Improving Blind Steganalysis in Spatial Domain Using a Criterion to Choose the Appropriate Steganalyzer Between CNN and SRM+EC

Jean-François Couchot, Raphaël Couturier, Michel Salomon

Improving Blind Steganalysis in Spatial Domain using a Criterion to Choose the Appropriate Steganalyzer between CNN and SRM+EC

Jean-Francois Couchot[1], Raphaël Couturier[1], and Michel Salomon[1]

FEMTO-ST Institute, CNRS - Univ. Bourgogne Franche-Comté (UBFC), France
{jean-francois.couchot,raphael.couturier,michel.salomon}@univ-fcomte.fr

Abstract. Conventional state-of-the-art image steganalysis approaches usually consist of a classifier trained with features provided by rich image models. As both features extraction and classification steps are perfectly embodied in the deep learning architecture called Convolutional Neural Network (CNN), different studies have tried to design a CNN-based steganalyzer. This work proposes a criterion to choose either the CNN designed by Xu et al. or the combination Spatial Rich Models (SRM) and Ensemble Classifier (EC) for an input image. Our approach is studied with three steganographic spatial domain algorithms: S-UNIWARD, MiPOD, and HILL, and exhibits detection capabilities better than each method alone. As SRM+EC and the CNN are only trained with MiPOD the proposed method can be seen as an approach for blind steganalysis.

Keywords: Steganalysis, Spatial Domain, CNN, SRM+EC

1 Introduction

During this past decade many steganographic algorithms have been proposed to hide a secret message inside a cover image. Such embedding schemes can operate in the spatial domain, like for example MiPOD [20], S-UNIWARD [7], HILL [15], WOW [8], HUGO [16], or STABYLO [3] but also in the frequency domain as J(PEG) counterpart of S-UNIWARD. When designing such an algorithm the objective is to provide an approach that changes the cover image as little as possible. The less the cover is modified, the less likely the stego image containing the message is to be detected and thus the more secure the steganographic scheme is. Obviously, assessing the security of steganographic tools has given rise to the dual challenge of detecting hidden information, also called steganalysis.

The wide majority of image steganalysis approaches are two-step. The first stage exhibits useful information on image content by computing a set of features and the second one uses them to train a machine learning tool to distinguish cover images from stego ones. For the first step, different Rich Models (RM) have been proposed for the spatial domain (SRM) [5] and the JPEG one [10], while for the second step the most common choice is Ensemble Classifier (EC) [12]. This combination RM+EC is used in many state-of-the-art image steganalysis

tools. As an illustration, in [5] stego images obtained with the steganographic algorithm HUGO have been detected with errors of 13% and 37%, respectively, for embedding payloads of 0.4 and 0.1 bpp. These errors were slightly reduced (12% and 36%) in [9] thanks to an improved rich model.

Deep learning [19, 14] has led to breakthrough improvements in various challenging tasks in computer vision, becoming the state-of-the-art for many of them. A key reason for this success is the current availability of powerful computing platforms, and more particularly GPU-accelerated ones. Among the different network architectures belonging to this family of machine learning methods, Convolutional Neural Networks (CNN) [13] are very efficient to solve image classification problems. As steganalysis is a similar problem, since the objective is to classify an input image as either a cover or a stego, the design of a CNN-based steganalyzer has received increasing attention for the past few years.

From an architecture point of view, a CNN is a feedforward network composed of two parts matching exactly the two steps used in conventional steganalysis. The first part, called the convolutional part, consists of one or several layers trained to extract feature maps becoming smaller with the layer depth. The second one is composed of some fully-connected layers trained simultaneously to perform the classification task. Hence, CNN does not only learn how to classify, but also how to automatically find a set of features giving a better representation of the input image thanks to 2D convolution kernels. A feature map is usually produced by a three-step process: a combination of filtered maps of the previous layer (or the input image for the first layer), a nonlinear processing by a neuron, and finally a size reduction through pooling (see [13] for more details).

The remainder of this paper proceeds as follows. Section 2 presents related works. The next section first recalls the CNN architecture designed by Xu *et al.* [23]. After an experimental study, we focus on why it sometimes fails to detect some stego images. Section 4 is devoted to our proposal: a criterion to choose the best suited method between CNN and SRM+EC. The paper ends with a section that summarizes the contributions and outlines suggestions for future work.

2 Related Works

2.1 Steganography

To be self-sufficient, this article recalls the key ideas of the three most secure steganographic tools, namely S-UNIWARD [7], MiPOD [20], and HILL [15]. For each of these algorithms, a distortion function ρ associates to each pixel the cost of modifying it. More formally, for a given cover X, let $\rho(X)$ be the matrix whose elements represent the cost of increasing or decreasing by 1 the corresponding pixels. By ranking pixels according to their value in $\rho(X)$, one can compute the set of pixels whose modification induces the smallest detectability. For instance the distortion function ρ_U of S-UNIWARD is defined by:

$$\rho_U(X) = \sum_{k=1}^{3} \frac{1}{|X \star K^k| + \sigma} \star |K^k|^{\frown}, \tag{1}$$

where $\star$ is a convolution mirror-padded product, $Y^{\curvearrowleft}$ is the result of a 180 degrees rotation of Y, K^k, $1 \leqslant k \leqslant 3$ are Daubechies-8 wavelet kernels in the three directions, and σ is a stabilizing constant. It should be noticed that the multiplicative inverse is element-wise applied. An element of $\rho_U(X)$ is small if and only if there are large variations of large cover wavelet coefficients in the three directions.

In MiPOD, the distortion function ρ_M is obtained thanks to a probabilistic approach. More precisely, let β be the matrix defined as the probabilities to increase by 1 the image pixels. The objective of such a scheme is then to find probabilities which minimize a *deflection coefficient*, $\Sigma\sigma^{-4}\beta^2$, where σ is the residual variance matrix of image pixels. Notice that the product is element-wise applied and the sum concerns all the elements of the matrix. Thanks to a Wiener filter and a Lagrangian method, β can be computed. Considering such pixel probabilities, the distortion cost ρ_M is defined by:

$$\rho_M(X) = \ln\left(\frac{1}{\beta} - 2\right). \tag{2}$$

Finally, the distortion function ρ_H of the HILL steganographic scheme is based on combinations of convolution products. However, contrary to the distortion function ρ_U of S-UNIWARD, this one combines a high-pass filter H_1 and two low-pass filters L_1 and L_2. More precisely, ρ_H is defined by:

$$\rho_H(X) = \frac{1}{|X \star H_1| \star L_1} \star L_2, \text{ where } H_1 = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix} \tag{3}$$

and L_1 (resp. L_2) is a 3×3 (resp. 5×5) mean matrix.

In all aforementioned schemes, the distortion function reflects the underlying image model. To summarize, ρ returns a large value in a easy-defined or smooth area, whereas in a textured or "chaotic" area, *i.e.*, with no model, it returns a small value.

2.2 CNN-based Steganalysis

The first attempt at designing a CNN-based steganalyzer for image steganalysis is due to Tan *et al.* [21]. Their proposal, a stacking of convolutional auto-encoders, yielded for HUGO a detection error more than twice as bad as the one given by SRM+EC: 31% compared to 14% for a payload of 0.4 bpp.

Qian *et al.* [18] have proposed for 256×256 input images a CNN consisting of a convolutional part of 5 layers producing at the end 256 features, which are then processed by a fully-connected part of two hidden layers and a final output one of two softmax neurons. The preliminary high-pass filtering is done using a 5×5 kernel, called F_0, similar to the 5×5 kernel predictor obtained in [6]. As noticed by Fridrich and Kodovský in [6], this kernel is inspired by a specific embedding algorithm, namely HUGO, but it worked well for the other steganographic algorithms they tested. The detection performance of this CNN was still

slightly lower than the state-of-the-art SRM+EC steganalyzer, but Pibre *et al.* [17] improved it thanks to a CNN with a different shape.

In comparison with the work of Pibre *et al.*, the CNN we designed in [2] being shallow, was quite different and calling into question some assumptions previously made. On the one hand, we proposed a convolutional part of two layers: a first layer reduced to a single 5×5 kernel trained to replace F_0, followed by a layer using large kernels (almost as large as the image size). On the other hand, the resulting set of 256 features (for an input image of 512×512 pixels) was so discriminating that the fully-connected network doing the classification task could be shortened to the two final softmax neurons. Unfortunately, our work, as well as the one of Pibre *et al.*, suffers from a crippling drawback: stego images were always obtained by using the same embedding key. The work by Qian *et al.* might suffer from the same drawback too.

More recently, the works [23] and [22] by Xu *et al.* have shown that CNN-based steganalysis remains competitive with conventional steganalysis. In [23] they first proposed a structural design of CNNs for steganalysis that is neither large, nor deep, and learns from noise residuals, since they considered as input image the one issued from high-pass filtering using the kernel F_0. The architecture of such convolutional networks, which is the basis of our work presented thereafter, will be described in detail in the next section. The experiments they completed have considered two spatial content-adaptive steganographic algorithms: S-UNIWARD and HILL. They have shown that the performance gained by an ensemble of five CNNs is comparable to the one of SRM+EC. In the following work [22], Xu *et al.* decided to study the merging of CNNs and ensemble classifier. The background idea is to train a second level classifier using information provided by CNNs. Furthermore, they also sligthly modified the architecture of the original CNN designed in [23]. This new CNN architecture has one more layer and changed pooling sizes in the previous ones. In addition to the ensemble method [23], called PROB, where EC will use the output of 16 CNNs instead of five, they defined two further ensemble methods. The first one, called PROB_POOL, is supposed to lower the loss of information induced by the pooling operation. Indeed, when the stride value is larger than one, some sampling operations are dropped. For a stride value $p > 1$, applying the pooling on a block of $p \times p$ pixels gives a single value, whereas for a stride of 1 the same block would have been replaced by $p \times p$ values. The idea is thus to also consider independently each remaining $p \times p - 1$ possible sampling. The second new ensemble method, called FEA, is simpler: it uses an architecture merging the convolutional part and the ensemble classifier. From the experiments done with these 6 ensemble scenarios (two possible sizes for the final vector of features and three methods), Xu *et al.* concluded that it might be interesting to replace the fully-connected part of the CNN by EC for image steganalysis.

Finally, we can notice the latest work [24] by Zeng *et al.* dealing with JPEG domain steganalysis. They propose to start by manually applying to the input image the first two phases of DCTR [10], namely a convolution followed by *Q*uantization & *T*runcation (*Q&T*). They use 25 residual images, where each is

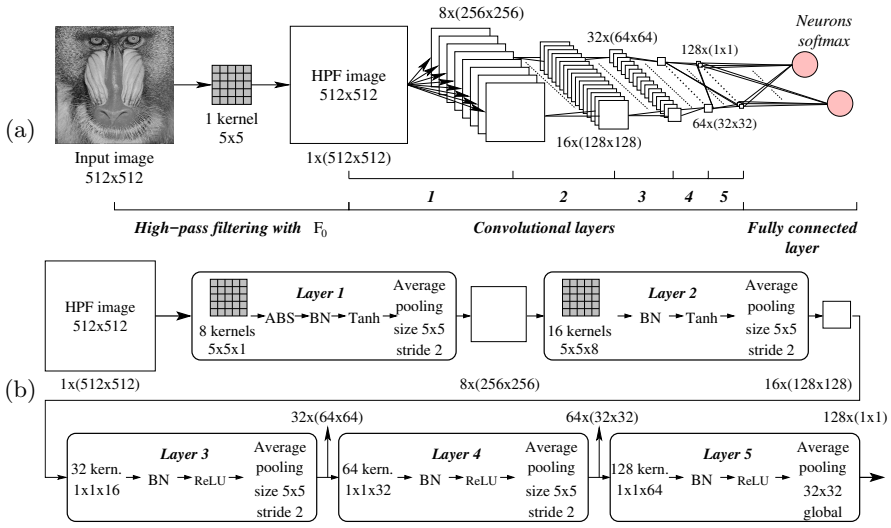


Fig. 1. CNN proposed by Xu *et al.* [23]: (a) overall architecture and (b) detailed view.

obtained by using a 5×5 DCT basis pattern, and three $Q\&T$ combinations. Then, for each group of residual maps for a given $Q\&T$ combination, a subnetwork corresponding to a simplified version of the convolutional part proposed by Xu *et al.* in [23] is trained to produce a feature vector of 512 components. To obtain the final prediction, the three vectors are concatenated and given as input to a three-layer fully connected network, which is trained together with the three subnets. Based on the experiments performed on more or less large databases of images issued from ImageNet, the authors claim that their proposal outperforms all other existing steganalysis approaches.

3 Convolutional Neural Networks for Image Steganalysis

3.1 The CNN Architecture Proposed by Xu *et al.* [23]

Like almost all the previous research works on CNNs for image steganalysis in the spatial domain, Xu *et al.* proposed an architecture that takes as input a high-pass filtered (HPF) version of the input image as shown in Fig. 1(a). Therefore, they used the kernel denoted by F_0 in [6, 18, 17] to highlight noise residuals. This filtering is obviously of great importance, since it provides the input information to the CNN, and thus must be suited to the classification task. The relevance of this kernel comes from its design for rich models. A classification part reduced to output neurons means that a linear classification is able to distinguish covers from stegos using the features produced by the final convolutional layer.

Starting with a HPF image of 512×512 pixels, the convolutional part results in 128 features, as shown in Fig. 1(b). Each of the four first layers successively halves the image size by generating feature maps using an average pooling, while

the fifth one replaces each feature map by a single value obtained through a global average pooling. Layers 1 and 2 learn 5×5 kernels, and the remaining layers 1×1 ones, the idea being to avoid an overfitting of the CNN to image content and/or stego noise. Layer 1 has also a specific function applied onto the outcome of the convolution, namely the absolute function (ABS), supposed to ensure that the model takes care of the symmetry in noise residuals like in rich models [6]. Batch normalization (BN) [11] is performed in every convolutional layer. A mixing of *Tanh* and $ReLU$ non-linear activation offered the best performance.

3.2 Detection Performance Evaluation of the CNN

To study and assess the performance of Xu *et al.* proposal, which was originally evaluated using a modified version of Caffe toolbox, the corresponding CNN has been implemented with the open source software library TensorFlow. The implementation is available on download from `GitHub`[1]. All the experiments are performed on a NVIDIA Tesla Titan X GPU, using as cover database the well-known BOSSBase [1]. Six stego images are associated to each cover image. They are obtained by embedding a message with S-UNIWARD, MiPOD, and HILL schemes considering two different payload values: 0.1 and 0.4 bpp. During a training execution a CNN is trained on a set of 5,000 cover-stego pairs and its detection performance assessed on a the remaining 5,000 pairs. Both training and testing sets are built by randomly picking pairs of images.

Notice that even if we implemented exactly a CNN according to the proposal, there is a major difference in comparison with the original work in the way the final prediction is obtained. In [23] Xu *et al.* generated from a training set five different non-overlapping 4,000/1,000 splits and each of them is used to train separately a CNN occurrence. The final prediction for a given test image is then obtained by averaging the five output probabilities.

Let us explain how the final prediction is computed with a set of T trained CNNs which are denoted as $\mathrm{CNN}_1, \mathrm{CNN}_2, \ldots, \mathrm{CNN}_T$. First of all, each CNN_i, $1 \leqslant i \leqslant T$, memorizes its L last versions provided by the L last training epochs obtained all along the program execution. These internal CNNs are denoted as $\mathrm{CNN}_i^1, \mathrm{CNN}_i^2, \ldots \mathrm{CNN}_i^L$. Each of these internal CNNs gives an answer, which is 0 if the tested image I is declared as cover and 1 otherwise. Finally, the average of all the values is computed, and a discrete answer is returned by each CNN depending on whether this average is greater or equal to 0.5 or not. This is formalized for each i, $1 \leqslant i \leqslant T$, by:

$$\mathrm{is\_stego}(I, \mathrm{CNN}_i) = \left\lfloor \frac{1}{L} \sum_{j=1}^{L} \mathrm{is\_stego}(I, \mathrm{CNN}_i^j) + 0.5 \right\rfloor . \tag{4}$$

The aggregation of these results must take into consideration the fact that an image I we want to classify is used in training step in some CNN_i or not. Let us consider the set $\mathcal{T}_I = \{i | 1 \leqslant i \leqslant T$ and I is used in testing step of $\mathrm{CNN}_i\}$ and

[1] https://github.com/rcouturier/steganalysis_with_CNN_and_SRM.git

T_I be its cardinality. The number T_I counts the number of times I is used as a testing image by some CNNs. The final answer is then the discrete answer of the average of all the CNNs that have used I as testing image. This is formalized by:

$$\text{is\_stego\_CNN}(I) = \left\lfloor \frac{1}{T_I} \sum_{i \in \mathcal{T}_I} \text{is\_stego}(I, \text{CNN}_i) + 0.5 \right\rfloor . \tag{5}$$

Indeed, as both training and testing sets are built by randomly picking images, the number of times an image I is in a test set varies.

Due to the huge computation cost we have only trained CNNs using MiPOD dataset and tested them directly on the S-UNIWARD and HILL ones. Hence, we can assess whether a CNN is competitive in a blind steganalysis approach.

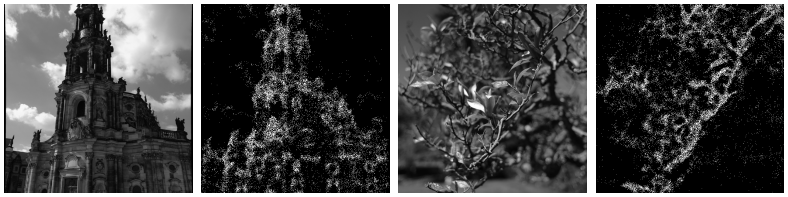
The key training parameters for reproducible experiments are discussed thereafter. First, a CNN is trained for a maximum number of training epochs $E_{\max}$ set to $1,000$ and 300, respectively, for embedding payloads of 0.1 and 0.4 bpp, without any overfitting control with a validation set. To compute the prediction given by a network CNN_i, $L = 20$ occurrences are used. Second, the network parameters are optimized by applying a mini-batch stochastic gradient descent, a typical choice in deep learning. We have used a mini-batch size of 64 samples. The gradient descent parameters are: a learning rate initialized to 0.001, but with no weight decay, and a momentum set to 0.9.

The obtained average detection errors are reported in Table 1. The first line labelled with "Caffe [23]" recalls values given in [23]. It should be noticed that "X" means that HILL has not been evaluated with the Caffe framework. The second line gives the average error rates from $T = 12$ independent training runs of the TensorFlow implementation for embedding payload of 0.4 bits per pixel. The third gives the average error rates for 200 runs with classical SRM+EC. In this latter context maxSRMd2 [4] has been used as a feature set. Finally, the last line gives the results obtained when the training stage is executed with images modified by MiPOD, whereas the testing stage is executed with images modified with another embedding scheme (SRM+EC and maxSRMd2 are still used).

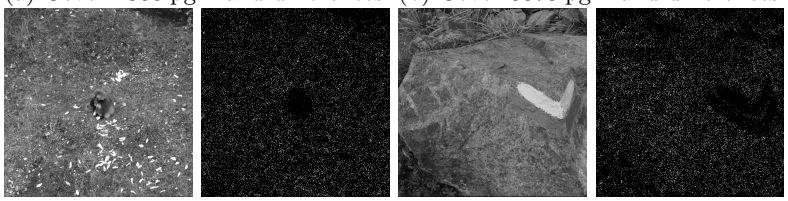
Table 1. Average detection error as a function of classifier (original Caffe by Xu *et al.*, our TensorFlow trained only with MiPOD at 0.4 bpp, and SRM+EC) and of payload.

	S-UNIWARD		MiPOD		HILL	
	0.1	0.4	0.1	0.4	0.1	0.4
Caffe [23]	42.67	19.76	X	X	41.56	20.76
TensorFlow	47.38	20.52	43.72	19.36	46.79	20.25
SRM + EC	39.84	18.06	41.18	21.42	42.96	23.31
SRM + EC (blind)	40.57	20.85	41.18	21.42	43.35	23.99

From the values given in this table we can draw several conclusions. First, despite the differences highlighted previously, for a payload of 0.4 bpp the TensorFlow implementation produces nearly the same performance for S-UNIWARD



(a) Cover 1388.pgm and differences. (b) Cover 8873.pgm and differences.



(c) Cover 1911.pgm and differences. (d) Cover 3394.pgm and differences.

Fig. 2. Examples of differences images between cover and corresponding stego when embedding is performed using MiPOD with a payload of 0.4 bpp.

and HILL than the original Caffe one, while for 0.1 bpp the performance is worse due to the blind context. Second, we observe that SRM+EC results in the best performances for S-UNIWARD in case of non-blind steganalysis. Third, for MiPOD the CNN approach is still competitive with SRM+EC. Fourth, the CNNs trained by only making use of the MiPOD dataset can provide a similar detection accuracy for S-UNIWARD and HILL, even if for the payload of 0.1 bpp a larger degradation of the accuracy can be noticed. Obviously, the lowest detection error is gained for the embedding scheme which has provided the training data. Fifth, CNNs outperform SRM+EC in blind steganalysis context with a payload of 0.4 bpp, which means that CNNs allow a better generalization to different steganographic algorithms.

3.3 Characterizing the mis-CNN-classified Images

Let us start with some illustrative examples of images describing the typical behavior of the CNN in the case of MiPOD with payload 0.4 bpp. Fig. 2 presents four case examples where for each we have the cover image and the corresponding differences between it and the stego one. From the images showing the differences we can distinguish two groups of images according to the pixels modified by the embedding process. It clearly appears that for both images shown on the upper line, 1388.pgm and 8873.pgm, MiPOD mainly modifies pixels corresponding to edges. For 1911.pgm and 3394.pgm, changes are scattered without obviously highlighting any underlying image edge. Consequently, since a CNN mainly learns to detect underlying edges, one can easily guess that the CNN-steganalyzer is able to detect both cover and stego for 1388.pgm and 8873.pgm, whereas it fails for the two other images. We are then left to provide a metric on images which reflects the difficulty to perform the CNN classification task.

Since the aforementioned steganographic schemes have their own distortion function ρ, we decided to study whether a metric can be deduced from it. Therefore, for each image I of the BossBase we have performed 200 classification procedures with SRM+EC (thanks to maxSRMd2) for the embedding algorithm MiPOD at payload 0.4 bpp. Fig. 3(a) presents the resulting scatter plot of $(\overline{\rho_U}(I), \overline{e_{SRM+EC}}(I))$ pairs and the curve linking the mean error of each class, whereas the bar displays its corresponding standard deviation. This figure is obtained for S-UNIWARD, while Fig. 3(c) and 3(e) are the corresponding ones for HILL and MiPOD. Similarly, Fig. 3(b), 3(d), and 3(f) show the scatter plots, curve, and error bars, for the CNN. In that case, $(\overline{\rho_U}(I), \overline{e_{CNN}}(I))$, $(\overline{\rho_H}(I), \overline{e_{CNN}}(I))$, and $(\overline{\rho_M}(I), \overline{e_{CNN}}(I))$ are the average testing error obtained after training 12 independent networks. This low number explains why in comparison with the SRM+EC steganalysis context the points are less vertically spread. The scalar $\overline{\rho_U}(I)$ is the mean of all the matrices $\rho_U(X)$ presented in equation (1), where U means S-UNIWARD. $\overline{\rho_M}(I)$ has a similar definition for MiPOD. Finally $\overline{\rho_H}(I)$ is not directly the mean of all the matrices $\rho_H(X)$ of HILL. Due to its definition (Eq. (3)), some extremely large values may result from an extremely small denominator and lead to a meaningless mean value. To avoid this behavior, extremely large values are excluded from the computation.

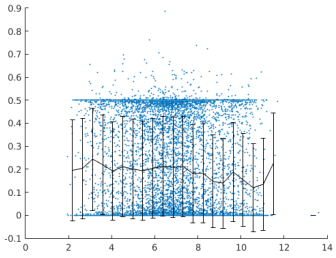
By focusing on Fig. 3(a), 3(c), and 3(e), it can be first deduced that the detection error of SRM+EC is quite independent of $\overline{\rho}$. Secondly, considering Fig. 3(b) and 3(d), we can deduce that the CNN testing error continuously decreases with respect to $\overline{\rho_U}(I)$ and with $\overline{\rho_H}(I)$. The good correlation between the prediction accuracy of the CNN for a given image I and the value of $\overline{\rho}(I)$ can be observed in the two former cases but not in the last one. The functions $\overline{\rho_U}$ and $\overline{\rho_H}$ are thus an indicator of the CNN accuracy. For instance, in Fig 2, for the misclassified images we obtain $\overline{\rho_U}(1911) = 2.1$ and $\overline{\rho_U}(3394) = 3.06$; on the other hand for the well detected images we get $\overline{\rho_U}(1388) = 7.05$ and $\overline{\rho_U}(8874) = 7.39$. Thus $\overline{\rho_U}$ and $\overline{\rho_H}$ enable to cluster the images in two groups which are in accordance with those noticed at the beginning of the section.

4 Taking the Best from CNN and SRM+EC Predictions

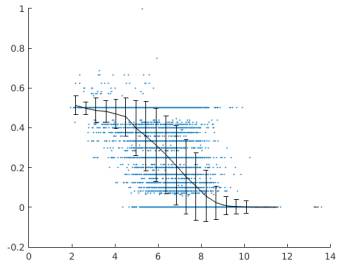
4.1 Choosing the Best Method for a Given Input Image

We have shown that the lower the distortion function mean $\overline{\rho_U}$ of an input image is, the more difficult it will be for the CNN to correctly detect whether the image is a cover or a stego. Conversely, SRM+EC gives rather regular detection errors, without showing too much sensitivity to $\overline{\rho_U}$, being robust against the image structure. A look at Fig. 3(a) and 3(b) shows that we can take advantage from these different behaviors to improve the detection performance on the BossBase.

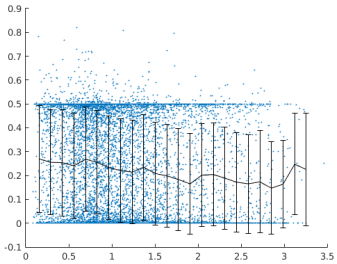
In fact, SRM+EC and the CNN can be combined due to complementary purposes. As can be seen in Fig. 4, from the largest $\overline{\rho_U}$ value up to the point where both curves intersect the CNN is the most competitive, whereas after, towards the lowest $\overline{\rho_U}$ value, it is SRM+EC which is the most accurate. Formally,



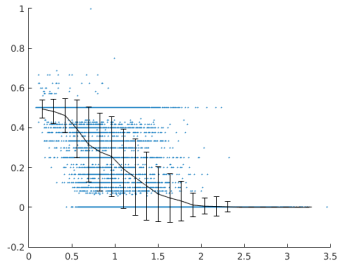
(a) Detection error w.r.t image $\overline{\rho_U}$ value for SRM+EC.



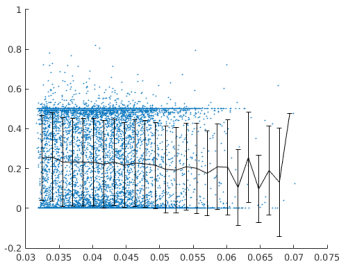
(b) Detection error w.r.t image $\overline{\rho_U}$ value for the CNN by *Xu et al.*



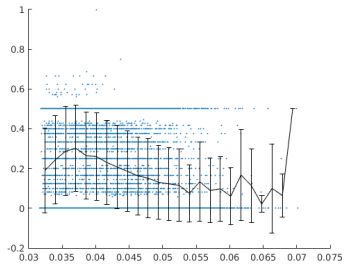
(c) Detection error w.r.t image $\overline{\rho_H}$ value for SRM+EC.



(d) Detection error w.r.t image $\overline{\rho_H}$ value for the CNN by *Xu et al.*



(e) Detection error w.r.t image $\overline{\rho_M}$ value for SRM+EC.



(f) Detection error w.r.t image $\overline{\rho_M}$ value for the CNN by *Xu et al.*

Fig. 3. Relation between testing errors and distortion function mean.

this can be expressed as follows for an input image I, once $\overline{\rho_U}(I)$ is computed:

$$\text{if } \overline{\rho_U}(I) < \overline{\rho_U^{\cap}} \text{ use SRM+EC prediction, otherwise use CNN prediction} \quad (6)$$

where $\overline{\rho_U^{\cap}}$ corresponds to the intersection abscissa. For Fig. 4, we have obtained $\overline{\rho_U^{\cap}} = 6.6$. Let us emphasize that the same approach can be applied to S-UNIWARD and HILL algorithms, leading to different values for $\overline{\rho_U^{\cap}}$.

Fig. 4. Average error of CNN and SRM+EC for MiPOD 0.4 bpp w.r.t $\overline{\rho_U}$.

Overall, the feature set generated by a spatial rich model is so large and diverse that it is able to give predictions yielding almost the same level of accuracy, regardless of the pixels modified by the embedding process. Moreover, the computing of the features is precisely defined. Conversely, the CNN learns to extract a set of features to fulfill its classification task according to the data given during the training step. Therefore, it will be well-suited to process images having the same kind of embedding than the main trend in the training set. In other words, images having low $\overline{\rho_U}$ values are so underrepresented in the BossBase that they have a limited influence during the training process.

4.2 Detection Performance Evaluation of the Proposal

Table 2 presents in its last column the average detection error obtained for the three steganographic algorithms. The first column gives the performance of SRM+EC computed on images I such that $\overline{\rho_U} < \overline{\rho_U^{\cap}}$, this last value is shown in the second column, while the third column shows the results gained from CNN for the remaining images. The proposal improves the detection performance for each embedding algorithm. For a payload of 0.4 bpp, S-UNIWARD has the lowest error rate with 14.82%, whereas for MiPOD and HILL we have values slightly below 17%. The lines labelled as non blind correspond to situations where SRM+EC was trained with the same algorithm than the one used to perform the embedding. Conversely, the lines denoted as blind mean that SRM+EC was trained with MiPOD and then used to detect S-UNIWARD or HILL. This also explains why for both blind and non blind situations the CNN gives the same error when both cases use the same value for $\overline{\rho_U^{\cap}}$. For the lower payload of 0.1 bpp, the improvements provided by our method are also clearly visible. These results are also somewhat surprising, since they are obtained by training only CNNs using images embedding hidden messages with MiPOD. This means that even if each steganographic algorithm has its own distortion function, there is certainly a high redundancy among the modifications they made on the same cover image.

Table 2. Average detection error according to $\overline{\rho_U^{\diamond}}$ for different embedding payloads.

	SRM+EC		$\overline{\rho_U^{\diamond}}$		CNN		CNN + SRM+EC	
Payload (bpp)	0.1	0.4	0.1	0.4	0.1	0.4	0.1	0.4
S-UNIWARD non blind	40.08	20.01	9.2	7.1	23.36	8.25	38.06	14.82
S-UNIWARD blind	41	22.05	9.2	6.9	23.36	9.5	38.88	15.87
MiPOD non blind	42.13	23.89	8	6.6	25.84	9.26	37.82	15.65
HILL non blind	43.48	24.51	8.9	6.6	21.88	9.78	40.24	16.22
HILL blind	44.30	25.41	8.3	6.6	27.72	9.78	40.64	16.61

A closer look on the performances of each steganalyzer on the subset of images it has to classify according to $\overline{\rho_U^{\diamond}}$ explains why our proposal is relevant. Indeed, in comparison with the performances shown in Table 1 we can remark that the SRM+EC error rate is slightly worse than on the whole dataset. Thus we take advantage from the low error rate of the CNN at a price of a slightly worse misclassification by SRM+EC. Another point to notice is the evolution in opposite directions of $\overline{\rho_U^{\diamond}}$ and payload values, which means that, as expected, the scatterness of the modified pixels increases and thus is more difficult to detect with the current CNN architecture. Nevertheless, our approach allows us to build a competitive blind steganalyzer, which gives lower detection errors than CNN based only or SRM+EC based only approaches.

5 Conclusion and Future Work

Over the past two years the design of deep learning based approaches for image steganalysis in spatial domain, using more particularly convolutional neural networks, has received an increasing attention due to their impressive successes on many classification tasks. Recently, Xu *et al.* have introduced a CNN architecture, which, to the best of our knowledge, is the most competitive one compared to rich models with ensemble classifier. In this paper, we have investigated when this CNN architecture fails in order to propose a method improving the detection performance on the BossBase for different spatial steganography algorithms.

Thanks to a TensorFlow implementation of the CNN, giving nearly the same detection performance than the original Caffe one for S-UNIWARD and HILL, we have found a metric strongly correlated with the CNN classification performance. This metric consists in the mean of all the elements in the cost matrix provided by the distortion function ρ of the considered steganographic algorithm for the input image. We have shown that the lower this latter value $\overline{\rho_U}$ for S-UNIWARD is, the more the CNN fails to correctly detect if the image is a cover or a stego. Fortunately, the CNN and SRM+EC detection errors evolve in different ways according to the metric function. By computing the intersection of

the corresponding curves we are then able to define a reliable criterion allowing to decide, for an input image, when to use the CNN or SRM+EC to obtain the most accurate prediction. The experiments done considering the steganographic algorithms S-UNIWARD, HILL, and MiPOD, have validated the proposed criterion, since it has always led to improved detection performance, regardless of the embedding payload value. Another contribution of this work is to have designed a steganalyzer insensitive to the embedding process (blind detection).

Our future work will focus on two aspects. First, it might be interesting to subdivide the BossBase in disjoint subsets according to the average distortion function value and to train several CNNs on them. However, to be able to train a CNN for low $\overline{\rho_U}$ values, the database should be expanded to include more images corresponding to this case. Second, CNNs dealing with spatial domain steganalysis work on a single high-pass filtered version of the input image. Therefore, we plan to replace the single filter by a filter bank, an approach which in the case of the JPEG domain steganalysis seems to be successful according to [24].

6 Acknowledgments

This article is partially funded by the Labex ACTION program (ANR-11-LABX-01-01 contract) and the Franche-Comté regional council. We would like to thank NVIDIA for hardware donation under CUDA Research Center 2014 and the Mésocentre de calcul de Franche-Comté for the use of the GPUs.

References

1. Bas, P., Filler, T., Pevný, T.: "Break Our Steganographic System": The Ins and Outs of Organizing BOSS, pp. 59–70. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
2. Couchot, J.F., Couturier, R., Guyeux, C., Salomon, M.: Steganalysis via a Convolutional Neural Network using Large Convolution Filters for Embedding Process with Same Stego Key. ArXiv e-prints (May 2016)
3. Couchot, J., Couturier, R., Guyeux, C.: STABYLO: steganography with adaptive, bbs, and binary embedding at low cost. Annales des Télécommunications 70(9-10), 441–449 (2015), http://dx.doi.org/10.1007/s12243-015-0466-7
4. Denemark, T., Sedighi, V., Holub, V., Cogranne, R., Fridrich, J.J.: Selection-channel-aware rich model for steganalysis of digital images. In: 2014 IEEE International Workshop on Information Forensics and Security, WIFS 2014, Atlanta, GA, USA, December 3-5, 2014. pp. 48–53. IEEE (2014), http://dx.doi.org/10.1109/WIFS.2014.7084302
5. Fridrich, J., Kodovsk, J.: Multivariate gaussian model for designing additive distortion for steganography. In: Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. pp. 2949–2953 (May 2013)
6. Fridrich, J.J., Kodovský, J.: Rich models for steganalysis of digital images. IEEE Trans. Information Forensics and Security 7(3), 868–882 (2012), http://dx.doi.org/10.1109/TIFS.2012.2190402

7. Holub, V., Fridrich, J., Denemark, T.: Universal distortion function for steganography in an arbitrary domain. EURASIP Journal on Information Security 2014(1), 1 (2014), http://dx.doi.org/10.1186/1687-417X-2014-1
8. Holub, V., Fridrich, J.J.: Designing steganographic distortion using directional filters. In: WIFS. pp. 234–239. IEEE (2012)
9. Holub, V., Fridrich, J.J.: Random projections of residuals for digital image steganalysis. IEEE Trans. Information Forensics and Security 8(12), 1996–2006 (2013), http://dx.doi.org/10.1109/TIFS.2013.2286682
10. Holub, V., Fridrich, J.J.: Low-complexity features for JPEG steganalysis using undecimated DCT. IEEE Trans. Information Forensics and Security 10(2), 219–228 (2015), http://dx.doi.org/10.1109/TIFS.2014.2364918
11. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
12. Kodovský, J., Fridrich, J.J., Holub, V.: Ensemble classifiers for steganalysis of digital media. IEEE Transactions on Information Forensics and Security 7(2), 432–444 (2012), http://dx.doi.org/10.1109/TIFS.2011.2175919
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
14. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature 521(7553), 436–444 (2015)
15. Li, B., Wang, M., Huang, J., Li, X.: A new cost function for spatial image steganography. In: 2014 IEEE International Conference on Image Processing (ICIP). pp. 4206–4210. IEEE (2014)
16. Pevný, T., Filler, T., Bas, P.: Using high-dimensional image models to perform highly undetectable steganography. In: Böhme, R., Fong, P.W.L., Safavi-Naini, R. (eds.) Information Hiding - 12th International Conference, IH 2010, Calgary, AB, Canada, June 28-30, 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6387, pp. 161–177. Springer (2010), http://dx.doi.org/10.1007/978-3-642-16435-4
17. Pibre, L., Jérôme, P., Ienco, D., Chaumont, M.: Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch. In: Media Watermarking, Security, and Forensics, EI: Electronic Imaging (2016)
18. Qian, Y., Dong, J., Wang, W., Tan, T.: Deep learning for steganalysis via convolutional neural networks. In: IS&T/SPIE Electronic Imaging. pp. 94090J–94090J. International Society for Optics and Photonics (2015)
19. Schmidhuber, J.: Deep learning in neural networks: An overview. Neural Networks 61, 85 – 117 (2015)
20. Sedighi, V., Cogranne, R., Fridrich, J.: Content-adaptive steganography by minimizing statistical detectability. IEEE Transactions on Information Forensics and Security 11(2), 221–234 (Feb 2016)
21. Tan, S., Li, B.: Stacked convolutional auto-encoders for steganalysis of digital images. In: Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA). pp. 1–4. IEEE (2014)
22. Xu, G., Wu, H.Z., Shi, Y.Q.: Ensemble of cnns for steganalysis: An empirical study. In: ACM Workshop on Information Hiding and Multimedia Security (2016)
23. Xu, G., Wu, H.Z., Shi, Y.Q.: Structural design of convolutional neural networks for steganalysis. IEEE Signal Processing Letters 23(5), 708–712 (2016)
24. Zeng, J., Tan, S., Li, B., Huang, J.: Large-scale JPEG steganalysis using hybrid deep-learning framework. ArXiv e-prints (Nov 2016)