https://doi.org/10.2352/ISSN.2470-1173.2017.7.MWSF-324

Pre-training via fitting deep neural network to rich-model features extraction procedure and its effect on deep learning for steganalysis

Jishen Zeng,[+] Shunquan Tan,[×] Bin Li,[+] and Jiwu Huang,[+] [+]College of Information Engineering, Shenzhen University, China. [×]College of Computer Science and Software Engineering, Shenzhen University, China. [+×]Shenzhen Key Laboratory of Media Security, Guangdong Province, 518060 China. (Corresponding author: Shunquan Tan, e-mail: tansq@szu.edu.cn).

Abstract

Recent studies have shown that the steganalytic approaches based on deep learning frameworks cannot surpass their rich-model features based companions in performance. According to our analysis, one of the main causes of the unsatisfactory performance of deep learning frameworks is that training procedure tends to get stuck at local plateaus or even diverge when starting from a non-ideal initial state. In this paper we will try to investigate how to fit deep neural network to a rich-model features set. We regard it as a pre-training procedure and study its effect on deep learning for steganalysis. The state-of-the-art JPEG steganalytic features set DCTR is selected as the target and its features extraction procedure is divided into multiple sub-models. A deep learning framework with similar sub-networks is proposed. In the pre-training procedure we train the framework from bottom to up, fitting the output of each sub-network to the actual output of the corresponding sub-module of DCTR. The motivation behind the scenario is that we reinforce the proposed framework learn to fit the nonlinear mapping implicit in DCTR and expect when it is trained from an initial state which represents an approximate solution of DCTR, we can get better performance compared to what DCTR has achieved.

Motivation

For the past few years, rich-model features combined with ensemble classifiers [1] reign supreme in image steganalysis. In spatial domain, SRM (34,671 dimensions) [2] performs beyond the majority of steganalytic algorithms. In JPEG domain, DCTR (8,000 dimensions) [3] achieves very promising performance with relatively low dimensionality, while PHARM (12,600 dimensions) [4] provides better performance with higher dimensionality. Their common characteristic is that they all include multiple sub-models with tens of thousands of features, which exhibits similarity to deep learning approaches [5, 6], e.g, the Convolutional Neural Network (CNN).

With rapid development of parallel computing ability provided by GPU (Graphics Processing Units) acceleration, deep learning seems to be a promising alternative [5]. Many researchers started to applying deep learning frameworks in image steganalysis. In [6], Tan and Li proposed a CNN based steganalytic detector and explored the effect of unsupervised pre-training on the performance of the proposed detector. In [7], Qian et al. proposed another CNN based steganalytic detector in which a hand crafted high passed filter (KB filter) is introduced in the pre-processing procedure. With the help of GPU, the performance of their proposed detector can get close to that of SRM. In [8], Pibre et al. verified that deep learning is a good steganalysis tool when embedding key is reused for different images. Xu et al. [9] proposed yet another CNN based steganalytic detector featured with its Batch Normalization (BN) layers. Although pre-processing with KB filter is still indispensable, their proposed framework outperforms SRM. However, our extensive experiments show that the performance improvement of their approach comes at the expense of more fragile to the cover-mismatch problem. Furthermore, all of the above methods focus on spatial-domain steganalysis. None of them addresses the application of deep learning frameworks in JPEG domain steganalysis. The feasibility of deep learning frameworks in images steganalysis still needs to be further addressed, especially in the JPEG domain.

One of the main causes of the unsatisfying performance of deep learning frameworks in steganalysis is that the training procedure tends to get stuck on local plateaus or even diverge when starting from a non-ideal initial state. Traditional deep learning frameworks use unsupervised pre-training to go over the obstacle. However, our previous paper [6] showed that the effect of unsupervised pre-training on the performance of deep learning frameworks in steganalysis is not obvious. In this paper we try another way and explore the possibility to fit CNN to a rich-model features set. We regard it as a pre-training procedure and study its effect on deep learning for steganalysis. The state-of-the-art JPEG steganalytic features set DCTR is selected as the targeted rich-model features. The features extraction procedure of DCTR is divided into multiple sub-models. A deep learning framework with similar sub-networks is proposed. In the pre-training procedure We train the framework from bottom to up, fitting the output of each sub-networks to the actual output of the corresponding sub-module of DCTR. The motivation behind the scenario is that we reinforce the proposed framework learn to fit the nonlinear mapping implicit in DCTR and expect when the deep learning framework is trained from an initial state which represents an approximate solution of DCTR, we can achieve better performance compared to what DCTR has achieved after the final supervised training procedure.

In the following section, we give a brief overview of DCTR features set and discuss the similarities and differences between rich models (including DCTR) and CNN. In the third section, we propose a supervised pre-training deep learning framework via fitting a deep CNN network to DCTR features extraction procedures. Then in the fourth section, results for experiments conducted on images extracted from ImageNet [10] dataset are presented. Finally, we conclude the proposed approach in the last section.

Preliminaries

We go through the DCTR features extraction procedure firstly. And then briefly explore the similarities and differences between rich features model and CNN to prepare the basis foundation of our proposed pre-training strategy for deep-learning steganalytic frameworks.

DCTR

Similar with other state-of-the-art JPEG rich-model based steganalytic algorithms [4], DCTR [3] takes decompressed JPEG images as input and its features extraction procedure can be divided from bottom to up into three steps:

- *Convolution*: Given a $M \times N$ JPEG image, we decompress it to the corresponding spatial-domain version $\mathbf{X} \in \mathbb{R}^{M \times N}$. Sixty-four 8×8 DCT basis patterns are defined as $\mathbf{B}^{(k,l)} = (B_{mn}^{(k,l)}), 0 \le k, l \le 7, 0 \le m, n \le 7$:

$$B_{mn}^{(k,l)} = \frac{w_k w_l}{4} cos \frac{\pi k(2m+1)}{16} cos \frac{\pi l(2n+1)}{16}, \qquad (1)$$

where $w_0 = \frac{1}{\sqrt{2}}$, $w_k = 1$ for $k > 0$. $\mathbf{X}$ is convolved with $\mathbf{B}^{(k,l)}$ to generate 64 noise residuals $\mathbf{U}^{(k,l)}, 0 \le k, l \le 7$:

$$\mathbf{U}^{(k,l)} = \mathbf{X} * \mathbf{B}^{(k,l)} \qquad (2)$$

The purpose of this step is to suppress the image contents as much as possible.
- *Quantization and truncation*: The elements in each $\mathbf{U}^{(k,l)}$ are quantized with a predefined quantization step q and then truncated with a given threshold T. The purpose of this step is to reduce the computational complexity.
- *Pooling operation*: The DCTR features set is eventually built by summing up the absolute values of the quantized and truncated elements in each $\mathbf{U}^{(k,l)}$, to further reduce features dimensions and obtain the final JPEG steganalytic features set.

From rich model to CNN

Basically, rich steganalytic models can be regarded as one-staged features extraction system with a cascade of convolution, quantization/truncation and pooling operation. Likewise, CNN can also be regarded as a cascade of several convolutional layers, regulation layers and pooling layers. By virtue of the structure, rich steganalytic models exhibit similarity to CNN [6]. However, unlike hand-crafted rich steganalytic models, CNN is a deep-learning framework with a great deal of learnable parameters. Regardless of the types of the layers, CNN is made up of units that have learnable weights and biases. Backpropagation algorithm is used in the training procedure of CNN.

Although rich steganalytic models and CNN have a lot in common, CNN is still hard to surpass the rich steganalytic models. This may due to the following reasons: Firstly, the training procedure of deep learning frameworks easily gets stuck on local plateaus or even diverges when starting from a non-ideal initial state. Secondly, CNN is commonly used to identify image contents. However in steganalysis the primary task is to identify stego noises. Image contents tend to be mixed up with stego noises, and traditional data pre-processing strategies used in CNN, including mean subtraction, normalization, and whitening cannot help to suppress image contents and retain stego noises at the same time. Therefore, An attractive proposition setting before us is that: Is it possible to take advantage of the domain knowledge of rich models in the training procedure of deep CNN steganalytic framework? Based on previous reports [6–9] and our own work [11] we have already noticed that Two crucial components of rich models can not be efficiently learned by CNN: the convolutional kernels used to extract diverse noise residuals and the threshold quantizers used to reduce model complexity. Based on the above reasons, we directly introduce hand-crafted convolutional kernels and threshold quantizers in the bottom layers of our proposed deep-learning framework, as reported in [11]. And furthermore, in this paper we try to use supervised pre-training scheme to pre-train the upper layers of our proposed deep-learning framework via fitting CNN to rich-model features extraction procedure, to make the performance of the proposed framework close to or even exceed what rich model can achieve.

Our proposed CNN steganalytic framework and the pre-training scheme

We imitate the DCTR JPEG steganalytic features set and propose a CNN steganalytic framework with a pre-training scheme.

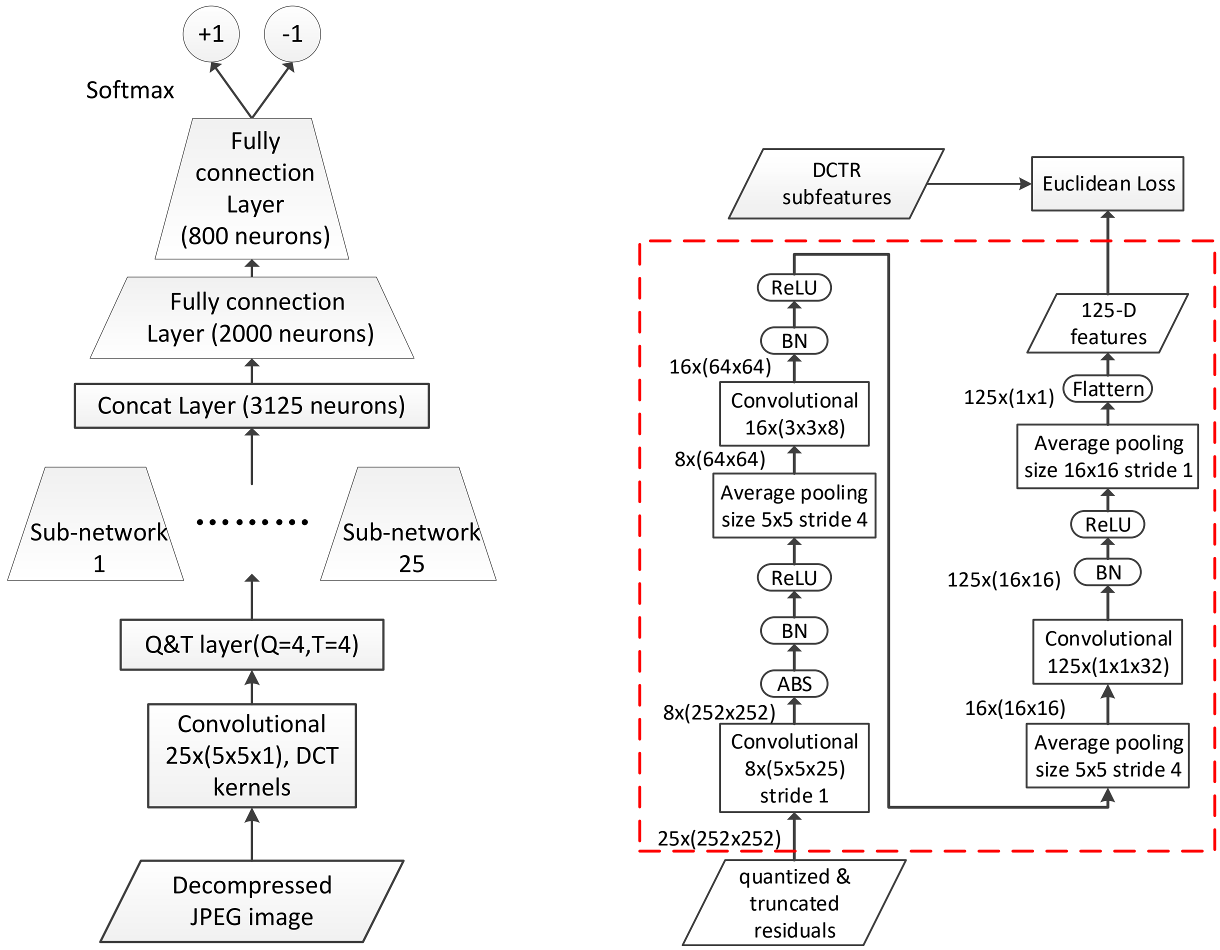
The framework structure

At the current stage, the size and the complexity of deep learning models in steganalysis is still constrained by limited GPU memory. Therefore the DCTR JPEG steganalytic features set is selected as our object of imitation due to its relative simplicity and satisfactory performance in the family of rich-model steganalytic features. However, the sixty-four residual sub-features adopted in DCTR features set is still too large from the perspective of saving GPU memory. Therefore we adopt twenty-five 5×5 DCT basis patterns in the bottom convolutional layer of our proposed deep learning framework, which is defined as $\mathbf{B}^{(k,l)} = (B_{mn}^{(k,l)}), 0 \le k, l \le 5, 0 \le m, n \le 5$:

$$B_{mn}^{(k,l)} = \frac{w_k w_l}{5} cos \frac{\pi k(2m+1)}{10} cos \frac{\pi l(2n+1)}{10}, \quad w_0 = 1, \; w_k = \sqrt{2} \text{ for } k > 0. \qquad (3)$$

On top of the the bottom convolutional layer, we introduce a specific layer which acts as a threshold quantizer. Opposite to what reported lately by Xu et al. [9], our extensive experiments show that quantization and truncation are hard to imitate by existing deep learning frameworks. Therefore this layer is hand-crafted and the quantization step and the truncation threshold in this layer are both set to 4, as those used in DCTR.

In the first convolutional layer twenty-five residual maps are obtained via convolving the target JPEG image with twenty-five 5×5 DCT basis patterns in (3). Then in the second layer, quantization and truncation both reduce the data range and bring in non-linearities. On top of the two bottom layers, we use twenty-five sub-networks with identical structure to collect 125 dimensional sub-features from each of the twenty-five quantized and truncated residual maps. We set the output dimension of each sub-network



(a) (b)

Figure 1. *Our proposed deep-learning framework. (a) illustrates the final assembled deep learning JPEG steganalytic detector. (b) illustrates the structure of the sub-networks. The network contained in the red dashed rectangle is the network structure to be pre-trained and later cloned in the final assembly process.*

to 125 for the reason that DCTR also set the output dimension of each sub-model to 125, and we will pre-train the sub-networks via fitting them to the features extraction procedure of each DCTR sub-model, as mentioned in the following subsection.

The output of the sub-networks, the twenty-five 125 dimensional sub-features are concatenated together to generate the final 3,125 dimensional features. They act as the input of the top three-layer fully-connected neural network which output the final prediction. Our proposed deep CNN steganalytic framework is illustrated in Fig. 1(a). The detailed structure of the twenty-five sub-networks is illustrated in Fig. 1(b).

The pre-training procedure

In the pre-training procedure, we try to train the sub-networks to imitate the output of the corresponding DCTR sub-models. We construct a compact sub-network prototype which takes a single output feature map of the second layer as input and output 125 dimensional feature vector. The prototype is trained by using the quantized and truncated DCTR residual maps from cover images as input, and minimizing the fitting errors between the outputs of the sub-network and the actual DCTR sub-features generated by DCTR sub-models with Euclidean Loss function, as illustrated in Fig. 1(b). We expect that with the supervised pre-training phase via fitting the sub-networks to the corresponding sub-models, the final assembled deep CNN JPEG steganalytic detector can absorb the domain knowledge provided by DCTR. So that it can converge better and jump out of the local plateaus in the classification space.

In the final assembly process, twenty-five clones of the pre-trained prototype are inserted into the framework as illustrated in Fig. 1(a). Each of them takes one of the twenty-five quantized and truncated feature maps of the second layer as input. Their outputs are concatenated together to constitute the final 3,125 dimensional output features. Those output features will be fed into the top fully-connected neural network. After the assembly process, the training data including cover and stego images will be fed into the assembled model to further fine-tune our proposed framework.

Experimental results

All of the experiments are conducted on a GPU cluster with eight NVIDIA® Tesla® K80 dual-GPU cards. Based on machine capacity considerations, we restrict the size of the target images to 256×256. 50,000 JPEG images with size larger than 256×256 are randomly picked out from ImageNet [10]. Their left-top 256×256 regions are cropped, greyed and then JPEG recompressed with quality factor 75. 50% of them are randomly selected for training and the rest 50% are for testing. Our implementation is based on the publicly available Caffe toolbox [13] with a hand-crafted convolutional layer (with twenty-five 5×5 DCT basis patterns) and a threshold quantizer, which are both implemented by ourselves. J-UNIWARD [12], a state-of-the-art JPEG domain steganographic scheme is our attacking target in the experiments. In the traing procedure of all of the models, we use mini-batch stochastic gradient descent with "step" learning rate starting from 0.01 (step-size=5000) and a momentum fixed to 0.9.

The effect of pre-training procedure

We further randomly picked out 5,000 JPEG images from the 50,000 converted images in the dataset. In order to fully explore the fitting ability of the pre-training procedure, the images are permuted. 80% of them are randomly selected for training and the rest 20% are for testing. From each one of them, twenty-five residual maps are generated by twenty-five 5×5 DCTR kernels. The corresponding twenty-five 125 dimensional DCTR sub-features are generated by the DCTR features extraction procedure as well. Therefore there are $25 \times 5,000 \times 0.8 = 100,000$ residual maps and the corresponding DCTR sub-features for training. The rest $25 \times 5,000 \times 0.2 = 25,000$ residual maps and the corresponding DCTR sub-features are for testing. The batch size in the pre-training procedure is 64 and the maximum number of the iterations is set to 50×10^4. The fitting error is measured by Euclidean Loss.

In Fig. 2, the outputs of the pre-trained sub-network prototype after 50×10^4 training iterations and the corresponding actual DCTR sub-features for some selected residual maps of an ImageNet image are plotted. It is amazing that in the pre-training procedure, the fitting errors between the outputs of the pre-trained sub-network prototype and the actual DCTR sub-features can converge to a quite small value, which indicates that the first-order statistics used in DCTR can be well represented by the pre-trained sub-network prototype from the perspective of minimizing Euclidean distance. The variation trend of the outputs of the pre-trained sub-network prototype is approximate to the actual DCTR features. In the experiment, The best average Euclidean Loss for all of the testing images is 0.153631.

We can directly concatenate the outputs of the twenty-five clones of the pre-trained sub-network prototype together to generate a 3,125 dimensional steganalytic features vector. One interesting problem is that is it possible to directly feed this features vector into ensemble classifier [1] and use it to replace the DCTR features set. In Tab. 1, we compare the performance of the original 8,000 dimensional DCTR features set, the 3,125 dimensional DCTR features set with twenty-five residual maps generated by 5×5 DCTR kernels, and the 3,125 dimensional steganalytic features vector concatenated from the outputs of the twenty-five clones of the pre-trained prototype. They are trained and tested on the 50,000 JPEG images we mention at the beginning of this section. The corresponding stego images are generated by J-UNIWARD [12] with 0.4bpnzAC. The results show that the original 8,000 dimensional DCTR features outperform the other two. The accuracy of the 3,125 dimensional DCTR features set is also about 2% better than the features vector concatenated from the outputs of the clones of the pre-trained prototype. Certainly, the result show that the output of the pre-trained prototype still can be directly used for steganalysis in a certain extent. We expect that the clones of the pre-trained prototype can behave better when they are integrated into our proposed deep CNN steganalytic detector.

TABLE 1. The performance of the original DCTR features set, the 3,125 dimensional DCTR features set and features vector concatenated from the outputs of the twenty-five clones of the pre-trained prototype. The results are all with ensemble classifier.

Steganalytic Features Sets	Dimension	Accuracy
DCTR	8000	62.59%
DCTR	3125	58.61%
Concatenation of the outputs of 25 sub-networks	3125	56.59%

Fig. 3 shows the effect of pre-training procedure on the performance of our proposed framework. In Fig. 3, we compare the performance of the framework in which the sub-networks are randomly initialized, and the framework in which the sub-networks are the clones of the pre-trained prototype. This experiment is also conducted on the 50,000 JPEG images. The corresponding stego images are also generated by J-UNIWARD [12] with 0.4bpnzAC. The training batch size here is set to 20 due to the computation capacity. The experimental results indicated that with or without pre-training procedure, the performance of our proposed deep CNN steganalytic detector can surpass that of DCTR as well as that of the ensemble classifier fed with 3,125 dimensional steganalytic features vector concatenated from the outputs of the twenty-five clones of the pre-trained prototype. However, we can also see that the framework with randomly initialized sub-networks requires a great deal of iterations to converge to a good place. The pre-traing procedure for the sub-networks can considerably boost the convergence speed, however with cost of slightly performance reduction. Certainly, the experimental results also show that the framework with pre-trained sub-networks can obtain more stability than that with randomly initialized sub-networks.

Comparison to prior arts

In Fig. 4, we compare the performance of our proposed frameworks (with pre-trained sub-networks) with other steganalytic models including two JPEG domain rich models (DCTR [3] and PHARM [4]), and a deep-learning steganalytic model proposed by Xu et al. [9] (referred as Xu's model in the context). The experiments are all conducted on the 50,000 JPEG images. The corresponding stego images are also generated by J-UNIWARD. From Fig. 4 we can see that our proposed framework (with pre-trained sub-networks) can obtain significant performance improvement compared with DCTR, but is still inferior to PHARM. By the way, in Fig. 4 we can also observe the the effect of BN layers on the performance of our proposed framework. BN layers

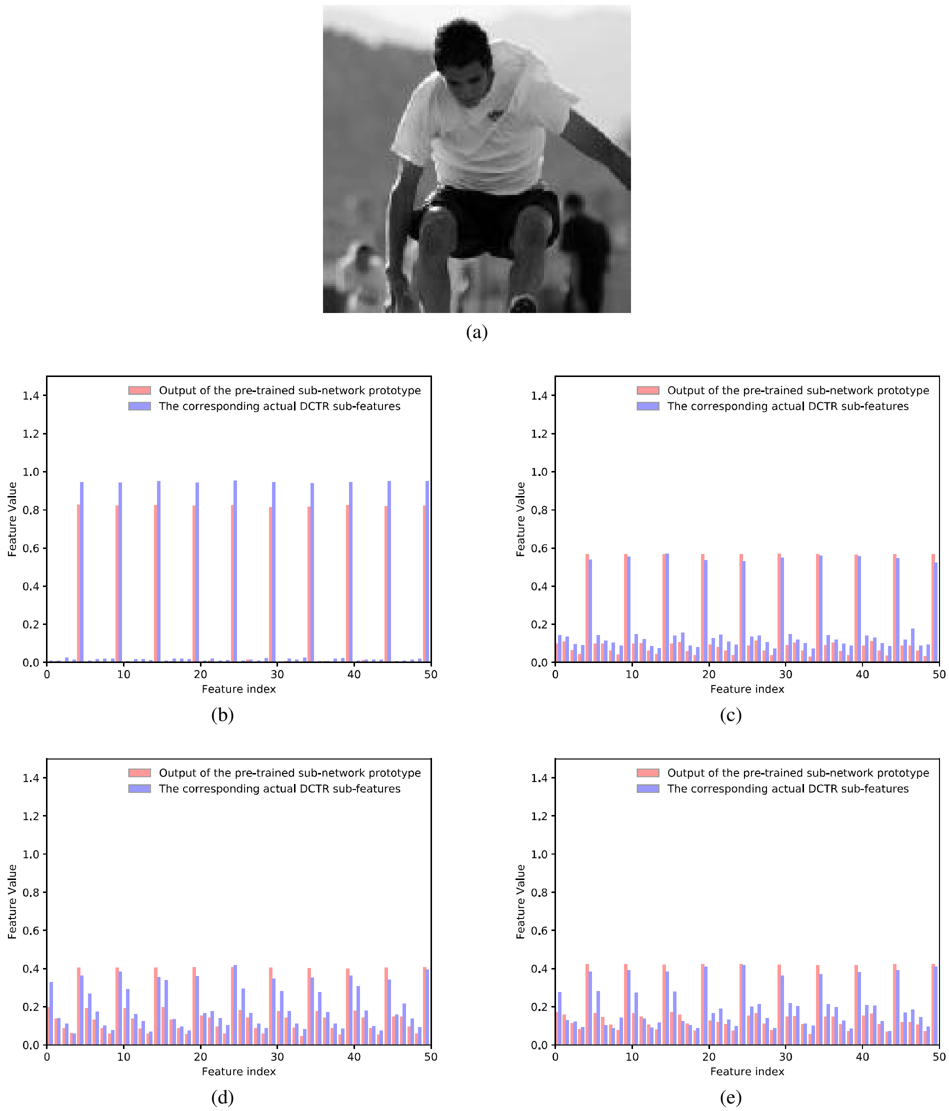


Figure 2. *Comparisons of the the outputs of the pre-trained sub-network prototype and the corresponding actual DCTR sub-features. (a) The converted version of ImageNet image "n00440382_15279.jpg". (b) The fitting result of DC (0,0) residual map (with Euclidean Loss=0.204945). (c) The fitting result of AC (0,3) residual map (with Euclidean Loss=0.103903). (d) The fitting result of AC (3,3) residual map (with Euclidean Loss=0.164526. (e) The fitting result of AC (4,1) residual map (with Euclidean Loss=0.133274. Please note that we only show the results for the features with index in interval* $[0,50]$ *for the sake of clarity.*

are also crucial to the performance of our proposed framework. The performance of our proposed framework without BN layers is poor.

Conclusion

Up to now, no domain knowledge in image steganalysis is utilized in the design and training of deep learning based steganalytic detector except the KB filter used in previous works. And furthermore, only unsupervised pre-training is concerned in the literature of deep machine learning. In this paper for the first time we explore how to fit deep neural network to a rich-model steganalytic features set. We regard it as a pre-training procedure and study its effect on deep learning for steganalysis. Using this way, we propose a supervised pre-training deep learning framework, which can learn the domain knowledge implied in the state-of-the-art DCTR features set. With the help of BN layer, well designed pre-training procedure and large amount of pre-training/training data, our proposed framework can get better performance compared with the DCTR features set. The contribution of pre-training procedure is obvious. Our future work will focus on finding more effective deep-learning steganalytic frameworks with higher detection accuracy.

Acknowledgments

This work was supported in part by the NSFC (61332012, 61402295, 61572329), Guangdong NSF (2014A030313557),

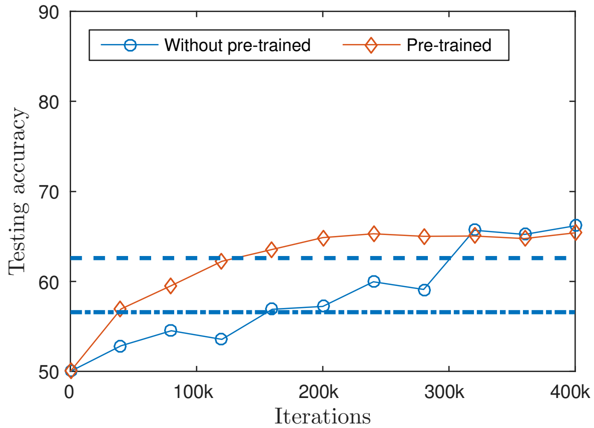


Figure 3. The effect of pre-training procedure on the performance of our proposed framework. ***Without pre-trained*** *denotes the performance of the framework with randomly initialized sub-networks on standalone testing set along with the increment of training iterations.* ***Pre-trained*** *denotes the performance of the the framework with pre-trained sub-networks. The dash reference line denotes the best testing accuracy of original DCTR features, while the dash-dot reference line denotes the best testing accuracy of the ensemble classifier fed with 3,125 dimensional steganalytic features vector concatenated from the outputs of the twenty-five clones of the pre-trained prototype.*

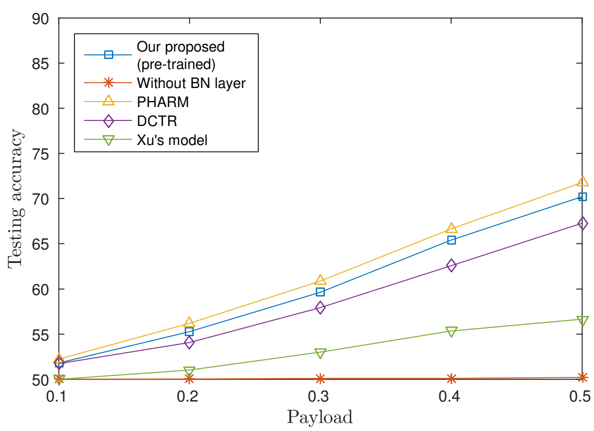


Figure 4. Comparison of the performance of our proposed frameworks (with pre-trained sub-networks) and other steganalytic models.

and Fundamental Research Program of Shenzhen City (JCYJ20140418182819173).

References

[1] J. Kodovský and J. Fridrich, "Ensemble classifiers for steganalysis of digital media," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, 2012.

[2] J. Fridrich and J. Kodovský, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012.

[3] V. Holub and J. Fridrich, "Low-complexity features for JPEG steganalysis using undecimated DCT," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 219–228, 2015.

[4] V. Holub and J. Fridrich, "Phase-aware projection model for steganalysis of JPEG images," in *Proc. SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics XVII*, vol. 9409, 2015.

[5] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[6] S. Tan and B. Li, "Stacked convolutional auto-encoders for steganalysis of digital images," in *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA'2014)*, Dec. 2014.

[7] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in *Proc. SPIE, Media Watermarking, Security, and Forensics 2015*, 2015, pp. 94 090J–1–94 090J–10.

[8] L. Pibre, P. Jérôme, 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 *Proc. Media Watermarking, Security, and Forensics, Part of IS&T International Symposium on Electronic Imaging (EI'2016)*, Feb. 2016.

[9] 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, 2016.

[10] "ImageNet," `http://image-net.org/`, [Online].

[11] J. Zeng, S. Tan, B. Li, and J. Huang, "Large-scale JPEG steganalysis using hybrid deep-learning framework," *arXiv:1611.03233*, 2016. [Online]. Available: `http://arxiv.org/abs/1611.03233`

[12] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP Journal on Information Security*, vol. 2014, no. 1, pp. 1–13, 2014.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv:1408.5093*, 2014. [Online]. Available: `http://arxiv.org/abs/1408.5093`

Author Biography

Jishen Zeng received the B.S. degree from the School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China, in 2015. He is currently pursuing the Ph.D. in Shenzhen University, Shenzhen, China. His current research interests include steganography, steganalysis, deep learning, and multimedia security.

Shunquan Tan received the B.S. degree in computational mathematics & applied software and the Ph.D. degree in computer software & theory from Sun Yat-sen University,Guangzhou, China, in 2002 and 2007, respectively. He is currently an Associate Research Fellow with College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include steganography, steganalysis, multimedia forensics, and deep machine learning.

Bin Li received the B.E. degree in communication engineering and the Ph.D. degree in communication and information system from Sun Yat-sen University, Guangzhou, China, in 2004 and 2009, respectively. He is currently an Associate Professor with College of Information Engineering, Shenzhen University, Shenzhen, China, where he joined in 2009. His current research interests include image processing, multimedia forensics, and pattern recognition.

Jiwu Huang received the B.S. degree, from Xidian University, Xi'an, China, in 1982, the M.S. degree from Tsinghua University, Beijing, China, in 1987, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, in 1998. He is IEEE fellow, the Distinguished Professor with College of Information Engineering, Shenzhen University, China. His current research interests include multimedia forensics and security.