ACM Information Hiding and Multimedia Security, June 22–24, 2020, due to the Covid-19 IH&MMSec was a virtual conference.

Pixels-off: Data-augmentation Complementary Solution for Deep-learning Steganalysis

Mehdi YEDROUDJ
mehdi.yedroudj@lirmm.fr
Montpellier University, LIRMM (UMR5506) / CNRS,
Montpellier, FRANCE

Marc CHAUMONT
marc.chaumont@lirmm.fr
Montpellier University, LIRMM (UMR5506) / CNRS,
Montpellier, FRANCE
Nîmes University, Nîmes, FRANCE

Frederic COMBY
frederic.comby@lirmm.fr
Montpellier University, LIRMM (UMR5506) / CNRS,
Montpellier, FRANCE

Ahmed OULAD AMARA
ahmed.oulad-amara@lirmm.fr
Montpellier University, LIRMM (UMR5506) / CNRS,
Montpellier, FRANCE

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

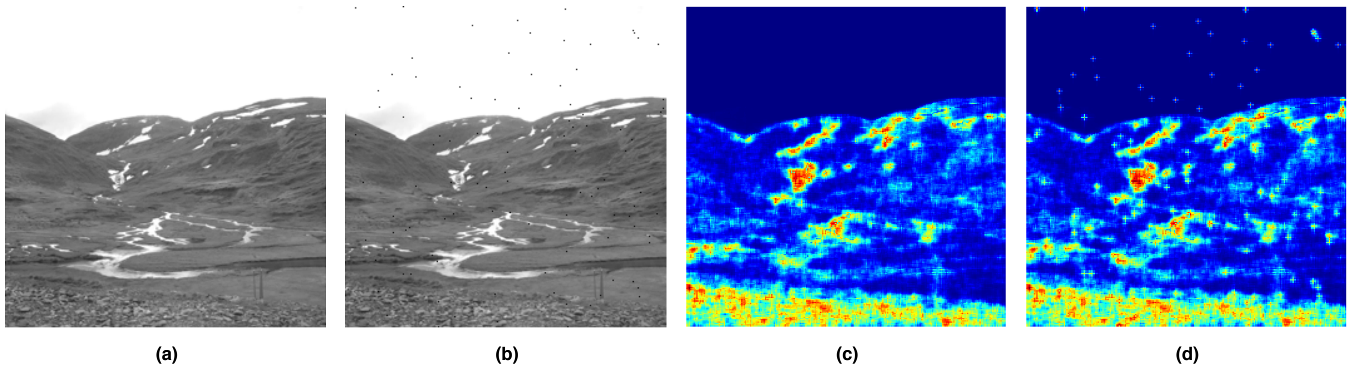

Figure 1: The "pixels-off" process illustrated on a 256×256 pixels image 1.pgm from BOSSBase [1]. Fig.a represents the cover and Fig.b its "pixels-off" version with 100 pixels switched-off. Fig.c (resp. Fig.d) is the embedding modification probabilities map for the cover (resp. "pixels-off" version) obtained from the S-UNIWARD model with a payload of 0.4bpp [12].

ABSTRACT

After 2015, CNN-based steganalysis approaches have started replacing the two-step machine-learning-based steganalysis approaches (feature extraction and classification), mainly due to the fact that they offer better performance.

In many instances, the performance of these networks depend on the size of the learning database. Until a certain point, the larger the database, the better the results. However, working with a large database with controlled acquisition conditions is usually rare or unrealistic in an operational context. An easy and efficient approach is thus to augment the database, in order to increase its size, and therefore to improve the efficiency of the steganalysis process.

In this article, we propose a new way to enrich a database in order to improve the CNN-based steganalysis performance. We have named our technique "pixels-off". This approach is efficient, generic, and is usable in conjunction with other data-enrichment approaches. Additionally, it can be used to build an informed database that we have named "Side-Channel-Aware databases" (SCA-databases).

, ,

CCS CONCEPTS

• **Applied computing** → **Computer forensics**; • **Computing methodologies** → Image processing; • **Computer systems organization** → Neural networks.

KEYWORDS

steganalysis; deep-learning; CNN; data-augmentation; steganography; pixels-off

1 INTRODUCTION

Modern numerical steganography is the art of concealing a message in a digital cover [7]. In other words, steganography is the set of techniques allowing secret communication between two protagonists, conventionally named Alice and Bob [19].

When we study steganography, one has to check the security of the secret communication occurring between Alice and Bob. This security analysis is named **steganalysis**, and we conventionally attribute this role to another entity named Eve.

Since the use of deep-learning for steganalysis purposes [4], many efficient models have been proposed, these are now seen as state-of-the-art references for spatial image steganalysis such as, Yedroudj-Net [24], SRNet [3], CovPool-Net [6], Zhu-Net [26], Calpa-Net [20].

For these networks, the number of images needed to reach the *region of good performance* (that is the performance of a Rich Model [8] with an Ensemble Classifier [14]), is about 10,000 images (5,000 covers and 5,000 stegos) for the learning phase. This is the case when there is no cover-source mismatch, and the images' size is 256×256 pixels [23].

However, this quantity of images is insufficient [23] in the sense that performance can be increased simply by augmenting the size of the training set. In steganalysis, the so-called *irreducible error region* [10] probably requires many more images than those normally used today. In the case of [25], it takes one million images, which corresponds to 100 times more images than those usually used for the learning phase.

Working with an extensive database is not necessarily the only solution to reach the *irreducible error region* faster. We can use transfer learning [17] and/or curriculum learning [22] to start learning from a network that has already been trained. We can also use a set of CNNs [21] or a network made of sub-networks [16], which can save a few percentage points on accuracy.

In this paper, we focus on the database enrichment in a clairvoyant scenario with no SCA. In this scenario, a CNN-steganalyzer is used inline to analyze test set images that have never been seen before (test phase). Prior to its inline deployment, during the learning phase, the CNN-steganalyzer has access to images whose distributions are similar to those of the test set (i.e. same sources, same development[1], same sizes, same embedding process, and same payload).

The database enrichment is usually done with virtual augmentation [15], or by adding another similar dataset such as BOWS2 [22], [23]. We can make new acquisitions with the same devices used to produce the test database, and perform images developments similar to the ones used to generate the test database [23].

Here, we propose to duplicate the learning set and to add a particular noise. Adding noise for database augmentation is a classical principle in machine learning and especially in image classification. Nevertheless, it did not give compelling results for CNN-based steganalysis. This can be explained by the fact that the added noise alters the initial surface statistical regularities [13], leading to a mismatch between the learning set and the test one.

[1]In this paper, the "development" stands for the numerical processes transforming a RAW color image to a 256×256 8-bits grey-levels image

We thus propose the *pixels-off* noise principle. After explaining the reasoning behind this approach, its principle and its statistical properties in section 2, we present current state-of-the-art CNNs (section 3); then, we give the experimental methodology in section 4. We intensively evaluate its accuracy and usability in section 5. This evaluation is performed alongside two state-of-the-art networks (Yedroudj-Net [24] and CovPool-Net [6]) using two well-established embedding algorithms (S-UNIWARD [12] and WOW [11]) at 0.2 bpp and 0.4 bpp. We use two well-known test databases (BOSS [1] and Alaska [5]), and we compare the *pixels-off* technique with many other enrichment propositions. In section 6, we further analyze the property of this technique, and we discuss its usage.

2 PIXELS-OFF PROPOSITION

2.1 The enrichment approaches

Using an *initial* learning set and a fixed test set, database augmentation (enrichment) consists in increasing the number of images of the *initial* learning set. In this paper, we explore an enrichment technique which improves the CNN performance, without requiring access: 1) to images other than those of the *initial* learning database and 2) to the original cameras (or the original raw images) or any knowledge about the development. The approach we are proposing is, therefore, always feasible.

As mentioned previously, the virtual augmentation (VA) which flips and rotates the learning set without interpolation [15], or eventually adding another database [22], can be used to enrich the *initial* learning set.

The steganalyzer, Eve, can also decide to duplicate the *initial* learning set and apply image processing on it with the purpose of enriching the *initial* learning set. For example, in [24], the authors propose two processes. The first one consists in applying a sub-pixel image translation, of 0.5 pixel, on the padded images, followed by a cropping operation to obtain a 256×256 pixel image. The second one consists in upsampling with the Lanczos3 filter to obtain a 512×512 pixel image and then downsampling it with the same interpolation kernel to get an image of 256×256 pixels. Unfortunately, these processes applied directly to the images give an important accuracy reduction. Also note that the usual approaches consisting of adding noise, like in image classification, are also not as efficient (see section 5). These signal processing approaches are not preserving, or are too far from the *initial* learning set statistics. In short, a cover-source mismatch phenomenon [9] occurs because the added images do not share the same surface statistical regularities [13], i.e. the same pixel distribution.

As experimentally observed in [13], the networks tend to learn surface statistical regularities, instead of high-level abstraction. As appended comments, the authors of [13] observe that:

(1) better results are obtained when tests are performed on an unfiltered database (unfiltered images) while training is performed on low-pass filtered images, compared to training and testing on unfiltered images.
(2) when merging the filtered and unfiltered images databases, the generalization capabilities improve, but results are not surpassing the results obtained when the learning and testing databases are similar.

These first observations, even if they are related to an image classification context, suggest that well-chosen noises, applied to cover images, could lead to beneficial enrichment of an *initial* training set, i.e. an improvement in the steganalysis accuracy.

Our proposition consists in switching off a small proportion of pixels from the duplicated *initial* database of cover images. We name this filtering the "pixels-off" technique. We thus keep the idea of random masking, but rather than applying it to the Fourier domain; we apply it directly to the spatial domain so that we keep the average cover statistics. Pixel zeroing is similar to a noise addition, but to our knowledge, no other noise had previously worked to augment a database for steganalysis purposes. The *pixels-off* technique can be seen as a simulation of dead pixels, or faulty sensors, which frequently occurs in various application domains.

2.2 The pixels-off technique

The **Algorithm 1** describes how the pixels-off technique is used for database enrichment.

Algorithm 1: *Pixels-off* database generation.

```
Data: training-set-list, P;
// P: numbers of pixels to nullify
Result: pixels-off-list
1  pixels-off-list = [];
2  cover-list = get-covers (training-set-list);
3  N = length (cover-list);
4  (h,w) = size (cover-list(0));
5  for $i = 1, i \leq N, i++$ do
6      x = cover-list(i); // x: a cover
7      z = dispatch-P-zeros (h,w, P);
       // randomly spread P zeros, in an image made
          of 1, of size $h \times w$
8      $x_{off} = x \odot z$; // $x_{off}$: a cover with P pixels-off
9      $y_{off}$ = embedding ($x_{off}$ , payload);
10     pixels-off-list.append([$x_{off}$ , $y_{off}$])
11 end
12 return pixels-off-list;
```

As shown in **Algorithm 1**, 2nd line, only the covers are retained out of the training set since they are the ones considered for pixels-off.

The loop **for** (line 5) iterates through the entire cover list. For each iteration, a cover, x, is selected (line 6). A binary image, z, whose size is equal to the cover's one, is generated (line 7) by randomly spreading P zeros in an image made of ones. Note that the variable P defines the number of pixels to be switched to zero. In practice, the value of this variable should be small. Therefore, it is chosen so that only a small portion of the image (between 1.5‰and 1.5%) is switched off.

The cover pixels-off version, x_{off}, is obtained by applying an element-wise multiplication between the cover, x, and the binary image, z. The obtained image, x_{off}, is thus a new cover. The latter is subsequently used to generate a stego through the same embedding algorithm and the same payload as those used to obtain the *training-set-list* stegos (line 9). These generated pairs (cover/stego) are then appended to the *pixels-off-list* (line 10), which will later be added to the training set.

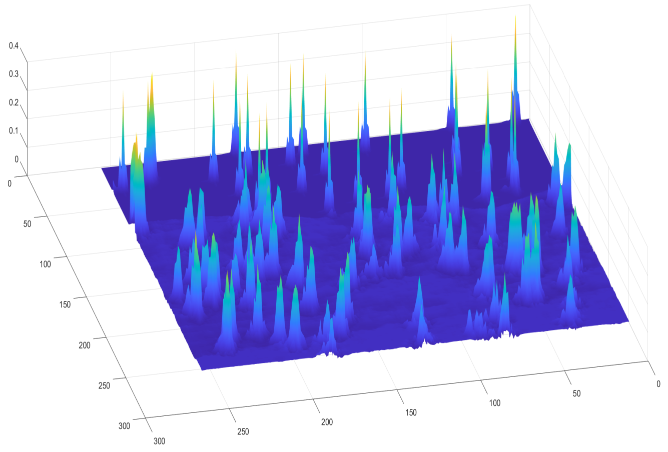

Figure 2: Visualization by elevation for the differences between the embedding modification probability maps of the cover (Figure 1.c) and its pixels-off version (Figure 1.d).

2.3 The impact on the covers/stegos statistics

At first glance, the "pixels-off" noise could seem to be a noise of a high power relative to the original cover image. However, this is not the case. Take for example the noisy image in Figure 1.b, its PSNR is 31,932. The number of modified pixels is very small with respect to the image size. The proportion of the modified pixels is indeed between 1.5‰ and 1.5%. This can be visually observed in Figure 1. The cover is given in Figure 1.a, and the cover with 1.5 ‰ pixels-off, is given in Figure 1.b. For these two images, the difference between their average is 0.23, the difference between their standard deviation is also 0.23, and the difference between their Entropy is 0.0081.

The first and second moments and the probability distribution are therefore approximately equal. There are nevertheless, some local differences between the two images, such as the local variance computed on a window close to pixels-off.

That said, the majority of the pixels of a pixels-off noisy cover keep the same correlation/dependencies as the original cover. Those correlations/dependencies are what is defining the source (i.e. the surface statistical regularity). So, the "pixels-off" technique only implies a small perturbation of the statistics of the covers, but preserves the average statistics. This is exactly what we need to enrich the *initial* learning set with additional covers.

The impact on the stegos is described below. Figure 1.c shows the embedding modification probability, when using S-UNIWARD [12] embedding model, for each pixel of the cover image in Figure 1.a. Figure 1.d shows the embedding modification probability, for the cover with pixels-off given in Figure 1.b. The red pixels stand for the high probabilities and the blue ones for low probabilities. Figure 2 better illustrates the difference between these two images with a visualization by elevation. The "peaks" stand for pixel areas where there is a difference between the embedding modification probability for the cover image and its corresponding pixels-off one.

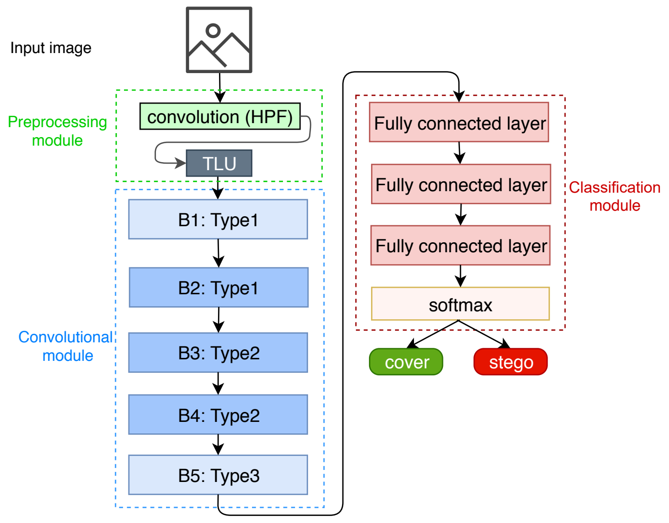

Figure 3: Yedroudj-Net CNN architecture.

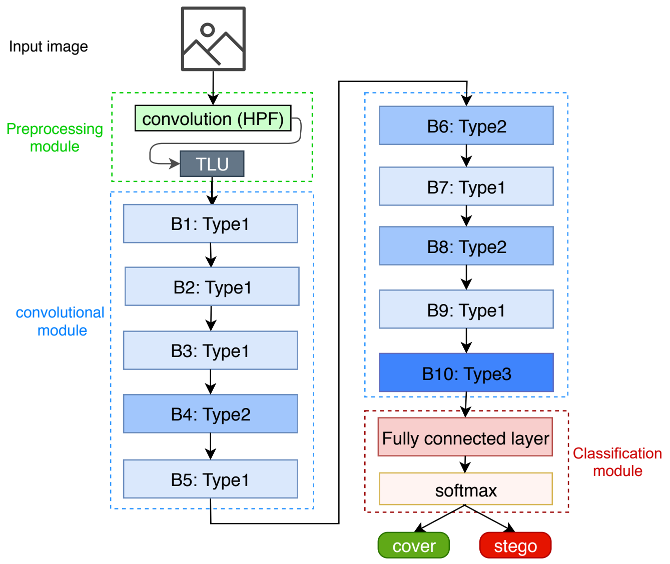

Figure 4: CovPool-Net CNN architecture.

The peaks only occur to the pixels-off positions, and they are all centred on the pixels-off positions.

One can observe that the probabilities between the two images are similar, except in areas close to a pixel-off. This occurs because a pixel-off area is producing a saliency in the image. It is captured by the filters computing the cost map, such as those of S-UNIWARD, and is identified as an area where the embedding is favorable, i.e. whose cost value is small, thus, whose modification probability is high. A stego image and its "pixels-off" noisy version will not have the exact same modified pixels when the same message is embedded. This property is also interesting because it produces noisy stegos different from *initial* stegos, but still statistically close.

3 PRESENTATION OF TWO STATE-OF-THE-ART CNNS FOR SPATIAL STEGANALYSIS

Our study on the pixels-off data augmentation technique, is conducted using two convolutional neural networks (CNNs) from existing state-of-the-art spatial steganalysis methods: Yedroudj-Net [24] (section 3.1), and CovPool-Net [6] (section 3.2), over two databases BOSS base [1] and ALASKA [5].

3.1 Yedroudj-Net

Yedroudj-Net [24] is a shallow network that converges fast. It can learn on small databases whilst obtaining good performances. It was published in 2018 to become, in a very short period of time, one of the most important CNNs for spatial steganalysis. We report in Figure 3 the global architecture of this network. We can observe that it is built in three parts, called modules: the pre-processing, the convolution and the classification module.

The pre-processing module includes one convolutional layer whose weights are initialized with high-pass filter kernels derived from the SRM linear filters [8]. Unlike the original paper, we have decided to add a truncation activation layer (TLU) on this module. This helps the network to converge even faster.

The convolutional module is composed of five convolutional blocks; each block starts with a convolutional layer and ends with an average pooling layer. In between, we find a batch normalization layer and an activation layer (block Type2). The average pooling layer is omitted from the first block to prevent premature signal loss (block Type1). The average pooling layer is replaced with a global average pooling in the last block (block Type3).

As for the classification module, it consists of three fully connected layers and a softmax activation function. The softmax function is used to normalize the two scores provided by the network between [0,1].

For more details on Yedroudj-Net, the reader can either check the original paper [24] or the online code at [2]. The pytorch version of yedroudj-net, which has been used in this paper, will soon be available at [3].

3.2 CovPool-Net

Global Covariance Pooling Network, referred to as CovPool-Net [6], is a deep network that is as efficient as SRNet [3], yet it requires no curriculum learning. Moreover, according to [6] authors, CovPool-Net converges faster than SRNet. CovPool-Net is a recent CNN that was published in July 2019. Thanks to its deep architecture design, this network has been able to achieve a good performance level and thus takes its place among the state-of-the-art spatial steganalysis models.

Similar to Yedroudj-Net, CovPool-Net is composed of a pre-processing module, a convolutional module and a classification

[2]online code: Caffe version
[3]online code: Pytorch version

module (see Figure 4). As for the pre-processing module, it is composed of two layers, a convolutional layer whose weights are initialized in the same way as in Yedroudj-Net, and a truncation activation layer (TLU) [22].

The CNN convolutional module is made up of 10 convolutional blocks. All of these ten blocks include a convolutional layer, a batch normalization layer and an activation layer. The average pooling layer is present only on blocks 4, 6 and 8, while block 10 includes a global covariance pooling which, according to [6] authors, improves the detection performance while preserving similar training speeds compared to a global average pooling. The blocks without an average pooling layer are noted Type1. Block Type2 refers to those with an average pooling layer. Last block, the one with the global covariance pooling layer, is noted as block Type3.

Unlike the Yedroudj-Net classification module, which is composed of three fully connected layers, the CovPool-Net classification module is composed of only one fully connected layer, which is followed by a softmax function that produces the probability distribution on the two-class labels.

4 EXPERIMENTAL METHODOLOGY

This section describes the common core of all the experiences reported in section 5.

4.1 Datasets

In this paper, to assess the reliability of our approach, we conduct various experiments on different databases. In the following paragraphs, we list those databases with a brief description.

4.1.1 the BOSS base: Break Our Steganographic System [1]. This database was made in 2011 for a steganalysis competition. This database has become the most known and used database for steganalysis, but also for steganography. It consists of 10,000 grayscale images of 512×512 pixels, uncompressed, and coming from 7 different cameras,

4.1.2 the BOWS2 base: Break Our Watermarking System [2]. This database was originally created for a watermarking competition held in 2008, but since 2017 it has been widely used as a complementary database to the BOSS base. In particular, it is used to augment the provided samples' number when training on the Boss base. This database consists of 10,000 grayscale images of 512×512 pixels, uncompressed, and whose distribution is close to that of BOSS base,

4.1.3 ALASKA-10K base: The ALASKA database was formerly created for the "in the wild" steganalysis competition [5]. The main objective of this competition was to bring steganalysis from research laboratories to real-life conditions.

Compared to BOSS and BOWS2, ALASKA is a large-scale database. It consists of 80,000 raw images from 51 different cameras (including smartphones and tablets...). This allows us to reflect better the wide media diversity that can be found "in the real world"; this diversity may be particularly fruitful for machine learning models. Moreover, a large database with many samples is important for training Deep Learning models dedicated to steganalysis, especially given that these models are getting deeper and deeper.

For the objectives of this paper, and in order to be able to fairly compare the results with those obtained on the BOSS base, we decided to randomly select 10,000 (the same number of images as on the BOSS database) of the 80,000 RAW images. Next, we applied the scripts provided with the ALASKA database to obtain 10,000 uncompressed gray-scale images in "tiff" format of 1024×1024 pixels. We referee to that base as ALASKA-10K.

Due to computational and time limitations, we conduct all the experiments on images of 256×256 pixels. To this end, we re-sampled all the images to 256×256 pixels, using the imresize() Matlab function with the default parameters (bi-cubic interpolation with anti-aliasing).

Note that the sub-sampling step supposes to make the pixels' value distribution of ALSAKA-10k database closer to that of BOSS'. However, by examining the steganalysis results in section 5, it appears that these two databases are statistically different. ALSAKA-10k is still a more "secure" database relative to BOSS, i.e. more difficult to steganalyze.

4.2 Software platform

In our experiments, we use Matlab for image manipulation and pre-processing tasks such as image re-sampling, data embedding, pixel manipulation, etc. For data embedding, two well-known content-adaptive spatial steganography algorithms with two payloads of 0.2 and 0.4 bit per pixel (bpp) are used, namely S-UNIWARD [3] and WOW [2].

Note that we use Matlab implementations (online codes[4]) with the simulator for embedding and a random key for each embedding. In this paper, both CNN's, i.e. Yedroudj-Net and CovPool-Net, are implemented using PyTorch [5], an open-source deep learning framework widely used by the scientific community. Also, note that the results presented in the original Yedroudj-Net paper were obtained using Caffe. This explains the slight variations in performance compared to those presented in this paper. All the experiments were run on an NVidia Titan X GPU card.

4.3 Descriptions of the different experimental scenarios

Below, we list all the scenarios that we have adopted in our tests with a brief description of how we create the training sets for each scenario. For a fair comparison, the test set remains the same on all scenarios.

Classical scenario. This represents the conventional scenario which is often used to obtain the baseline performance of a steganalysis model. Basically, this scenario consists of evaluating a steganalysis model on a given database. Therefore, we split the database of pairs (cover/stego) into three sets. 40% of the cover/stego pairs are reserved for the model's training, 10% are assigned to the validation set, while the remaining 50% are reserved for the test set. When referring to this scenario, we use the name of the used database, e.g. if we use Yedroudj-Net on ALASKA-10k, the results based on this scenario will be noted as ALASKA-10k.

[4]http://dde.binghamton.edu/download/
[5]https://pytorch.org/

Database enhancement scenario. In the field of steganalysis, the databases used are still relatively small. To overcome this limitation, researchers are studying database augmentation techniques and their impact on the overall performance of steganalysis models. One of these techniques is what we call **database enhancement**. In practical terms, **database enhancement** consists of providing a training set of the classical scenario with cover/stego pairs coming from a different database (a complementary database). To refer to this scenario, we put side-by-side the names of the original base and the complementary base. For example, BOSS+BOWS2 is when we enrich the BOSS training set with images from BOWS2.

Virtual augmentation scenario. This is the most commonly used technique for database enrichment, whether for machine learning or deep learning tasks. In steganalysis, VA consists of performing label-preserving flips and rotations on both covers and stegos of the training set, thus increasing the size of the learning set by a factor of 8. In this paper, we refer to this scenario as **Database name_VA**. If we take the previous example BOSS+BOWS2, when applying VA, this scenario will be noted "BOSS+BOWS2_VA". Note that the test set of this configuration remains unchanged (no virtual augmentation).

Pixels-off scenario. This scenario is similar to that of database enhancement, but rather than enriching the training set with cover/stego pairs from a complementary database, we use the *pixels-off* technique outlined in Algorithm 1 and discussed in section 2.2. First, we generate a pixels-off version(s) of the training set's covers. Once the covers are generated, we proceed to the generation of their corresponding stegos. These new resulting base of cover/stego pairs are then added to the training set. This increases the size of the learning set by a factor of 2. Note that it is possible to further increase the size of the training set by changing the P parameter each time (see Algorithm 1) and thus generating more pixels-off versions. To refer to this scenario, we use the following notation “Database name+P_1_P_2 .._P_n-off”, where $P_{i\in\{1,..,n\}}$ is the number of pixels that are switched to zero, and n is the number of duplicated versions. For example, "BOSS+100_400-off" refers to the scenario in which we augment the Boss training set with two **pixels-off** versions, $n = 2$ with $P_1 = 100$, $P_2 = 400$.

5 EXPERIMENTAL RESULTS

In the following section, we present the obtained results for different setups. Note that the maximum number of epoch is fixed to 200 (resp. 400) epochs for CovPool-Net (resp.Yedroudj.Net). Nevertheless, the two networks reach convergence before attaining this number. Therefore, to give a rough estimate, we provide convergence times for each setup.

5.1 Setup 1: No pixels-off, Yedroudj-Net, BOSS base

In Table 1, we report the error probability obtained using Yedroudj-Net on WOW and S-UNIWARD at 0.2 bpp and 0.4 bpp. Different scenarios are tested, the classical scenario, virtual augmentation scenario and the database enhancement scenario, plus the combination of database enhancement and virtual augmentation scenarios.

Table 1: Steganalysis error probability of Yedroudj-Net on BOSS base with two embedding algorithms WOW and S-UNIWARD at 0.2 bpp and 0.4 bpp under different scenarios

	WOW		SUNIWARD		examples	conv
	0.2bpp	0.4bpp	0.2bpp	0.4bpp	(pairs)	time
BOSS	27.71	15.27	35.42	22.7	4,000	4-5h
BOSS_VA	24.2	13.01	34.2	18.4	32,000	4-5h
BOSS+BOWS2	23.12	12.84	33.06	17.8	14,000	14-15h
BOSS+BOWS2_VA	20.85	10.13	29.12	16.37	112,000	14-15h

Regardless of payload size and embedding algorithms, this virtual augmentation scenario (BOSS_VA) provides superior results to the classical scenario (BOSS), with a 2-3% performance improvement. Note that the training time (to reach convergence) of Yedroudj-Net under the virtual augmentation scenario is the same as under the classical scenario; this is due to the fact that we use **online virtual augmentation**. Thus, in theory, more images are available for network training (32,000 pairs), but in practice, for each epoch, the network will see as many images as in the classical scenario (4,000 pairs). In other words, for each epoch, we randomly select 4,000 pairs from the 3,2000 pairs.

As for the database enhancement scenario, noted BOSS+BOWS, the CNN's performance is even better. As we can see, the steganalyzer efficiency has increased by 2 to 5%. However, more time is needed to train (14-15h versus 4-5h in the VA scenario). This is because, in contrast to the VA scenario, the number of images provided to the network at each epoch in the database enhancement scenario increases. The results are even better when we apply both augmentation techniques (database enhancement and virtual augmentation); the error probability is reduced by 5-7% compared to the classical scenario.

5.2 Setup 2: With pixels-off, Yedroudj-Net, BOSS base

In order to study the importance of the proposed technique, several tests have been carried out. First, as a reference test, we tested the classical scenario noted **B** (**BOSS**). Then, we tested the "pixels-off" scenario. For that, four pixels-off versions of the BOSS (B) training set were generated (100-off, 256-off, 400-off, 1024-off). These generated versions were then added, gradually, to the classic scenario training set **B (BOSS)**; the training set size is thus increased by 4,000 each time. Accordingly, the final training set ($\mathbf{B}_4$) contains 20,000 cover/stego pairs.

The results are presented in Table 2. For the experiment with the training set noted $\mathbf{B}_1$ (**B+100-off**), when the initial learning set is increased using the 100-off version, the Yedroudj-Net error probability is decreased -on average- by 2%. The performance is further improved when we increase the initial training set twice by using both the 100-off and the 256-off versions ($\mathbf{B}_2$). The detection

Table 2: Steganalysis error probability of Yedroudj-Net on BOSS base with two embedding algorithms WOW and S-UNIWARD at 0.2 bpp and 0.4 bpp for pixels-off scenario

	WOW		SUNIWARD		examples	conv
	0.2bpp	0.4bpp	0.2bpp	0.4bpp	(pairs)	time
B = BOSS	27.71	15.27	35.42	22.70	4,000	4-5h
$\mathbf{B}_1$ = **B**+100-off	25.31	14.3	33.1	19.4	8,000	9-10h
$\mathbf{B}_2$ = $\mathbf{B}_1$+256-off	23.95	13.41	29.8	17.8	12,000	13-14h
$\mathbf{B}_3$ = $\mathbf{B}_2$+400-off	23.5	13.44	29.3	16.95	16,000	17-18
$\mathbf{B}_4$ = $\mathbf{B}_3$+1024-off	23.8	13.65	29.2	16.98	20,000	21-22
$\mathbf{B}_5$ = $\mathbf{B}_1$_VA	21.5	12.4	31.4	14.8	64,000	9-10h

error probability is reduced by 2-4% for WOW (resp. 5-6% for S-UNIWARD) in comparison to the classical scenario. However, more time is needed to train (∼ 14h).

The $\mathbf{B}_3$ training set slightly reduces the steganalysis error probability, for both payloads, compared to the $\mathbf{B}_2$ training set (0,4% for WOW at 0.2bpp, and 0.8% for S-UNIWARD at 0.4bpp). Furthermore, no significant improvement is observed from $\mathbf{B}_3$ to $\mathbf{B}_4$.

Table 3: Steganalysis error probability of Yedroudj-Net on BOSS base with two embedding algorithms WOW and S-UNIWARD at 0.4 bpp with different numbers *P* of pixels-off.

	WOW 0.4 bpp	SUNIWARD 0.4 bpp
BOSS	15.27	22.70
BOSS+100-off	14.3	19.4
BOSS+256-off	14.6	19.3
BOSS+400-off	13.9	18.6
BOSS+1024-off	14.8	19.9

As shown in Table 3, whatever the value of the variable P used for the enrichment (100-off, 256-off, 400-off, or 1024-off) the steganalysis performance is improved, with $P = 400$ (400-off) gives the best results. When grouping all those enrichment, as observed in Table 2, with $\mathbf{B}_4$, the results are even better. Thus, the optimal parameters is roughly around $P = 400$ pixels-off, but combining various enrichments with P between 100 and 1024 allows us to improve even more the steganalysis efficiency. Note that this behavior is also observed in the other experiments detailed hereafter.

In the case of $\mathbf{B}_1$**_VA** (Table 2) where the enrichment is done by applying, in addition to the pixels-off technique, the virtual data augmentation, a significant improvement is observed. The detection error probability is reduced by 3-6% for WOW (and 4-5% for S-UNIWARD) compared to the classical scenario. These results show that virtual data augmentation can be freely used in parallel with pixels-off techniques. This conclusion is very interesting as it shows that the pixels-off technique can offer a good and straightforward solution to further improve the performance of CNN steganalysis, while continuing to apply classical enrichment methods.

5.3 Setup 3: With pixels-off, Yedroudj-Net, ALASKA base

To investigate the usability of the pixels-off technique to different databases, we conducted similar tests to those presented in Table 2, except that we used a different database, ALASKA-10k. The results are presented in Table 4. For the pixels-off scenario using $\mathbf{A}_1$ (ALASKA+100-off) training set, the error probability is reduced by an average of 1% in comparison with the classical scenario **A** (**ALASKA**). The performance keeps improving as we further enrich the training set ($\mathbf{A}_2$). Better results are obtained for $\mathbf{A}_3$, as we achieve an improvement in performance of 2-4% on WOW, and 2-3% on S-UNIWARD with respect to the no-enrichment scenario (ALASKA).

The trend for scenario $\mathbf{A}_4$ is similar to those obtained with BOSS base with scenario B4 (Table 2). The results between $\mathbf{A}_3$ and $\mathbf{A}_4$ are comparable or even a little bit improved with $\mathbf{A}_4$. Again, combining various enrichments by varying the parameter P between 100 and 1024 improves the performances. From previous results, we can conclude that the pixels-off technique improves the performance of the model, regardless of the database.

Table 4: Steganalysis error probability of Yedroudj-Net on ALASKA-10K base with two embedding algorithms WOW and S-UNIWARD at 0.2 bpp and 0.4 bpp for pixels-off scenario

	WOW		SUNIWARD		examples	conv
	0.2bpp	0.4bpp	0.2bpp	0.4bpp	(pairs)	time
A = Alaska	40.11	30.4	42.85	32.45	4,000	4-5h
$\mathbf{A}_1$ = **A**+100-off	38.78	28.92	41.87	31.69	8,000	9-10h
$\mathbf{A}_2$ = $\mathbf{A}_1$+256-off	38.15	28.09	40.55	30.24	12,000	13-14h
$\mathbf{A}_3$ = $\mathbf{A}_2$+400-off	38.05	27.91	40.05	29.54	16,000	17-18h
$\mathbf{A}_4$ = $\mathbf{A}_3$+1024-off	37.5	28.02	40.01	28.94	20,000	21-22h

Before moving to the next section, where we will examine the performance of the pixels-off technique using another state-of-the-art steganalyzer CovPool-Net, we would like to point out two observations in agreement with those completed in [9] and [18].

Firstly, we can notice that the intrinsic security of ALASKA-10k database is better than that of BOSS database. As an example, the accuracy of S-UNIWARD 0.4 bpp, steganalyzed with Yedroudj-Net, is 32% on ALASKA (A), whereas it is 22% on BOSS (B). This can be explained by the fact that ALASKA-10k is made up of images that come from many different camera models, and that the raw images have been developed with a various number of developments, among them some are producing a high level of mismatch [9].

Secondly, the superiority, in terms of security, of S-UNIWARD versus WOW, when evaluated on BOSS, is less apparent when the evaluation is done on ALASKA-10K. At 0.4 bpp, S-UNIWARD is 7% more secure than WOW on BOSS, whereas the gap falls to 2% on ALASKA-10K. As reported in [18], most of the adaptive algorithms have been tuned on the BOSS base, and the internal parameters

Table 5: Steganalysis error probability of CovPool-Net on BOSS base with two embedding algorithms WOW and S-UNIWARD at 0.2 bpp and 0.4 bpp for pixels-off scenario

	WOW		SUNIWARD		examples	conv
	0.2bpp	0.4bpp	0.2bpp	0.4bpp	(pairs)	time
B = BOSS	26.08	15.60	31.89	18.32	4,000	5-6h
$\mathbf{B}_1$ = **B**+100-off	25.33	14.63	28.54	16.25	8,000	10-11h
$\mathbf{B}_2$ = $\mathbf{B}_1$+256-off	24.88	13.11	26.61	15.00	12,000	14-15h
$\mathbf{B}_3$ = $\mathbf{B}_2$+400-off	23.34	13.02	26.64	15.44	16,000	19-20h
$\mathbf{B}_4$ = $\mathbf{B}_1$_VA	17.5	9.23	21.58	10.54	64,000	10-11h

are not optimized for other databases. This can explain the security gap reduction between WOW and S-UNIWARD.

5.4 Setup 4: With pixels-off, CovPool-Net, BOSS

In order to analyse the performance of the pixels-off technique when using different steganalyzer, additional tests were carried out.

In this section, we present the obtained results when pixels-off enrichment technique is used with another steganalysis model, CovPool-Net. For embedding, we use WOW and S-UNIWARD with BOSS base. As in section 5.2, we start with the classical scenario (no enrichment) and then gradually extend the training set size by applying the pixels-off algorithm, the results are presented in Table 5. For $\mathbf{B}_1$, we see an improvement of 2-3% for S-UNIWARD and less than 1% for WOW compared to the classical scenario. Better results are achieved for $\mathbf{B}_2$ (3-5% for S-UNIWARD and 1-2% for WOW)

For $\mathbf{B}_3$, the pixels-off algorithm increases the performance by 3-2% for S-UNIWARD and 2% for WOW. Like on Yedroudj-Net, CovPool-Net shows a considerable improvement when we combine both pixels-off and virtual augmentation ($\mathbf{B}_4$). The error rate can be reduced by 10-7% for S-UNIWARD (resp. 8-6% for WOW) compared to classical scenario (BOSS). Note that CovPool-Net takes one to two hours longer for training compared to Yedroudj-Net.

6 DISCUSSION AND ADVANCED ANALYSIS

In an attempt to better understand the underlying properties resulting from the use of the pixels-off technique, we first evaluate two other noises and compare their performance to the pixels-off enrichment. Second, we detail the class of problems addressed by the pixels-off technique.

6.1 Discussion on the noise power

We evaluate the impact of two classical additive noises when used for database enrichment. The first noise is the intensity spikes noise, known as the salt and pepper noise. We set the density to the default value, d = 0.05. The second noise is the Gaussian one. We use it with a zero mean and a variance of 0.01. We use BOSS base with WOW and S-UNIWARD at a payload of 0.4 bpp. For the steganalysis we use Yedroudj-Net.

The results are presented in Table 6. Note that for a fair comparison, the tests are always performed on the same test set. Also, note that the training sets have the exact same number of cover/stego pairs. One can observe that the results are worse when we perform a database enrichment using Gaussian noise (BOSS+Gaussian), compared to the *no-enrichment* scenario. An increase of 1% in detection error probabilities is reported for both WOW and S-UNIWARD. These results suggest that the addition of high-power noise, such as Gaussian noise, can lead to an important modification of the initial statistical models of both cover and stego contents. Thus, it provokes a mismatch between the training and the test set and so causes a drop in performance.

Table 6: Steganalysis error probability of Yedroudj-Net on BOSS base with two embedding algorithms WOW and S-UNIWARD at 0.4 bpp for different types of noise enrichment

	WOW0.4	SUNIWARD0.4
BOSS	15.27	22.70
BOSS+100-off	14.3	19.4
BOSS+Gaussian	16.08	23.25
BOSS+salt&pepper (d = 0.05)	15.16	22.25
BOSS+salt&pepper (d = 0.0016)	14.76	19.92

For enrichment with the salt and pepper noise and a density d = 0.05, the results are similar to those of the classic scenario (BOSS), and significantly inferior to those of the pixels-off scenarios (BOSS+100-off), regardless of the embedding algorithms used. The density d = 0.05 stands for the default density value (in Matlab) and the value commonly used in machine learning for data enrichment, however for steganalysis this value may be too high. Indeed, for an image of 256×256 pixels, with d= 0.05, about 3,300 pixels are affected by the salt and pepper noise, that is three times more modifications than when the maximum value of the P parameter is used (in pixel-off technique P_{max} = 1024 pixels).

It appears that only low-power noise affecting a very small percentage of pixels (less than 1.5%) can be useful for data-enrichment. To investigate this assumption, we test the salt and pepper noise but this time with a lower noise density, d=0.0016. Therefore, for an image of 256×256 pixels, about 100 pixels will be affected by the salt and pepper noise. In this case, the performance is significantly improved compared to the "BOSS+salt&pepper (d = 0.05)" scenario, while it is quite similar to that of the the "BOSS+100-off" scenario. This is because the salt and pepper noise, with d=0.0016, is inherently close to the pixels-off noise, as it randomly scatters white or black pixels in the image.

In the past, the steganalysis community did not obtain successful results with salt and pepper noise, this is mainly due to the use of the same density value used in the machine learning literature, which was actually too high for steganalysis' purpose. However, this type of noise gives satisfactory results when the lower density value is used. Finally, note that we have also observed that some other weak noises can be used fruitfully. Instead of switching off 100 pixels (setting their values to 0), we tested adding a noise of "+/-1" to 100 pixels randomly picked. We obtained an improvement, almost equal to that of "BOSS+salt&pepper (d = 0.0016)".

6.2 Class of problems addressed by the pixels-off technique

A signal like the pixels-off noise slightly changes the statistical properties of both cover and stego images [13]. As a consequence, the enrichment of a database with pixels-off noisy images should introduce a mismatch between the learning and test set, and thus imply a drop off in steganalysis performance. Nevertheless, the results that we reported previously, show that this type of enrichment improves CNN steganalysis accuracy.

What can appear as a counter-intuitive behaviour is, in fact, specific to deep-learning based steganalysis. Indeed, feature-based steganalysis, for its part, is badly impacted by the pixel-off noise. This was checked by evaluating the impact of a pixels-off-100 enrichment on the BOSS base with WOW at 0.4 bpp using an Ensemble Classifier [14] with Rich Model SRM [8]. Instead of improving the accuracy, the error probability increases from 25.5% to 28.3%. This confirms that the pixels-off technique does not produce images sharing the exact same source-model, and/or does not produce stegos that comes from the exact same stego-source-model. The pixels-off noise is breaking a sufficient number of pixel dependencies (which are due to the development processing) to results in an image realization that cannot come from the source that produced the initial database.

One should conclude that pixels-off enrichment incites the learning to be more sensitive to areas around switched-off pixels since these areas are now associated with strong embedding probabilities. Indeed, an area with a pixel switched off is creating a local abrupt signal change, for the covers and the stegos, and we think that it encourages the convolutional kernels to learn local weights more sensitive to those very singular areas. Knowing that modern adaptive embedding algorithms are mainly modifying the pixels from the textured areas, pixels-off may also encourage the network to focus more attention on these areas.

In order to challenge this rationale, we have forced the pixels that are switched-off to occur in the pixels of higher embedding probability. We conducted two additional experiments to check if the enrichment of the database can be guided by the adaptability of the embedding algorithm. In order to do so, we modify the pixels-off Algorithm 1 such that pixels switched-off are chosen by taking into account the Side Channel Information, which is in our case the embedding modification probabilities map. More precisely, instead of randomly switching off 100 pixels of the whole image, we switch off the same number of pixels while considering only the 10% pixels with the highest embedding probability (experiment noted BOSS+100_off-highP), then we test while considering only the 10% pixels with the lowest embedding probability (experiment noted BOSS+100_off-lowP). These results are presented in Table 7.

We can observe that the detection accuracy is reduced by 1% for the Boss+100_off-lowP scenario compared to the classical pixels-off scenario BOSS+100_off . On the other hand, the results are improved for the +100_off-highP scenario compared to the Boss+100_off scenario.

These results are interesting because they could lead to another way of doing SCA steganalysis, by generating SCA training sets. The usual approaches, such as Ye-Net [22], SRNet [3], are using the probability map as an input of the network. On the contrary, with the *adaptive* pixels-off method, the side information is only used to enrich the database. In order to better understand how noises, like the pixels-off one, could be used as another way to do SCA learning, additional experiments can be carried out, such as selecting the pixels-off by drawing them from a scaled version of the probability map, or compare the results with a CNN-based informed steganalysis. We postpone the numerous additional experiments for future works.

Table 7: Steganalysis error probability of Yedroudj-Net on BOSS base with two embedding algorithms WOW and S-UNIWARD at 0.4 bpp for different scenarios of pixels-off

	WOW0.4	SUNIWARD0.4
BOSS	15.27	22.70
BOSS+100_off	14.3	19.4
BOSS+100_off-lowP	15.17	20.85
BOSS+100_off-highP	13.65	18.15

7 CONCLUSION

Today, CNN-based steganalyzer architectures are getting deeper and deeper. Therefore, larger databases are needed in order to reach the region of good performance (irreducible error region). However, having a larger database in the steganalysis field can be a challenging option. This is where database enrichment techniques can help. This consists in increasing the number of images of the initial training set. Among existing techniques, we find the virtual data augmentation technique and the database enhancement technique.

In this paper, we propose and explore a novel technique for database enrichment for CNN-based steganalyzer. We call this enrichment technique "pixels-off". This technique is similar, in concept, to a noise addition, but it is performed in such a manner that it ensures that the pixel distribution of the resulting image remains as close as possible to the original one. Thus it avoids triggering a Cover Source Mismatch phenomenon (CSM).

The proposed technique is simple to implement, with low complexity and, most importantly, it improves the detection accuracy of the CNN-based steganalyzer. In addition, our technique is compliant with other techniques, so there is no restriction when it is performed in conjunction with other techniques such as virtual augmentation. Furthermore, the combination of the pixels-off technique with other data enrichment techniques leads to an even better performance.

We expect this paper to lead to fruitful research avenues that we plan to pursue. In future work, it would be interesting to study thoroughly how data enrichment impacts CNN-based steganalyzer performances. For this, we could examin the confusion matrix to understand the origin of the performance improvements. We could examine the learned filters of CNN, before and after data enrichment. We shall also study other techniques, and evaluate their scope for improving CNN-based SCA or not-SCA steganalysis performance.

ACKNOWLEDGMENTS

The authors would like to thank the University of Montpellier (LIRMM) and the University of Nîmes. We would like to thank the French Direction Générale de l'Armement (DGA) for its support through the Alaska project ANR (ANR-18-ASTR-0009), and the French ANR DEFALS program (ANR-16- DEFA-0003). We would also like to thank the Algerian Ministry of Higher Education / Scientific Research, for its scholarship support.

REFERENCES

[1] Patrick Bas, Tomas Filler, and Tomas Pevný. 2011. 'Break Our Steganographic System': The Ins and Outs of Organizing BOSS. In *Proceedings of 13th International Conference on Information Hiding, IH'2011 (Lecture Notes in Computer Science, Springer)*, Vol. 6958. Prague, Czech Republic, 59–70.

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

[3] Mehdi Boroumand, Mo Chen, and Jessica Fridrich. 2019. Deep Residual Network for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security* 14, 5 (May 2019), 1181 – 1193.

[4] Marc Chaumont. 2020. Deep Learning in steganography and steganalysis. In *Digital Media Steganography: Principles, Algorithms, Advances*, M. Hassaballah (Ed.). Vol. abs/1904.01444. Elsevier, Chapter 14, 46. http://arxiv.org/abs/1904.01444

[5] Remi Cogranne, Quentin Giboulot, and Patrick Bas. 2019. The ALASKA Steganalysis Challenge: A First Step Towards Steganalysis. In *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security* (Paris, France) *(IH&MMSec'2019)*. Paris, France, 125–137.

[6] Xiaoqing Deng, Bolin Chen, Weiqi Luo, and Da Luo. 2019. Fast and Effective Global Covariance Pooling Network for Image Steganalysis. In *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security* (Paris, France) *(IH&MMSec'2019)*. Paris, France, 230–234.

[7] Jessica Fridrich. 2009. *Steganography in Digital Media*. Cambridge University Press. Cambridge Books Online.

[8] Jessica Fridrich and Jan Kodovsky. 2012. Rich Models for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security, TIFS* 7, 3 (June 2012), 868–882.

[9] Quentin Giboulot, Rémi Cogranne, Dirk Borghys, and Patrick Bas. 2020. Roots and Solutions of Cover-Source Mismatch in Image Steganalysis: a Comprehensive Study. *submitted to Elsevier Signal Processing: Image Communication* (2020).

[10] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. 2017. Deep Learning Scaling is Predictable, Empirically. In *Unpublished - ArXiv*, Vol. abs/1712.00409. arXiv:1712.00409 http://arxiv.org/abs/1712.00409

[11] Vojtech Holub and Jessica Fridrich. 2012. Designing Steganographic Distortion Using Directional Filters. In *Proceedings of the IEEE International Workshop on Information Forensics and Security, WIFS'2012*. Tenerife, Spain, 234–239.

[12] Vojtech Holub, Jessica Fridrich, and Tomas Denemark. 2014. Universal Distortion Function for Steganography in an Arbitrary Domain. *EURASIP Journal on Information Security, JIS* 2014, 1, Article 1 (2014).

[13] Jason Jo and Yoshua Bengio. 2017. Measuring the tendency of CNNs to Learn Surface Statistical Regularities. *CoRR* abs/1711.11561 (2017). arXiv:1711.11561 http://arxiv.org/abs/1711.11561

[14] Jan Kodovský, Jessica Fridrich, and Vojtech Holub. 2012. Ensemble Classifiers for Steganalysis of Digital Media. *IEEE Transactions on Information Forensics and Security* 7, 2 (2012), 432–444.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceeding of Advances in Neural Information Processing Systems 25, NIPS'2012*. Curran Associates, Inc., Lake Tahoe, Nevada, USA, 1097–1105.

[16] Bin Li, Weihang Wei, Anselmo Ferreira, and Shunquan Tan. 2018. ReST-Net: Diverse Activation Modules and Parallel Subnets-Based CNN for Spatial Image Steganalysis. *IEEE Signal Processing Letters* 25, 5 (May 2018), 650–654.

[17] Yinlong Qian, Jing Dong, Wei Wang, and Tieniu Tan. 2016. Learning and Transferring Representations for Image Steganalysis Using Convolutional Neural Network. In *Proceedings of IEEE International Conference on Image Processing, ICIP'2016*. Phoenix, Arizona, 2752–2756.

[18] Vahid Sedighi, Jessica J. Fridrich, and Rémi Cogranne. 2016. 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* (San Francisco, California, USA). San Francisco, California, USA, 1–9.

[19] Gustavus J. Simmons. 1983. The Subliminal Channel and Digital Signatures. In *Proceeding of Crypto'83*. New York, Plenum Press, Santa Barbara, CA, 51–67.

[20] Shunquan Tan, Weilong Wu, Zilong Shao, Qiushi Li, Bin Li, and Jiwu Huang. 2020. CALPA-NET: Channel-pruning-assisted Deep Residual Network for Steganalysis of Digital Images, In under submission. *ArXiv* abs/1911.04657. arXiv:1911.04657 https://arxiv.org/abs/1911.04657

[21] Guanshuo Xu, Han-Zhou Wu, and Yun Q. Shi. 2016. Ensemble of CNNs for Steganalysis: An Empirical Study. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec'2016)*. Vigo, Galicia, Spain, 103–107.

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

[23] Mehdi Yedroudj, Marc Chaumont, and Frédéric Comby. 2018. 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). Burlingame, California, USA, 7.

[24] Mehdi Yedroudj, Frédéric Comby, and Marc Chaumont. 2018. 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). Calgary, Alberta, Canada, 2092–2096.

[25] Jishen Zeng, Shunquan Tan, Bin Li, and Jiwu Huang. 2018. Large-Scale JPEG Image Steganalysis Using Hybrid Deep-Learning Framework. *IEEE Transactions on Information Forensics and Security* 13, 5 (May 2018), 1200–1214.

[26] Ru Zhang, Feng Zhu, Jianyi Liu, and Gongshen Liu. 2020. 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*). *IEEE Transactions on Information Forensics and Security, TIFS* 15 (2020), 1138–1150.