization of features.

Nor will we discuss the batch steganography and pooled steganalysis with CNNs which has not been adress yet, although the work presented in [104] using two-stage machine learning can be extended to deep learning.

1.5.1 THE SPATIAL STEGANALYSIS WITH NO SIDE CHANNEL INFORMED

In early 2018 the most successful spatial steganalysis approach is the Yedroudj-Net [101] approach. The experiments are done on the BOSS database made of 10,000 images sub-sampled in 256×256. For a fair comparison, the experiments were performed by comparing the approach to the Xu-Net without Ensemble [95], to the Ye-Net network in its not-informed version [99], and also to the Ensemble Classifier [48] fed by the Spatial-Rich-Models [25]. Note that the Zhu-Net [107] (not yet published when writing this book chapter) offers three improvements to the Yedroudj-Net that allows it to be even more efficient. The improvements reported in Zhu-Net [107] are the update of the kernels filters of the pre-processing module (in the same vein as what was proposed by Matthew Stamm's team in Forensics [6]), replacing the first two convolution blocks with two modules of *Depthwise Separable Convolutions* as proposed in [16], and finally replace the global average pooling with a *Spatial Pyramid Pooling (SPP)* module as in [32].

In May 2018 the ReST-Net [58] approach has been proposed. It consists of agglomerating three networks to form a *super-network*. Each sub-net is a modified Xu-Net like network [95] resembling the Yedroudj-Net [101] network, with an Inception module on block 2 and block 4. This Inception module contains filters of the same size with a different activation function for each "path" (TanH, ReLU, Sigmoid). The first subnet performs a pre-processing with 16 Gabor filters, the second sub-network does a pre-processing with 16 SRM linear filters, and the third network does a pre-processing with 14 non-linear residuals (min and max calculated on SRM). The learning process requires four steps (one step per subnet and then one step for the *super-network*). The results are 2-5% better than Xu-Net for S-UNIWARD [39], HILL [57], CMD-HILL [59] on the BOSSBase v1.01 [4] 512×512. At the sight of the results, it is the concept of Ensemble that improves the results. Taken separately, each sub-net has a lower performance. At the moment, no comparison in a fair framework was made between an Ensemble of Yedroudj-Net and ReST-Net.

In September 2018 the SRNet [9] approach was available online. It proposes a network longer than the previous networks, which is composed of 12 convolution blocks. The network does not perform pre-processing (the filters are learned) and sub-samples the signal only from the 8th convolution block. To avoid the problem of vanishing gradient the blocks 2 to 11 use the shortcut mechanism. The Inception mechanism is also implemented from layer 8 during the pooling sub-sample phase. The learning base is augmented with the BOWS2 database as in [99] or [100], and a curriculum training mechanism

[99] is used to change from a standard payload size of 0.4 bpp to other payload sizes. Finally, gradient descent is performed by Adamax [45]. The network can be used for spatial steganalysis, for informed (SCA) spatial steganalysis (see Section 1.5.2) and for JPEG steganalysis (see Sections 1.5.3 not-SCA or SCA). Overall the philosophy remains similar to the previous networks, with three parts: pre-processing (with learned filters), convolution blocks, and classification blocks. With a simplified vision, the network corresponds to the addition of 5 blocks of convolution without pooling, just after the first convolution block of the Yedroudj-Net network. To be able to use this large number of blocks on a modern GPU, authors must reduce the number of feature maps to 16, and in order to avoid the problem of vanishing gradient, they must use within the blocks the trick of residual shortcut proposed in [33]. Note that preserving the size of the signal in the first seven blocks is a radically different approach. This idea had been put forward in [72] where the suppression of pooling had clearly improved the results. The use of modern brick like shortcuts or Inception modules also enhances performance.

It should also be noted that the training is done end-to-end without particular initialization (except when there is a curriculum training mechanism). This initial network was not compared to Yedroudj-Net [101], nor to Zhu-Net [107] at the time of writing this chapter but one can think that the update of Yedroudj-Net (i.e. Zhu-Net), and this network (SRNet) have similar performances.

1.5.2 THE SPATIAL STEGANALYSIS WITH SIDE CHANNEL INFORMED

At the end of 2018, two approaches are integrating the knowledge of the selection channel, the SCA-Ye-Net which is the SCA version of Ye-Net [99] and the SCA-SRNet which is the SCA version of SRNet [9]. The idea is to use a network used for non-informed steganalysis and to inject not only the image to be steganalyzed, but also the modification probability map. It is thus assumed that Eve knows or can have a good estimation [80] of the modification probability map, i.e. Eve has access to side-channel information.

The modification probability map is given to the pre-processing block SCA-Ye-Net [99], and equivalently to the first convolution block for SCA-SRNet [9] but the kernels values are replaced by their absolute values. After the convolution, each feature map is summed point-wise with the corresponding convolved "modification probability map". Note that the activation function of this first convolution (pre-processing block or first block) is (if it is not already the case) replaced by a ReLU. In SCA-Ye-Net the truncation activation function (*truncated linear unit (TLU)* in the article) is indeed replaced by ReLU. This makes possible to propagate (forward pass) "virtually" throughout the network information related to the image and another related to the modification probability map.

Note that this procedure to transform a non-SCA-CNN in an SCA-CNN is inspired by the propagation of the modification probability map proposed in [21] and [18]. Those two papers come as an improvement of the previous maxSRM Rich Models [20]. In maxSRM, instead of accumulating the number of occurrences in the co-occurrence matrix, an accumulation of the maximum of a local probability was used. In [21] and [18], the idea was to transformed the modification probability map in a similar way as the filtering, and then to update the co-occurrence matrix using the transformed version of the modification probability map, instead of the original modification probability map. The imitation of this principle was initially integrated into Ye-Net for CNN steganalysis, and this concept is easily transposable to most of the modern CNN.

1.5.3 THE JPEG STEGANALYSIS

The best JPEG CNN at the end of 2018 is SRNet [9]. Note that this network is the only one that has been proposed with a Side Channel Informed version.

It can be interesting to list and rapidly discuss the previous CNNs used for JPEG steganalysis. The first network, published in February 2017, was the Zeng *et al.* network and was evaluated with a million of images, and does a little evaluation of the stego-mismatch [106] [105]. Then in IH&MMSec'2017, in June 2017, two networks have been proposed: the PNet [15], and the Xu-Net-Jpeg [96]. Finally, SRNet [9] has been put online in Septembre 2018.

In Zeng et al. network [106] [105], the pre-processing block takes as input a de-quantized (real values) image, then convolved it with 25 DCT basis, and then quantize and truncate those 25 images. This pre-processing block, uses handcrafted filters kernel (DCT basis), the kernels' values are fixed, and those filters are inspired by the DCTR Rich Models [37]. There are three different quantizations, so, the pre-processing block gives 3×25 residual images. The CNN is then made of 3 subnetworks which are each producing a feature vector of dimension 512. The subnetworks are inspired by Xu-Net [95]. The three feature vectors, outputs by the three subnetworks, are then given to a fully connected structure, and the final network ends with a softmax layer.

Similarly to what has been done for spatial steganalysis, this network is using a pre-processing block inspired by a Rich Models [37]. Note that the most efficient rich models today is the Gabor Filter Rich Models [93]. Also, note that this network takes advantage of the notion of ensemble of features, which comes from the different sub-networks. The network of Zeng et al. is less efficient than Xu-Net-Jpeg [96] but gives an interesting first approach guided by the Rich Models.

The PNet main idea (and also VNet which is less efficient but takes less memory) [15] is to imitate the Phase-Aware Rich Models, such as DCTR [37], PHARM [38], or GFR [93], and thus to have a decomposition of an input image into 64 features maps which stands for the 64 phases of a Jpeg images. The pre-

processing block takes as input a de-quantized (real values) image, convolves it with four filters, the "SQUARE5×5" from the Spatial Rich Model [25], a "point" high-pass filter (referenced as "catalyst kernel") which complement the "SQUARE5×5", and two directional Gabor Filters (with angles 0 and pi).

Just after the second block of convolution, a "PhaseSplit Module" splits the residual image into 64 feature maps (one map = one phase), similarly to what was done in the Rich Models. Some interesting tricks have been used such as (1) the succession of the fixed convolutions of the pre-processing block, and a second convolution having learnable values, (2) a clever update of the BN parameters, (3) the use of the "Filter Group Option" which virtually builds sub-networks, (4) a bagging on 5-cross-validation, (5) to take the 5 last evaluations in order to give the mean error for a network, (6) to shuffle the database at the beginning of each epoch, to have a better BN behavior, and to help to the generalization, and (7) eventually to use an Ensemble. With such know-how, PNet beat the classical two-step machine learning approaches in a no-SCA and also in an SCA version (Ensemble Classifier + GFR).

The Xu-Net-Jpeg [96] was even more attractive since the approach was even slightly better than PNet and was not requiring a strong domain inspiration as in PNet. The Xu-Net-Jpeg is strongly inspired by ResNet [33], a well-established network from the machine learning community. ResNet allows the use of deeper networks thanks to the use of shortcuts. In Xu-Net, the pre-processing block takes as input dequantized (real values) images, then convolved the image with 16 DCT basis (in the same spirit as Zeng et al. network [106] [105]), and then apply an absolute value, a truncation, and a set of convolution, BN, ReLU until obtaining a feature maps of 384 dimension, which is given to a fully connected block. We can note that the max pooling or average pooling are replaced by convolutions. This network is thus really simple and was in 2017 the state-of-the-art. In a way, this kind of results shows us that the networks proposed by the machine learning are very competitive and there is not so much domain-knowledge to integrate to the topology of a network in order to obtain a very efficient network.

In 2018 the state-of-the-art CNN for JPEG steganalysis (which can also be used for spatial steganalysis) was SRNet [9]. This network has been presented in the previous Section 1.5.1. Note that for the side channel aware version of SRNet, the embedding change probability per DCTs coefficient is first map back in the spatial domain using absolute values for the DCT basis. This *side-channel* map then enter the network and is convolved with each kernel (this first convolution act as a pre-processing block). The convolutions are such that the filters kernels are modified to their absolute values. After having passed the convolution, the features maps are summed with the square of the convolved side-channel map. Note that this idea is similar to what was exposed in SCA Ye-Net version (SCA-TLU-CNN) [99] about the integration of a Side-Channel map, and to the recent proposition for Side-Channel Aware steganalysis in

JPEG with Rich Models, where the construction of Side-Channel map and especially the quantity $\delta_{uSA}^{1/2}$ [8] was defined.

1.5.4 DISCUSSION ABOUT THE MISMATCH PHENOMENON SCENARIO

The mismatch (cover-source mismatch or stego-mismatch) is a phenomenon present in machine learning, and which sees the classification performances decrease because of the inconsistency between the distribution of the learning base and the distribution of the test base. The problem is not due to an inability to generalize of machine learning algorithms, but to the lack of similar examples between the train and test base. The problem of mismatch is a problem that goes well beyond the scope of steganalysis.

In steganalysis the phenomenon can be caused by many factors. The cover-source mismatch can be be caused by the use of different photo-sensors, by different digital processing, by different camera settings (focal length, ISO, lens, etc), by different image sizes, by different image resolutions, etc [28], [8]. The stego-mismatch can be caused by different amounts of embedded bits, by different embedding algorithms.

Even if not yet really well treated and understood, the mismatch (cover-source mismatch (CSM) or stego mismatch) is a major stake of the coming years for the discipline. The results of the Alaska challenge[9] published at the ACM conference IH&MMSec'2019 will continue the reflexion.

In 2018, the CSM has been known for 10 years [12]. There are two majors currents of thought, as well as a more exotic one:

- The first current of thought is the so-called **holistic** current (that is to say, global, macroscopic, or systemic), and consists in learning all distributions [64], [63]. The use of a single CNN with millions of images [105] is in the logical continuation of this current of thought. Note that this scenario does not consider that the test set can be used during the learning. This scenario can be assimilated to an *online scenario* where the last player (from a game theory point of view) is the steganographer because in an online scenario the steganographer can change its strategy while the steganalyzer is set.

- The second current of thought is **atomistic** (= partitioned, microscopic, analytical, of type divide-and-conquer, or individualized) and consists in partitioning the distribution [67] that is to say to create a partition, and to associate a classifier for each cell of the partition. Note that

[8] uSA stand for Upper bounded Sum of Absolute values).

[9] Alaska: A challenge of steganalysis into the wilderness of the real world. `https://alaska.utt.fr/`.

an example of an atomistic approach for stego-mismatch management, using a CNN multi-classifier, is presented in [11]. The ideas given in [11] have been used by the winners of Alaska challenge. Note that again, this scenario does not consider that the test set can be used during the learning. This scenario can be assimilated to an *online scenario* where the last player (from a game theory point of view) is the steganographer because in an online scenario the steganographer can change its strategy while the steganalyst is set.

- Finally, the exotic current considers that there is a base of test (with much more than one image), and that the base is available, and usable (without the labels) during the learning. This scenario can be assimilated to an *offline scenario* where the last player (from a game theory point of view) is the steganalyser because in this an offline scenario the steganalyser is more in a forensics scenario. In this current, there are approaches of type domain adaptation, or a transfer of features GTCA [61], IMFA [50], CFT[22], where the idea is to define an invariant latent space. Another approach is ATS [56] which performs an unsupervised classification using only the test database and requires the embedding algorithm in order to re-embed a payload in the images from the test database.

Those three currents can help deriving approaches by CNN that integrate the ideas presented here. That said, the ultimate solution may be to detect the phenomenon of mismatch and raise the alarm or prohibit the decision [46]. In short, to integrate a mechanism a little smarter than just holistic or atomistic.

1.6 STEGANOGRAPHY BY GAN

In Simmons' founding article [84], steganography and steganalysis are defined as a *3-player game*. The steganographers, usually named Alice and Bob, want to exchange a message without being suspected by a third party. They must use a harmless medium, such as an image, and hide the message in this medium. The steganalyst, usually called Eve, observes the exchanges between Alice and Bob. Eve must check whether these images are natural, that is, cover images, or whether they incorporate a message, i.e. stego images.

This notion of *game* between Alice, Bob and Eve corresponds to that found in game theory. Each player tries to find the strategy that maximizes his winnings. For this, we express the problem as a min-max problem that we seek to optimize. The solution to the optimum, if it exist, is called the solution at the Nash equilibrium. When all the players are using a strategy at the Nash equilibrium, any change of strategy of a player, leads a counter attack of the

other players allowing them to increase their gains.

In 2012, Schöttle and Böhme [77], [78] have for example modeled with simplifying hypotheses a problem of steganography and steganalysis and proposed a formal solution. Schöttle and Böhme have named this approach the *optimum adaptive steganography* or *strategic adaptive steganography* in opposition to the so-called *naive adaptive steganography* that corresponds to what is done in algorithms like HUGO (2010) [69], WOW (2012) [40], S-UNIWARD / J-UNIWARD / SI-UNIWARD (2013) [39], HILL (2014) [57], MiPOD (2016) [79], Synch-Hill (2015) [19], UED (2012) [30], IUERD (2016) [66], IUERD-*UpDist-Dejoin2* (2018) [60], etc.

That said, the mathematical formalization of the steganography / steganalysis problem by game theory is difficult and often far from practical reality. Another way to determine a Nash equilibrium is to "simulate" the game. From a practical point of view, Alice plays the entire game alone, meaning that she does not interact with Bob or Eve to build his embedding algorithm. The idea is that she uses 3 algorithms (2 algorithms in a simplified version) that we name *agents*. Each of those agents will play the role of Alice, Bob [10] and Eve, and each agent runs at Alice's home. Let us note these three algorithm running at Alice's home: *Alice-agent*, *Bob-agent*, and *Eve-agent*. Alice-agent's role is to embed a message into an image so that the resulting stego image is undetectable by Eve-agent, and so Bob-agent can extract the message.

Alice can launches the game that is to say the simulation, and the agents are " fighting ". Once the agents have reach a Nash's equilibrium, Alice stops the simulation and can now keep the Alice-agent which is her *strategic adaptive embedding* algorithm and can send the Bob-agent i.e the extraction algorithm (or any equivalent information) to Bob [11]. The secret communication between Alice and Bob is now possible through the use of the Alice-agent algorithm for the embedding and the Bob-agent algorithm for the extraction.

The first precursor approaches aimed at simulating a *strategic adaptive equilibrium*, and thus proposing strategic embedding algorithms date from 2011 and 2012. The two approaches are MOD [23] and ASO [52] [51]. Whether for MOD or ASO, the game is made by competing Alice-agent and Eve-agent. In this game, Bob-agent is not used since Alice-agent is simply generating a cost map, which is then used for coding and embedding the message thanks to an STC [24]. Alice can generate a cost map for a source image with the Alice-agent, and then she can easily use the STC [24] algorithm to embed her message and to obtain the stego image. From his side, Bob just has to use the STC [24] algorithm to retrieve the message from the stego image.

---

[10] Bob is deleted in the simplified version.

[11] Note that the exchange of any secret information between Alice and Bob, prior to the use of Alice-agent and Bob-agent, requires the use of another steganographic channel. Also note that this initial sending from Alice to Bob before been able to use Alice-agent and Bob-agent is equivalent to the classical the stego-key exchange problem.

In both MOD or ASO, the "simulation" is such that the two following actions are iterated until a stop criterion is reached:

i) Alice-agent updates its embedding costs map by asking an Oracle (the Eve-agent) how best to update each embedding cost, to be even less detectable.

   In MOD (2011) [23] , the Eve-agent is an SVM. Alice-agent updates its embedding costs by reducing the SVM margin separating the covers and the stegos.

   In ASO (2012) [52] , the Eve-agent is an Ensemble Classifier [48] and is named an Oracle. Alice-agent updates its embedding costs by transforming a stego in a cover.

   In both cases, the idea is to find a displacement in the latent space (feature space) in the direction of the hyperplane separating the cover-class and the stego-class. Note that in the nowadays terminology introduced by Ian Goodfellow in 2014 [29], the Alice-agent run an adversarial attack, and the Oracle (the Eve-agent) is named a discriminator, or the classifier to be fooled.

ii) The Oracle (Eve-agent) updates its classifier. Reformulated with the terminology from machine learning, this equates to the discriminant update by re-learning it, in order to steganalysis once more the stego images generated by the Alice-agent.

In 2014, Goodfellow *et al.* [29] used neural networks to "simulate" a game with an *image generator network* and a *discriminating network* whose role was to decide whether an image was real or synthesized. The authors have named this the Generative Adversarial Networks (GAN approach). The terminology used in this paper was subsequently widely adopted. Moreover, the use of neuron networks makes the expression of the min-max problem easy. The optimization is then carried out via the back-propagation optimization process. Moreover, thanks to deep-learning libraries it is now easy to build a GAN type system. As we already mentioned before, the concept of game simulation existed already in steganography / steganalysis with MOD [23] and ASO [52] but the implementation and the optimization become easier with neural networks.

From 2017, after a period of 5 years of stagnation, the concept of simulated game is again studied in the field of steganography / steganalysis, thanks to the emergence of deep learning and GAN approaches. At the end of 2018, we can define four group or four families[12] of approaches some of which will probably merge:

[12] " Deep Learning in Steganography and Steganalysis since 2015 ", tutorial given at the " Image Signal & Security Mini-Workshop ", the 30th of October 2018, IRISA / Inria Rennes,

- The family by synthesis,
- The family by generation of the modifications probability map,
- The family by adversarial embedding by GAN (approaches misleading a discriminant),
- The family by 3-players game,

1.6.1 APPROACHES BY SYNTHESIS

The first approaches based on the *image synthesis* via a GAN [29] generator proposed the generation of images cover and then use them to make insertion by modification. Those early propositions were thus approaches *by modifications*. The argument put forward for such approaches is that the generated base would be safer. A reference often cited is that of SGAN [92] found on ArXiv, which was rejected at ICLR'2017 and subsequently never published. This unpublished paper has a lot of unspoken and errors. We should rather prefer the reference to SSGAN [83] that was published in September 2017, and that proposes the same thing: generate images and then hide a message in it. Anyway, this protocol seems to complicate the matters. It is more logic that Alice herself chooses natural images that are safe for embedding, i.e. images that innocuous, never broadcast before, adapted to the context, with lots of noise or textures [82], not well classified by a classifier [51] or with a small deflection coefficient [79], rather than generating images and then using them to hide a message.

A much more interesting approach using *synthesis* is to directly generate images that will be considered stego. To my knowledge, the first approach exploiting the GAN mechanism for image synthesis using the principle of steganography *without modifications* [27] is proposed in the article of Hu *et al.* [41] and published in July 2018.

The first step consists of learning to a network to synthesize images. In this paper, the DCGAN generator [76] is used to synthesize images with a preliminary learning thanks to GAN methodology. Thus, when fed with a vector of fixed-size uniformly distributed in $[-1, 1]$ the generator synthesizes an image. The second step consists of learning to another network to extract a vector from a synthesized image; the extracted vector must correspond to the vector given at the input of the generator which synthesizes the image. Finally, the last step consists of sending to Bob the network that extracts. Now, Alice can map a message to a fixed-size uniformly distributed vector, and then synthesize an image given the vector, and send it to Bob. Bob can extract the vector and retrieve the corresponding message.

France, DOI: 10.13140/RG.2.2.25683.22567, *http://www.lirmm.fr/ chaumont/publications.* See the slides here, and the video of the talk here.

The approaches with *no modifications* date for many years, and it is known that one of the problems is that the number of bits that can be communicated is lower compared to the approaches with modifications. That said, the gap between the approaches by *modifications* versus *no-modifications* is beginning to narrow.

Here a rapid analysis of the efficiency of the method. In the paper of Hu *et al.* [41], the capacity is around 0.018 bits per pixel (bpp) with images 64×64 [13]. In the experiment carried out, the synthesized images are either faces or food photos. An algorithm like HILL[57] (one of the most powerful algorithm on the BOSS database [82]) is detected by SRNet [9] (one of the most successful steganalysis approaches by the end of 2018) with a probability of error of Pe = 31.3% (note that a Pe of 50% is equivalent to a random detector) on a 256×256 BOSS Base for a payload size of 0.1 bpp. Due to the square root law, the Pe would be higher for the 64×64 BOSS database.

There is therefore around 0.02 bpp for the unmodified synthetic approach of Hu *et al.* [41] whose security is not yet enough evaluated, against something around 0.1 bpp for HILL, with less than one chance in three to be detected with a *clarivoyant* steganalysis i.e. a laboratory steganalysis (to contrast with real-world steganalysis [44]). There is therefore still a margin in terms of the number of bits transmitted between the *no-modification* synthesis-based approaches, such as that of Hu *et al.* approach [41], and *modification* approaches such as S-UNIWARD [39], HILL [57], MiPod [79] or even Synch-Hill [19], but this margin is reduced [14]. Also, note that there are still some issues to be addressed to ensure that approaches such as the one proposed by Hu *et al.* are entirely safe. In particular, it must be ensured that the detection of synthetic images [75] does not compromise the communication channel in the long term. It must also be ensured that the absence of a secret key does not jeopardize the approach. Indeed, if one considers that the generator is public, is it possible to use this information to deduce that a synthesis approach without modification is used.

---

[13] The vector dimension is 100. This vector is used to synthesize images of size $64 \times 64 \times 3$. There are 100×3 bits (see the mapping) per image, i.e. about 0.02 bits per pixel (bpp). The Bit Error Rate is BER $= 1 - 0.94 = 6\%$. It is, therefore, necessary to add an Error Correcting Code (ECC) so that the approach be without errors. With the use of a Hamming code $[15, 11, 3]$ that correct at best 6% of errors, the payload size is therefore around 0.018 bpp.

[14] The other families of steganography per GAN, which are *modification* based, will probably help to maintain this performance gap still during few years.

1.6.2 THE FAMILY BY GENERATION OF THE MODIFICATIONS PROBABILITY MAP

The family by generation of the modification probability map is summarized in the late 2018 in two papers: ASD-GAN [90], and UT-6HPF-GAN [98]. In this approach, there is a generator network and a discriminant network. From a cover, the generator network generates a map which is named modifications probability map. This modification probability map is then passed to an equivalent of the random draw function used in the STC [24] simulator. We then obtain a map whose values belong to {-1, 0, +1}. This map is called the modification map and corresponds to the so-called stego-noise. The discriminant network takes as inputs a cover or an image resulting from the summation (point-to-point sum) of the cover and the stego-noise generated by the generator. The discriminant's objective is to distinguish between the cover and the "*cover* + *stego-noise*" image. The generator's objective is to generate a modification map which makes it possible to mislead the discriminant the most. Of course, the generator is forced to generate a non-zero probability map by adding in the loss term a term constraining the size of the payload in addition to the term misleading the discriminant.

In practice, taking the latest approach UT-6HPF-GAN [98], the generator is a U-Net type network, the draw function is obtained by a differentiable function *double Tanh*, and the discriminant is the Xu-Net [95] enriched with 6 high-pass filters for the pre-processing in the same spirit as Ye-Net [99] or Yedroudj-Net [101].

The system learns on a first database, and then security comparisons have been made on the 256×256 BOSS database. For the moment, even if the approach is promising, the experiments are not carried out by embedding a real message by using STC, and nothing proves that the obtained modifications probability map has a meaning. For the moment, there is no guarantee that the obtained probability map would beat in practice the security performance of HILL or SUNIWARD with STC. Nor is it clear whether the generator's loss has to integrate both a security-related term and a payload-size term. Usually, one of the two criteria is fixed so that we just have to be in a payload-limited sender scenario or a security-limited sender scenario. Besides, it is not entirely sure if there is not a mismatch phenomenon with an impact on the generator (the learning database and the database used by the generator during its deployment can be different). Anyway, it is a very promising family.

1.6.3 THE FAMILY BY ADVERSARIAL-EMBEDDING BY GAN

The family by adversarial-embedding by GAN re-uses the concept of a *game simulation* which has been presented in the beginning of the Section 1.6 with a simplification of the problem since there is only two-players: Alice-agent and

Eve-agent. Historically MOD [23] and ASO [52] were the first algorithms of that type.

Recently some papers have use the adversarial concept[15] by generating a fooling example (see for example [108]), but those approaches are *not adversarial attack by GAN*. Those approaches are not dynamic, there is no game simulation, they are not trying to reach a Nash equilibrium, they are not using a GAN simulation, there is no learning alternation between the embedder and the extractor.

A paper more in the spirit of a simulation of a game which takes the principle of ASO [52], and whose objective is to update the costs map is the algorithm ADV-EMB [89] (previously named AMA on *ArXiv* arXiv:1803.09043). In this article, the authors propose to make an adversarial-embedding by GAN, by letting Alice-agent access to the gradient of the loss of the Eve-Agent (similarly to ASO, where Alice-agent has access to its Oracle (the Eve-agent)). In ADV-EMB, Alice-agent uses the gradient, of the direction to the class frontier (between classes cover and stego), to modify the costs map, and in ASO, Alice-agent uses directly the direction to the class frontier to modify the cost map.

In ADV-EMB [89], the costs map is initialized with the costs of in S-UNIWARD (for ASO it was the costs of HUGO [69]). During the iterations, the costs map is updated, but there is only a β percentage of values that are updated [16]. When the ADV-EMB iterations are stopped, the cost map is composed of a $\beta - 1$ percent of positions having a cost defined by S-UNIWARD, and β percent of positions having a cost coming from a change in the initial cost given by S-UNIWARD.

Note that updating a cost causes a cost asymmetry since the cost of a +1 change is no longer equal to the cost of a -1 change, as in ASO. Besides, the update of the two costs of a pixel is rather rough since it is a simple division by 2 for a direction (+1 or -1) and multiplication by 2 for the other direction. The sign of the gradient of the loss, calculated by choosing the cover label, for a given pixel, makes it possible to determine for each of the two directions (+ 1 / -1) if one should reduce or increases the cost. The idea is as in ASO, to deceive the discriminant since when one decides to reduce the value of a cost, it is to favor the direction of modification associated with this cost, and thus we promote to get closer to the cover class.

With such a scheme, security is improved. The fact that there are only a small number of modifications of the initial cost map probably makes it possible to preserve the initial embedding approach, and thus not to introduce

[15] An adversarial attack does not necessarily require to use a deep learning classifier.

[16] In STC, before coding the message, the pixels position of the image are shuffled thanks to the use of a pseudo-random shuffler, seeded by the secret stego-key. Note that this stego-key is shared between Alice and Bob. After the shuffling, ADV-EMB selects the last β percent pixels of the *shuffled* image, and modify their associated costs and only those one.

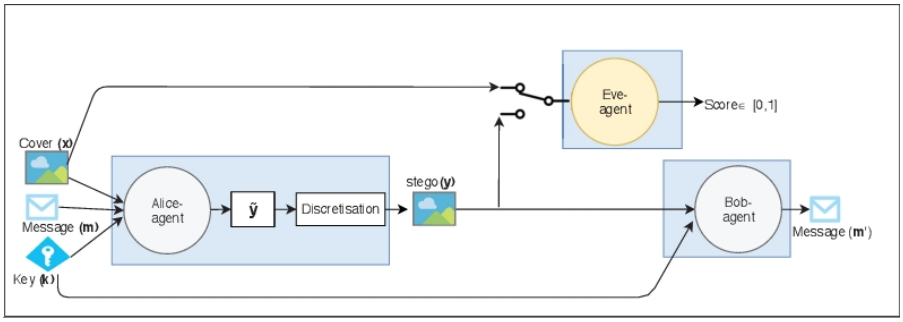

Figure 1.3 The overall architecture of the *3-players game*.

too many traces that could be detected by another steganalyzer [47]. That said, the update of the costs is probably to be refined to better take into account the value of the gradient. The approach should allow choosing the pixels that will be modified, eventually by looking to their initial costs. Finally, as it was the case for ASO, if the discriminant is not powerful enough to carry out a steganalysis then it can be totally counterproductive for the Alice-agent. There are therefore many open questions regarding the convergence criterion, the stopping criterion, the number of iterations in the alternation between Alice-agent and Eve-agent, the definition of a metric for measuring the relevance of Eve-agent, etc.

1.6.4 THE FAMILY BY 3-PLAYERS GAME

The 3-players game concept is an extension of the previous family (see the family "adversarial-embedding by GAN"). There, the three agents: Alice-agent, Bob-agent, and Eve-agent are present (see Section 1.6 for a recall of the game). Note that Alice-agent and Bob-agent are "linked" since Bob-agent is only there to add a constraint on the solution obtained by Alice-agent. Thus, the primary "game" is an antagonistic (or adversarial) game between Alice-agent and Eve-agent, while the "game" between Alice-agent and Bob-agent is rather cooperative since these two agents share the common purpose of communicating (Alice-agent and Bob-agent both want Bob-agent to be able to extract the message without errors). Figure 1.3 from [102] summarizes the principle of the 3-players game family. Alice-agent takes a cover image, a message and a stego-key, and after a discretisation step generate a stego image. This stego image is used by Bob-agent to retrieve a message. In the other side, Eve-agent has to decide wheter an image is cover or stego; this agent outputs a score.

Historically, after MOD and ASO, which only included two players, we can see the premise of the idea of three players appear in 2016 with the paper of

Abadi and Andersen [1]. In this paper, Abadi and Andersen [1] from Google Brain proposed a cryptographic toy-example for an encryption based on the use of three neural networks. The use of neural networks makes it easy to obtain a *strategic equilibrium* since the problem is expressed as a min-max problem and its optimization can be done by the back-propagation process. Naturally, this 3-players game concept can be transposed to steganography with the use of deep learning.

In December 2017 (GSIVAT; [31]), and September 2018 (HiDDeN; [109]), two different teams from the machine learning community proposed, in NIPS'2017, then in ECCV'2018, to achieve a *strategic embedding* thanks to 3 CNNs, iteratively updated, who play the role of Alice-agent, Bob-agent, and agent Eve-agent. These two articles overfly the concepts of the 3-players game, and their assertions are wrong, mainly because the concept of security and its evaluation are not correctly handled. If one places oneself in the standard framework to evaluate the empirical security of an embedding algorithm, that is to say with a clairvoyant Eve, the two approaches are very detectable. The significant issues with those two papers are first, neither of the two approaches uses a stego-key; it is equivalent to always use the same key, and it leads to very detectable schemes [72], second, there is no discretization of pixels values issued from Alice-agent, third, the computational complexity due to the use of fully connected blocks leads to un-practical approaches, and fourthly, the security evaluation is not done with a state-of-the-art steganalyzer.

At the beginning of 2019, Yedroudj et al. [102] redefine the 3-players concept, integrate the possibility to use a stego-key, treat the problem of discretization, goes through convolution modules to have a scalable solution, and use a suitable steganalyzer. The proposition is not comparable to classical adaptive embedding approaches, but there is a real potential to such an approach. The Bit Error Rate is sufficiently small to be nullified, the embedding is done in the textures parts, and the security could be improve in the future. As an example, the probability of error with a steganalysis by Yedroudj-Net[101], under equals errors prior, for a real payload size 0,3 bpp[17] for images of from BOWS2 database is 10,8%. This can, for example, be compared to the steganalysis of WOW[40] in the same conditions, which give a probability of error of 22.4%. There is still a security gap, but this approach paves the way to many research. There are still open questions on the link between Alice-agent and Bob-agent, on the use of GANs, and on the definition of losses and the tuning of the compromises between the different constraints.

[17] A Hamming error correcting codes ensures a null BER theoretically for most of the images, and thus a rate of 0.3 bpp for those images.

CONCLUSION

In this book chapter, we have practically done a complete presentation of the subject of the deep learning in steganography and steganalysis, since its appearance in 2015. As a reviewer of lots of the papers related to the subject during this period, I think and I hope this chapter will help the community to better understand what has been done and what are the next things to treat.

In this chapter, we had recalled the main bricks of a CNN. We had discussed the memory complexity, the time complexity, and practical problems for the efficiency. We had done the link with some past approaches sharing similitudes with what is currently done in a CNN. We had presented the various main networks until the beginning of 2019, and the multiple scenarios, finally we had enumerated the recent approaches for steganography with the GANs.

As recalled in this chapter, many things are not solved yet, and the major one is to be able to play with more realist hypothesis to be more "into the wild". The "holy grail" is the cover-source mismatch and the stego-mismatch, but in a way, the mismatch is a problem shared by all the machine learning community. CNNs are now well present in the steganalysis community, and the next question is probably: how to go a step farther and produce clever networks?

ACKNOWLEDGMENTS

I thank the PhD students (and the Masters' students) who directly or indirectly worked on the topic, Sarra Kouider, Amel Tuama, Jerôme Pasquet, Hasan Abdulrahman, Lionel Pibre, Mehdi Yedroudj, Ahmad Zakaria, during this period (2015-2018). Without all of them, this chapter would never have been possible.

I also thank my two colleagues, Frédéric Comby and Gérard Subsol, who help me supervising all this nice small-world.

I thank the French working group, Caroline Fontaine, Patrick Bas, Rémy Cogranne, and the defence ministry guys, with whom I have many interesting discussion and who encourage me to write this chapter.

I thank the LIRMM (the lab), ICAR (my team - with all the members), the Montpellier University and the Nîmes university, HPC@LR, for all the given resources which allowed me to run such a work.

Finally, I would like to thanks my wife, Nathalie, my four little smurfs, Noam, Naty, Coline, Mila, and Louis, who are the guardian of my mental healthiness : −)

REFERENCE

[1] Martín Abadi et David G. Andersen : Learning to protect communications with adversarial neural cryptography. *In ArXiv; Rejected from the 5th International Conference on Learning Representations, ICLR'2017.*, volume abs/1610.06918, 2016. URL http://arxiv.org/abs/1610.06918.

[2] Hasan Abdulrahman, Marc Chaumont, Philippe Montesinos et Baptiste Magnier : Color image steganalysis based on steerable gaussian filters bank. *In Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '16, pages 109–114, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4290-2. URL http://doi.acm.org/10.1145/2909827.2930799.

[3] M. A. Alcorn, Q. Li, Z. Gong, C. Wang, L. Mai, W.-S. Ku et A. Nguyen : Strike (with) a Pose: Neural Networks Are Easily Fooled by Strange Poses of Familiar Objects. *arXiv e-prints*, novembre 2018.

[4] P. Bas, T. Filler et T. Pevný : 'Break Our Steganographic System': The Ins and Outs of Organizing BOSS. *In Information Hiding, 13th International Conference, IH'2011*, volume 6958 de *Lecture Notes in Computer Science*, pages 59–70, Prague, Czech Republic, mai 2011. Springer.

[5] P. Bas et T. Furon : BOWS-2 Contest (Break Our Watermarking System), 2008. Organized between the 17th of July 2007 and the 17th of April 2008. http://bows2.ec-lille.fr/.

[6] Belhassen Bayar et Matthew C. Stamm : A deep learning approach to universal image manipulation detection using a new convolutional layer. *In Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec'2016, pages 5–10, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4290-2. URL http://doi.acm.org/10.1145/2909827.2930786.

[7] Yoshua Bengio, Aaron C. Courville et Pascal Vincent : Representation Learning: A Review and New Perspectives. *IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI*, 35(8):1798–1828, 2013.

[8] Dirk Borghys, Patrick Bas et Helena Bruyninckx : Facing the cover-source mismatch on jphide using training-set design. *In Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2018*, pages 17–22. ACM, juin 2018. URL https://doi.org/10.1145/3206004.3206021.

[9] M. Boroumand, M. Chen et J. Fridrich : Deep Residual Network for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security*, 14 (5):1181 – 1193, mai 2019. ISSN 1556-6013.

[10] Mehdi Boroumand et Jessica Fridrich : Nonlinear Feature Normalization in Steganalysis. *In Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '17, pages 45–54, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5061-7. URL http://doi.acm.org/10.1145/3082031.3083239.

[11] Jan Butora et Jessica J. Fridrich : Detection of diversified stego sources with cnns. *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2019, Part of IS&T International Symposium on Electronic Imaging, EI'2019*, Burlingame, California, USA, janvier 2019. Ingenta.

[12] G. Cancelli, G. J. Doërr, M. Barni et I. J. Cox : A comparative study of +/-1 steganalyzers. *In Workshop Multimedia Signal Processing, MMSP'2008*, pages 791–796, 2008.

[13] M. Chaumont et S. Kouider : Steganalysis by Ensemble Classifiers with Boosting by Regression, and Post-Selection of Features. *In Proceedings of IEEE International*

Conference on Image Processing, ICIP'2012, pages 1133–1136, Lake Buena Vista (suburb of Orlando), Florida, USA, septembre 2012.

[14] Mo Chen, Mehdi Boroumand et Jessica J. Fridrich : Deep learning regressors for quantitative steganalysis. *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018*, Burlingame, California, USA, 28 Jan - 2 Feb 2018. Ingenta.

[15] Mo Chen, Vahid Sedighi, Mehdi Boroumand et Jessica Fridrich : JPEG-Phase-Aware Convolutional Neural Network for Steganalysis of JPEG Images. *In Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec'17, pages 75–84, Drexel University in Philadelphia, PA, juin 2017.

[16] François Chollet : Xception: Deep learning with depthwise separable convolutions. *In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pages 1800–1807, Honolulu, HI, USA, juillet 2017. URL `https://doi.org/10.1109/CVPR.2017.195`.

[17] R. Cogranne, T. Denemark et J. Fridrich : Theoretical Model of the FLD Ensemble Classifier Based on Hypothesis Testing Theory. *In Proceedings of IEEE International Workshop on Information Forensics and Security, WIFS'2014*, pages 167–172, Atlanta, GA, décembre 2014.

[18] T. Denemark, M. Boroumand et J. Fridrich : Steganalysis features for content-adaptive jpeg steganography. *IEEE Transactions on Information Forensics and Security*, 11(8):1736–1746, août 2016.

[19] T. Denemark et J. Fridrich : Improving steganographic security by synchronizing the selection channel. *In Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '15, pages 5–14, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3587-4. URL `http://doi.acm.org/10.1145/2756601.2756620`.

[20] T. Denemark, V. Sedighi, V. Holub, R. Cogranne et J. Fridrich : Selection-channel-aware rich model for steganalysis of digital images. *In Proceedings of IEEE International Workshop on Information Forensics and Security, WIFS'2014*, pages 48–53, Atlanta, Georgia, USA, décembre 2014.

[21] Tomás Denemark, Jessica J. Fridrich et Pedro Comesaña Alfaro : Improving selection-channel-aware steganalysis features. *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2016*, pages 1–8, San Francisco, California, USA, février 2016. Ingenta.

[22] C. Feng, X. Kong, M. Li, Y. Yang et Y. Guo : Contribution-based feature transfer for jpeg mismatched steganalysis. *In Proceedings of IEEE International Conference on Image Processing, ICIP'2017*, pages 500–504, septembre 2017.

[23] T. Filler et J. Fridrich : Design of adaptive steganographic schemes for digital images. *Proc. SPIE*, 7880:78800F–78800F–14, 2011. URL `http://dx.doi.org/10.1117/12.872192`.

[24] T. Filler, J. Judas et J. Fridrich : Minimizing additive distortion in steganography using syndrome-trellis codes. *IEEE Transactions on Information Forensics and Security*, 6(3):920–935, Sept 2011. ISSN 1556-6013.

[25] J. Fridrich et J. Kodovsky : Rich models for steganalysis of digital images. *Information Forensics and Security, IEEE Transactions on*, 7(3):868–882, June 2012. ISSN 1556-6013.

[26] J. Fridrich, J Kodovský, V. Holub et M. Goljan : Breaking HUGO - The Process Discovery. *In Information Hiding, 13th International Conference, IH'2011*, volume 6958 de *Lecture Notes in Computer Science*, pages 85–101, Prague, Czech Republic, mai 2011. Springer.

[27] Jessica Fridrich : *Steganography in Digital Media*. Cambridge University Press, 2009. ISBN 9781139192903. URL `http://dx.doi.org/10.1017/CBO9781139192903`. Cambridge Books Online.

[28] Quentin Gibouloto, Rémi Cogranne et Patrick Bas : Steganalysis into the wild: How to define a source? *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018*, Burlingame, California, USA, 28 Jan - 2 Feb 2018. Ingenta.

[29] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville et Yoshua Bengio : Generative adversarial nets. *In* Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence et K. Q. Weinberger, éditeurs : *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`.

[30] L. Guo, J. Ni et Y. Q. Shi : An efficient jpeg steganographic scheme using uniform embedding. *In Information Forensics and Security (WIFS), 2012 IEEE International Workshop on*, pages 169–174, Dec 2012.

[31] Jamie Hayes et George Danezis : Generating steganographic images via adversarial training. *In* I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan et R. Garnett, éditeurs : *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 1951–1960, décembre 2017.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren et Jian Sun : Spatial pyramid pooling in deep convolutional networks for visual recognition. *In* David Fleet, Tomas Pajdla, Bernt Schiele et Tinne Tuytelaars, éditeurs : *Proceedings of the European Conference on Computer Vision, ECCV'2014*, pages 346–361, Zurich, Switzerland, 2014. Springer International Publishing.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren et Jian Sun : Deep residual learning for image recognition. *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'2016*, pages 770–778, Las Vegas, Nevada, juin 2016.

[34] J. Hestness, S. Narang, N. Ardalani, G. Diamos, H. Jun, H. Kianinejad, M. M. A. Patwary, Y. Yang et Y. Zhou : Deep Learning Scaling is Predictable, Empirically. *ArXiv e-prints*, décembre 2017.

[35] V. Holub et J. Fridrich : Optimizing Pixel Predictors for Steganalysis. *In Proceedings of SPIE Media Watermarking, Security, and Forensics, Part of IS&T/SPIE 22th Annual Symposium on Electronic Imaging, SPIE'2012*, volume 8303, pages 830309–830309–13, San Francisco, California, USA, février 2012. URL `http://dx.doi.org/10.1117/12.905753`.

[36] V. Holub et J. Fridrich : Random projections of residuals for digital image steganalysis. *Information Forensics and Security, IEEE Transactions on*, 8(12):1996–2006, Dec 2013. ISSN 1556-6013.

[37] V. Holub et J. Fridrich : Low-complexity features for jpeg steganalysis using undecimated dct. *Information Forensics and Security, IEEE Transactions on*, 10(2):219–228, Feb 2015. ISSN 1556-6013.

[38] V. Holub et J. Fridrich : Phase-Aware Projection Model for Steganalysis of JPEG Images. *In Proceedings of SPIE Media Watermarking, Security, and Forensics 2015, Part of IS&T/SPIE Annual Symposium on Electronic Imaging, SPIE'2015*, volume 9409, page 11, San Francisco, California, USA, février 2015.

[39] V. Holub, J. Fridrich et T. Denemark : Universal Distortion Function for Steganography in an Arbitrary Domain. *EURASIP Journal on Information Security, JIS*, 2014 (1), 2014.

[40] V. Holub et J. J. Fridrich : Designing Steganographic Distortion Using Directional Filters. *In Proceedings of the IEEE International Workshop on Information Forensics and Security, WIFS'2012*, pages 234–239, Tenerife, Spain, décembre 2012. IEEE.

[41] Donghui Hu, Liang Wang, Wenjie Jiang, Shuli Zheng et Bin Li : A novel image steganography method via deep convolutional generative adversarial networks. *IEEE Access*, 6:38303–38314, juillet 2018. ISSN 2169-3536.

[42] Xiaosa Huang, Shilin Wang, Tanfeng Sun, Gongshen Liu et Xiang Lin : Steganalysis of adaptive jpeg steganography based on resdet. *In Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA'2018*, volume 2018, pages 12–15, novembre 2018.

[43] Sergey Ioffe et Christian Szegedy : Batch normalization: Accelerating deep network training by reducing internal covariate shift. *In Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org, 2015. URL `http://dl.acm.org/citation.cfm?id=3045118.3045167`.

[44] A. D. Ker, P. Bas, R. Böhme, R. Cogranne, S. Craver, T. Filler, J. Fridrich et T. Pevný : Moving Steganography and Steganalysis from the Laboratory into the Real World. *In Proceedings of the 1st ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2013*, pages 45–58, Montpellier, France, juin 2013. ACM. ISBN 978-1-4503-2081-8. URL `http://doi.acm.org/10.1145/2482513.2482965`.

[45] D. P. Kingma et L.J. Ba : Adam: A method for stochastic optimization. *In Proceedings of Conference on Learning Representations, ICLR'2015*. Ithaca, mai 2015.

[46] Mustafa Anil Koak, David Ramirez, Elza Erkip et Dennis Shasha : Safepredict: A meta-algorithm for machine learning that uses refusals to guarantee correctness. *CoRR*, abs/1708.06425, 2017.

[47] J. Kodovsky, J. Fridrich et V. Holub : On dangers of overtraining steganography to incomplete cover model. *In Proceedings of the Thirteenth ACM Multimedia Workshop on Multimedia and Security*, MM&Sec '11, pages 69–76, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0806-9. URL `http://doi.acm.org/10.1145/2037252.2037266`.

[48] J. Kodovský, J. Fridrich et V. Holub : Ensemble classifiers for steganalysis of digital media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, 2012.

[49] Jan Kodovský et Jessica J. Fridrich : Quantitative steganalysis using rich models. *In Proceeding of SPIE Media Watermarking, Security, and Forensics, Part of IS&T/SPIE 23th Annual Symposium on Electronic Imaging, SPIE'2013*, volume 8665 de *SPIE Proceedings*, page 0O 111, San Francisco, California, USA, février 2013. SPIE.

[50] Xiangwei Kong, Chaoyu Feng, Ming Li et Yanqing Guo : Iterative multi-order feature alignment for jpeg mismatched steganalysis. *Journal of Neurocomputing*, 214

(C):458–470, novembre 2016. ISSN 0925-2312. URL https://doi.org/10.1016/j.neucom.2016.06.037.

[51] S. Kouider, M. Chaumont et W. Puech : Technical points about adaptive steganography by oracle (ASO). *In Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 1703–1707, Aug 2012.

[52] S. Kouider, M. Chaumont et W. Puech : Adaptive steganography by oracle (ASO). *In Multimedia and Expo (ICME), 2013 IEEE International Conference on*, pages 1–6, July 2013.

[53] A. Krizhevsky, I. Sutskever et G. E. Hinton : ImageNet Classification with Deep Convolutional Neural Networks. *In* F. Pereira, C.J.C. Burges, L. Bottou et K.Q. Weinberger, éditeurs : *Advances in Neural Information Processing Systems 25, NIPS'2012*, pages 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[54] Artur Kuzin, Artur Fattakhov, Ilya Kibardin, Vladimir Iglovikov et Ruslan Dautov : Camera model identification using convolutional neural networks. *In Proceedings of the 2nd International Workshop on Big Data Analytic for Cyber Crime Investigation and Prevention, co-located with IEEE Big Data 2018*, décembre 2018.

[55] Yann LeCun, Yoshua Bengio et Geoffrey Hinton : Deep learning. *Nature*, 521 (7553):436–444, mai 2015.

[56] Daniel Lerch-Hostalot et David Megías : Unsupervised steganalysis based on artificial training sets. *Engineering Applications of Artificial Intelligence*, 50(C):45–59, avril 2016. ISSN 0952-1976. URL https://doi.org/10.1016/j.engappai.2015.12.013.

[57] B. Li, M. Wang, J. Huang et X. Li : A new cost function for spatial image steganography. *In Image Processing (ICIP), 2014 IEEE International Conference on*, pages 4206–4210, Oct 2014.

[58] B. Li, W. Wei, A. Ferreira et S. Tan : Rest-net: Diverse activation modules and parallel subnets-based cnn for spatial image steganalysis. *IEEE Signal Processing Letters*, 25(5):650–654, May 2018. ISSN 1070-9908.

[59] Bin Li, Ming Wang, Xiaolong Li, Shunquan Tan et Jiwu Huang : A strategy of clustering modification directions in spatial image steganography. *IEEE Transaction on Information Forensics and Security*, 10(9):1905–1917, 2015.

[60] Weixiang Li, Weiming Zhang, Kejiang Chen, Wenbo Zhou et Nenghai Yu : Defining joint distortion for jpeg steganography. *In Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec'18, pages 5–16, New York, NY, USA, 2018. ACM.

[61] X. Li, X. Kong, B. Wang, Y. Guo et X. You : Generalized transfer component analysis for mismatched jpeg steganalysis. *In Proceedings of IEEE International Conference on Image Processing, ICIP'2013*, pages 4432–4436, septembre 2013.

[62] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan L. Yuille, Jonathan Huang et Kevin Murphy : Progressive Neural Architecture Search. *In Proceedings of the European Conference on Computer Vision, ECCV'2018*, volume 11205 de *Lecture Notes in Computer Science*, pages 19–35. Springer, septembre 2018.

[63] I. Lubenko et A. D. Ker : Going from small to large data in steganalysis. *In Media Watermarking, Security, and Forensics III, Part of IS&T/SPIE 22th Annual Sym-*

posium on Electronic Imaging, SPIE'2012, volume 8303, San Francisco, California, USA, février 2012.

[64] I. Lubenko et A. D. Ker : Steganalysis with mismatched covers: do simple classifiers help? *In Multimedia and Security Workshop, MM&Sec'2008, Proceedings of the 14th ACM multimedia*, MM&Sec'2012, pages 11–18, Coventry, UK, septembre 2012. ISBN 978-1-4503-1417-6. URL `http://doi.acm.org/10.1145/2361407.2361410`.

[65] Stéphane Mallat : Understanding Deep Convolutional Networks. *Philosophical Transactions of the Royal Society. Series A, Mathematical, physical, and engineering sciences*, 374, 2016.

[66] Yuanfeng Pan, Jiangqun Ni et Wenkang Su : Improved Uniform Embedding for Efficient JPEG Steganography. *In* Xingming Sun, Alex Liu, Han-Chieh Chao et Elisa Bertino, éditeurs : *Proceedings of the International Conference on Cloud Computing and Security*, volume 10039 de *Part of the Lecture Notes in Computer Science book series (LNCS)*, pages 125–133, Cham, 2016. Springer International Publishing.

[67] Jérôme Pasquet, Sandra Bringay et Marc Chaumont : Steganalysis with cover-source mismatch and a small learning database. *In Proceedings of the 22nd European Signal Processing Conference, EUSIPCO'2014*, pages 2425–2429, septembre 2014.

[68] T. Pevný : Co-occurrence Steganalysis in High Dimensions. *In Proceeding of SPIE Media Watermarking, Security, and Forensics, Part of IS&T/SPIE 22th Annual Symposium on Electronic Imaging, SPIE'2012*, volume 8303, pages 83030B–83030B–13, San Francisco, California, USA, février 2012. URL `http://dx.doi.org/10.1117/12.908914`.

[69] T. Pevný, T. Filler et P. Bas : Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. *In* R. Bhme, P.W.L. Fong et R. Safavi-Naini, éditeurs : *Information Hiding, 12th International Conference, IH'2010*, volume 6387 de *Lecture Notes in Computer Science*, pages 161–177, Calgary, Alberta, Canada, juin 2010. Springer.

[70] T. Pevný et A. D. Ker : The Challenges of Rich Features in Universal Steganalysis. *In Proceeding of SPIE Media Watermarking, Security, and Forensics, Part of IS&T/SPIE 23th Annual Symposium on Electronic Imaging, SPIE'2013*, volume 8665, pages 86650M–86650M–15, San Francisco, California, USA, février 2013. URL `http://dx.doi.org/10.1117/12.2006790`.

[71] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le et Jeff Dean : Efficient Neural Architecture Search via Parameters Sharing. *In Proceedings of Thirty-fifth International Conference on Machine Learning, ICML'2018*, juillet 2018.

[72] L. Pibre, J. Pasquet, D. Ienco et M. Chaumont : Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch. *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2016, Part of I&ST International Symposium on Electronic Imaging, EI'2016*, pages 1–11, San Francisco, California, USA, février 2016.

[73] Y. Qian, J. Dong, W. Wang et T. Tan : Learning and transferring representations for image steganalysis using convolutional neural network. *In Proceedings of IEEE International Conference on Image Processing, ICIP'2016*, pages 2752–2756, Phoenix, Arizona, septembre 2016.

[74] Yinlong Qian, Jing Dong, Wei Wang et Tieniu Tan : Deep Learning for Steganalysis via Convolutional Neural Networks. *In Proceedings of Media Watermarking, Security, and Forensics 2015, MWSF'2015, Part of IS&T/SPIE Annual Symposium on Elec-*

tronic Imaging, SPIE'2015, volume 9409, pages 94090J–94090J–10, San Francisco, California, USA, février 2015.

[75] W. Quan, K. Wang, D. Yan et X. Zhang : Distinguishing Between Natural and Computer-Generated Images Using Convolutional Neural Networks. *IEEE Transactions on Information Forensics and Security*, 13(11):2772–2787, novembre 2018. ISSN 1556-6013.

[76] Alec Radford, Luke Metz et Soumith Chintala : Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *In Proceedings of the International Conference on Learning Representations, ICLR'2016*, mai 2016.

[77] Pascal Schöttle et Rainer Böhme : A game-theoretic approach to content-adaptive steganography. *In* Matthias Kirchner et Dipak Ghosal, éditeurs : *Proceedings of the 14th International Conference on Information Hiding, IH'12*, volume 7692, pages 125–141. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-36372-6.

[78] P. Schttle et R. Bhme : Game theory and adaptive steganography. *IEEE Transactions on Information Forensics and Security*, 11(4):760–773, April 2016.

[79] Vahid Sedighi, Rémi Cogranne et Jessica Fridrich : Content-adaptive steganography by minimizing statistical detectability. *Information Forensics and Security, IEEE Transactions on*, 11(2):221 – 234, Feb 2016. ISSN 1556-6013.

[80] Vahid Sedighi et Jessica Fridrich : Effect of imprecise knowledge of the selection channel on steganalysis. *In Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec'2015, pages 33–42, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3587-4. URL `http://doi.acm.org/10.1145/2756601.2756621`.

[81] Vahid Sedighi et Jessica J. Fridrich : Histogram Layer, Moving Convolutional Neural Networks Towards Feature-Based Steganalysis. *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2017*, pages 50–55, San Francisco, California, USA, février 2017. Ingenta.

[82] Vahid Sedighi, Jessica J. Fridrich et Rémi Cogranne : Toss that BOSSbase, Alice! *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2016*, pages 1–9, San Francisco, California, USA, février 2016. Ingenta.

[83] Haichao Shi, Jing Dong, Wei Wang, Yinlong Qian et Xiaoyu Zhang : SSGAN: Secure Steganography Based on Generative Adversarial Networks. *In Proceedings of the 18th Pacific-Rim Conference on Multimedia, PCM'2017*, volume 10735 de *Lecture Notes in Computer Science*, pages 534–544. Springer, septembre 2017.

[84] G. J. Simmons : The subliminal channel and digital signatures. *In* Edited by D. Chaum, éditeur : *Proceeding of Crypto'83*, pages 51–67. New York, Plenum Press, août 1983.

[85] K. Simonyan et A. Zisserman : Very Deep Convolutional Networks for Large-Scale Image Recognition. *In Proceeding of International Conference on Learning Representations, ICLR'2015*, mai 2015.

[86] Xiaofeng Song, Fenlin Liu, Chunfang Yang, Xiangyang Luo et Yi Zhang : Steganalysis of Adaptive JPEG Steganography Using 2D Gabor Filters. *In Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'20015*, pages 15–23, Portland, Oregon, USA, juin 2015. ISBN 978-1-4503-3587-4. URL `http://doi.acm.org/10.1145/2756601.2756608`.

[87] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke et Andrew Rabinovich : Going deeper with convolutions. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'2015*, pages 1–9, juin 2015.

[88] S. Tan et B. Li : Stacked convolutional auto-encoders for steganalysis of digital images. *In Proceedings of Signal and Information Processing Association Annual Summit and Conference, APSIPA'2014*, pages 1–4, Siem Reap, Cambodia, décembre 2014.

[89] W. Tang, B. Li, S. Tan, M. Barni et J. Huang : Cnn-based adversarial embedding for image steganography. *IEEE Transactions on Information Forensics and Security*, janvier 2019. ISSN 1556-6013.

[90] W. Tang, S. Tan, B. Li et J. Huang : Automatic Steganographic Distortion Learning Using a Generative Adversarial Network. *IEEE Signal Processing Letters*, 24(10): 1547–1551, octobre 2017. ISSN 1070-9908.

[91] Clement Fuji Tsang et Jessica J. Fridrich : Steganalyzing images of arbitrary size with cnns. *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018*, Burlingame, California, USA, 28 Jan - 2 Feb 2018.

[92] Denis Volkhonskiy, Ivan Nazarov, Boris Borisenko et Evgeny Burnaev : Steganographic Generative Adversarial Networks. never published, 2017.

[93] Chao Xia, Qingxiao Guan, Xianfeng Zhao, Zhoujun Xu et Yi Ma : Improving GFR Steganalysis Features by Using Gabor Symmetry and Weighted Histograms. *In Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '17, pages 55–66, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5061-7. URL `http://doi.acm.org/10.1145/3082031.3083243`.

[94] Chao Xia, Qingxiao Guan, Xianfeng Zhao, Zhoujun Xu et Yi Ma : Improving GFR Steganalysis Features by Using Gabor Symmetry and Weighted Histograms. *In Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec'17, page 11, Drexel University in Philadelphia, PA, juin 2017.

[95] G. Xu, H. Z. Wu et Y. Q. Shi : Structural Design of Convolutional Neural Networks for Steganalysis. *IEEE Signal Processing Letters*, 23(5):708–712, mai 2016.

[96] Guanshuo Xu : Deep Convolutional Neural Network to Detect J-UNIWARD. *In Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec'17, pages 67–73, Drexel University in Philadelphia, PA, juin 2017.

[97] Guanshuo Xu, Han-Zhou Wu et Yun Q. Shi : Ensemble of CNNs for Steganalysis: An Empirical Study. *In Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec'16, pages 103–107, Vigo, Galicia, Spain, juin 2016. ISBN 978-1-4503-4290-2.

[98] Jianhua Yang, Kai Liu, Jiwu Huang, , Yun-Qing Shi et Xiangui Kang : **Under Review** ; *an arxiv preliminary version was online in april 2018, http://arxiv.org/abs/1804.07939, and was named spatial image steganography based on generative adversarial network* the new name of the paper could be *a novel embedding cost learning framework using gan. IEEE Transactions on Information Forensics and Security, TIFS*, XXX(XXX):XXX–XXX, 2019.

[99] Jian Ye, Jiangqun Ni et Y. Yi : Deep learning hierarchical representations for image steganalysis. *IEEE Transactions on Information Forensics and Security, TIFS*, 12 (11):2545–2557, novembre 2017.

[100] Mehdi Yedroudj, Marc Chaumont et Frédéric Comby : How to augment a small learning set for improving the performances of a CNN-based steganalyzer? *In Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018*, Burlingame, California, USA, 28 Jan - 2 Feb 2018.

[101] Mehdi Yedroudj, Frdric Comby et Marc Chaumont : Yedrouj-Net: An efficient CNN for spatial steganalysis. *In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'2018*, Calgary, Alberta, Canada, avril 2018. IEEE.

[102] Mehdi Yedroudj, Frdric Comby et Marc Chaumont : Can we model steganography with a 3 players game? *In* **Submitted to** *Proceedings of the 7th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019*, Paris, France, juillet 2019. ACM.

[103] Ahmad Zakaria, Marc Chaumont et Gérard Subsol : Quantitative and Binary Steganalysis in JPEG: A Comparative Study. *In Proceedings of the European Signal Processing Conference, EUSIPCO'2018*, pages 1422–1426, septembre 2018.

[104] Ahmad Zakaria, Marc Chaumont et Gérard Subsol : Under submission Pooled Steganalysis in JPEG:how to deal with the spreading strategy? *In* **Submitted to** *Proceedings of the 7th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019*, Paris, France, juillet Forthcoming 2019. ACM. submitted.

[105] J. Zeng, S. Tan, B. Li et J. Huang : Large-scale jpeg image steganalysis using hybrid deep-learning framework. *IEEE Transactions on Information Forensics and Security*, 13(5):1200–1214, May 2018. ISSN 1556-6013.

[106] Jishen Zeng, Shunquan Tan, Bin Li et Jiwu Huang : Pre-training via fitting deep neural network to rich-model features extraction procedure and its effect on deep learning for steganalysis. *In Proceedings of Media Watermarking, Security, and Forensics 2017, MWSF'2017, Part of IS&T Symposium on Electronic Imaging, EI'2017*, page 6, Burlingame, California, USA, janvier 2017.

[107] R. Zhang, F. Zhu, J. Liu et G. Liu : Under submission Depth-wise separable convolutions and multi-level pooling for an efficient spatial CNN-based steganalysis; (previously named "Efficient feature learning and multi-size image steganalysis based on CNN" on ArXiv). *XXX*, Forthcoming 2019. submitted.

[108] Yiwei Zhang, Weiming Zhang, Kejiang Chen, Jiayang Liu, Yujia Liu et Nenghai Yu : Adversarial examples against deep neural network based steganalysis. *In Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, IH&MMSec '18, pages 67–72, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5625-1. URL `http://doi.acm.org/10.1145/3206004.3206012`.

[109] Jiren Zhu, Russell Kaplan, Justin Johnson et Li Fei-Fei : HiDDeN: Hiding Data With Deep Networks. *In* Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu et Yair Weiss, éditeurs : *Proceedings of the 15th European Conference on Computer Vision, ECCV'2018*, volume 11219 de *Lecture Notes in Computer Science*, pages 682–697. Springer, septembre 2018.