This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2017.2710946, IEEE Transactions on Information Forensics and Security

Deep Learning Hierarchical Representations for Image Steganalysis

Jian Ye, Jiangqun Ni*, *Member, IEEE,* Yang Yi

***Abstract*—Nowadays the prevailing detectors of steganographic communication in digital images mainly consist of three steps, i.e., residual computation, feature extraction and binary classification. In this paper, we present an alternative approach to steganalysis of digital images based on convolutional neural network (CNN), which is shown to be able to well replicate and optimize these key steps in a unified framework and learn hierarchical representations directly from raw images. The proposed CNN has a quite different structure from the ones used in conventional computer vision tasks (CVs). Rather than a random strategy, the weights in first layer of the proposed CNN are initialized with the basic high-pass filter set used in calculation of residual maps in Spatial Rich Model (SRM), which acts as a regularizer to suppress the image content effectively. To better capture the structure of embedding signals, which usually have extremely low SNR (stego signal to image content), a new activation function called truncated linear unit (TLU) is adopted in our CNN model. Finally, we further boost the performance of the proposed CNN based steganalyzer by incorporating the knowledge of selection channel. Three state-of-the-art steganographic algorithms in spatial domain, e.g., WOW, S-UNIWARD and HILL are used to evaluate the effectiveness of our model. Compared to SRM and its selection-channel-aware variant maxSRMd2, our model achieves superior performance across all tested algorithms for a wide variety of payloads.**

***Index Terms*—Steganalysis, Convolutional Neural Networks, Feature Learning.**

I. Introduction

IMAGE steganography is the science and art to conceal secret messages in the images through slightly modifying the pixel values (in spatial domain) or DCT coefficients (in JPEG domain). Nowadays, the most secure steganographic schemes are content-adaptive ones, which tend to embed the secret data in the regions with complex content where the embedding traces are less detectable. Examples in spatial domain include HUGO [1], WOW [2] and S-UNIWARD [3].

Corresponding to the development of image steganography, substantial progress has also been made in steganalysis, with the aim of revealing the presence of the hidden message in images. The state-of-the-art steganalyzers in spatial domain are the Spatial Rich Model (SRM) [4] and its several variants [5], [6]. These steganalysis tools are constructed by assembling a rich model as a union of many diverse submodels formed by joint distributions of neighboring samples from quantized image noise residuals obtained using linear and non-linear high-pass filters. Recently, to better cope with the emerging content-adaptive steganographic schemes with ever-increasing security, some selection-channel-aware steganalysis feature sets [6], [7] were proposed, among which, based on the rich media model and by utilizing the selection channel, the maxSRM [6] improved the detection of all content-adaptive steganographic schemes in spatial domain to a varying degree.

Currently the best image steganalyzers are built using feature-based steganalysis and machine learning and share the same pipeline: namely, noise residual computation, feature construction and binary classification. The success of the feature-based steganalysis depends heavily on the process of feature engineering, i.e., using the domain knowledge, e.g., the cover source model and the behavior of its opponent, to create features that make machine learning algorithms work. For the pipeline described above, the residuals help to improve the SNR of stego signal, and the state-of-the-art feature sets are unions of co-occurrences of different filter residuals, so-called rich models, which tend to be high-dimensional (e.g., 30,000 or more). From the perspective of steganalysis, to obtain a more complete description of cover source, high-dimensional representation is an inevitable trend, indicating that the features for steganalysis become increasingly complicated. In addition, note that current state-of-the-art steganalysis features are heuristically designed and the optimization of classifier is independent of the feature extraction step. In other words, the pipeline of steganalysis has barely been optimized in a unified framework.

In this paper, we show that the pipeline of steganalysis can be alternately implemented by a deep convolutional neural network (CNN) [8] to learn the optimized deep hierarchical representations for image steganalysis. An important property of CNN is that it can extract complex statistical dependencies from high-dimensional sensory input and efficiently learn deep (hierarchical) representations by re-using and combining intermediate concepts, allowing it to generalize well across a wide variety of computer vision (CV) tasks, including image classification [9], face recognition [10], and many others. It naturally motivates us to consider training a CNN to distinguish covers from stegos. In this way, the raw image to be detected can be directly mapped to a binary label (cover

This work is supported by National Natural Science Foundation of China (No. 61379156 and 60970145), the National Research Foundation for the Doctoral Program of Higher Education of China (No. 20120171110037) and the Key Program of Natural Science Foundation of Guangdong (No.S2012020011114).

J. Ye is with the School of Electronic and Information Technology, Sun Yat-Sen University, Guangzhou, 510006, China (e-mail: yejian2@mail2.sysu.edu.cn)

J. Ni is with the School of Data and Computer Science, Sun Yat-Sen University, Guangdong 510006, China, and is also with the Xinhua College of Sun Yat-Sen University, Guangdong 510520, China (e-mail: issjqni@mail.sysu.edu.cn).

Y. Yi is with the School of Data and Computer Science, Sun Yat-Sen University, Guangzhou, 510006, China (e-mail: issyy@mail.sysu.edu.cn).

*Corresponding author.

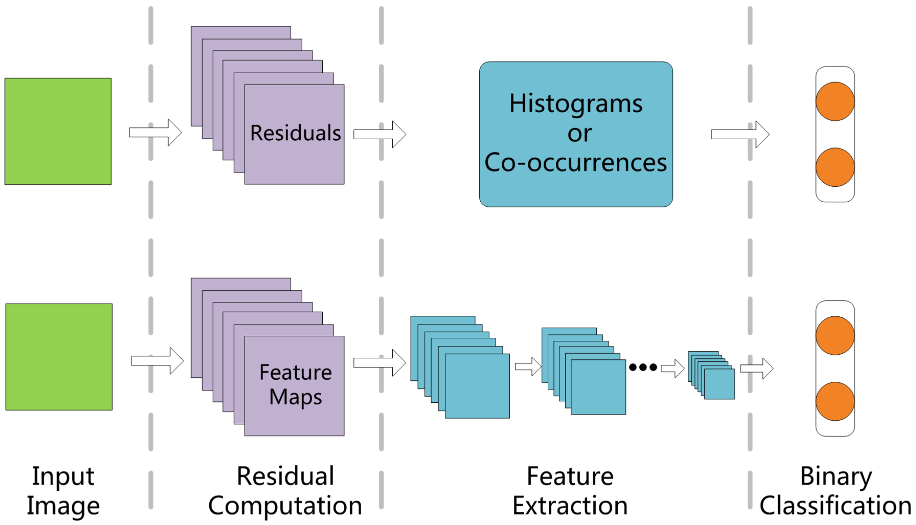


Fig. 1. The framework of image steganalysis methods and its similarity with the convolutional neural networks.

or stego) using the trained CNN. Furthermore, the feature extraction can be optimized together with the classifier, which helps to relieve us from the complicated feature-design step.

The first attempt of using CNNs to steganalysis is Qian *et al.*'s work [11], where a Gaussian-Neuron Convolutional Neural Network (GNCNN) was proposed for image steganalysis in spatial domain. By using the Gaussian function instead of the ReLU or sigmoid in conventional CNNs as the activation function, the GNCNN achieves a comparable performance to SRM on BOSSbase [12]. Recently, Xu *et al.* [13] investigated the design of CNN structure specific for image steganalysis applications, which is featured in (1) embedding an absolute activation (ABS) layer in the first convolutional layer to improve the statistical modeling in the subsequent layers; (2) applying the TanH activation function at early stages of networks to prevent overfitting; (3) performing batch normalization (BN) immediately before each nonlinear activation layer. Their results show that a well-designed CNN has the potential to provide better detection performance in steganalysis. The authors extended their works later in [14] by employing the network in [13] as base learners for ensemble classifier and obtained the results that can rival the SRM.

In this paper, we develop a supervised CNN model specific to steganalysis applications. The proposed CNN shows several prominent characteristics different from other CNNs, which are summarized as follows: (1) The first layer in the proposed CNN serves as the pre-processing module for noise residuals computation. Instead of the random strategy, the weights of the first layer are initialized with all the 30 basic filters used in the computation of residual maps in SRM [4], which corresponds to 30 output feature maps of the first layer and helps to accelerate the convergence of the network. (2) We employ a set of hybrid activation functions in the proposed CNN, where, in addition to the conventional ReLU function, a new function called truncated linear unit (TLU) is introduced to the first few layers of the network. Actually, different from the conventional CV tasks, the process of steganography can be regarded as the one of adding extremely low SNR embedding signals to the cover. The adoption of the TLU in the first few layers contributes to the adaptation to the distribution of the embedding signals and enforces the CNN to learn the high-pass filter in a more efficient manner. (3) We finally further boost the steganalysis performance by making use of the selection channel in training of the proposed CNN. The effectiveness of the proposed CNN is verified with evidence from thorough experiments using several state-of-the-art steganographic tools for a wide variety of payloads. The proposed CNN achieves considerable performance improvement in terms of detection accuracy when compared with previous CNN based steganalyzers [11], and outperforms the current state-of-the-art hand-crafted feature sets, e.g., SRM [4] and maxSRMd2 [6], by a clear margin.

The rest of the paper is organized as follows. In Section II, we present a brief review of the framework of the prevailing image steganalysis methods in spatial domain and the convolutional neural networks (CNNs). The structure of the proposed CNN is described in Section III, which is followed by the experimental results and analysis in Section IV. Finally, the concluding remarks are drawn in Section V.

II. Preliminaries

A. The Framework of Prevailing Image Steganalysis Methods

The well-established paradigm [4], [15], [16] of image steganalysis consists of three major steps, i.e., noise residual computation, feature extraction and binary classification as shown in Fig. 1.

1) Noise residual computation: The embedding operation in steganography can be viewed as adding extremely low amplitude noise to the cover. Therefore, it is wiser to model the noise residuals instead of raw pixels in steganalysis. Such an idea was initially proposed in [17] and was later adopted and developed in several subsequent methods [18], [16], [15], [4]. For a test image $X = (x_{ij})$, a popular strategy in steganalysis is to compute the noise residuals $R = (r_{ij})$ from a pixel predictor:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2017.2710946, IEEE Transactions on Information Forensics and Security

$$r_{ij} = Pred(N(x_{ij})) - lx_{ij}, \tag{1}$$

where $N(x_{ij})$ is a set of neighboring pixels of x_{ij}, $l \in \mathbb{N}$ is the residual order, and $Pred(\cdot)$ is the adopted predictor. In practice, many steganalysis schemes [17], [19] implement the predictor by convolving a finite impulse response filter K' with image X:

$$R = X * K' - lX = (r_{ij}) = (\sum_{r,c} x_{ij}^{rc} k'^{rc} - lx_{ij}), \tag{2}$$

where $*$ denotes the convolution operator, and r, c are the index of the kernel K'. According to the distributive law, the residuals above can be reformulated as:

$$R = X * K = (r_{ij}) = (\sum_{r,c} x_{i,j}^{r,c} k^{rc}). \tag{3}$$

There are plenty of choices for the filters (linear or non-linear) in steganalysis, which can be used to generate different residuals and capture different dependencies among neighboring pixels. The diversity of residuals is fundamental to the success of the so-call rich media models (RM).

2) Feature extraction: This is critical in steganalysis. With more discriminative features, it would be much easier to distinguish cover images from stego ones. In this step, the joint or conditional probability distributions of neighboring residuals are modeled through histograms or co-occurrences. For SRM and its several variants, the features are built on the basis of fourth order co-occurrence matrixes. Take horizontal co-occurrence for example, we have:

$$\begin{array}{c} c^h_{d_0 d_1 d_2 d_3} = \sum\limits_{i,j=1}^{n_1, n_2 - 3} [r_{i,j+k} = d_k, \forall k = 0,1,2,3] \cdot \varphi(\beta_{i,j}) \\ d_k \in \{-Tq, (-T+1)q, \cdots, Tq\} \end{array}, \tag{4}$$

where $[\cdot]$ is Iverson bracket whose result is 1 when statement inside is true and 0 otherwise. And $\varphi(\beta_{i,j})$ is a statistical measure of the corresponding embedding probability. For different versions of SRM, there are different values for $\varphi(\beta_{i,j})$:

$$\varphi(\beta_{i,j}) = \begin{cases} 1, & \text{SRM} \\ max(2\beta_{i,j+k}), k = 0,1,2,3, & \text{maxSRM} \\ [\beta_{ij} \geq \beta_{threshold}], & \text{tSRM} \end{cases}. \tag{5}$$

Note that the $\varphi(\beta_{i,j})$s in maxSRM and tSRM vary from one place to another, indicating that both of them are selection-channel-aware.

3) Binary classification: The final step of steganalysis is to classify an image as a cover or a stego using an elaborately designed classifier (support vector machine (SVM) or ensemble classifier), which needs to be trained through supervised learning prior to practical application.

B. Convolutional Neural Network Architecture

A convolutional neural network consists of one or several convolutional layers, followed by some fully-connected layers of neurons. The input and output of a convolutional layer are sets of arrays called feature maps, while each convolutional layer usually produces feature maps by a three-step process, i.e., convolution, non-linear activation and pooling. The first step performs some filtering using k kernels leading to k new feature maps. Therefore, each kernel is applied on the existing feature maps resulting from the previous layer. Let us denote by $F^n(X)$ the output feature map in layer n with the kernel (filter) and bias defined by W^n and B^n, respectively, we have:

$$F^n(X) = pooling(f^n(F^{n-1}(X) * W^n + B^n)), \tag{6}$$

where $F^0(X) = X$ is the input data, $f^n(\cdot)$ is a non-linear activation function that applies to each element of its input, e.g., TanH or ReLU function, and $pooling(\cdot)$ represents the pooling operation, including mean-pooling or max-pooling, etc. Generally speaking, the non-linear activation and pooling operation are optional in a specific layer. For a classification problem, a complete network usually contains several cascaded convolutional layers and ends with one fully-connected layer followed by a softmax classifier.

Obviously, the three key steps in the framework of modern steganalysis can be well simulated by a CNN model described above. According to (3), the residual computation is actually implemented through convolution which can be achieved by a convolutional layer. The cascade of multiple convolutional layers in a CNN can be trained to learn or extract high-level and discriminative representation or features of the original data, which explains the success of CNNs in many image and video recognition problems, and that also coincides with the objective of the feature extraction in steganalysis. As for the classification step, the softmax classifier in CNN acts like the SVM or ensemble classifier. In fact, a CNN based steganalyzer would allow to automatically unify residual computation, feature extraction and classification steps in one unique architecture without any a prior feature selection and to be optimized simultaneously as a whole framework.

III. The Proposed Convolutional Neural Network for Steganalysis

From the analysis in Section II, a CNN model is shown to be able to well simulate the three key steps in the framework of modern steganalysis. Therefore, it is not only realistic but also attainable to develop image steganalyzer via convolutional neural networks. However, the steganalysis task is quite different from the ones in computer vision, where the CNNs have made great success. The stego noise to deal with in steganalysis usually cannot be perceived by human perceptual system. In fact, with elaborately designed steganographic schemes, the stego usually closely resembles the cover not only visually but also statistically. As a result, the feature representations in CNN based steganalyzer should be a lot different than the ones in conventional CV tasks. In

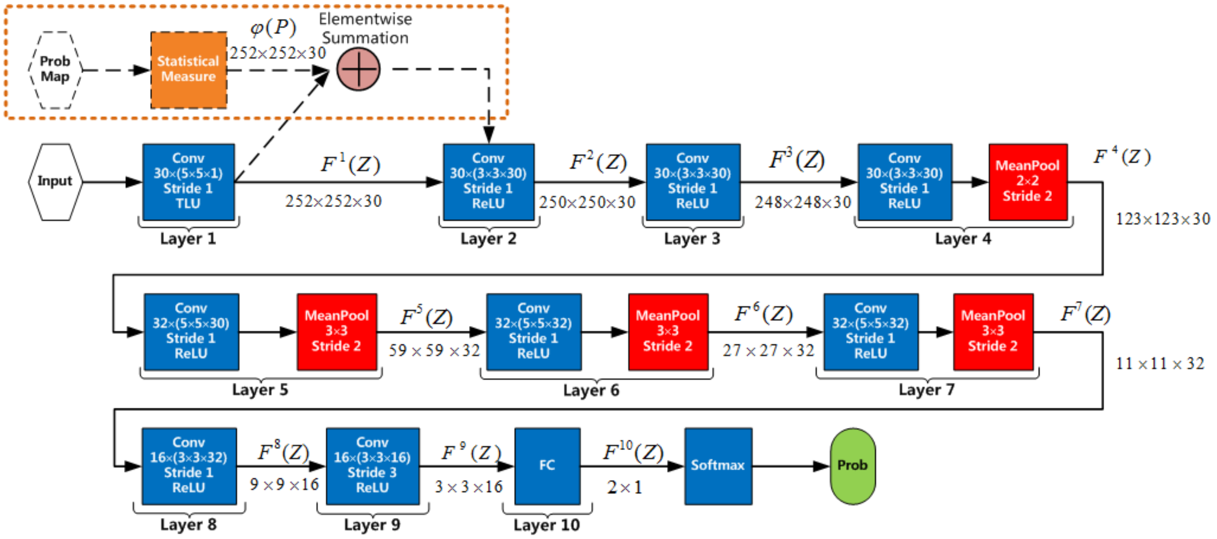


Fig. 2. The architecture of the proposed 10-layer convolutional neural network. For each convolutional layer, the input feature maps are the output of its previous layer. The layers in the dotted box only exist in the selection-channel-aware version of the proposed network.

light of this, it is not surprising to find that a CNN with random initialized weights usually cannot converge when it is trained as a steganalyzer (see Table I). Therefore, some customized designs specific to steganalysis are required in order to incorporate the domain knowledge into the learning of CNN based steganlyzer.

A. The Architecture

As illustrated in Fig. 2, the proposed CNN consists of 10 layers and ends with a fully-connected layer with a 2-way softmax, which produces the distribution over 2 class labels. Non-linear activation is applied after every convolution operation. And the pooling operations from 1st to 3th layers are suppressed. Different from other conventional CNN architectures that employ two or more fully-connected layers, we use only one necessary 2-way fully-connected layer at the end of our network. This is because the fully-connected layer usually involves too many parameters to be trained, and this could easily lead to overfitting, especially when the training set is not big enough, which is the case for our task. Besides, except the layers illustrated in Fig. 2, there is no other layer, such as Local Response Normalization (LRN) [9], dropout [9], Batch Normalization (BN) [20] or Local Contrast Normalization (LCN) [21], used in the network. The components inside the dotted box show the first two operations when the knowledge of selection channel is explored, which will be detailed in Section III-D. The depth and width of the network and the size of filters are determined by experiments based on the tradeoff between performance and model complexity.

We then proceed to discuss the naming conventions for the CNN and the way we repeat our experiments on the proposed models, which will be adopted throughout this paper. For the network whose non-linear activation functions are ReLUs, we call it ReLU-CNN, while if some of the activation functions are replaced by the new introduced truncated linear unit (TLU), which will be later elaborated in Section III-B, it is referred as TLU-CNN. In addition, the TLU-CNN becomes its Selection-Channel-Aware version SCA-TLU-CNN, when the statistical measure of embedding probabilities is incorporated in the network design. For all of the experiments on the CNN models, we repeat the training and testing procedure for three times using three different training/validation/test sets. The final experimental results are obtained by averaging three test results. Unless otherwise specified, the involved networks for the experiments in this paper are trained with resampled images on BOSS+BOWS2+AUG dataset and tested with resampled images on BOSS_test (see Section IV-B for details).

For the rest of this Section, we elaborate several key techniques employed in the design of our proposed convolutional neural network.

B. Initialization with High-Pass Filters in SRM

As mentioned earlier in Section II, the computation of residuals can be well simulated by a convolutional layer. Inspired by this, we initialize the weights of the first convolutional layer in our convolutional neural network with high-pass filter kernels used in SRM instead of random values. Although such strategy was also utilized in Qian *et al.*'s work [11], only the "square 5×5" filter in SRM was used to initialize their first layer in the GNCNN model. According to our understanding, the residuals in SRM help to improve the SNR (stego signal to image content) and it is the combination of different filter residual models that makes the success of the rich models (RM) in steganalysis. Consequently, we propose to increase the width of the first convolutional layer and initialize the weights with the kernels of all the 30 basic linear filters (the "spam" filters and their rotated counterparts) used in calculating residual maps in SRM.

The basic filters above correspond to 7 residual classes in SRM, which include 8 filters in class "1st", 4 in class "2nd", 8 in class "3rd", 1 in class "SQUARE 3×3", 1 in class "SQUARE 5×5", 4 in class "EDGE 3×3" and 4 in class "EDGE 5×5", for a total of 30 basic filters with maximum

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2017.2710946, IEEE Transactions on Information Forensics and Security

TABLE I
THE PERFORMANCE OF DIFFERENT INITIALIZATION STRATEGIES OF THE FIRST CONVOLUTIONAL LAYER IN TERMS OF DETECTION ERROR (P_E) FOR RELU-CNN AND TLU-CNN($T = 3$) ON THREE STEGANOGRAPIC SCHEMES AT A PAYLOAD OF 0.2 BPP ON RESAMPLED IMAGES. THE INVOLVED NETWORKS ARE TRAINED ON BOSS+BOWS2+AUG AND TESTED ON BOSS_TEST.

Algorithm	Model	Random	Fixed	Learned
WOW	ReLU-CNN	0.5	0.2259	0.2136
	TLU-CNN	0.5	0.2261	**0.1982**
S-UNIWARD	ReLU-CNN	0.5	0.2968	0.2937
	TLU-CNN	0.5	0.2807	**0.2540**
HILL	ReLU-CNN	0.5	0.2980	0.2971
	TLU-CNN	0.5	0.3068	**0.2761**

kernel size of 5×5. Therefore, we set the kernel size of weighting matrix in the first convolutional layer of our CNN all to 5×5 as shown in Fig. 2. Let $W_{CNN}^{5\times5}$ and $W_{SRM}^{m\times n}$ be the weight matrix and filter kernel in SRM, respectively, we initialize the central part of W_{CNN} with W_{SRM} and leave the remaining elements of W_{CNN} to be zeros. In another word, we pad $W_{SRM}^{m\times n}$ to be the $W_{CNN}^{5\times5}$ with zeros. It is worth noting that for all the SRM filters, we do not normalize them by dividing the residual orders l in formula (1).

The above initialization strategy acts as a regularization term in machine learning, which dramatically narrows down the feasible parameter space and helps to facilitate the convergence of the network. Besides, these high-pass filters make our network concentrate on the embedding artifacts introduced by steganography rather than the complex image content. To the best of our knowledge, all of the CNN models that are trained for steganalysis adopt a similar initialization strategy [11] [13] [22].

However, the initialization with the 30 SRM filters in the first layer serves as a good starting point for the network training but not the best end point. According to our experiments (see Table I), keeping these filters unchanged during training usually leads to worse results than updating them. As a result, all the filters in the first convolutional layer should be optimized through training together with other parameters in the network.

C. Truncated Linear Unit

1) Motivation: The activation function $f(\cdot) : \mathbb{R} \rightarrow \mathbb{R}$ introduces non-linearity to neural networks, which can significantly increase the capability of feature representation. Various choices are possible for $f(\cdot)$, such as the conventional sigmoid and hyperbolic tangent function, or the recently emerged ReLU (Rectified Linear Unit) function. Among them, ReLU is a notable choice for the convolutional layer in CNN and it can be formulated as

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}. \tag{7}$$

The ReLU function has been successfully applied to a wide variety of tasks in CV. For CV tasks, e.g., object classification, the target object can usually be distinguished easily from the background. In another word, the signals in such tasks are of high SNR. Under this circumstance, applying ReLU to neurons can make them selectively respond to useful signals in the input, resulting the so-called sparse features. Both theoretical and empirical arguments show that sparse representations are more likely to be linearly separable and have better generalization ability [23]. However, the picture is completely different for our task in steganalysis. The steganographic embedding procedure can be viewed as adding low-amplitude additive noises to cover images. And the embedding signals have much low amplitude compared to the image content, which means extremely low SNR. In contrast to CV tasks or some high SNR applications, where the ReLU function can well adapt to the distributions of the object signals, the activation function adopted in steganalysis should take into account the structure of the embedding signals, especially in the first several convolutional layers. Notice that in image steganography, the embedding signals are usually in the range of [-1,1], a new activation function known as truncated linear unit (TLU), which is slightly modified from ReLU, is introduced in our proposed CNN and defined as follows:

$$f(x) = \begin{cases} -T, & x < -T \\ x, & -T \leq x \leq T \\ T, & x > T \end{cases}, \tag{8}$$

where $T > 0$ is the parameter determined by experiments.

In the proposed CNN, the weighting kernels in the first convolutional layer are initialized with the basic high-pass filters in SRM, which contributes to the suppression of image content and extraction of embedding signals. The utilization of TLU in the outputs of the first convolution operation helps to: (1) adapt to the distribution of embedding signals (with low SNR); and (2) enforce the CNN to learn more effective high-pass filters in the first layer. According to our experiments, for other layers of the CNN based steganalyzer, the distributions of the input signals tend to be more consistent with the ones in conventional CV tasks. Therefore, the ReLU function is more preferable in those layers.

2) Experimental Verification and Analysis: We then proceed to verify the effectiveness of TLU in steganalysis and determine the proper T through experiments. In addition, we also try to explain the function of TLU with the help of visualization tools.

We conducted the comparisons based on the deep convolutional model shown in Fig. 2. The CNN with ReLU as activation function in all its layers, i.e., ReLU-CNN, is trained as the baseline model. The TLU-CNN with ReLU replaced by TLU in its first convolutional layer is also adopted for comparison.

Both ReLU-CNN and TLU-CNN with different T are trained for HILL, S-UNIWARD and WOW, at the payload of 0.2 bpp. The experimental results are summarized in Table II.

It is observed that, for all the three involved steganographic schemes, the TLU-CNN achieves consistently better performance in terms of detection error than the baseline ReLU-

TABLE II
THE PERFORMANCE OF RELU AND TLU ON RESAMPLED IMAGES IN TERMS OF DETECTION ERROR (P_E) WITH DIFFERENT T SETTINGS. INVOLVED NETWORKS ARE TRAINED ON BOSS+BOWS2+AUG AND TESTED ON BOSS_TEST. THE EMBEDDING PAYLOAD IS 0.2 BPP.

Algorithm	ReLU	TLU				
		$T = 3$	$T = 7$	$T = 15$	$T = 63$	$T = \infty$
WOW	0.2136	0.1982	**0.1966**	0.2142	0.2139	0.2170
S-UNIWARD	0.2937	**0.2540**	0.2624	0.2653	0.2921	0.2990
HILL	0.2971	**0.2761**	0.2812	0.2894	0.2956	0.2955

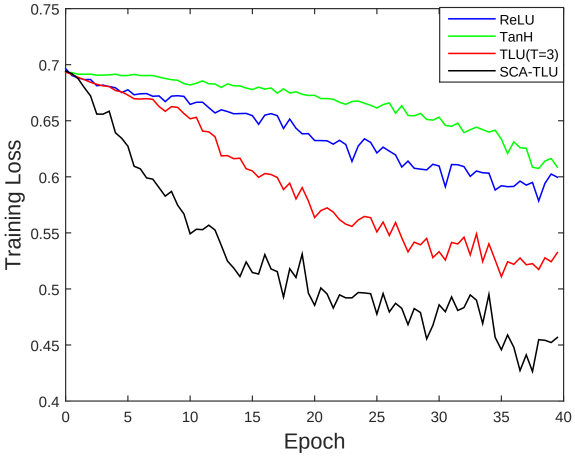


Fig. 3. The convergence performance for training the four involved 10-layer CNN models against S-UNIWARD at 0.2 bpp on resampled BOSS+BOWS2+AUG images. Normalization of the initial high-pass filters is necessary for training with TanH unit. Except for the first layer, all the activation functions are ReLUs.

CNN for most of the tested parameter values, with the best performance obtained when $T = 3$ or $T = 7$. With increasing value of T, the effect of TLU to suppress image content gradually decreases, leading to the loss in performance. Note that in the extreme case when $T = \infty$, TLU becomes an identity function (linear activation function). It is interesting to see that, even with the linear activation function which will decrease the non-linearity, the TLU-CNN can still have comparable performance to ReLU-CNN. This may result from the effect of ReLU which sets all the negative inputs to be zeros, leading to about 50% information loss in the embedding signals.

As an added bonus, it is also observed that the TLU-CNN can be trained much faster than its counterparts with ReLU and TanH unit. The convergence performance when training the three involved networks (TLU with $T = 3$, ReLU and TanH) is illustrated in Fig. 3, which shows the evolution of training error versus the number of epochs on the training images, obtained for S-UNIWARD at 0.2 bpp. The TLU is specifically designed according to the distribution of the embedding signals. Therefore, it is able to make better use of the information to train the network more efficiently.

To better illustrate the superiority of the TLU-CNN for steganalysis, we then visualize the filters in the first convolutional layer trained with TLU and ReLU, respectively. Fig. 4 (a) and (b) show the visualizations of filters (30 filters in total) in the first convolutional layers with ReLU and TLU ($T = 3$). It is clear to see that the adoption of TLU results in more distinctive features and fewer “dead” filters (filters corresponding to the feature boxes that are almost pure white). Note that although the learnt filters have similar shapes with the original SRM filters, they actually have different values. From the results in Table I, it can be easily verified that the network can indeed fine-tune the SRM filters in the first layer.

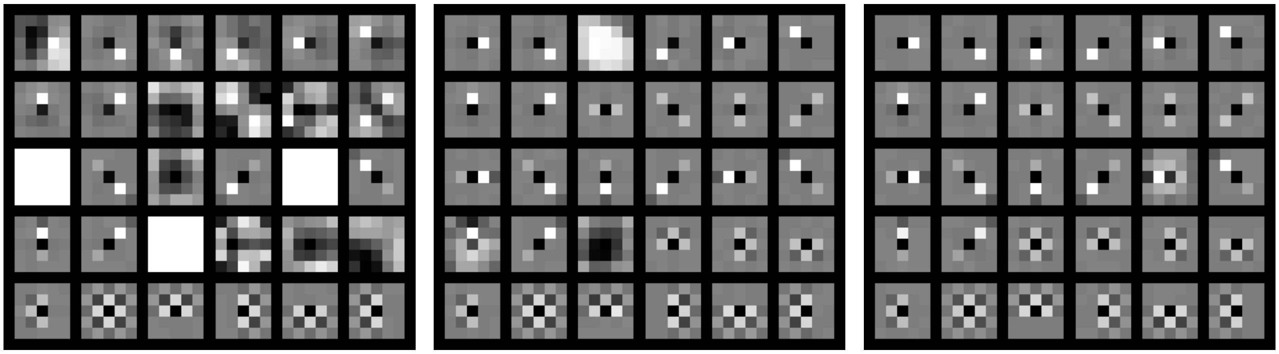
It is also interesting to evaluate the performance of the proposed CNN based steganalyzer when the ReLUs in first couple of layers are replaced by TLUs. The network is named as TLU_n if the ReLUs in its first n convolutional layers are replaced by TLUs and the ReLUs in other layers remain unchanged. As shown in Table III, despite of the similar results from TLU_1 to TLU_3, the detection accuracy becomes worse from TLU_4 and beyond. This is because when the network becomes deeper, the distribution of TLU output tends to be consistent with the one of ReLU. Therefore it is preferable to use ReLUs in relatively deep convolutional layers. Considering that the computation of ReLU can be accelerated by CuDNN library and TLU_3 needs more training epochs, we choose TLU_1 as the TLU-CNN model in our implementation so as to achieve a good compromise between training time and detection performance.

TABLE III
THE PERFORMANCE ON RESAMPLED IMAGES IN TERMS OF DETECTION ERROR (P_E) OF OUR CNN MODEL WHEN THE FIRST SEVERAL RELUS ARE REPLACED BY TLUS($T = 3$). INVOLVED NETWORKS ARE TRAINED ON BOSS+BOWS2+AUG AND TESTED ON BOSS_TEST. THE EMBEDDING PAYLOAD IS 0.2 BPP.

	TLU_1	TLU_2	TLU_3	TLU_4	TLU_5
S-UNIWARD	0.2540	0.2409	**0.2389**	0.2660	0.2851

D. Incorporating the Knowledge of Selection Channel

1) Problem Formulation: According to Kerckhoffs’s principle [24] from cryptography, to evaluate the security performance of a steganographic scheme, each element of the scheme (embedding, detection, etc) should be declared public except for the secret key. For image steganography, the probability of each pixel being modified, i.e., the so-called selection channel, when executing embedding could also be known by the steganalyst. By incorporating the knowledge of selection channel, the performance of steganalyzers against



(a) (b) (c)

Fig. 4. Visualizations of 1st convolutional layer filters in 3 different models. (a) Filters in ReLU-CNN. (b) Filters in TLU-CNN ($T = 3$). (c) Filters in SCA-TLU-CNN. Using TLU non-linearity and incorporating the knowledge of selection channel can result in more distinctive filters and fewer "dead" filters.

modern content-adaptive steganographic schemes is expected to be improved. For those recently proposed selection-channel-aware SRM feature sets, e.g., tSRM, maxSRM and σSRM [25], some statistical measures of the embedding probabilities are accumulated when calculating the co-occurrences or histograms of the corresponding residuals. For our CNN model, however, we do not explicitly compute the co-occurrences or histograms. Therefore, we have to find another way to exploit the embedding probabilities in the design of CNN based steganalyzer.

Inspired by the work in [25], we choose to take the upper bound of the expectation of L_1 norm of the residual distortion as the statistical measure of selection channel. For a cover image $X = (x_{ij})$, and the corresponding stego $Y = (y_{ij})$, we denote the difference between stego and cover by

$$N = Y - X = (y_{ij} - x_{ij}) = (n_{ij}). \tag{9}$$

For most of the existing steganographic schemes, a pixel x_{ij} is modified into $x_{ij} + 1$ and $x_{ij} - 1$ with the same probability β_{ij}, we then have $n_{ij} \in \{-1, 0, 1\}$ with probability $\{\beta_{ij}, 1 - 2\beta_{ij}, \beta_{ij}\}$, therefore,

$$E[n_{ij}] = 0, E[|n_{ij}|] = 2\beta_{ij}. \tag{10}$$

Recall that residuals can be computed by convolutions with high-pass filters. For filter kernel K, the residual distortion can be formulated as:

$$\begin{aligned} D &= K * Y - K * X = K * (Y - X) \\ &= K * N = \sum_{r,c} k^{rc} n_{ij}^{rc} = (d_{ij}) \end{aligned} , \tag{11}$$

where r, c are the index of the filter kernel. And it is easy to verify that:

$$E[d_{ij}] = \sum_{r,c} k^{rc} \cdot E[n_{ij}^{rc}] = 0. \tag{12}$$

To make use of the selection channel, the standard deviation $Std[d_{ij}]$ or the expectation of the L_1 norm of d_{ij} seems to be a natural choice:

$$\begin{aligned} Std[d_{ij}] &= \sqrt{2 \sum_{r,c} (k^{rc})^2 \beta_{ij}^{rc}} \\ E[|d_{ij}|] &= E[| \sum_{r,c} k^{rc} n_{ij}^{rc} |] \end{aligned} , \tag{13}$$

where β_{ij} is the embedding probability for x_{ij}. Note the fact that the computation of both $Std[d_{ij}]$ and $E[|d_{ij}|]$ is not computationally efficient on a GPU platform, we then turn to resort to the upper bound of $E[|d_{ij}|]$ for the statistical measure of embedding probabilities:

$$\begin{aligned} \varphi(\beta_{ij}) = 2 \sum_{r,c} |k^{rc}| \cdot \beta_{ij}^{rc} &= E[\sum_{r,c} |k^{rc}| \cdot |n_{ij}^{rc}|] \\ \geq E[| \sum_{r,c} k^{rc} n_{ij}^{rc} |] &= E[|d_{ij}|] \end{aligned} . \tag{14}$$

The $\varphi(\beta_{ij})$ above can be easily obtained by convolving the probability map $P = (p_{ij}) = (2\beta_{ij})$ with the absolute value of the residual filter K. Therefore, the upper bound map for the L_1 norm of the residual distortion d_{ij}, i.e., $\varphi(P)$ is obtained:

$$\varphi(P) = P * |K|. \tag{15}$$

We then try to take advantage of the selection channel in the design of a convolutional neural network. For maxSRM [6] and σSRM [25], the statistical measure of probabilities are accumulated in the bins of co-occurrences as the final features. Therefore, we should also try to propagate the computed $\varphi(P)$ through the whole network and make it contribute to the final features extracted by the CNN. There are two simple ways that can achieve this. One is applying an elementwise summation between $\varphi(P)$ and the output feature maps of the first convolutional layer, and the other is applying an elementwise multiplication. Our experimental results indicated that the elementwise summation approach always performs much better than the elementwise multiplication. It is most likely that with elementwise multiplication, the distribution of the output feature maps in the first layer (act as the residuals in SRM) will be changed too much, which inhibits to a great extent the feature extraction in the subsequent layers. Therefore, the elementwise summation is adopted in our network and the output of the second convolutional layer then becomes (refer to (6)):

$$\begin{aligned} F^2(Z) &= f^2((F^1(Z) + \varphi(P)) * W^2 + B^2) \\ &= f^2(F^1(Z) * W^2 + B^2 + \varphi(P) * W^2) \end{aligned}. \quad (16)$$

Note that, for our proposed CNN, except for the first convolutional layer, the non-linear activation functions in other layers are ReLUs, which means that for neurons that are activated in the second layer, their outputs can be expressed as:

$$F^2(Z)_{activated} = (F^1(Z) * W^2 + B^2) + \varphi(P) * W^2, \quad (17)$$

where the first term above is the original output of the neuron and the second one can be regarded as a weighted sum of the statistical measure of selection channel. Recall that, for the activated neurons, the ReLU is a linear operator, thus the outputs $F^n(Z)_{activated}$ of the n^{th} layer can also be factorized into two terms as the one in (17), and the effect of the statistical measure is accumulated hierarchically through the forward propagation along the network. As a result, the obtained features, or the input of the fully-connected softmax classifier (see Fig. 2), are the combination of features from two separate sources, one of which is extracted from the test image and the other from the selection channel of the same image. Note that according to (15), the statistical measure $\varphi(P)$ is computed based on the learnt filter K, and therefore, during training, $\varphi(P)$ does not participate in back propagation to update filter K.

2) Experimental Verification and Analysis: We then proceed to conduct several experiments to verify the effectiveness of the CNN based steganalyzer when the knowledge of selection channel is properly utilized. In our experiments, the SCA-TLU-CNN stands for the network that makes use of selection channel. First, the SCA-TLU-CNN can be trained more efficiently when incorporated with the information of selection channel as illustrated in Fig. 3. It is also observed that the SCA-TLU-CNN can learn more effective filters as shown in Fig. 4 (c), where the filters in the first convolutional layer of SCA-TLU-CNN are more distinct than the ones of TLU-CNN and ReLU-CNN, and there are no “dead” filters any more.

Moreover, compared to the TLU-CNN, the detection error can also be further decreased as expected. Table IV shows the performance comparison between these two CNN models against three state-of-the-art content-adaptive steganographic schemes, i.e, S-UNIWARD, HILL, and WOW at 0.2 bpp. It is clear to see that there is about 3% decrease in terms of detection error for all the involved embedding schemes.

TABLE IV
THE DETECTION ERROR (P_E) OF TLU-CNN AND ITS SELECTION-CHANNEL-AWARE COUNTERPART SCA-TLU-CNN. THE EMBEDDING PAYLOAD IS 0.2 BPP. IMAGES (BOSS+BOWS2+AUG) ARE RESIZED TO 256 × 256.

Algorithms	TLU-CNN	SCA-TLU-CNN
WOW	0.1982	**0.1691**
S-UNIWARD	0.2540	**0.2224**
HILL	0.2761	**0.2538**

E. Curriculum Learning for Low Payload Steganalysis

Nowadays it is still quite challenging to distinguish cover images from stego ones that are generated by some state-of-the-art steganographic schemes at very low payload, e.g., 0.1 bpp or below. In fact, we found that the proposed CNN described in Fig. 2 usually cannot converge if we train the network from scratch on those images with very low embedding rate. In our work, however, the issue above can be well solved by adopting a curriculum learning [26] or transfer learning strategy [27]. It benefits from the observation that humans can learn much better when the examples are not randomly presented but organized in a meaningful order which illustrates gradually more complex concepts. To put the strategy into practice, we train the network from easy aspects of the steganalysis task, and gradually increase the difficult level. In another word, we first train a network on a dataset generated at a higher embedding rate and then fine-tune it on another dataset generated at a relatively low embedding rate, and so on. Unless otherwise specified, the results on low payload steganalysis (0.1 bpp and 0.05 bpp), which are presented later in Section IV, are all based on the CNN models trained with the strategy of curriculum learning. For instance, to train a steganalyzer at 0.1 bpp, we first train a model from scratch on dataset at 0.2 bpp, then fine-tune the final model on dataset at 0.1 bpp. And similarly, the model at 0.05 bpp is obtained based on the final model at 0.1 bpp and so on. Note that this curriculum learning strategy also applies to the training of networks with images generated by subsampling (detailed in Section IV-B).

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this Section, extensive experiments are carried out to demonstrate the feasibility and effectiveness of our proposed CNN model. We compare our model with the state-of-the-art hand-crafted feature set SRM and its selection-channel-aware variant maxSRMd2. For fair comparison, all the involved steganalysis methods are tested on the same datasets.

A. The Steganographic Schemes

In our experiments, several state-of-the-art content-adaptive steganographic methods in the spatial domain, e.g., S-UNIWARD, WOW and HILL, are employed to evaluate the performance of the involved steganalyzers. And all the embedding algorithms are implemented with STC simulator based on the publicly available codes. Note that in our implementation, instead of the C++ code version of the simulator tools (S-UNIWARD, WOW) with fixed embedding key, we use the ones in Matlab code with random embedding key. This is because we found in our experiments that, although the CNN based steganalyzer could achieve extraordinary detection performance (say, detection error is less than 0.1 for WOW in 0.2 bpp) if the CNN was trained on dataset generated by simulator with a fixed embedding key, its performance would decrease dramatically (detection error is close to 0.5) when the test dataset was generated using another embedding key.In another word, the trained CNN was overfitted to the specific embedding key in training set and had no generalization capability at all. The similar problem was also reported in [22], where the authors made use of the same embedding key to create stego images for training.

B. The Datasets and Data Augmentation

In this paper, the involved experiments are carried out on two image sources. The first comes from the BOSSbase 1.01 [12], which contains 10,000 $512\times512\times8$-bit grayscale images with different texture characteristics and is widely used in steganalysis. The other one is BOWS2 [28], which is used for BOWS2 contest and consists of downsampled and cropped natural and grayscale images of size $512 \times 512 \times 8$-bit. Constrained by our available GPU computing platform, conducting experiments on full resolution images of 512×512 pixels can be extremely time consuming. As a result, we decide to evaluate the performance of our CNN based steganalyzer on test images of 256×256 pixels. To this end, we generated 3 image datasets from both image databases above in different ways as described below:

a) resample all the images into the size of 256×256 pixels (using "imresize()" in Matlab with default settings);
b) crop the central part of the original images into size of 256×256 pixels;
c) subsample the original images to 256×256 by skipping every other pixels.

For each image dataset, we then prepared 3 training sets and 1 testing set separately as follows:

1) training set BOSS: it contains 5,000 images randomly selected from BOSSbase (including 1,000 randomly selected images as validation set);
2) training set BOSS+BOWS2: it contains the images in training set BOSS and another 10,000 images from BOWS2;
3) training set BOSS+BOWS2+AUG: it is obtained by performing some label-preserving transformations, such as transposing and rotating, on the images in BOSS+BOWS2, which increases the size of BOSS+BOWS2 training set by a factor of 8;
4) testing set BOSS_test: it contains the remaining 5,000 images in BOSSbase other than the ones in training set BOSS.

TABLE V
DETECTION ERROR (P_E) OF THREE STEGANALYSIS SCHEMES TRAINED ON DIFFERENT DATASETS AND TESTED ON BOSS_TEST OF RESAMPLED IMAGES, FOR WOW AT 0.2 BPP.

Algorithms	BOSS	BOSS+BOWS2	BOSS+BOWS2+AUG
SRM	0.3266	**0.3228**	N/A
maxSRMd2	0.2424	**0.2325**	N/A
TLU-CNN	0.3364	0.2693	**0.1982**

TABLE VI
DETECTION ERROR (P_E) OF THREE STEGANALYSIS SCHEMES TRAINED ON DIFFERENT DATASETS AND TESTED ON BOSS_TEST OF CROPPED IMAGES, FOR WOW AT 0.2 BPP.

Algorithms	BOSS	BOSS+BOWS2	BOSS+BOWS2+AUG
SRM	0.3865	**0.3853**	N/A
maxSRMd2	**0.3075**	0.3092	N/A
TLU-CNN	0.4205	0.3512	**0.2808**

TABLE VII
DETECTION ERROR (P_E) OF THREE STEGANALYSIS SCHEMES TRAINED ON DIFFERENT DATASETS AND TESTED ON BOSS_TEST OF SUBSAMPLED IMAGES FOR WOW AT 0.8 BPP.

Algorithms	BOSS	BOSS+BOWS2	BOSS+BOWS2+AUG
SRM	**0.2300**	0.2332	N/A
maxSRMd2	0.1813	**0.1807**	N/A
TLU-CNN	0.1991	0.1569	**0.1182**

For CNN based steganalysis, it is preferable to adopt a larger training set to avoid overfitting. Tables V - VII summarizes the performance of our CNN based steganalyzer and other two competing steganalysis schemes trained on different training sets and tested on BOSS_test of resampled, cropped and subsampled images respectively, for WOW at 0.2 or 0.8 bpp. It is observed that, the proposed TLU-CNN suffers from substantial overfitting when it is trained on BOSS. Its performance is improved to a certain degree when the training set is replaced with BOSS+BOWS2. And the best performance is achieved if we train the network using BOSS+BOWS2+AUG. However, things are different for the involved hand-crafted feature sets, e.g., SRM and maxSRMd2. For resampled images, the better choice is BOSS+BOWS2. But for cropped and subsampled images, there is no clear difference in performance on BOSS and BOSS+BOWS2. The experiments for SRM and maxSRMd2 on BOSS+BOWS2+AUG are pointless because these features are already symmetrized. The rotated or mirrored images will cause duplicated features and singular matrices in the base FLD learners of the ensemble classifer. Therefore, for the image dataset adopted in this paper and in the interest of fairness, the training set for SRM and maxSRMd2 is BOSS+BOWS2, while our CNN models use the BOSS+BOWS2+AUG. And the performance of all involved schemes is evaluated on BOSS_test described above. For repeating the experiments, we create three different training sets and test sets. Every CNN model will be trained independently on the three training sets using the same hyper-parameters and tested on their associated test sets. Then the test results are averaged as the final performance of this model.

C. Implementation Details

We implemented the proposed CNN models using Caffe [29] with necessary modifications. It is worth noting that, instead of SGD, we use AdaDelta [30] to train our networks as we found in our early experiments that with AdaDelta the networks can learn much faster and achieve better results. Accordingly, all the following parameters we described are based on AdaDelta: the mini-batch size is 32, which contains 16 pairs of cover and stego images; the momentum value is

0.95 and the weight decay is 5×10^{-4}; the "delta" parameter for AdaDelta is 1×10^{-8}. Data augmentation is conducted during training and the same rotated or mirrored operation is applied to a pair of images within a mini-batch. "Xavier" initialization [31] is used to initialize the weights from 2nd to 9th layers and their initial biases are set to be 0.2. The last fully-connected layer is initialized with random values obtained from a Gaussian source of zero mean and standard deviation 0.01 and the initial bias is set to be zero. Based on the above settings, the networks are then trained to minimize the cross-entropy loss.

During training, we use the "multistep" policy in Caffe to adjust the learning rate. When the training iteration is equal to one of the specified step values, the learning rate will be divided by 5. Take TLU-CNN for WOW at 0.2 bpp on resampled images as an example, with an initial value of 0.4, the learning rate will be decreased to 0.08, 0.016 and 0.0032 at iterations 500,000, 600,000 and 650,000 respectively[1]. Note that with different embedding schemes at different payloads, we are actually training the CNNs for tasks of varying difficulties, which means that different configurations should be applied to control the learning rate. Owing to space constraints, it is impractical for us to give all the step values for each involved CNN models. As an alternative, we turn to elaborate the rules on how to determine those step values. The key to tackling the problem lies in monitoring the error and accuracy on the validation set during training. When neither the error decreases nor the accuracy increases, the learning rate should be changed. A similar policy was also adopted in [9] and [32]. Note that for each model, we create three different training/validation/test sets using the way described in Section IV-B and choose the step values for this model from the first trining/validation set under the rules described above. The same step values are used when training on the other two training sets.

For TLU-CNN models corresponding to resampled and cropped images at payloads from 0.2 to 0.5 bpp and subsampled images at 0.8 bpp, the network parameters are trained from scratch and are stopped at 100 epochs with an initial learning rate of 0.4. The strategy of curriculum learning is applied to train the models corresponding to other stego images at lower embedding rates, to be specific, the resampled and cropped images at payloads from 0.05 to 0.1 bpp, and the subsampled images at payloads from 0.05 to 0.5 bpp, where the networks are fine-tuned from their previous trained ones with an initial learning rate of 0.05. All the fine-tuning procedures will be stopped at 35 epochs except for the training of subsampled images at 0.5 bpp, which will be stopped at 70 epochs.

D. Comparison with the state-of-the-art Steganalyzers in Spatial Domain

In this subsection, we compare the performance of the proposed CNN-based models with two state-of-the-art steganalyzers in spatial domain, i.e., SRM and maxSRMd2, for a wide variety of payloads. Tables VIII-X show the performance comparison in terms of detection error (P_E) for all the tested schemes on resampled, cropped and subsampled images. In Fig. 5 to Fig. 7, we further illustrate the detection errors of three state-of-the-art steganographic schemes in spatial domain, i.e., WOW, S-UNIWARD and HILL, as a function of payload (ranging from 0.05 bpp to 0.5 bpp), for all the involved steganalyzers on all the 3 image datasets.

It is observed in Fig. 5 to Fig. 7 that, the proposed TLU-CNN and SCA-TLU-CNN consistently outperform the other two hand-crafted rich models by a clear margin, irrespective of the embedding method, payload and image dataset (resampled, cropped and subsampled). On one hand, when the knowledge of selection channel is not incorporated, our TLU-CNN model can achieve significant performance gains over SRM for the involved embedding schemes, tested payloads and image datasets. This is particularly evident for resampled and subsampled images. For instance, in contrast to SRM, the TLU-CNN decreases the detection error of WOW by 12.46% on the resampled images when payload is 0.2 bpp as shown in Fig. 5 (a). It is also shown in Fig. 6 that the performance gain of TLU-CNN decreases somewhat for cropped images. This is because the cropped central images are usually the most complex regions of the original images, which makes the detection of stego images more difficult for both CNN based and hand-crafted steganalyzers. On the other hand, for those selection-channel-aware schemes, our SCA-TLU-CNN model also convincingly outperforms the maxSRMd2 as shown in Fig. 5 to Fig. 7, and the performance gap becomes most pronounced for S-UNIWARD at 0.3 bpp on the resampled images, where the detection error is decreased by 10.4%. It is believed that the regularization for initialization with the high-pass filters in SRM, the use of TLU non-linearity and the unified optimization framework of the CNN model contribute much to the superior performance of CNN based staganalyzers over the conventional heuristic feature sets.

It is worth noticing that, although the TLU-CNN does not explicitly take advantage of the selection channel, it still defeats the selection-channel-aware maxSRMd2 algorithm in most cases. This surprising result shows that the proposed TLU-CNN is able to learn the distribution of selection channel for a specific embedding scheme implicitly, if it is trained on a sufficient large and diverse training set. It may explain the reason why the SCA-TLU-CNN does not outperform much its non-selection-channel-aware version TLU-CNN. It is also observed that, although the SCA-TLU-CNN consistently works better than TLU-CNN, the performance tends to be more and more similar at high payloads, especially for WOW and S-UNIWARD on resampled images as shown in Fig. 5 (a)-(b). This is because, with the increase of data payload, both the WOW and S-UNIWARD become less adaptive, and therefore, the SCA-TLU-CNN could not exemplify its advantage in the knowledge of selection channel. For HILL, however, the adoption of a series of filtering operations allows it to exhibit some "adaptivity" at different payload, which contributes to the superior performance of SCA-TLU-CNN over TLU-CNN at high data payload, especially on resampled and subsampled images.

[1] These values correspond to the repeated "stepvalue" in Caffe when using "multistep" policy.

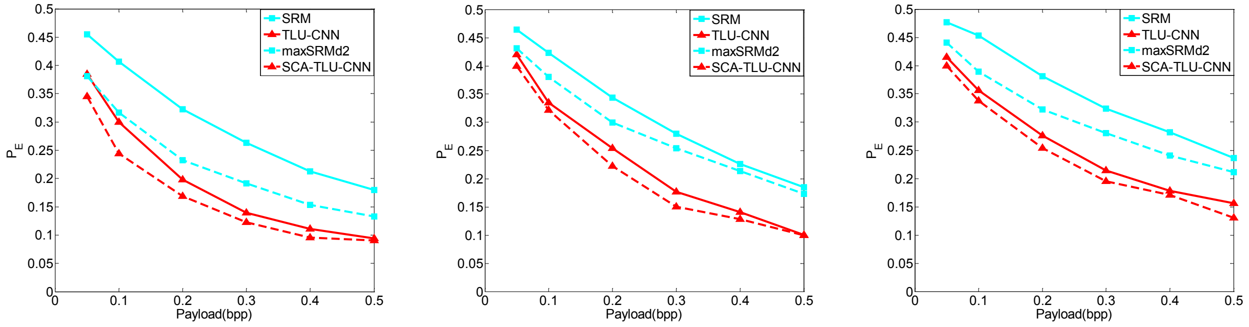
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2017.2710946, IEEE Transactions on Information Forensics and Security

TABLE VIII
PERFORMANCE COMPARISON OF THE INVOLVED STEGANALYZERS IN TERMS OF DETECTION ERROR (P_E) FOR 3 STATE-OF-THE-ART STEGANOGRAPHIC SCHEMES AT DIFFERENT PAYLOADS ON RESAMPLED IMAGES.

Algorithm	Payload (bpp)	SRM (P_E)	TLU-CNN (P_E)	maxSRMd2 (P_E)	SCA-TLU-CNN (P_E)
WOW	0.05	0.4551	0.3850	0.3810	**0.3450**
	0.1	0.4066	0.3000	0.3163	**0.2442**
	0.2	0.3228	0.1982	0.2325	**0.1691**
	0.3	0.2633	0.1394	0.1918	**0.1229**
	0.4	0.2127	0.1109	0.1536	**0.0959**
	0.5	0.1800	0.0938	0.1331	**0.0906**
S-UNIWARD	0.05	0.4641	0.4200	0.4316	**0.4000**
	0.1	0.4232	0.3350	0.3806	**0.3220**
	0.2	0.3437	0.2540	0.2999	**0.2224**
	0.3	0.2798	0.1772	0.2542	**0.1502**
	0.4	0.2260	0.1410	0.2136	**0.1281**
	0.5	0.1848	0.1003	0.1732	**0.1000**
HILL	0.05	0.4765	0.4150	0.4409	**0.4000**
	0.1	0.453	0.3560	0.3894	**0.3380**
	0.2	0.3811	0.2761	0.3226	**0.2538**
	0.3	0.3236	0.2145	0.2804	**0.1949**
	0.4	0.2818	0.1782	0.2410	**0.1708**
	0.5	0.2363	0.1561	0.2115	**0.1305**

TABLE IX
PERFORMANCE COMPARISON OF THE INVOLVED STEGANALYZERS IN TERMS OF DETECTION ERROR (P_E) FOR 3 STATE-OF-THE-ART STEGANOGRAPHIC SCHEMES AT DIFFERENT PAYLOADS ON CROPPED IMAGES.

Algorithm	Payload (bpp)	SRM (P_E)	TLU-CNN (P_E)	maxSRMd2 (P_E)	SCA-TLU-CNN (P_E)
WOW	0.05	0.4772	0.4139	0.4199	**0.3874**
	0.1	0.4460	0.3488	0.3730	**0.3240**
	0.2	0.3853	0.2808	0.3092	**0.2435**
	0.3	0.3337	0.2450	0.2686	**0.2036**
	0.4	0.2887	0.2044	0.2361	**0.1707**
	0.5	0.2496	0.1680	0.2041	**0.1445**
S-UNIWARD	0.05	0.4750	0.4460	0.4571	**0.4390**
	0.1	0.4439	0.4040	0.4206	**0.3938**
	0.2	0.3823	0.3318	0.3614	**0.3218**
	0.3	0.3287	0.2850	0.3132	**0.2571**
	0.4	0.2805	0.2374	0.2721	**0.1955**
	0.5	0.2411	0.1959	0.2355	**0.1660**
HILL	0.05	0.4845	0.4540	0.4536	**0.4325**
	0.1	0.4618	0.4129	0.4211	**0.3806**
	0.2	0.4129	0.3494	0.3638	**0.3288**
	0.3	0.3645	0.3018	0.3253	**0.2885**
	0.4	0.3236	0.2470	0.2874	**0.2291**
	0.5	0.2810	0.2100	0.2520	**0.1977**

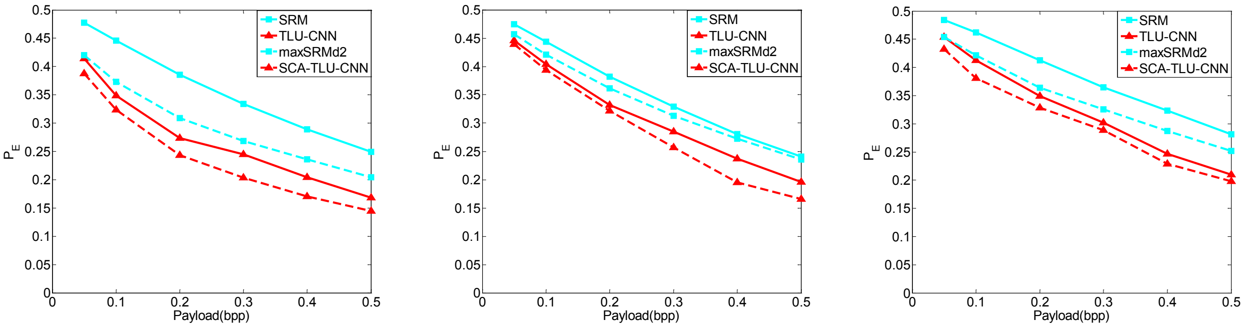


(a) (b) (c)

Fig. 5. Detection errors P_E of 3 state-of-the-art steganographic schemes as a function of payload for the involved steganalysis methods.Images are resized to 256×256. (a) WOW. (b) S-UNIWARD. (c) HILL.

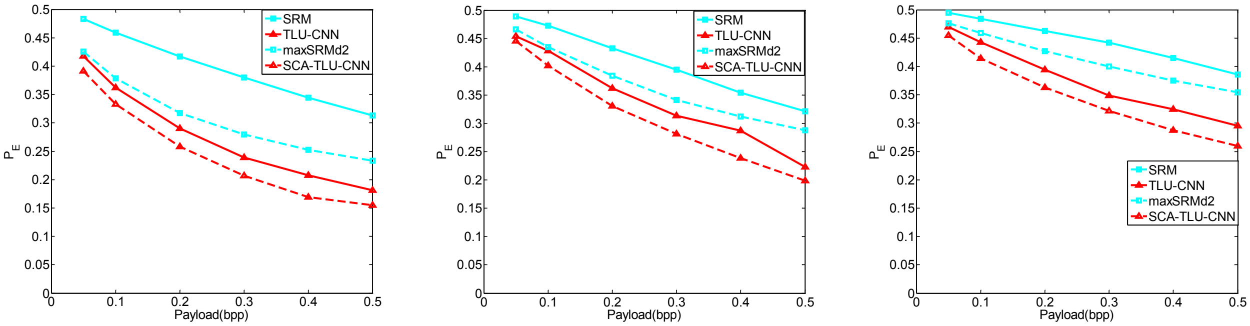
TABLE X
PERFORMANCE COMPARISON OF THE INVOLVED STEGANALYZERS IN TERMS OF DETECTION ERROR (P_E) FOR 3 STATE-OF-THE-ART STEGANOGRAPHIC SCHEMES AT DIFFERENT PAYLOADS ON SUBSAMPLED IMAGES.

Algorithm	Payload (bpp)	SRM (P_E)	TLU-CNN (P_E)	maxSRMd2 (P_E)	SCA-TLU-CNN (P_E)
WOW	0.05	0.4831	0.4176	0.4254	**0.3916**
	0.1	0.4592	0.3622	0.3788	**0.3333**
	0.2	0.4171	0.2900	0.3176	**0.2585**
	0.3	0.3797	0.2391	0.2796	**0.2070**
	0.4	0.3443	0.2077	0.2523	**0.1691**
	0.5	0.3132	0.1812	0.2335	**0.1547**
S-UNIWARD	0.05	0.4893	0.4541	0.4662	**0.4452**
	0.1	0.4722	0.4283	0.4347	**0.4020**
	0.2	0.4323	0.3618	0.3842	**0.3307**
	0.3	0.3949	0.3137	0.3416	**0.2814**
	0.4	0.3544	0.2872	0.3120	**0.2387**
	0.5	0.3213	0.2226	0.2881	**0.1988**
HILL	0.05	0.4948	0.4697	0.4761	**0.4551**
	0.1	0.4840	0.4430	0.4592	**0.4140**
	0.2	0.4629	0.3940	0.4269	**0.3632**
	0.3	0.4416	0.3490	0.3998	**0.3216**
	0.4	0.4146	0.3245	0.3747	**0.2877**
	0.5	0.3859	0.2950	0.3541	**0.2596**



(a) (b) (c)

Fig. 6. Detection errors P_E of 3 state-of-the-art steganographic schemes as a function of payload for the involved steganalysis methods.Images are cropped into 256×256. (a) WOW. (b) S-UNIWARD. (c) HILL.



(a) (b) (c)

Fig. 7. Detection errors P_E of 3 state-of-the-art steganographic schemes as a function of payload for the involved steganalysis methods. Images are subsampled to 256×256. (a) WOW. (b) S-UNIWARD. (c) HILL.

V. Conclusion

The paradigm of modern steganalyzer mainly consists of three steps, i.e., residual computation, feature extraction and binary classification. In this paper, we propose a CNN based steganalyzer, which is shown to be able to well simulate and optimize these key steps in a unified network architecture. The proposed CNN has a quite different structure compared to the ones designed for CV tasks, and is capable of detecting several state-of-the-art steganographic schemes in spatial domain for a wide variety of payloads with high accuracy. Instead of a random strategy, the weights in the first layer of the proposed CNN are initialized with the basic high-pass filters used in computation of residual maps in SRM, which helps to find a better local minima as a regularizer. Considering that the embedding signals usually have an extremely low SNR, a set of hybrid activation functions is adopted in our CNN model, where, in addition to the conventional ReLU function, a new function called truncated linear unit (TLU) is introduced to the first few layers of our network to well adapt to the distribution of the embedding signals. And finally, the performance of the proposed CNN is further boosted by incorporating the knowledge of selection channel. Extensive experiments have been carried out, which demonstrates the superior performance of the proposed CNN based steganalyzer over other state-of-the-art steganalysis methods.

Acknowledgment

The authors would like to thank the anonymous reviewers and associate editor for their comments that greatly improved the paper.

References

[1] T. Pevný, T. Filler, and P. Bas, “Using high-dimensional image models to perform highly undetectable steganography,” in *International Workshop on Information Hiding*, 2010, pp. 161–177.

[2] V. Holub and J. Fridrich, “Designing steganographic distortion using directional filters,” in *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2012, pp. 234–239.

[3] 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.

[4] 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.

[5] V. Holub and J. Fridrich, “Random projections of residuals for digital image steganalysis,” *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 12, pp. 1996–2006, 2013.

[6] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, “Selection-channel-aware rich model for steganalysis of digital images,” in *International Workshop on Information Forensics and Security (WIFS)*, 2014, pp. 48–53.

[7] W. Tang, H. Li, W. Luo, and J. Huang, “Adaptive steganalysis against wow embedding algorithm,” in *Proceedings of the 2nd ACM workshop on Information hiding and multimedia security*, 2014, pp. 91–96.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[10] O. M. Parkhi, A. Vedaldi, and A. Zisserman, “Deep face recognition,” in *British Machine Vision Conference*, vol. 1, no. 3, 2015, p. 6.

[11] Y. Qian, J. Dong, W. Wang, and T. Tan, “Deep learning for steganalysis via convolutional neural networks,” in *SPIE/IS&T Electronic Imaging*, 2015, pp. 94 090J–94 090J.

[12] P. Bas, T. Filler, and T. Pevný, “ break our steganographic system: The ins and outs of organizing boss,” in *International Workshop on Information Hiding*, 2011, pp. 59–70.

[13] 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.

[14] G. Xu, H.-Z. Wu, and Y. Q. Shi, “Ensemble of cnns for steganalysis: An empirical study,” in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, 2016, pp. 103–107.

[15] T. Pevný, P. Bas, and J. Fridrich, “Steganalysis by subtractive pixel adjacency matrix,” *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 215–224, 2010.

[16] D. Zou, Y. Q. Shi, W. Su, and G. Xuan, “Steganalysis based on Markov model of thresholded prediction-error image,” in *2006 IEEE International Conference on Multimedia and Expo*, 2006, pp. 1365–1368.

[17] S. Lyu and H. Farid, “Detecting hidden messages using higher-order statistics and support vector machines,” in *International Workshop on Information Hiding*, 2002, pp. 340–354.

[18] I. Avcibas, N. Memon, and B. Sankur, “Steganalysis using image quality metrics,” *IEEE transactions on Image Processing*, vol. 12, no. 2, pp. 221–229, 2003.

[19] M. Goljan, J. Fridrich, and T. Holotyak, “New blind steganalysis and its implications,” in *SPIE/IS&T Electronic Imaging*, 2006, pp. 607 201–607 201.

[20] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International Conference on Machine Learning*, 2015.

[21] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. Lecun, “What is the best multi-stage architecture for object recognition?” in *International Conference on Computer Vision*, 2009, pp. 2146 – 2153.

[22] 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 sourcemismatch,” *Electronic Imaging*, vol. 2016, no. 8, pp. 1–11, 2016.

[23] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” *Journal of Machine Learning Research*, vol. 15, 2011.

[24] A. Kerckhoffs, “La cryptographie militaire,” *Journal Des Sciences Militaires*, vol. IX, pp. 5–83, 1883.

[25] T. Denemark, J. Fridrich, and P. Comesaña-Alfaro, “Improving selection-channel-aware steganalysis features,” *Electronic Imaging*, vol. 2016, no. 8, pp. 1–8, 2016.

[26] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[27] Y. Qian, J. Dong, W. Wang, and T. Tan, “Learning and transferring representations for image steganalysis using convolutional neural network,” in *IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 2752–2756.

[28] P. Bas and T. Furon, “Bows-2,” http://bows2.gipsa-lab.inpg.fr, July 2007.

[29] 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 preprint arXiv:1408.5093*, 2014.

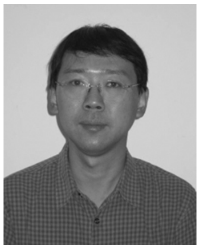
[30] M. D. Zeiler, “Adadelta: An adaptive learning rate method,” *Computer Science*, 2012.

[31] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks.” in *Aistats*, vol. 9, 2010, pp. 249–256.

[32] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

Jian Ye receved the B.S. degree in School of Information Science and Technology, Sun Yat-Sen University, Guangzhou, China, in 2014. He is currently pursuing the M.S. degree with the School of Electronics and Information Technology, Sun Yat-Sen University, Guangzhou, China.

His current research interests include image steganalysis and machine learning.



Jiangqun Ni (M'12) received the Ph.D. degree in electronic engineering from The University of Hong Kong, Hong Kong, in 1998.

He was a Post-doctoral Fellow for a joint program between the Sun Yat-Sen University and the Guangdong Institute of Telecommunication Research from 1998 to 2000. Since 2001, he has been with the School of Data and Computer Science, Sun Yat-Sen University, Guangzhou, China, where he is currently a Professor. His research interests include data hiding and digital image forensics, image-based modeling and rendering, and image/video processing. He has authored over 50 papers in these areas.

Yang Yi received the B.Sc. degree in Electronic Engineering from Fudan University, and the M.Sc. and Ph.D. degrees in Machine Learning and Optimization from Northeastern University, respectively.

From September 2005 to August 2006, she was a visiting scholar with the Computer Science Department, Eastern Washington University, USA. Currently, she is with the School of Data and Computer Science, Sun Yat-Sen University. Meanwhile, she is also a Fellow of Guangdong Province Key Laboratory of Big Data Analysis and Processing, China. She has published over 80 papers in international conferences or journals. And her current research interests include computer vision and digital image processing by machine learning and pattern recognition.