Yedroudj-Net: An Efficient CNN for Spatial Steganalysis

Mehdi Yedroudj, Frédéric Comby, Marc Chaumont

PAPER ACCEPTED TO IEEE ICASSP'2018, CALGARY, ALBERTA, CANADA, 15-20 APRIL 2018.

YEDROUDJ-NET: AN EFFICIENT CNN FOR SPATIAL STEGANALYSIS

Mehdi YEDROUDJ (1), *Frédéric COMBY* (1), *Marc CHAUMONT* (1),(2)

(1) Montpellier University, LIRMM (UMR5506) / CNRS, FRANCE,
(2) Nîmes University, FRANCE.

ABSTRACT

For about 10 years, detecting the presence of a secret message hidden in an image was performed with an Ensemble Classifier trained with Rich features. In recent years, studies such as Xu *et al.* have indicated that well-designed convolutional Neural Networks (CNN) can achieve comparable performance to the two-step machine learning approaches.

In this paper, we propose a CNN that outperforms the state-of-the-art in terms of error probability. The proposition is in the continuity of what has been recently proposed and it is a clever fusion of important bricks used in various papers. Among the essential parts of the CNN, one can cite the use of a pre-processing filter-bank and a Truncation activation function, five convolutional layers with a Batch Normalization associated with a Scale Layer, as well as the use of a sufficiently sized fully connected section. An augmented database has also been used to improve the training of the CNN.

Our CNN was experimentally evaluated against S-UNIWARD and WOW embedding algorithms and its performances were compared with those of three other methods: an Ensemble Classifier plus a Rich Model, and two other CNN steganalyzers.

Index Terms— Steganalysis, Deep Learning, Convolutional Neural Network.

1. INTRODUCTION

The first attempt to use Deep Learning methods for steganalysis dates back to 2014 [1] with auto-encoders. One year later Qian *et al.* [2] and Pibre *et al.* [3] proposed to use Convolutional Neural Networks. In 2016, the first results, close to those of the state-of-the-art, were obtained with an ensemble of CNNs [4]. The Xu-Net[1] [5] CNN is used as *base learner* of the ensemble of CNNs. Other networks have been proposed in 2017, this time for JPEG steganalysis. In [6], authors proposed a pre-processing inspired by the Rich Models, and the use of a big learning database. The results were close to those of the state-of-the-art. In [7], the network is built with a *phase-split* inspired by the JPEG compression process. An ensemble of CNNs was required to obtain results that were slightly better than those of the state-of-the-art. In [8], a CNN inspired by ResNet [9] with the *shortcut connection* trick and 20 layers also improved the results in term of accuracy.

These results were highly encouraging but regarding the gain obtained in other image processing tasks using Deep Learning methods [10], the steganalysis results were not "10% better" compared to the classical approaches that use an Ensemble Classifier [11] with a Rich Model [12, 13] or a Rich Model with a Selection-Channel Awareness [14, 15]. In 2017, the main trends to improve CNN results are: using an ensemble of CNNs, modifying the topology by mimicking the Rich Models extraction process, or using ResNet. In most of the cases, the design or the experimental effort is very high for a very small improvement of the performance.

By looking back to the *good practices* in deep learning as well as the recent studies, we experimentally designed a CNN for spatial steganalysis whose efficiency is naturally better than the state-of-the-art. This is performed without resort to either a design specific to the nature of images (spatial, jpeg, ...) or a CNN ensemble (which is known to improve the results). We focused on the design of the CNN, avoiding the use of tricks known to improve the performances such as transfer learning [16] or virtual augmentation of the database [17], etc. Additionally, the proposed network is not sensitive to the initialization of hyperparameters and thus easily converges, which will be later discussed in Section 3. We named this network the "Yedroudj-Net" CNN, and will compare it with Xu-Net [5], Ye-Net [17] and also with the Ensemble Classifier [11] fed with the Spatial-Rich-Models [12] for spatial steganalysis.

2. YEDROUDJ-NET

Fig. 1 illustrates the overall architecture of our CNN. The network is composed of a *pre-processing block*, five *convolutional blocks*, and a *fully connected block* made of three fully connected layers followed by a *softmax*. The network produces a probability distribution over the two class labels.

The *pre-processing block* filters the input cover/stego image with a predefined high-pass filter in order to extract the noise component residuals. The pre-processed image then feeds the network. Previous studies [2, 3] observed that without this preliminary high-pass filter the CNN converges more slowly. This pre-processing largely suppresses the image content, narrows the dynamic range, and thus increases the signal-to-noise ratio between the weak stego signal (if present) and the image signal. As a result, the CNN can learn on a more compact and *robust* signal.

Inspired by the benefit of *diversity* [12], and similarly to [17], we use the 30-basic high-pass filters from SRM [12], instead of using only one filter such as [2, 3, 5], in order to pre-process the input image. Note that the filters kernel values of the *preprocessing block*, i.e. the weights, are not optimized/learned during the training. This pre-processing has been integrated into a lazy fashion, directly into the CNN, such that the size of all kernels (weighting matrix) are set to 5×5. Their central part is initialized with the weights of the SRM kernels and the remaining elements are padded to zero. No normalization of the kernels' values is performed.

The rest of our CNN can be divided into a convolutional module, dedicated to features representation, that transforms the input image into a *feature vector*, and a classification module, consisting

This work was supported by the Algerian Ministry of Higher Education / Scientific Research, and the University of Montpellier (LIRMM).

[1] In this paper, we reference *Xu-Net* a CNN similar to the one given in [5] and not to the ensemble version [4].

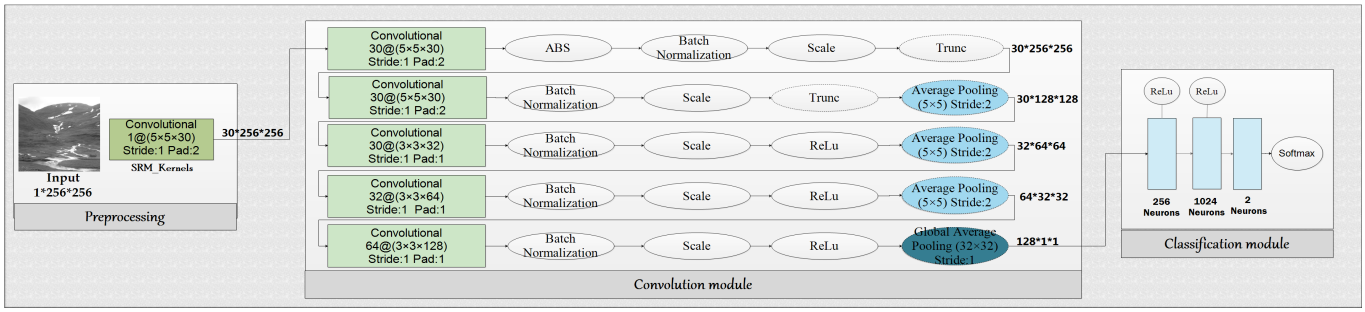

Fig. 1. Yedroudj-Net CNN architecture.

of three fully-connected layers and a softmax layer, which produces the classification decision (cover or stego).

Similarly to Xu-Net, the convolutional module has five blocks marked as 'Block 1' through 'Block 5' to extract effective features for cover and stego images discrimination; see Fig. 1. Each block is made of the following steps:

1. a *Convolution Layer*. Similar to Xu-Net [5], we set the size of the convolutional kernels to 5×5 for Blocks 1 and 2, but we reduced it to 3×3 for the Blocks 3 through 5. For all the convolution layers and similarly to Res-Net [9] and Xu-Net [5], no biases are used. Biases terms are set to false on the convolution layer and moved to the Scale Layer.
2. an *Absolute Value* activation (ABS) layer. This ABS layer is only used in Block 1 similarly to Xu-Net. It forces the statistical modeling to consider the sign symmetry of the noise residuals. The relevance of this layer was observed in Xu-Net [5].
3. a *Batch Normalization (BN)*. The BN normalizes the distribution of each feature to a zero-mean and a unit-variance, and eventually, scales and translates the distribution. The benefit of using a BN layer is that it desensitizes the training to the parameters initialization [18], allows the use of a larger learning rate which speeds up the learning, and improves the detection accuracy [7]. Note that similarly to ResNet [9], and in contrast to Xu-Net, we provide a BN layer accompanied by a scale layer. The latter attempts to learn the scaling and translation parameters more efficiently. Those two parameters can be well learned by the independent *Scale Layer*. Similarly to ResNet, we observe a very slight increase in the network's accuracy.
4. a non-linear *Activation layer*. For the Blocks 1 and 2, a *Truncation function* is used to limit the range of data values and prevent the deeper layers from modeling large values. Indeed, these values are sparse and not statistically significant. The formula of the truncation function (*Trunc*) is given in Eq. 1, and is parameterized by $T \in \mathbb{N}$, a threshold:

$$Trunc(x) = \begin{cases} -T, & x < -T, \\ x, & -T \leq x \leq T, \\ T, & x > T. \end{cases} \quad (1)$$

This **outlier** suppression process, proposed in [17], can also be seen as the use of a *robustness function*. For the Blocks, 3 through 5, the classical *Rectified Linear Unit* (ReLU) is used because it yields good performances and its gradient computation is fast.

5. An *Average pooling*. This average pooling layer is exclusively used in Blocks 2 through 5. This allows to downsample the feature maps, and thus reduces the dimensionality. For the last block, a *global* average pooling is performed to generate a one by one element for each corresponding feature map, thereby preventing the statistical modeling from grasping the location information of embedded pixels from the training data [19]. There is no pooling in the first block to avoid information loss at the beginning of the network.

The features extracted from the convolutional module feed the classification module which consists of three fully connected layers. The number of neurons in the first and second layers is 256 and 1024 respectively, and the last fully connected layer has only two neurons corresponding to the number of classes of the network's output. At the end of this module, a softmax activation function is used to produce a distribution over the two class labels.

3. EXPERIMENTS

3.1. Dataset and software platform

We use S-UNIWARD [20], and WOW [21], two well-known content-adaptive methods for the embedding in the spatial domain and their Matlab implementations (online codes[2]) with the simulator for the embedding and a random key for each embedding. We thus avoid any wrong use of the C++ codes, i.e. a fixed and unique embedding key, as reported in [3].

Our steganalysis CNN, *Yedroudj-Net*, is compared with the state-of-the-art approaches: *Xu-Net* CNN [5], *Ye-Net* CNN [17], and with *SRM + EC* which stands for the hand-crafted feature set Spatial-Rich-Model [12] and the Ensemble Classifier [11]. For a fair comparison, all the involved steganalysis methods are tested on the same subsampled images from the BOSSBase database v.1.01 [22]. All CNNs experiments were performed with the publicly available *Caffe* toolbox [23] with necessary modifications, plus digits V5. All tests were run on an NVidia Titan X GPU card.

3.2. Training, Validation, Test

Due to our GPU computing platform and time limitation, we conduct all the experiments on images of 256×256 pixels, similarly to [17]. To this end, we resampled all the 512×512 images to 256×256 images, using the *imresize()* Matlab function with the default parameters. Then, our 256×256 BOSSBase is split into two sets, 50% (resp. the other 50%) of the cover/stego pairs is assigned to the training

[2]http://dde.binghamton.edu/download/

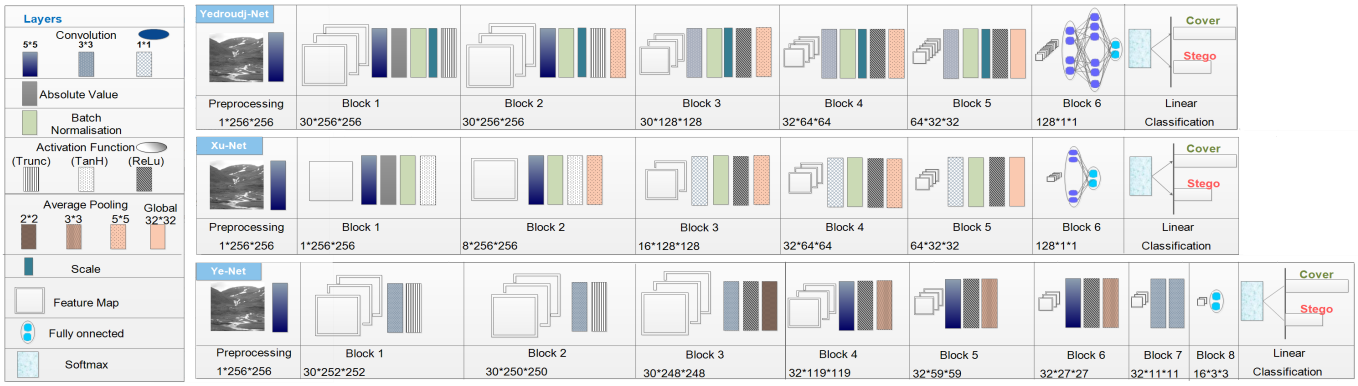

Fig. 2. Comparison of Yedroudj-Net, Xu-Net, and Ye-Net architectures.

(resp. testing) set. 4000 out of the 5000 training set pairs are randomly selected for training, the remaining 1000 pairs are set aside for validation. The testing set is left untouched during the training stage.

During the CNNs training, we fixed a maximum of 900 epochs. Nevertheless, most of the time, we manually stopped the training when an over-fitting phenomenon appeared (usually before the epoch 200 for WOW and 300 for S-UNIWARD), i.e. when the *Loss* continues to decrease on the training set but starts to increase on the validation set. In practice, observing the *Loss* curve computed on the validation test leads us to keep two versions of the CNN: the CNN's models with minimum Loss (resp. maximum) on the validation set over the previous five epochs. Those two CNN's models are evaluated on the testing set, and we report the average error probability of detection for these two CNN's models.

For *SRM + EC* we use the SRM feature set of dimension=34 671 [12], and the Ensemble Classifier [11]. We report the minimum error probability under equals prior, averaged over 10 tests.

3.3. Hyper-parameters

We apply a mini-batch stochastic gradient descent (SGD) to train our CNN. The momentum is fixed to 0.95 and the weight decay to 0.0001. No dropout is used. The batch size in the training procedure is set to 16, due to GPU memory limitation (8 cover/stego pairs). All layers are initialized using **Xavier method**: the weights follow a Gaussian distribution and are chosen so that the variance for both input and output among each layer remains the same [24]. During the training, we use the *step policy* of Caffe to adjust the learning rate (initialized to 0.01). With this *policy*, each 10% of the total number of epochs, our learning rate is decreased by a factor gamma equal to 0.1. The threshold T, for the Truncations functions (see Equ. 1) is set to 3 for the first layer and 2 for the second layer, and the 30-basic high-pass SRM filters are not normalized. Note that the source codes and the materials files are available at http://www.lirmm.fr/~chaumont/DemoAndSources.html.

3.4. Difference between the 3 CNNs

In this section we will briefly discuss the differences between our CNN *Yedroudj-Net*, the *Xu-Net* CNN and *Ye-Net* CNN, the state-of-the-art CNNs for the spatial steganalysis. In our comparisons, *Xu-Net* is a CNN similar to the one given in [5] that takes images of size of 256×256 instead of 512×512. We thus suppressed the *average pooling* from the first Block, which is a favorable measure since it avoids an early down-sampling. We also set a ReLU activation function among the Fully connected layers. Fig.2 shows the overall architectures of all CNNs. We summarize below the major similarities and differences between the CNNs:

- Both Yedroudj-Net and Xu-Net use 5 convolution layers. Yedroudj-Net has nevertheless two times more features (256) at the input of the fully connected section. Ye-Net has more convolution layers.
- Both Yedroudj-Net and Xu-Net use a *Batch Normalization* layer; the Ye-Net does not.
- Both Yedroudj-Net and Xu-Net use the *Absolute Value* layer; the Ye-Net does not.
- Both Yedroudj-Net and Ye-Net use a 30 filter bank for pre-processing; the Xu-Net does not
- Both Yedroudj-Net and Ye-Net Net use a Truncation activation function in Block 1 and 2 (We have found "Experimentally" that using Truncation activation function only in the Blocks 1 and 2 is the best choice in term of detection accuracy, those experiments are not reported here); the Xu-Net does not.
- Yedroudj-Net has three (resp. Xu-Net two, and Ye-Net one) fully connected layer.

3.5. Results without using any tricks

3.5.1. General performance comparisons

In Table 1, we report the error probability obtained when steganalyzing WOW and S-UNIWARD embedding algorithms at 0.2 bpp and 0.4 bpp. The steganalysis methods are Yedroudj-Net, Xu-Net, Ye-Net, and SRM+EC [11, 12].

Table 1. Steganalysis error probability comparison of Yedroudj-Net, Xu-Net, Ye-Net, and SRM+EC for two embedding algorithms WOW and S-UNIWARD at 0.2 bpp and 0.4 bpp.

	BOSS 256×256			
	WOW [21]		**S-UNIWARD** [20]	
Payload / **Steganalysis**	0.2 bpp	0.4 bpp	0.2 bpp	0.4 bpp
SRM+EC [11, 12]	36.5 %	25.5 %	**36.6** %	24.7 %
Yedroudj-Net	**27.8** %	**14.1** %	**36.7** %	**22.8** %
Xu-Net [5]	32.4 %	20.7 %	39.1 %	27.2 %
Ye-Net [17]	33.1 %	23.2 %	40.0 %	31.2 %

For WOW algorithm, Yedroudj-Net has an error probability 8% lower (resp. 11%) at 0.2 bpp (resp. 0.4 bpp) compared to SRM+EC. The results are also favorable for S-UNIWARD steganalysis with an equal error probability at 0.2 bpp and 2% lower at 0.4 bpp.

Compared to the other CNN algorithms, our proposed CNN achieves far superior results. Yedroudj-Net is 2% to 6% better compared to Xu-Net for the two embedding algorithms and the two payloads. The results are even better when compared to Ye-Net, where Yedroudj-Net is 3% to 9% better. Let us note that the two other CNNs are not always superior when compared to the SRM+EC. To beat SRM+EC, those approaches require using an ensemble of CNN, as proposed in [4], or increasing the learning database, as proposed in [6], and showed in section below.

Note that extreme caution must be taken for the initialization of the learning rate of the Ye-Net and the management of its evolution through the epochs. Indeed, a bad initialization prevents the network from converging. In Yedroudj-Net and Xu-Net, the use of the Batch Normalization ensures less sensitivity to such a parameter setting.

To conclude on these general comparisons, in a classical clairvoyant scenario without any channel-awareness, and without using an ensemble, a larger database, a virtual augmentation of the database, or a transfer learning, Yedroudj-Net has a clear advantage over all the state-of-the-art methods.

3.6. Results with a Base augmentation

Many tricks exist for improving the results of CNN but the *base augmentation* seems to be a very important measure to apply in order to better exploit the capacity of Deep Learning approaches.

Table 2. Base Augmentation influence: error probability comparison of Yedroudj, Xu and Ye nets on WOW at 0.2 bpp with a learning base augmented with BOWS2, and Virtually Augmented.

	BOSS	BOSS+BOWS2	BOSS+BOWS2+VA
Yedroudj-Net	**27.8** %	**23.7** %	**20.8** %
Ye-Net	33.1 %	26.1 %	22.2 %
Xu-Net	32.4 %	30.3 %	30.5 %

In machine learning, and this is also true for CNNs, it is important to use a training base large enough to ensure a good generalization but also to avoid over-training. Some authors are prone to use big databases [2, 6, 17] in order to reach the state-of-the-art results. In the above experiment, we attempt to investigate the improvement brought by increasing the learning database size without modifying the testing set. It means that the learning set does not only contain images of the same kind as in the test set: e.g. the settings of cameras, the scenes of the learning set, can all be different from those of the testing set. We show the effects of increasing the image database on the error probability in Table2. To increase the size of our training set, two scenarios have been tested inspired by [17].

In the first scenario, noted BOSS+BOWS2, we embedded the payload in the subsampled BOSSBase database v.1.01 [22]. We split this base into two sets: 50% of the cover/stego pairs to the training set, the rest to the testing set. Then, 10 000 additional pairs of cover/stego pair (obtained by subsampling BOWS2Base [25]) were added to the training set. The learning database now contains 15 000 pairs of cover/stego images minus 1000 pairs from BOSS, set aside for validation.

In the second scenario, noted BOSS+BOWS2+VA, the database is virtually augmented by performing the label-preserving flips and rotations on the BOSS+BOWS2 training set. The size of the BOSS+BOWS2 training set is thus increased by a factor of 8, which virtually gives a final learning database made of 112 000 pairs of cover/stego images plus 1000 pairs from BOSS used for validation.

Table 2 shows the performance comparisons in terms of detection error probability for Yedroudj-Net, Xu-Net [5], Ye-Net [17], against the embedding algorithm WOW [21] at payload 0.2 bpp. For all algorithms, better performances are achieved using BOSS+BOWS2 compared to using only BOSSBase. The Yedroudj-Net obtains the best results and decreases its detection error probability by 4%. Ye-Net and Xu-Net respectively decrease their detection error probability by 7% and 2%. At this point, it was not clear if the improvement was only due to a lack of data or also because the additional images came from the same cameras. We have nevertheless conducted additional experiments, reported in the paper [26], and it seems that in order to improve the performance, one must increase the database with images coming from the same sources and with a development process respecting the pixels resolutions and ratios.

When virtually augmenting the entire BOSS+BOWS2 learning set (i.e. BOSS+BOWS2+VA) thanks to the 8 combinations of rotations and flips that do not introduce interpolation, the performances are again increased. The Yedroudj-Net keeps the best results and decreases its detection error probability by 7% (Ye-Net decreases it by 11%, and Xu-Net by 2%) compared to the case of only using BOSSBase for the training. Comparing to RM+EC [11, 12], whose error probability is 36.5% with a learning on the BOSSBase, the Yedroudj-Net obtain an error probability of 20.8% which give an improvement of 16%. The Ye-Net obtains an improvement of 14% and the Xu-Net an improvement of 6%.

These tests reveal how important it is to have a large database when using CNN of 5-7 blocks. The number of parameters (without taking into account the BN and/or scale) goes approximately from 50 thousand (Xu-Net) to 500 thousand (Yedroudj-Net). Such a huge number of unknown requires bearing enough samples. The experiments show that the CNNs still do not have enough learning samples. For a steganalysis of BOSSBase with CNNs of 5-7 blocks, even 112 000 pairs of images (BOSS+BOWS2 virtually augmented) is not enough. Consequently, using a bigger base allows our CNN to achieve better performances even if the convergence time increases.

Using a GPU card of the previous generation (Nvidia TitanX) on an Intel Core i7-5930K CPU 3.50GHz$\times$12 with 32G of RAM, it takes less than one day for learning Yedroudj-Net CNN on BOSSBase, three days on BOSS+BOWS2, and more than seven days on BOSS+BOWS2+VA.

4. CONCLUSION

This article presents the evaluation of the Yedroudj-Net CNN, designed for spatial steganalysis. This CNN gathers some recent design propositions in order to build a simple approach beating the state-of-the-art approaches in a classical clairvoyant scenario without knowledge of the selection channel.

The key to the steganalysis performance improvement is the combination of the following elements: a bank of filters for the preprocessing step, a Truncation activation function, and a Batch Normalization associated with a Scale Layer.

An additional experiment dealing with the problem of the learning base size showed that by adding BOWS2 and virtually augmenting the learning database, the results become extremely satisfactory. An experiment on WOW at 0.2 bpp led to an error probability decrease of 16% compared to the RM+EC.

5. REFERENCES

[1] S. Tan and B. Li, "Stacked convolutional auto-encoders for steganalysis of digital images," in *Proceedings of Signal and Information Processing Association Annual Summit and Conference, APSIPA'2014*, Siem Reap, Cambodia, Dec. 2014, pp. 1–4.

[2] Yinlong Qian, Jing Dong, Wei Wang, and Tieniu Tan, "Deep Learning for Steganalysis via Convolutional Neural Networks," in *Proceedings of Media Watermarking, Security, and Forensics 2015, MWSF'2015, Part of IS&T/SPIE Annual Symposium on Electronic Imaging, SPIE'2015*, San Francisco, California, USA, Feb. 2015, vol. 9409, pp. 94090J–94090J–10.

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

[4] Guanshuo Xu, Han-Zhou Wu, and Yun Q. Shi, "Ensemble of CNNs for Steganalysis: An Empirical Study," in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, Vigo, Galicia, Spain, June 2016, IH&MMSec'16, pp. 103–107.

[5] G. Xu, H. Z. Wu, and Y. Q. Shi, "Structural Design of Convolutional Neural Networks for Steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, May 2016.

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

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

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

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'2016*, Las Vegas, Nevada, June 2016, pp. 770–778.

[10] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[11] J. Kodovský, J. Fridrich, and V. Holub, "Ensemble Classifiers for Steganalysis of Digital Media," *IEEE Transactions on Information Forensics and Security, TIFS*, vol. 7, no. 2, pp. 432–444, 2012.

[12] J. Fridrich and J. Kodovský, "Rich Models for Steganalysis of Digital Images," *IEEE Transactions on Information Forensics and Security, TIFS*, vol. 7, no. 3, pp. 868–882, June 2012.

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

[14] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, "Selection-channel-aware rich model for steganalysis of digital images," in *Proceedings of IEEE International Workshop on Information Forensics and Security, WIFS'2014*, Atlanta, Georgia, USA, Dec. 2014, pp. 48–53.

[15] T. Denemark, M. Boroumand, and J. Fridrich, "Steganalysis features for content-adaptive jpeg steganography," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1736–1746, Aug. 2016.

[16] Y. Qian, J. Dong, W. Wang, and T. Tan, "Learning and transferring representations for image steganalysis using convolutional neural network," in *Proceedings of IEEE International Conference on Image Processing, ICIP'2016*, Phoenix, Arizona, Sept. 2016, pp. 2752–2756.

[17] Jian Ye, Jiangqun Ni, and Y. Yi, "Deep learning hierarchical representations for image steganalysis," *IEEE Transactions on Information Forensics and Security, TIFS*, vol. 12, no. 11, pp. 2545–2557, Nov. 2017.

[18] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 2015, pp. 448–456.

[19] Min Lin, Qiang Chen, and Shuicheng Yan, "Network in network," in *International Conference on Learning Representations, ICLR 2014*, Banff, Canada, Apr. 2014, p. 10.

[20] V. Holub, J. Fridrich, and T. Denemark, "Universal Distortion Function for Steganography in an Arbitrary Domain," *EURASIP Journal on Information Security, JIS*, vol. 2014, no. 1, 2014.

[21] V. Holub and J. Fridrich, "Designing Steganographic Distortion Using Directional Filters," in *Proceedings of the IEEE International Workshop on Information Forensics and Security, WIFS'2012*, Tenerife, Spain, Dec. 2012, pp. 234–239.

[22] P. Bas, T. Filler, and T. Pevný, "'Break Our Steganographic System': The Ins and Outs of Organizing BOSS," in *Proceedings of the 13th International Conference on Information Hiding, IH'2011*, Prague, Czech Republic, May 2011, vol. 6958 of *Lecture Notes in Computer Science*, pp. 59–70, Springer.

[23] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, Orlando, Florida, USA, Nov. 2014, ACM, pp. 675–678.

[24] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS'2010*, Chia Laguna Resort, Sardinia, Italy, May 2010, vol. 9 of *Proceedings of Machine Learning Research*, pp. 249–256.

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

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