A Novel Convolutional Neural Network for Image Steganalysis With Shared Normalization

Songtao Wu, Sheng-hua Zhong, Member, IEEE, and Yan Liu

Abstract—Image steganalysis is to discriminate innocent images (cover images) and those suspected images (stego images) with hidden messages. The task is challenging since modifications to cover images due to message hiding are extremely small. To handle this difficulty, modern approaches proposed using convolutional neural network (CNN) models to detect steganography with paired learning, i.e., cover images and their stegos are both in training set. In this paper, we explore an important technique in CNN models, the batch normalization (BN), for the task of image steganalysis in the paired learning framework. Our theoretical analysis shows that a CNN model with multiple batch normalization layers is difficult to be generalized to new data in the test set when it is well trained with paired learning. To address this problem, we propose a novel normalization technique called shared normalization (SN) in this paper. Unlike the BN layer utilizing the mini-batch mean and standard deviation to normalize each input batch, SN shares consistent statistics for training samples. Based on the proposed SN layer, we further propose a novel neural network model for image steganalysis. Extensive experiments demonstrate that the proposed network with SN layers is stable and can detect the state-of-the-art steganography with better performances than previous methods.

Index Terms—Steganalysis, steganography, convolutional neural network, batch normalization, shared normalization.

I. INTRODUCTION

STEGANOGRAPHY is the technique to hide secret messages into multimedia signals such as audio, image and video [1], [2], or some special signals including high dynamic range image [3] and encrypted image [4]. Steganalysis, from an opponent’s perspective, is the art of revealing the presence of secret messages embedded in the digital medias [5]. Among all steganalytic techniques, image steganalysis plays an important role in many information security systems and attracts increasing interests in recent years [6]–[8].

Designing effective features that are sensitive to message embedding is key to image steganalysis [9]–[11]. Traditional feature-based methods, Spatial Rich Model (SRM) [9] steganalysis and its selection-channel version [10], assemble various handcrafted features and learn a binary classifier to detect steganography. However, designing effective features turns out to be a difficult task which needs strong domain knowledge of steganography and steganalysis. Recently, several interesting works have been proposed to detect steganography based on deep convolutional neural networks [12]–[22]. Compared with traditional methods that extract handcrafted features, CNN based steganalysis automatically learns effective features using various network architectures for discriminating cover images and stego images. Tan and Li [12] first proposed to detect the presence of secret messages based on a deep stacked convolutional auto-encoder network. Qian et al. [13] proposed a model for steganalysis using the standard CNN architecture with Gaussian activation function and further proved that transfer learning is beneficial for a CNN model to detect a steganographic algorithm at low payloads in [14]. Xu et al. [15] designed a compact and effective CNN architecture with multiple batch normalization layers. This network now becomes a basic model for several more complex CNN models [16]–[18]. Wu et al. [19], [20] proposed a novel CNN model to detect steganography based on residual learning and achieved low detection error rates when cover images and stego images are paired. Ye et al. in [21] firstly incorporated the truncation technique into the design of steganalystic CNN models. With the help of selection channel knowledge and data augmentation, their model obtained significant performance improvements than the classic SRM on resampled and cropped images. Except for the spatial domain steganalysis, Xu et al. [22] proposed a network based on residual learning and achieved better detection accuracy than traditional methods in compressed domain.

Training deep neural network is difficult since data at different layers follow different distributions [23]. Batch normalization is a standard technique widely used in many CNN models [24], [25], which can effectively handle this difficulty. By normalizing input data with mini-batch statistics, elements in feature maps are forced to be distributed with zero mean and unit standard deviation. Batch normalization not only reduces the training time of CNN models significantly, but also regularizes the network implicitly. As an important component of a CNN model, batch normalization has shown promising performances in different
image related tasks, such as classification [24], image denoising [25], and image captioning [26].

Steganalysis is different from natural image classification. It is challenging since steganalysis aims to discriminate cover images and stego images which are the results of adding weak noise into covers to form stego signals. This characteristic leads to the result that a cover image is more statistically similar to its stego image than other cover images. In order to learn discriminative features between cover images and stego images, existing steganalytic methods [9], [10] often use paired training, i.e. cover images and their stegos should be both in the training set, for steganalytic feature extraction. For deep networks [12]–[22], they use pairs of images in a mini-batch for feature learning. In this paper, we analyze how batch normalization affects the performance of CNN-based image steganalysis. Our theoretical analysis shows that the batch normalization can capture the characteristic of paired learning, which makes the model fall into a dilemma. For one hand, a CNN model with batch normalization layers can detect steganography accurately once training batch and test batch have consistent statistical properties, i.e. cover images and their stegos are paired both in training set and test set. However, the requirement of paired learning and paired testing is too restrict to be satisfied in real applications. For the other hand, the model trained with paired learning is hard to be generalized to the test data once cover images and their stegos are not paired in a test batch.

To address the limitation of batch normalization for image steganalysis, a novel normalization technique called Shared Normalization (SN) is proposed. Unlike the batch normalization that uses mini-batch mean and standard deviation to normalize the data, SN shares consistent statistics for training samples. With new SN layer, CNN models are forced to learn features to discriminate cover images and stego images, thus improves the model’s generalization ability. Based on the proposed SN technique, we further introduce a novel CNN model\footnote{The code to train and test our model: https://github.com/Steganalysis-CNN/CNN-without-BN.} to detect steganography. Our CNN model recursively uses a processing unit, which contains a convolutional layer, a SN layer, a ReLU layer and an average pooling layer, to extract effective features for image steganalysis. Experiments demonstrate that the proposed CNN model can not only learn more effective features compared with classical CNN architecture, but also detect steganographic algorithms with better performance than SRM and other CNN models.

The rest of this paper is organized as follows. In Section II, we introduce the preliminary knowledge about convolutional neural network and batch normalization. In Section III, we introduce the importance of paired learning for image steganalysis. In Section IV, we analyze the limitation of batch normalization for image steganalysis in theory. To overcome the shortcoming of batch normalization, we propose the shared normalization in Section V and discuss their main differences. In Section VI, we further introduce a novel convolutional neural network model based on the proposed shared normalization layer for image steganalysis. In Section VII, we validate the effectiveness of the proposed model on several state-of-the-art steganographic algorithms. The paper is finally closed with the conclusion in Section VIII.

II. PRELIMINARIES

A. Rationality of Using Convolutional Neural Network for Image Steganalysis

In recent years, CNN has achieved great success in many image related tasks. A series of breakthroughs have been made for discriminative learning, including image classification [24], [27], [28], image denoising [25], and image super-resolution [29]. In addition, several recent work show that CNN models are successfully applied for generative learning, including real image generation [30] and texture synthesis [31]. These successes indicate that CNN can not only extract effective features for discriminating different images but also provide a good description for representing real images. All the evidences show that CNN can well describe the distribution of natural images. These results motivate us to use CNN for image steganalysis, since its purpose is to discriminate the “natural” images (cover images) against the “unnatural” images contaminated by embedded secret messages (stego images). The following three characteristics of CNN models further demonstrate that they are suitable for the task of image steganalysis:

- Convolutional kernels in CNN models can exploit the strong spatially local correlation present in input images [32]. This local correlation among image pixels is distorted when secret messages are embedded, making it different from the correlation in natural images. The difference between natural images and distorted images can be effectively captured by CNN models;
- The convolution operation is actually to sum image pixels in a local region, which would accumulate the weak embedded signal of this region to be a large value. This may lead to stego images be more easily detected against cover images;
- Nonlinear mappings in CNN models make them able to extract rich features for classifying cover images and stego images. These features, which are automatically learned by updating the network, can hardly be designed by hand.

B. Batch Normalization for Convolutional Neural Network

Batch normalization is a standard technique that is widely used in CNN models for image classification [23]. Training a deep neural network model is often difficult not only because of the gradient vanishing/exploding but also because the distribution of data changes in different layers, which is called the “internal covariate shift” phenomenon. Batch normalization is such a technique that can relieve this phenomenon. Before feeding the data to the next processing, batch normalization normalizes the data by introducing several operations as follows:

$$\mu_g \leftarrow \frac{1}{m} \sum_{i=1}^{m} I_i$$  \hspace{1cm} (1)
where $I_i$ denotes the $i$-th training sample, $m$ is the number of samples in the batch, $B = \{I_1,...,m\}$ denotes the input data in a mini-batch, $\mu_B$ and $\sigma_B$ represents the mean and standard deviation of the mini-batch $B$ respectively. $\epsilon$ is a small constant to avoid zero dividing, $\gamma$ and $\beta$ are the parameters. With these operations, the output data $\hat{I}_i$ in the mini-batch is distributed with fixed mean and standard deviation at any depth after the batch normalization. Thus, deviations to the mean and variance can be eliminated by the batch normalization, which makes the network overcome the “internal covariate shift”.

III. PAIRED TRAINING FOR IMAGE STEGANALYSIS

Steganalysis is usually formulated as a binary classification problem [9], [10]. This technique, which is called “universal/blind steganalysis”, becomes the main stream among most current steganalytic algorithms. Fig. 1 illustrates the process of universal steganalysis. In the training phase, effective features sensitive to message embedding are extracted to highlight potential manipulation by steganographer. Then, a binary classifier is learned based on pairs of images, i.e. cover images and their corresponding stegos, which aims to find a boundary to detect steganography. In testing phase, the trained classifier is used to predict labels of new input images. Previous research [34] showed that, in the training phase, it is rather important to force cover features and stego features to be paired, i.e. steganalytic features of cover images and their stego images should be preserved in the training set. Otherwise, breaking cover-stego pairs in different sets would introduce biased error and lead to a suboptimal performance [35]. To demonstrate the importance of paired training for image steganalysis, we compare the statistics of pixels in secret messages and cover images after high pass filtering.\(^2\) The size of testing image is 100 and the Spatial UNIversal W Avelet Relative Distortion (S-UNIWARD) [38] steganography is used for validation.

Table I shows that both mean values of secret messages and cover images are very small, almost equal to zeros. However, the variance of cover image, 37.89, is much larger than that of secret message, 0.31. This indicates that the amplitude of embedded secret messages is very small compared to cover images. Consequently, when cover images and their stegos are not paired in training set/batch, it is rather difficult for a model to learn effective features to detect steganography since cover images drown out secret messages. However, when cover images and their stegos are paired in training set/batch, steganalysis becomes easier because cover images are similar to their stegos in appearance and the variance introduced by cover images is reduced. In this case, the model focuses on learning effective features to capture the difference between a cover image and its stego, which is the purpose of image steganalysis. For these reasons, either traditional hand-craft feature based methods [9], [10] or deep learning based methods [12]–[22] in the mainstream image steganalysis schemes use paired training.

IV. LIMITATION OF BATCH NORMALIZATION FOR IMAGE STEGANALYSIS

In this section, we analyze the limitation of BN for image steganalysis in the scheme of paired learning. Firstly, we describe the behavior of the BN layer for steganalysis when cover images and their stego versions are paired both in training set and testing set. Secondly, we explain why the performance of deep network with BN layers degrades when the model is trained with paired samples but tested in unpaired manner. Finally, we analyze the real case that a deep network with BN layers is trained with paired samples but tested with fixed BN parameters (these BN parameters are learned in the training phase but fixed in testing).

TABLE I

| Statistics         | Secret message | Cover image |
|--------------------|----------------|-------------|
| Mean               | $-2.86 \times 10^{-8} \approx 0$ | $-2.41 \times 10^{-4} \approx 0$ |
| Variance           | 0.31           | 37.89       |

\(^2\)High pass filtering is a necessary step for image steganalysis. The purpose is to enlarge the signal to noise ratio between secret message and cover image.
and the batch statistics are used to normalize the data. To illustrate this phenomenon, we provide mathematical analysis here. Assume the network is fed with a batch which only contains a cover image $x$ and its stego image $y$:

$$y = x + s$$  \hspace{1cm} (5)$$

where $s$ denotes the embedded signal. For the block “Conv+BN+ReLU”, the output of the cover image is:

$$x^{op} = \text{ReLU} \left( \frac{Wx - \mu}{\sigma} \right) = \left[ \frac{Wx - \mu}{\sigma} \right] \circ \mathcal{H} \left[ \frac{Wx - \mu}{\sigma} \right]$$  \hspace{1cm} (6)$$

where $x^{op}$ represents the output of cover image in paired case. $\circ$ represents the pointwise product. $W$ is the weights of the convolutional kernel. $\mu$ denotes the mean value of all pixels in $x$ and $y$, $\sigma$ represents its standard deviation, where $\mu$ can be written as:

$$\mu = \frac{1}{E} \left[ Wx + Wy \right] = E \left[ Wx \right] + \frac{1}{2} E \left[Ws\right]$$  \hspace{1cm} (7)$$

where $E[\cdot]$ denotes the expectation operator. In Eq.(6), $\mathcal{H}(\cdot)$ represents Heaviside step function:

$$\mathcal{H}(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$  \hspace{1cm} (8)$$

For simplicity, we have omitted the bias term and the scaling term in the BN layer. Similarly, the output of stego image $y^{op}$ is:

$$y^{op} = \text{ReLU} \left( \frac{Wy - \mu}{\sigma} \right) = \left[ \frac{W(x + s) - \mu}{\sigma} \right] \circ \mathcal{H} \left[ \frac{W(x + s) - \mu}{\sigma} \right]$$  \hspace{1cm} (9)$$

For Eq.(6) and Eq.(9), we consider the expectation of batch normalization layer’s outputs:

$$E \left[ \frac{Wx - \mu}{\sigma} \right] = E \left[ \frac{Wx - E[Wx] + \frac{1}{2} E[Ws]}{2\sigma} \right] = -\frac{E[Ws]}{2\sigma}$$  \hspace{1cm} (10)$$

$$E \left[ \frac{Wy - \mu}{\sigma} \right] = E \left[ \frac{Wy - E[Wy] + \frac{1}{2} E[Ws]}{2\sigma} \right] = \frac{E[Ws]}{2\sigma}$$  \hspace{1cm} (11)$$

Eq.(10) and Eq.(11) indicate that the feature map of $x$ and the feature map of $y$ are distributed across 0 on average. This property makes elements in the cover feature map and elements in the stego feature map be automatically separated after the ReLU layer, once the $W$ has been optimized to discriminate cover images and their stegos. Consequently, the classification of cover images and stego images becomes easy for a CNN model with “Conv+BN+ReLU” block.

For a network with many “Conv+BN+ReLU” blocks, cover images and stego images are also easily classified when they are paired in a batch. In the paired learning case, the output of a cover image and its stego image is:

$$x^{op}_n = f^{pc}_n \circ \mathcal{H}[f^{pc}_n]$$  \hspace{1cm} (12)$$

where $x^{op}_n$ and $y^{op}_n$ represents the output of cover image and stego image after $n$-th “Conv+BN+ReLU” blocks respectively. $f^{pc}_n$ and $f^{ps}_n$ represents the feature map of cover image and stego image after the batch normalization layer:

$$f^{ps}_n = \frac{1}{\sigma_n} \left[ W_n x^{op}_{n-1} - \frac{1}{2} E \left[ W_n x^{op}_{n-1} + W_n y^{op}_{n-1} \right] \right]$$  \hspace{1cm} (14)$$

$$f^{ps}_n = \frac{1}{\sigma_n} \left[ W_n y^{op}_{n-1} - \frac{1}{2} E \left[ W_n x^{op}_{n-1} + W_n y^{op}_{n-1} \right] \right]$$  \hspace{1cm} (15)$$

where $W_n$ denotes the $n$-th convolution kernel and $\sigma_n$ is the standard deviation. By introducing a variable which is defined as:

$$s^{op}_{n-1} = y^{op}_{n-1} - x^{op}_{n-1}$$  \hspace{1cm} (16)$$

Obviously, $s^{op}_{n-1}$ represents the embedded signal at $n-1$ layer, which is introduced by $s$. For Eq.(14) and Eq.(15), we take expectation to $f^{pc}_n$ and $f^{ps}_n$ and obtain:

$$E[f^{pc}_n] = -\frac{E[W_n s^{op}_{n-1}]}{2\sigma_n}$$  \hspace{1cm} (17)$$

$$E[f^{ps}_n] = \frac{E[W_n s^{op}_{n-1}]}{2\sigma_n}$$  \hspace{1cm} (18)$$

Similar to Eq.(10) and Eq.(11), elements in the feature map of the cover image and the stego image are distributed across 0, which can be easily classified after the ReLU layer.

To demonstrate the fact that a CNN model with many BN layers can detect steganography easily when cover images and their stego are paired in a batch, we use the residual network proposed in [20] and S-UNIWARD at 0.4 bpp are used for demonstration. The batch statistics are used to normalize input data both in training and testing.
the detection errors when cover images and their stego images are paired both in training and testing.

In summary, we analyze that a CNN model with multiple BN layers can detect steganography at high accuracy when covers and their stegos are paired in training and testing. However, cover images and their stegos should be paired in testing is a strict assumption that might not be satisfied in real applications. Hence, in the next subsection, we will analyze the behavior of the CNN model with batch normalization layers when testing samples are not paired.

B. Paired Training and Unpaired Testing for a CNN Model With Multiple Batch Normalization Layers

In this subsection, we analyze the case that the network with BN layers is trained by paired learning but tested with unpaired cover/stego samples. The following analysis demonstrates that the network can hardly predict the label of unpaired images if the batch statistics is used to normalize the data. Assume the block is fed with a cover image \(x\) and a stego image \(y'\), where \(y' = x' + s\) and \(x' \neq x\). For the “Conv+BN+ReLU” block, the outputs of \(x\) and \(y'\) are:

\[
\begin{align*}
\hat{x} &= \frac{Wx - \mu}{\sigma} \circ H \left[ \frac{Wx - \mu}{\sigma} \right] \\
y' &= \frac{W(x' + s) - \mu}{\sigma} \circ H \left[ \frac{W(x' + s) - \mu}{\sigma} \right]
\end{align*}
\]

where \(\hat{x}\) and \(y'\) denotes the output of cover image and stego image in unpaired case respectively, \(\mu\) and \(\sigma\) represents the mean and the standard deviation. \(\mu\) is:

\[
\mu = \frac{1}{2} E(\hat{x})
\]

Similar to Eq.(10) and Eq.(11), we take expectation to the output of batch normalization layer in the unpaired case and obtain:

\[
\begin{align*}
E \left[ \frac{Wx - \mu}{\sigma} \right] &= \frac{1}{2\sigma} \left[ E \left( \frac{W(x - x') - 1}{2} Ws \right) \right] \\
E \left[ \frac{Wx' - \mu}{\sigma} \right] &= \frac{1}{2\sigma} \left[ E \left( \frac{W(x' - x) + 1}{2} Ws \right) \right]
\end{align*}
\]

Compared with the paired learning case, on average, the expected output of BN layer in Eq.(22) and Eq.(23) not only depends on \(Ws\) but also on \(W(x - x')\). Fig. 3 has shown the distribution of elements in \(W(x - x')\) and \(Ws\). It is observed that the amplitude of secret message \(Ws\) is much smaller than \(W(x - x')\). Therefore, in unpaired case, the output of the “Conv+BN+ReLU” block is dominated by the difference \((x - x')\) rather than the secret message \(s\). The result implies that the amplitude of the “Conv+BN+ReLU” output is determined by cover images. For the model with multiple “Conv+BN+ReLU” blocks, the iterative equations for \(x\) and \(y'\) at a deep depth are similar with Eq.(12)-Eq.(15) except that their difference is dominated by the term introduced by \((x - x')\) at any depth. This interference term, however, cannot be eliminated as the network depth increases. Consequently, the feature map generated by the unpaired case completely different from the feature map generated by the paired case, thus the detection error rate would increase.

C. Paired Training but Tested With Globally Learned Statistics

In previous two subsections, we analyze behaviors of a deep network with BN layers for steganalysis in paired and unpaired cases. Both analysis use batch statistics to normalize feature maps in training and testing. However, in real case, BN only uses batch statistics to normalize input data in training but uses global statistics, which are learned by training samples, to normalize the data in testing. In this section, we analyze how the the network with BN layers behaves when the globally learned statistics are used at testing phase. For a BN layer, global statistics are learned by each batch statistics with the exponential moving average method [46] (we only consider the mean here):

\[
\mu(k + 1) = (1 - \lambda) \mu(k) + \lambda \mu_k
\]

where \(\mu_k\) represents the globally learned mean of BN layer at the \(k\)-th iteration, \(\lambda\) is the moment to learn global statistics, and \(0 < \lambda < 1\). \(\mu_k\) represents the batch mean at the \(k\)-th iteration, which satisfies:

\[
\mu_k = 0, \text{ when } k \leq 0
\]

Mathematically, Eq.(24) can be written into the following compact form:

\[
\mu \equiv \mu_L = (1 - \lambda) \sum_{k=0}^{\infty} \lambda^k \mu_{L-k}
\]

where \(\mu_L\) denotes the converged mean of the BN layer. \(L\) is the number of total iterations. One fact of BN parameters including \(\mu_L\) is that they are updated in model learning. Once the learning is finished, these parameters are preserved and directly used for test images.

![Fig. 3. Histogram of elements in \(Ws\) (paired case) and \(W(x - x')\) (unpaired case).](image-url)
For $\mu_{L-k}$ in Eq.(26), it can be rewritten as:

$$\mu_{L-k} = E \left[ \frac{1}{|B|} \sum_{i \in B} W_{L-k} (x_{L-k}^i + y_{L-k}^i) \right]$$

(27)

where $x_{L-k}^i$ and $y_{L-k}^i$ denote the $i$-th cover image and its stego in a training batch $B$ at $(L-k)$-th iteration. $W_{L-k}$ is the weight of convolution layer at $(L-k)$-th iteration.

For a test cover image $x_{te}$, its centralized output by subtracting $\mu_g$ after a BN layer is:

$$c_{te} = W_L x_{te} - \mu_g$$

(28)

where $W_L$ denotes the converged weight of the convolutional layer after $L$ iterations. Combining Eq.(26) and Eq.(27), and considering the fact that $x_{L-k}^i$ is paired with $y_{L-k}^i$, $c_{te}$ can be approximated by following equation:

$$c_{te} \approx \hat{c}_{te} = W_L x_{te} - (1 - \lambda) \sum_{k=0}^{\infty} \frac{\lambda^k}{|B|} E \left[ \sum_{i \in B} W_{L-k} x_{L-k}^i \right]$$

(29)

For $\hat{c}_{te}$, we use $y_{L-k}^i$ to approximate $x_{L-k}^i$. In paired training scheme, the approximation works since the difference between a cover image and its stego is very small. After some mathematical operations, the expectation of $\hat{c}_{te}$ can be written as:

$$E(\hat{c}_{te}) = (1 - \lambda) \sum_{k=0}^{\infty} \frac{\lambda^k}{|B|} E \left[ \sum_{i \in B} (W_{L-k} x_{L-k}^i - W_L x_{te}) \right]$$

(30)

For the term in the $\sum_{i \in B}$ operator, its variance satisfies the following approximation:

$$var \left( W_{L-k} x_{L-k}^i - W_L x_{te} \right) = var \left[ (W_{L-k} - W_L) x_{L-k}^i + W_L (x_{L-k}^i - x_{te}) \right] \approx var \left[ W_L (x_{L-k}^i - x_{te}) \right]$$

(31)

where $var(\cdot)$ represents the variance of the element in a feature map. The approximation in Eq.(31) is promising when $k$ is a small value, because $W_{L-k}$ is almost converged to $W_L$ in this case. When $k$ is a large value, $\lambda^k$ is very small and the contribution of $var \left( W_{L-k} x_{L-k}^i - W_L x_{te} \right)$ to Eq.(30) can be neglected. In subsection B, we have already analyzed that the variance of $W(x - x')$ is much larger than the variance of secret message $\mathbf{W}$. Similarly, the feature map of the difference between two images $W_L (x_{L-k}^i - x_{te})$ also has large variance than secret message. Therefore, we can conclude that $c_{te}$ is dominated by the difference between test image $x_{te}$ and each training sample, which draws out the secret message. This, however, makes features extracted in testing mismatch features learned in training, thus the performance degrades. When the input is a stego image $y_{te}$, same conclusion can also be obtained as the case of the cover image $x_{te}$.

V. FROM BATCH NORMALIZATION TO SHARED NORMALIZATION

Previous analysis shows that a CNN model with multiple BN layers can detect stegography when test images are paired and batch statistics are used to normalize the data. However, the network becomes hard to be generalized to the test data when these conditions are violated, including the case that images are tested unpaired with batch statistics or the case that globally learned statistics are used in testing. To address this difficulty, we propose to use shared statistics rather than batch statistics to normalize the input data in model training. Different from a BN layer, the proposed SN layer uses shared statistics to normalize the input batches:

$$x_{sn} = SN(x_i) = x_i - \mu^* \frac{\sigma^*}{\sigma + \epsilon}$$

(32)

$$y_{sn} = SN(y_i) = y_i - \mu^* \frac{\sigma^*}{\sigma + \epsilon}$$

(33)

where $\mu^*$ and $\sigma^*$ represent the estimated mean and standard deviation of cover images and stego images, rather than the mean and standard deviation of a mini-batch. $\epsilon$ is a small value to prevent the zero denominator. In our implementation, initial values of $\mu^*$ and $\sigma^*$ are estimated by a set of paired image $\{(x_i, y_i)\}_{i \in S}$:

$$\mu^* = \frac{1}{|S|} \sum_{i \in S} E (x_i + y_i)$$

(34)

$$\sigma^* = \frac{1}{|S|} \sum_{i \in S} ||vec(x_i) - \mu^*||^2 + ||vec(y_i) - \mu^*||^2$$

(35)

where $S$ is the set for calculating $\mu^*$ and $\sigma^*$, $vec(\cdot)$ denotes the vectorization operator, and $||\cdot||^2$ represents the norm of a vector. In order to make the estimation accurate, we set size $S$ much larger than the size of a mini-batch $B$ in a BN layer. For a SN layer at the $n$-th layer of a network, the mean $\mu_n^*$ and the standard deviation $\sigma_n^*$ for each feature map are initialized by:

$$\mu_n^* = \frac{1}{|S|} \sum_{i \in S} E (vec(f_{n-1}^{xi}) + vec(f_{n-1}^{yi}))$$

(36)

$$\sigma_n^* = \frac{1}{|S|} \sum_{i \in S} ||vec(f_{n-1}^{xi}) - \mu_n^*||^2 + ||vec(f_{n-1}^{yi}) - \mu_n^*||^2$$

(37)

where $f_{n-1}^{xi}$ and $f_{n-1}^{yi}$ denote the feature map for cover image $x_i$ and $y_i$ at previous layer respectively.

To capture the change of the network and estimate accurate statistics of input data in model training, we do not fix $\mu_n^*$ and $\sigma_n^*$ but update them according to following equation:

$$\mu_n^*(t + 1) = (1 - \alpha_1) \mu_n^*(t) + \alpha_1 \mu_n^B(t)$$

(38)

$$\sigma_n^*(t + 1) = (1 - \alpha_1) \sigma_n^*(t) + \alpha_1 \sigma_n^B(t)$$

(39)

where $\alpha_1$ is the learning rate for SN, $\mu_n^B(t)$ and $\sigma_n^B(t)$ are the mean and standard deviation calculated by the mini-batch $B$ at $t$-th iteration. In testing, the model uses learned SN statistics to normalize input data, either the input data are fed in pairs or one by one.

BN layer uses the statistics from mini-batch to normalize each training sample in the training phase. And the statistics are fluctuated with batch in the training stage. To each batch, these statistics are precise, but unfortunately it makes the learning model relying on these statistics. Different from BN layer, SN
Fig. 4. The proposed network for image steganalysis. The preprocessing subnetwork consists of preprocessing, feature learning and classification layers. In training phase, the network are fed with paired cover images and stego images. In testing phase, the model with learned parameters are validated with one-by-one input.

layer uses a larger batch to obtain the initial statistics, and these statistics are updated with the learning progress. Compared with BN, the statistics are not fluctuated with each batch, and it also shares more similarity between the training and the test phases. This difference makes the network with SN layers obtain two advantages. On one hand, the proposed SN layer can extract stable and consistent statistical properties from training set, avoid the limitation in a BN layer that feature maps learned in the training are different from the feature maps used in testing when inaccurate statistics are used to normalize the data. On the other hand, cover images and their stego are not automatically separated after SN normalization, making the model insensitive to normalization statistics. Consequently, CNN models with SN layers are forced to learn discriminative patterns between cover images and stego images through convolutional layers, thus the generalization ability is improved.

VI. A NOVEL CONVOLUTIONAL NEURAL NETWORK FOR IMAGE STEGANALYSIS BASED ON SHARED NORMALIZATION

In this section, a novel CNN model with SN layers is proposed for image steganalysis. Details about the network and the learning are introduced in this section.

A. Network Architecture

Based on the SN layer, we propose a novel CNN model for image steganalysis. Fig. 4 shows the overall architecture of the proposed network. The network contains the preprocessing subnetwork, the feature learning subnetwork and the classification subnetwork, which are introduced in the following parts.

The preprocessing subnetwork is to extract the high frequency component from input cover/stego images. It contains two layers, i.e. the High-Pass-Filtering (HPF) layer and the truncation layer. For HPF layer, it contains several highpass filters to extract high frequency signals from input images, aiming to remove most of image contents thus amplify the Signal-to-Noise Ratio (SNR) between the signal introduced by secret message and the cover image. For those highpass filters, we follow the setting in [21] and use the 2nd order, 3rd order, EDGE-3 × 3, KV kernel and their rotations in our model. For the truncation layer, similar to [21], [22], we use Eq.(39) to constrain the dynamic range of input feature map, in order to further remove the image content and improve the convergence speed:

\[
\text{Trunc}(x) = \begin{cases}
    -T, & x < -T \\
    x, & -T \leq x \leq T \\
    T, & x > T
\end{cases}
\]  

Fig. 5. A Processing Unit (PU) contains a convolutional layer, a SN layer, a ReLU activation layer and a pooling layer.
convolutional kernels of current block and the number of convolutional kernels of previous block. The size of convolutional kernels in a PU block is $3 \times 3$ and the number of convolutional kernel is 24, 48, 96 and 192. To make the network learn discriminative features for steganalysis consistently, we force that the size of pooling layer is same to the convolutional kernel, i.e. $3 \times 3$. After several PUs, an average pooling layer with large size, $32 \times 32$, transforms feature maps into feature vectors. The purpose of using a large size pooling layer is to reduce the information loss introduced by many processing layers.

The classification subnetwork maps extracted features into binary labels. Two output nodes, which corresponds to the label of cover and stego, are fully connected to the feature map that are averaged pooled in the feature learning subnetwork.

B. Parameter Learning

Parameters of the proposed network are learned by minimizing the softmax loss function:

$$L(x_i, \theta) = -\sum_{k=1}^{K} 1\{y_i = k\} \cdot \log \left( \frac{e^{\theta \alpha_i k(x_i, \theta)}}{\sum_{k=1}^{K} e^{\theta \alpha_i k(x_i, \theta)}} \right)$$

(41)

where $\theta$ denotes the parameters of the network, including weight matrices $W$ and the bias vectors $b$. $K$ is the number of labels, where $K = 2$ in our model. $y_i$ is the label of $x_i$, $1\{\cdot\}$ is the indicator function. $\alpha_i k(x_i, \theta)$ represents the output of the network for the sample $x_i$. $W$ and $b$ of the network parameter $\theta$ are updated by the mini-batch stochastic gradient descending (SGD):

$$W(t + 1) = W(t) - \alpha_2 \frac{1}{|B|} \sum_{i \in B} \frac{\partial L(x_i, \theta)}{\partial W}$$

(42)

$$b(t + 1) = b(t) - \alpha_2 \frac{1}{|B|} \sum_{i \in B} \frac{\partial L(x_i, \theta)}{\partial b}$$

(43)

where $|B|$ is the size of a mini-batch $B$, $\alpha_2$ is the learning rate.

VII. Experiments

In this section, comprehensive experiments are conducted to demonstrate the effectiveness of the proposed SN layer and our new CNN model for image steganalysis. We not only compare the proposed network with the state-of-the-art rich model based methods but also compare it with several recently proposed CNN models. We further conduct sensitivity experiments to test our model when the embedding payload or steganographic algorithm is mismatched. We finally demonstrate that some techniques widely used in deep learning, including data augmentation and model ensemble, can also improve our CNN model’s detection accuracy.

A. Dataset and Steganographic Schemes

The dataset used for validation is the BOSSbase 1.01 [33], which is a standard database for evaluating steganography and steganalysis. The BOSSbase contains 10,000 uncompressed natural images with the size of $512 \times 512$, including human beings, natural scenes, animals and buildings, which are shown as Fig. 6.

The detection error rate $P_E$, which is widely used to measure the security of steganographic algorithms [8]–[10], is adopted for performance evaluation:

$$P_E = \frac{1}{2} (P_{MD} + P_{FA})$$

(44)

where $P_{MD}$ is the miss detection probability and $P_{FA}$ represents the false alert probability.

Four states of the art steganographic algorithms, including the Highly Undetectable steGanography with Bounding Distortion (HUGO-BD) [39], the Wavelet Obtained Weights steganography (WOW) [40], S-UNIWARD [38], and the High-pass Low-pass steganography (HILL) [41], are used for performance validation. To avoid that secret data are embedded with the same key, we adopt the MATLAB implementation of steganographic algorithms rather than the C++ version for performance evaluation. In all experiments, we use bit-per-pixel (bpp) to represent the size of secret data embedded into cover images.

B. Parameter Setting

For the proposed network, each element $W_{ij}$ in the weight matrix $W$ is initialized by the improved “Xavie” method [42], i.e. Gaussian distribution with zero mean and the standard deviation inversely proportional to the number of network’s connections:

$$W_{ij} \sim \mathcal{N}\left(0, \frac{2}{c_n}\right)$$

(45)

where $c_n$ denotes the number of connections at $n$-th layer. The bias $b$ of a convolutional layer is initialized to be zero. The truncation value $T$ is set to 5.0 in all experiments. For the initialization of SN layer parameters, they are calculated by Eq.(36) and Eq.(37) from layer to layer. In order to make the estimation of $\mu_n^c$ and $\sigma_n^c$ accurate, the size of $S$ should be set to a relatively large value. In our implementation, $|S|$ is set to 200, including 100 cover images and 100 their stegos. And the image pairs in $S$ were randomly selected from the training set. $\epsilon$ in a SN layer is set to $10^{-5}$. To make SN layers capture the change of the network, statistics of SN should be updated in the same rate of its parameter, i.e. $\alpha_1$ is set same to $\alpha_2$.

For the learning rate $\alpha_2$ and batch size $|B|$ of SGD, we conduct experiments to show how our model performs at given batch sizes and learning rates. The batch size and learning rate for searching are set to [20, 30, 40, 50, 60] and [0.1, 0.05, 0.01, 0.005, 0.001] respectively. Similar to existing deep CNN models [20], instead of using a fixed learning rate, we adaptively adjust $\alpha_2$ in training phase. Specifically, the learning rate for $W$ is set to $\alpha_2$ in first 150 training epoches and is decreased to $0.1 \cdot \alpha_2$ in resting 50 epoches. The purpose of decreasing the learning rate is to make the network escape the error plateaus. In
the learning phase, the momentum and the weight decay in our network are set to 0.9 and 0.0001 respectively. Fig. 7 shows the detection error rates of S-UNIWARD steganography at 0.4 bpp with different batch sizes and learning rates. Results in Fig. 7 show that when $|B|$ and $\alpha_2$ and $T$ are selected to 40 and 0.01 the proposed model obtains the best performance. In the following experiment, we use these settings to optimize the proposed model for all steganographic algorithms.

### C. Comparison: Network With Batch Normalization Layers and Network With Shared Normalization Layers

In this experiment, we demonstrate the effectiveness of the proposed shared normalization layer over the batch normalization layer for image steganalysis. We compare two models in this experiment: the proposed network with BN layer and with SN layers; the Xu-network [15] with BN layers and SN layers. For Xu-network, we implement it on Matconvnet platform and follow the setting of original paper. To make the result comparable, the network with BN layers has exactly same architecture with the network with SN layers except that all shared normalization layers are replaced with batch normalization. In addition, BN versions and SN versions utilize same training images and testing images in the experiment.

Fig. 8 and Fig. 9 show the detection error rates of two networks with BN layers and with SN layers on S-UNIWARD steganography at 0.4 bpp respectively. For simplicity, detection error rates in training set and testing set are simply called as “training error” and “testing error” in the figure. For networks with BN layers, the error decreases fast as the training proceeds. However, the testing error vibrates greatly and no clear connection between the training error and the testing error can be found. This indicates that features learned by the network with BN layers mismatch features extracted from testing data, making the model hard to classify cover images and stego images accurately. For networks with SN layers, the testing error systematically decreases as the training error decreases, showing that the proposed network with SN layers can effectively learn discriminative features to classify unknown cover images and stego images. Table II shows the detection error rates of Xu-network\(^3\) and proposed network at S-UNIWARD and HILL at 0.4 bpp. Results in the table demonstrate that the proposed SN improves deep network’s generalization ability for image steganalysis.

### D. Performance Comparison With Prior Arts

In this subsection, we conduct comprehensive experiments to demonstrate the effectiveness of the proposed CNN model. We compare the proposed network with the classical SRM [9] and its select-channel-aware version, the maxSRM steganalysis [10]. SRM steganalysis extracts many handcrafted features that are sensitive to message embedding and combine them into a long feature vector for classification. An ensemble classifier [35] is trained based on the extracted features and is used for predicting the label of an input image. With embedding probability maps, the maxSRM steganalysis focuses more on image pixels with lower embedding distortions. Four steganographic algorithms, i.e. HUGO-BD, WOW, S-UNIWARD, and HILL, at five different payloads, i.e. [0.05, 0.1, 0.2, 0.3, 0.4], are used for validation. In the experiment, we randomly select 5,000 cover images and their stego images to train an optimized model, while the rest 5,000 cover images and their stego images are used for testing.

For network training, we use transfer learning [14], [21], [22] to learn effective features for image steganalysis at low payloads. This technique is widely used in many deep convolutional neural network models [43], [44], showing promising performances for closely related tasks. Specifically, our CNN model for a lower payload steganalysis, e.g. 0.3 bpp, are finetuned based on the network trained at a higher payload, e.g. 0.4 bpp. The reason is that directly learning discriminative features proves to be hard for CNN based steganalysis at low payloads [22], while transfer learning can regularize the feature space and utilize auxiliary information from stego images at higher payloads [21]. To avoid training samples are reused for testing at different payloads, we force that cover images for network training/testing at a lower payload are same to those for network training/testing at a higher payload.

Table III gives performance comparisons between the proposed network and SRM, maxSRM at different configurations. Compared with the maxSRM steganalysis, our model achieves lower detection error rates in most cases. An important fact is that the maxSRM method uses extra information, embedding probability map, for image steganalysis. Each element in an embedding probability map denotes the probability that a secret message is embedded at this position. However, our model does not utilize such information to detect steganographic algorithms but still outperforms the maxSRM in most of settings. Besides, we also provide ROC curves as Fig. 10. Compared with other steganalytic methods, the proposed network shows performance improvements on all steganographic schemes at all payloads. From the table and the figure, we have an important observation.

\(^3\)The averaged performance on S-UNIWARD and HILL reported in [15] are 20.97% ± 0.67% and 22.42% ± 0.30%. The performance of our implementation, 21.54% and 22.67%, which are consistent with the reported result.
that the performance improvement between the proposed network and maxSRM becomes smaller as the payload decreases. This phenomenon has also been observed in [22], indicating that CNN models are not effective to detect steganography at low payloads even though transfer learning is used. To illustrate this phenomenon, two embedding probability maps at extreme payloads, i.e. the largest payload 0.4 bpp and the smallest payload 0.05 bpp, are drawn as Fig. 11. Unlike the embedding probability map at 0.4 bpp (as Fig. 11(c) shows) that secret data can be almost embedded in the whole building area, the map at 0.05 bpp (as Fig. 11(b) shows) only hides secret data in very complex regions such as sharp edges or textured patterns. Consequently, we conclude that two reasons may limit the performance of CNN models to detect steganography at low payloads. For the first, these very complex regions do not take a large proportion in most of natural images. In this case, CNN models become easily over-fitted because they do not have enough training data to model the distribution of complex regions. For the second, very complex regions often show some properties like random “noise” [45]. CNN models are proved to have strong ability extract regular patterns among natural images, but may not be effective to model this type of noisy patterns.

Additionally, we compare the proposed network with two states of the art CNN based steganalytic methods, including
TABLE III
DETECTION ERROR RATES OF SRM, maxSRM AND THE PROPOSED NETWORK FOR FOUR STEGANOGRAPHIC ALGORITHMS AT FIVE DIFFERENT PAYLOADS. THE BOSSBASE DATASET IS USED FOR VALIDATION

| Steganography | Detection algorithm | 0.05 bpp | 0.1 bpp | 0.2 bpp | 0.3 bpp | 0.4 bpp |
|---------------|---------------------|----------|---------|---------|---------|---------|
| HUGO-BD       | SRM + ensemble      | 42.60%   | 37.26%  | 28.78%  | 22.54%  | 18.23%  |
|               | maxSRM + ensemble   | 36.83%   | 31.32%  | 24.53%  | 20.37%  | 16.47%  |
|               | The proposed network | 36.76%   | 30.81%  | 23.72%  | 19.25%  | 15.43%  |
| WOW           | SRM + ensemble      | 45.63%   | 40.15%  | 32.31%  | 25.66%  | 20.08%  |
|               | maxSRM + ensemble   | 35.39%   | 30.18%  | 23.84%  | 18.92%  | 15.40%  |
|               | The proposed network | 35.87%   | 30.02%  | 23.48%  | 17.43%  | 14.26%  |
| S-UNIWARD     | SRM + ensemble      | 45.38%   | 40.38%  | 32.54%  | 25.51%  | 20.70%  |
|               | maxSRM + ensemble   | 41.98%   | 35.63%  | 28.04%  | 22.35%  | 18.84%  |
|               | The proposed network | 42.13%   | 35.21%  | 26.82%  | 20.71%  | 16.53%  |
| HILL          | SRM + ensemble      | 47.32%   | 43.71%  | 36.47%  | 29.39%  | 24.57%  |
|               | maxSRM + ensemble   | 42.31%   | 37.76%  | 30.95%  | 25.71%  | 21.63%  |
|               | The proposed network | 42.15%   | 36.86%  | 29.63%  | 23.60%  | 19.87%  |

Fig. 10. ROC curves for SRM, maxSRM and the proposed network for different steganographic algorithms. (a) HUGO-BD steganography. (b) WOW steganography. (c) S-UNIWARD steganography. (d) HILL steganography.
Fig. 11. A cover image and its embedding probability maps at different payloads. The S-UNIWARD steganography at 0.05 bpp and 0.4 bpp are used for demonstration. (a), the “433.pgm” cover image in the BOSSbase dataset; (b), the embedding probability at 0.05 bpp; (c), the embedding probability at 0.4 bpp.

TABLE IV
DETECTION ERROR RATES FOR CNN MODELS ON THREE STEGANOGRAPHIC ALGORITHMS AT PAYLOAD 0.4 bpp. “—” DENOTES THAT THE RESULT IS NOT REPORTED IN THE PAPER

| CNN Models      | WOW  | S-UNIWARD | HILL |
|-----------------|------|-----------|------|
| Qian-network [14] | 21.95% | 22.05% | —    |
| Xu-network [15]  | —    | 20.97% | 22.42% |
| Proposed network | **14.26%** | **16.53%** | **19.87%** |

TABLE V
DETECTION ERROR RATES OF PROPOSED NETWORK AND THE SRM MODEL WHEN THE RATIO OF TRAINING SET CHANGES. THE S-UNIWARD STEGANOGRAPHY AT 0.4 bpp IS USED FOR PERFORMANCE EVALUATION

| Ratio of training set | 50% | 60%  | 80%  |
|-----------------------|-----|------|------|
| Proposed network      | 16.53% | 14.72% | 11.26% |
| The SRM model         | 20.70% | 19.22% | 18.39% |

E. Sensitivity Experiments

Most existing work including our previous experiment assume that the data embedding algorithm and the payload are known to the steganalyzer. However, it is often hard to obtain such information in real applications. For steganalytic algorithms, they are often difficult to detect steganography when the payload or the embedding algorithm is mismatched. Here, “mismatch” means that the payload or the embedding algorithm used to train the model is different from that is used to test the model. This reality motivates us to investigate whether the proposed network is sensitive to the payload mismatch and the embedding algorithm mismatch. We use the following experiments to validate our model:

- The network is trained and tested with the same steganographic algorithm but with different payloads;
- The network is trained and tested with the same payload but with different steganographic algorithms.

1) Sensitivity to the Payload Mismatch: In this experiment, we assess the sensitivity of proposed network by testing it with the payload different from that for network training. Specifically, we first train the proposed network from 0.05 bpp to 0.4 bpp and obtain five optimized networks. Then, each optimized network is validated at these five different payloads. The network for the S-UNIWARD steganography is reported in this subsection. For all cases, 5,000 cover images and their stegos are used for training, while the rest images and their stegos are for testing. In addition, we make sure that cover images in testing are not used to train the network in each payload mismatched case.

Results are reported in the Table VI. In this table, the payload in each row denotes the payload for network training, while the payload in each column denotes the payload for network testing. The diagonal detection error rates actually denote the case that the training payload and the testing payload are matched. Compared with the payload matched case, we observe that the detection error rate increases gradually as the payload mismatch amplitude increases. More importantly, the loss of detection error rate is quite small when the network is tested at an adjacent mismatched payload, indicating that the proposed network is robust to the payload mismatch.
TABLE VI
Sensitivity Experiment When Payloads Are Mismatched. The S-UNIW ARD Steganography Is Used for Demonstration. Diagonal Elements Denote the Payload Matched Case

| Test Payload | 0.05 bpp | 0.1 bpp | 0.2 bpp | 0.3 bpp | 0.4 bpp |
|--------------|----------|---------|---------|---------|---------|
| Train Payload|          |         |         |         |         |
| 0.05 bpp     | 42.13%   | 36.81%  | 31.40%  | 28.25%  | 26.61%  |
| 0.1 bpp      | 42.22%   | 35.21%  | 28.74%  | 25.37%  | 23.23%  |
| 0.2 bpp      | 44.23%   | 36.75%  | 26.82%  | 22.10%  | 19.48%  |
| 0.3 bpp      | 45.91%   | 39.07%  | 27.85%  | 20.71%  | 16.91%  |
| 0.4 bpp      | 47.46%   | 41.42%  | 29.97%  | 21.54%  | 16.53%  |

Fig. 12. Sensitivity experiment when steganographic algorithms are mismatched. In mismatched case, the network is trained by S-UNIW ARD steganography but tested by other three steganographic algorithms. The training payload and testing payload are fixed at 0.4 bpp.

2) Sensitivity to the Steganographic Algorithm Mismatch: In this experiment, we assess the sensitivity of the proposed network by testing it with a steganographic algorithm that is different from the algorithm for the network training. The network, which is trained on the S-UNIW ARD steganography, is tested on three other steganographic algorithms. In order to solely investigate whether the proposed network is sensitive to the change of steganographic algorithms, we force that the payload for training and testing is fixed at 0.4 bpp. Same to the setting in the previous experiments, 5,000 cover images and their stegos are used for training, while the rest images and their stegos are used for testing.

Fig. 12 shows detection errors of steganographic algorithm matched and mismatched cases. In mismatched cases, we use the network trained on S-UNIW ARD steganography to detect other three steganographic algorithms. For HUGO-BD and HILL, the detection error rate of mismatched case degrades significantly to the matched case. The reason is that distortion functions of these two steganographic algorithms for message embedding, which are depicted by Fig. 13(a) and Fig. 13(b), are different from S-UNIW ARD, which is depicted by Fig. 13(c). This indicates that secret messages in HUGO-BD and HILL are embedded in completely different patterns to S-UNIW ARD, making the network trained by S-UNIW ARD hard to detect stego images generated by them. For WOW steganography, the performance loss of mismatched case is relatively small because distortion functions used for S-UNIW ARD and WOW are similar to each other, as shown in Fig. 13(c) and Fig. 13(d). Those observations imply that, compared to the payload mismatch, the proposed CNN model is more sensitive to steganographic algorithm mismatch.

F. Data Augmentation and Ensemble Learning

In the last subsection, we demonstrate whether some techniques widely used in deep learning, including data augmentation and ensemble learning, can improve the detection accuracy of our network. Three settings are conducted here:

1) Data Augmentation (Aug.): in this setting, 10,000 BOSSbase samples are randomly split into 5,000 training images and 5,000 testing images. For training images, we rotate them with 90 degree, 180 degree, and 270 degree, generating a new training set with 20,000 samples. Then, the S-UNIW ARD steganography is used to embed secret messages into the augmented training set and the test set. The proposed network is trained on this new training set with 20,000 covers/stegos and finally validated on the test set with 5,000 covers/stegos.

2) Ensemble Learning (Ens.): similar to the first setting, we first split the BOSSBase dataset into 5,000 training images and 5,000 test images randomly. Then, we use the S-UNIW ARD steganography to embed secret messages into images and obtain 5,000 training pairs and 5,000 testing pairs. Then, we generate five new training sets by randomly selecting image pairs from the original training sets, where each new set contains 4,000 pairs. Finally, five CNN models are trained based on new training sets, and they are tested on the original test pairs. The overall performance of ensemble learning is obtained by voting five optimized models.

3) Data Augmentation + Ensemble Learning (Aug. + Ens.): same to the first setting, a training set with 20,000 pairs and a test set with 5,000 pairs are created. Same to the second setting, five new training sets are generated by random selection, where each set contains 16,000 pairs. Then, five CNN models are trained based on five augmented training sets respectively. The overall detection accuracy is obtained by voting these optimized CNN models.

The detection error rates of three schemes are reported in Table VII. In this table, performance at configuration (1, 4,000/16,000) means the average detection error rate of five models with 4,000/16,000 training pairs, while (5, 4,000/16,000) denotes the voted performance of five models with their own 4,000/16,000 training pairs which randomly selected from the same training set. For the table, we can observe that data augmentation decreases the detection error rate from 16.53% to 14.86%. For model ensembling, the detection error rate of single model decreases from 18.23% to 16.79%. In the experiment, model trained with 4,000 pairs in ensembling, 16.79%, is worse than original performance, 16.53%. The performance loss mainly caused by the fact that each model in ensemble setting has smaller number of training samples than original model. This result is also consistent with Table V that the amount of training set is crucially important for deep learning based image steganalysis. Nevertheless, combining model ensemble and data augmentation, the detection error rate has been further reduced to
14.07%. These results demonstrate that data augmentation, ensemble learning and their combination can improve the performance of the proposed network. The results also indicate that those deep learning techniques are indeed beneficial for CNN based steganalysis.

VIII. CONCLUSION

Even though convolutional neural network based steganalysis develops a lot in recent years, what kinds of deep learning techniques suitable for image steganalysis waits to be carefully examined. In this paper, we deeply explore the batch normalization, a popular technique used in general image classification, for the steganalysis task. Theoretically, we analyze that a CNN model with multiple batch normalization layers can detect steganography at very high accuracy when cover images and their stegos are paired training and testing, and their batch statistics are obtained to normalize the data. However, the network becomes unstable and fails to detect stego images when cover-setgos are unpaired. To handle this difficulty, we propose a novel normalization technique called shared normalization to cover-setgos are unpaired. To handle this difficulty, we propose a novel normalization technique called shared normalization to normalize input data. Compared with the batch normalization, the proposed normalization method can stabilize network learning and make the network obtain better generalization ability. Based on the SN, we further propose a novel CNN model for image steganalysis. The experiment demonstrates that the proposed model achieved better performances than traditional rich model method and state-of-the-art CNN models. In future works, we would extend our model to detect steganography in compressed domain.

### TABLE VII

| Single model | Original | Aug. | Ens. | Aug. + Ens. |
|--------------|---------|------|------|------------|
| (# models, # pairs) | (1, 5000) | (1, 20000) | (1, 4000) | (1, 16000) |
| Detection error rate | 16.53% | 14.86% | 18.23% | 15.36% |

| Ensembled models | Original | Aug. | Ens. | Aug. + Ens. |
|-------------------|---------|------|------|------------|
| (# models, # pairs) | — | — | (5, 4000) | (5, 16000) |
| Detection error rate | — | — | 16.79% | 14.07% |

Fig. 13. A cover image and its embedding probability maps of different steganographic algorithms. (a) a crop of “13.gpm” cover image from the BOSSbase dataset; (b) the embedding probability of HUGO; (c) the embedding probability of HILL; (d) the embedding probability of S-UNIWARD; (e) the embedding probability of WOW.

### REFERENCES

[1] A. Cheddad, J. Condell, K. Curran, and P. M. Kevitt, “Digital image steganography: Survey and analysis of current methods,” Signal Process., vol. 90, no. 3, pp. 727–752, 2010.

[2] W. Zhang, Z. Zhang, L. Zhang, H. Li, and N. Yu, “Decomposing joint distortion for adaptive steganography,” IEEE Trans. Circuits Syst. Video Technol., vol. 27, no. 10, pp. 2274–2280, Oct. 2016.

[3] Y. Liao, C. Wang, W. Chen, F. Lin, and W. Liu, “A novel data hiding algorithm for high dynamic range images,” IEEE Trans. Multimedia, vol. 19, no. 1, pp. 196–211, Jan. 2017.

[4] W. Zhang, H. Wang, D. Hou, and N. Yu, “Reversible data hiding in encrypted images by reversible image transformation,” IEEE Trans. Multimedia, vol. 18, no. 8, pp. 1469–1479, Aug. 2016.

[5] H. Wang and S. Wang, “Cyber warfare: Steganography vs. steganalysis,” Commun. ACM, vol. 47, no. 10, pp. 76–82, 2004.

[6] H. Yin, W. Hui, H. Li, C. Lin, and W. Zhu, “A novel large-scale digital forensics service platform for internet videos,” IEEE Trans. Multimedia, vol. 14, no. 1, pp. 178–186, Feb. 2012.

[7] N. Provos and P. Honeyman, “Detecting steganographic content on the Internet,” in Proc. Netw. Distrib. Syst. Secur. Symp., 2002, pp. 1–13.

[8] J. Fridrich and M. Goljan, “Practical steganalysis of digital images—State of the art,” Proc. SPIE, vol. 4675, pp. 1–13, 2002.

[9] J. Fridrich and J. Kodovsky, “Rich models for steganalysis of digital images,” IEEE Trans. Inf. Forensics Secur., vol. 7, no. 3, pp. 868–882, Jun. 2012.

[10] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, “Selection-channel-aware rich model for steganalysis of digital images,” in Proc. IEEE Workshop Inf. Forensics Secur., 2014, pp. 48–53.

[11] G. Lin, Y. Chang, and W. Lie, "A framework of enhancing image steganography with picture quality optimization and anti-steganalysis based on simulated annealing algorithm," IEEE Trans. Multimedia, vol. 12, no. 5, pp. 345–357, Aug. 2010.

[12] S. Tan and B. Li, “Stacked convolutional auto-encoders for steganalysis of digital images,” in Proc. Ann. Summit Conf. Asia–Pacific Signal Inf. Process. Assoc., 2014, pp. 1–4.

[13] Y. Qian, J. Dong, W. Wang, and T. Tan, “Deep learning for steganalysis via convolutional neural networks,” Proc. SPIE, vol. 9409, 2015, Art. no. 94090J.

[14] Y. Qian, J. Dong, W. Wang, and T. Tan, “Learning and transferring representations for image steganalysis using convolutional neural network,” in Proc. IEEE Int. Conf. Image Process., 2016, pp. 2752–2756.

[15] G. Xu, H. Z. Wu, and Y. Q. Shi, “Structural design of convolutional neural networks for steganalysis,” IEEE Signal Process. Lett., vol. 23, no. 5, pp. 708–712, May 2016.

[16] J. Zhang, S. Tan, B. Li, and J. Huang, “Large-scale JPEG image steganalysis using hybrid deep-learning framework,” IEEE Trans. Inf. Forensics Secur., vol. 13, no. 5, pp. 1200–1214, May 2018.

[17] W. Tang, S. Tan, B. Li, and J. Huang, “Automatic steganographic distortion learning using a generative adversarial network,” IEEE Signal Process. Lett., vol. 24, no. 10, pp. 1547–1551, Oct. 2017.

[18] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich, “JPEG-phase-aware convolutional neural network for steganalysis of JPEG images,” in Proc. ACM Workshop Inf. Hiding Multimedia Secur., 2017, pp. 75–84.

[19] S. Wu, S. Zhong, and Y. Liu, “Deep residual learning for image steganalysis,” Multimedia Tools Appl., vol. 77, pp. 10437–10453, 2018.
[20] S. Wu, S. Zhong, and Y. Liu, “Residual convolution network based ste- 
ganomaly analysis with adaptive content suppression,” in Proc. IEEE Int. Conf. 
Multimedia Expo, 2017, pp. 241–246.

[21] J. Ye, J. Ni, and Y. Yi, “Deep learning hierarchical representations for 
image steganalysis,” IEEE Trans. Inf. Forensics Secur., vol. 12, no. 11, 
pp. 2545–2557, Nov. 2017.

[22] G. Xu, “Deep convolutional neural network to detect J-UNIWARD,” Proc. 
5th ACM Workshop Inf. Hiding Multimedia Secur., 2017, pp. 67–73.

[23] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network 
training by reducing internal covariate shift,” in Proc. 32nd Int. Conf. 
Mach. Learn., 2015, pp. 448–456.

[24] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image 
recognition,” in Proc. IEEE Conf. Comput. Vision Pattern Recognit., 2016, 
pp. 770–778.

[25] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, “Beyond a Gaussian 
denoiser: Residual learning of deep CNN for image denoising,” IEEE 
Trans. Image Process., vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[26] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural 
image caption generator,” in Proc. IEEE Conf. Comput. Vision Pattern 
Recognt., 2015, pp. 3156–3164.

[27] K. Simonyan and A. Zisserman, “Very deep convolutional networks for 
large-scale image recognition,” Proc. 3rd IAPR Asian Conf. Pattern Recogni- 
tion., 2014, pp. 730–734.

[28] C. Szegedy et al., “Going deeper with convolutions,” Proc. IEEE Conf. 
Comput. Vision Pattern Recognit., 2014, pp. 1–9.

[29] K. Simonyan and A. Zisserman, “Very deep convolutional networks for 
large-scale image recognition,” in Proc. 3rd IAPR Asian Conf. Pattern Recogni- 
tion. 2014, pp. 730–734.

[30] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using 
deep convolutional networks,” IEEE Trans. Pattern Anal. Mach. Intell., 
vol. 38, no. 2, pp. 295–307, Feb. 2016.

[31] E. Denton, S. Chintala, A. Szlam, and R. Fergus, “Deep generative image 
models using a Laplacian pyramid of adversarial networks,” in Proc. Adv. 
Neural Inf. Process. Syst., 2015, pp. 1486–1494.

[32] L. A. Gatys, A. S. Ecker, and M. Bethge, “Texture synthesis using convo- 
lutional neural networks,” in Proc. Adv. Neural Inf. Process. Syst., 2015, 
pp. 262–270.

[33] A. Mahendran and A. Vedaldi, “Visualizing deep convolutional neural 
networks using natural pre-images,” Int. J. Comput. Vision, vol. 12, no. 3, 
pp. 233–255, 2016.

[34] P. Bas, T. Filler, and T. Pevny, “Break our steganographic system: The ins 
and outs of organizing BOSS,” in Proc. Int. Workshop Inf. Hiding, 2011, 
pp. 59–70.

[35] V. Schwamberger and M.O. Franz, “Simple algorithmic modifications for 
 improving blind steganalysis performance,” in Proc. 12th ACM Multime- 
dia Secur. Workshop, 2010, pp. 225–230.

[36] J. Kodovsky, J. Fridrich, and V. Holub, “Ensemble classifiers for steganal- 
ysis of digital media,” IEEE Trans. Inf. Forensics Secur., vol. 7, no. 2, 
pp. 432–444, Apr. 2012.

[37] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual 
networks,” in Proc. Eur. Conf. Comput. Vision, 2016, pp. 630–645.

[38] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger, “Deep net- 
works with stochastic depth,” in Proc. Eur. Conf. Comput. Vision, 2016, 
pp. 646–661.

[39] V. Holub, J. Fridrich, and T. Denemark, “Universal distortion function 
for steganography in an arbitrary domain,” EURASIP J. Inf. Secur., vol. 1, 
no. 1, p. 1–13, 2014.

[40] T. Filler and J. Fridrich, “Gibbs construction in steganography,” IEEE 
Trans. Inf. Forensics Secur., vol. 5, no. 4, pp. 705–720, Dec. 2010.

[41] V. Holub and J. Fridrich, “Designing steganographic distortion using di- 
rectional filters,” in Proc. IEEE Workshop Inf. Forensic Secur., 2012, 
p. 234–239.

[42] B. Li, M. Wang, J. Huang, and X. Li, “A new cost function for spatial 
image steganography,” in Proc. IEEE Int. Conf. Image Process., 2014, 
pp. 4206–4210.

K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Sur- 
passing human-level performance on ImageNet classification,” in Proc. 
IEEE Int. Conf. Comput. Vision, 2015, pp. 1026–1034.

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are 
features in deep neural networks?” in Proc. Adv. Neural Inf. Process. Syst., 
2014, pp. 3320–3328.

R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Region-based convo- 
lutional networks for accurate object detection and segmentation,” IEEE 
Trans. Pattern Anal. Mach. Intell., vol. 38, no. 1, pp. 142–158, Jan. 2015.

D. Zoran and Y. Weiss, “Scale invariance and noise in natural images,” in 
Proc. IEEE Int. Conf. Comput. Vision, 2009, pp. 2209–2216.

G. Box, G. M. Jenkins, and G. C. Reinsel, Time Series Analysis: Fore- 
casting and Control, 3rd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 
1994.

Songtao Wu received the B.Sc. degree in electronic information engineering from Lanzhou University, Lanzhou, China, in 2009, the M.S. degree in circuits and systems from Peking University, Beijing, China, in 2012, and the Ph.D. degree from the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, in 2017. He is currently a visiting scholar with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His research interests include steganography and ste- 
ganalysis, and artificial intelligence.

Sheng-hua Zhong received the Ph.D. degree from the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, in 2013. She was a Postdoctoral Research Associate with the Department of Psychological and Brain Sciences, The Johns Hop- 
kins University, Baltimore, MD, USA, from 2013 to 2014. She is currently an Assistant Professor with the College of Computer Science and Software Engi-
neering, Shenzhen University, Shenzhen, China. Her research interests include multimedia content analy-
sis, brain science, and machine learning.

Yan Liu received the B.Eng. degree from the De- 
artment of Electronic Engineering, Southeast Uni-
versity, Nanjing, China, the M.Sc. degree from the School of Business, Nanjing University, Nanjing, 
China, and the Ph.D. degree from the Department of 
Computer Science, Columbia University, New York, 
NY, USA. She is currently an Associate Professor 
with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. As the Director of Cognitive Computing Laboratory, she focuses her 
research in brain modeling, ranging from image/video 
content analysis, music therapy, manifold learning, deep learning, and EEG 
data analysis.