Multimed Tools Appl (2018) 77:27955–27979
https://doi.org/10.1007/s11042-018-6010-9

Steganography algorithms recognition based on match image and deep features verification

Xiaoyu Xu[1] · **Yifeng Sun**[1] · **Jiang Wu**[1] · **Yi Sun**[1]

Received: 19 June 2017 / Revised: 15 February 2018 / Accepted: 16 April 2018 /
Published online: 16 May 2018

Abstract Steganography algorithms recognition is a sub-section of steganalysis. Analysis shows when a steganalysis detector trained on one cover source is applied to images from an unseen source, generally the detection performance decreases. To tackle with this problem, this paper proposes a steganalytic scheme for steganography algorithms recognition. For a given testing image, a match image of the testing image is achieved. The match image is generated by performing a Gaussian filtering on the testing image to remove the possible stego signal. Then the match image is embedded in with recognized steganography algorithms. A CNN model trained on a training set is used to extract deep features from testing image and match images. Computing similarity between features with inner product operation or weighted-χ^2, the final decision is made according to similarity between testing feature and each class of match feature. The proposed scheme can also detect steganography algorithms unknown in training set. Experiments show that, comparing with directly used CNN model, the proposed scheme achieves considerable improvement on testing accuracy when detecting images come from unseen source.

Keywords Steganography algorithms recognition · CNN · Match image · Deep feature

1 Introduction

Steganography is to conceal information communication by means of hiding secret messages in public media covers, such as images. Steganalysis is counter-technology of steganography, which aims at detecting the very presence of secret message in cover medium, recognizing steganography algorithms and extracting concealed secret message.

Current steganalysis methods can be divided into two categories: The first one is the traditional steganalysis method, which is mainly based on hand-extracted features and machine

✉ Yifeng Sun
yfsun001@163.com

[1] Institute of Information Science and Technology, Zhengzhou 450001, China

learning classifier. The commonly used hand-extracted steganalysis features are SPAM [30], SRM [9], PSRM [12] and maxSRM [4], etc. The commonly used classifiers are Fisher Linear Discrimination (FLD) [6], Support Vector Machine (SVM) [2] and ensemble classifier [18], etc. The second one is steganalysis based on deep learning [22] technology. With the development of big data and computing performance, deep learning technology has been applied to steganalysis. CNN [21] is one of the typical deep learning models, it is able to learn representation of image with multiple levels of abstraction, and has been proved to be an effective tool for steganalysis. Tan and Li [37] firstly proposed a CNN for steganalysis which comprises three stages of alternating convolutional and max pooling layers, and sigmoid nonlinear activations. They involved a high-pass kernel [9, 16] into parameter initialization of the first convolutional layer, and pre-training all of the parameters with unsupervised learning. Qian et al. [32] presented a CNN equipped with a high-pass filtering (HPF) layer, Gaussian non-linear activations, and average pooling for steganalysis. The reported detection error rates of CNNs proposed by Tan and Qian are still inferior to that of SRM with ensemble classifiers. However, along this direction, Xu et al. [39] proposed a CNN that tries to incorporate the knowledge of steganalysis. The reported detection error rates are 1%–4% lower than those achieved by the SRM with ensemble classifiers on the BOSSbase when detecting S-UNIWARD [13] and HILL [24]. This is a significant boost in performance. Moreover, Qian et al. [33] proposed a framework based on transfer learning to help the training of CNN for steganalysis, and showed that feature representations learned with a pre-trained CNN for detecting a steganographic algorithm with a high payload can be efficiently transferred to improve the learning of features for detecting the same steganographic algorithm with a low payload. In paper [40] Xu et al. employed CNNs as base learners and test several different ensemble strategies to improve the detect accuracy. In our former work [41], a designed CNN was used to learn on hand-extracted SRM features and showed comparable performance with ensemble classifier. Using deep learning technology, the above researchers focus on detecting the presence of secret information in cover medium. However, to our best knowledge, there is no published paper on deep learning for recognition of steganography algorithms. Existing research about steganography algorithms recognition are mainly based on hand-extracted features and machine learning classifier, such as [3, 5, 17, 25, 28, 29, 38].

On the other hand, study shows that when a model, which is trained on a specific image set, is applied to another image set, the testing accuracy decreases noticeably. As stated in paper [19], this phenomenon is so-called cover source mismatch (CSM). The CSM problem exists in both two kinds of steganalysis methods. In order to solve the problem, Lubenko et al. [26] concluded that simpler classifiers, such as the perceptron or the ensemble classifier, indeed appear to produce more robust detectors than more complex machine learning tools. Kodovský et al. [19] employs a bank of detectors trained on multiple different sources and tests on a detector trained on the closest source. Li and Kong [23] [20] propose a framework based on transfer learning to reduce difference of feature statistics between training set and test set.

This work firstly analyzes the reason for problems of CSM in steganalysis. Then based on deep learning, this work proposes a spatial steganalytic scheme for steganography algorithms recognition, which aims at improving the testing accuracy on images from unseen source. For a given testing image, the proposed scheme firstly generates a match image by performing a Gaussian filtering on the testing image to remove the possible stego signal. Then the corresponding stegos of the match image are obtained using recognized steganography algorithms. A CNN trained on a known image set is used to extract features from the testing

image and the match image and its corresponding stegos. The final decision is made according to the similarity between the testing feature and each match feature.

2 Related works

The related works are comprised of two parts: residual filters in steganalysis and convolutional neural networks (CNN) for steganalysis.

2.1 Residual filters in steganalysis

As stated in [15], to avoid attack, some steganography researchers proposed to model the cover-characteristics, since preserving the cover-distribution is essential in providing security. It could be possible to code the secret information in such a way that it is statistically indistinguishable from true random noise. The secret information could then be placed in highly noisy image regions. They substitute insignificant parts (e.g. the noise component) of the cover with the secret message.

For this reason, steganalysis researchers focus on only the noise component (noise residual) of images. Usually, a residual filter suppressing the image content and exposing the stego noise is applied. Examples of residual filters are shown in Fig. 1. The signal to noise ratio is improved by this method. Here, “signal” is the stego noise and “noise” is the image content.

2.2 Convolutional neural networks for steganalysis

Deep learning technology allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction [22]. Convolutional neural network (CNN) is a kind of deep learning models. A typical CNN is composed of two kinds of layers: the convolutional layer, which usually follows the activation and pooling operation, and the inner product layer, which follows the activation operation, as shown in Fig. 2. Convolutional layers combine two architectural ideas: local regions and shared weights. Firstly, using shared weight and local region strategy, convolutional layers have much fewer connections and weight parameters, compared to standard feed forward neural networks. Hence, CNNs are much easier to train, while theoretically-best performance is likely to be only slightly worse [21]. Secondly, convolutional layers perform fltering operations which search some kind of local feature. Hence, CNNs are very suitable for the

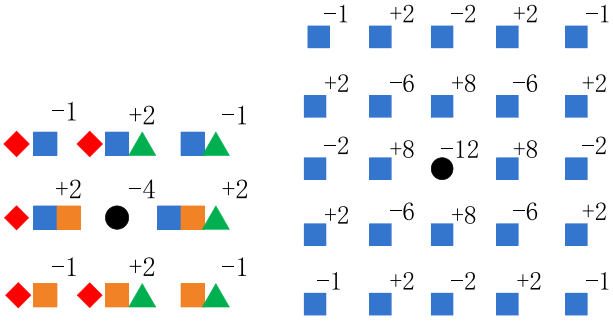

Fig. 1 Examples of residual filters used in steganalysis

input data with strong local correlation, such as images. So CNNs can have deeper layers which correspond to more abstract level data representation.

In paper [10]-[33, 39]-[41], researchers made many constructive designs on the structure of CNN, and proved CNNs being high-performance tools for steganalysis. A fixed high-pass filtering (HPF) layer is used to remove image content and improve signal to noise ratio for steganalysis; batch Normalization (BN) layers preserve the spatial correlations between elements; An absolute activation (ABS) layer prevent interference between feature values at early stages of statistical modeling. Besides, ensemble strategies of CNN models are also proposed for steganalysis.

3 Problem analysis and motivation

In this section, we try to analyze the causes of problem of CSM, and point out problem of CSM exists objectively in steganalysis. Based on the analysis, the motivation of the designed scheme is explained.

3.1 Analysis of CSM in steganalysis

In steganalysis, one problem has been noticed by many researchers. As stated in paper [19], when a steganalysis detector trained on one cover source is applied to images from a different source, generally the detection error increases. This issue is recognized as so-called cover source mismatch (CSM) by general public in steganalysis community.

The difficulty of steganalysis is that it is difficult to get rid of the interference of image content signals, and focuses on the analysis of the possible stego signals. Previous studies have shown that the two signals are difficult to quantify and distinguish after embedding. The residual filter method in stated in section 2.1 aims at removing image content signal and exposing the possible stego signal using residual filters. An example of a simple residual filter is

$$z_{ij} = x_{i,j+1} - x_{i,j}, \tag{1}$$

which computes the difference between a pair of horizontally neighboring pixels. This method weakens the pixel value itself, and retains the difference between adjacent pixels. The residual

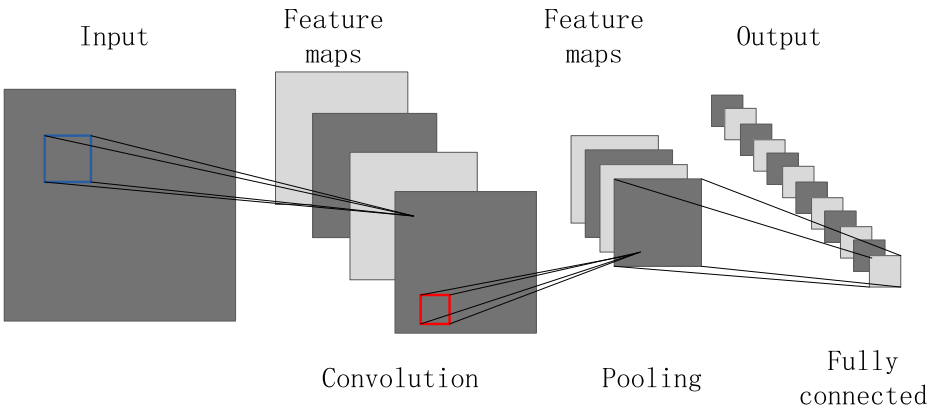

Fig. 2 Convolutional neural network example

differences are the noise part of the original image. However, steganography algorithms with high security often hide secret information statistically indistinguishable from true random noise. Modification on pixels could be $\{-1, 0, +1\}$. For a given pixel, the modification probability of -1 and $+1$ are equal. It is assumed that stego signal S is a random signal that obeys the Gauss distribution.

$$S \sim N\left(0, \sigma^2\right) \tag{2}$$

$x_{m,n}$ and $x_{p,q}$ are two pixels in cover image, and their residual values $z_{m,n}$ and $z_{p,q}$ are computed as

$$z_{m,n} = x_{m,n+1} - x_{m,n} \qquad z_{p,q} = x_{p,q+1} - x_{p,q} \tag{3}$$

$x'_{m,n}$ and $x'_{p,q}$ are the two corresponding pixels in stego image. $s_{m,n}$ and $s_{p,q}$ are observational values of random variables S.

$$x'_{m,n} = x_{m,n} + s_{m,n} \qquad x'_{m,n+1} = x_{m,n+1} + s_{m,n+1} \tag{4}$$

$$x'_{p,q} = x_{p,\mathrm{q}} + s_{p,q} \qquad x'_{p,q+1} = x_{p,q+1} + s_{p,q+1} \tag{5}$$

Residual values of $x'_{m,n}$ and $x'_{p,q}$ are $z'_{m,n}$ and $z'_{p,q}$, which are computed as

$$z'_{m,n} = x'_{m,n+1} - x'_{m,n} = x_{m,n+1} - x_{m,n} + s_{m,n+1} - s_{m,n} \tag{6}$$

$$z'_{p,q} = x'_{p,q+1} - x'_{p,q} = x_{p,q+1} - x_{p,q} + s_{p,q+1} - s_{p,q} \tag{7}$$

The relationship between the two residual values can be described as

$$r_1 = z_{m,n} - z_{p,q} = \left(x_{m,n+1} - x_{m,n}\right) - \left(x_{p,q+1} - x_{p,q}\right) \tag{8}$$

$$r_2 = z'_{m,n} - z'_{p,\mathrm{q}} = \left(x_{m,n+1} - x_{m,n} + s_{m,m+1} - s_{m,n}\right) - \left(x_{p,q+1} - x_{p,q} + s_{p,q+1} - s_{p,q}\right), \tag{9}$$

in which r_1 represents the relationship between the two residual values in cover image, while r_2 represents the relationship between the two residual values in stego image. Define the variable r to measure the differentiability of the relationship between the two residual values before and after embedding.

$$r = r_1 - r_2 = \left(s_{m,m+1} - s_{m,n}\right) - \left(s_{p,q+1} - s_{p,q}\right) \tag{10}$$

$$E(r) = E\left(\left(s_{m,m+1} - s_{m,n}\right) - \left(s_{p,q+1} - s_{p,q}\right)\right) = 0 \tag{11}$$

$E(.)$ represents computing mathematical expectation. $E(r) = 0$, so it could be difficult to distinguish the relationship between the two residual values before and after embedding. This inference can be extended to other residuals filtering.

In fact, method of residual filter can suppressing the image content to some extent, however, the true random noise of image remains in the filtered image. As shown in Fig. 3, the profile and texture of the original image content are retained in the filtered image. The residual true random noise is also a kind of image content, it 'preserves' stego signals from being detected and have a negative impact on steganalysis.

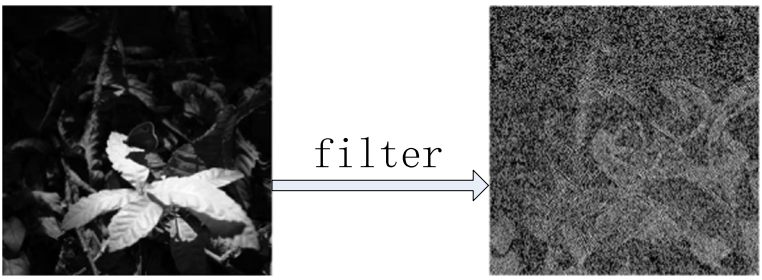

Fig. 3 Original image and filtered image

The negative impact may be one of the causes of problem of CSM, since the residual image content signals can be easily learned by the detector. In eq. (12), *Det* represents a trained detector, *Tra(.)* represents process of training.

$$Det = Tra(struc, ini\_w, E, steg, payload, key) \tag{12}$$

struc represents the structure of detector, *ini_w* represents initialized parameters before training, and *E* represents the set of images used for training. *Steg*, *payload* and *key* represent the used steganographic algorithms, payload value and key values respectively. **Variable-controlling approach** is performed to control *struc*, *ini_w*, *Steg*, *payload* and *key* remain unchanged. An obvious phenomenon: if different detectors are trained on different image sets respectively, the weight parameters in trained detectors will be very different, even if the training set is composed of residual filtered images. This phenomenon implies that detectors can learn and memorize some knowledge related to image sets in training phase, and rely on this knowledge to implement classification to some extent. This knowledge related to image set has nothing to do with steganography. It leads to a decrease in the testing accuracy when the training image and the testing image come from different source, since the statistical properties of image content are different between testing image and training image.

3.2 Motivation

In machine learning field, deep CNN model has been proved to be effective in many classification tasks. Moreover, CNN has been proved to perform better on testing accuracy than the traditional machine learning methods in steganalysis. The motivation of this paper: Firstly, we take advantage of CNN model for its inherent excellent performance in testing accuracy; secondly, based on the problem analysis in the previous sub-section, we propose an anti-CSM scheme specifically for steganalysis, which improves the generalization ability of the detector model.

Based on the above discussion, this paper attempts to provide a ‘reference’ for the trained CNN model to help reduce the negative impact of residual image content signals. For a given testing image, the ‘reference’ should be a match image satisfying the following two conditions:

Firstly, the match image should not contain stego signals;
Secondly, the match image and the testing image should be similar in image content.

The match image and testing image satisfying the above conditions are successively input to a trained CNN, match feature and testing feature are obtained from the output. The image content information contained in match feature will be similar to that in testing feature, and

stego related information does not contain in match feature. Further, the image content information in two features cancels each other out, highlighting the possible stego information in the testing feature.

Based on the above idea, a specific scheme to solve the CSM problem caused by the interference of image content signals is proposed. We refer to paper [36] in face verification domain, and revised the scheme to steganography algorithms recognition. The proposed scheme is detailed in the next section.

4 Proposed steganalytic scheme

In this section, we details the proposed method used for steganography algorithms recognition. Steganography algorithms recognition is essentially a multi-classification problem:

$$class = \{empty, A, B, \ldots\}_n.$$

Here *class* is the set of steganography algorithms, *empty* means no embedding and *A*, *B*, … mean different steganography algorithms. The proposed steganalytic scheme assumes that the Warden masters the recognized steganography algorithms.

For a given testing image, the proposed steganalytic scheme is comprised of three parts. In the first part, we obtain a match image. We generate the match image by performing a Gaussian filtering on the testing image to remove the possible stego signal. Then the corresponding stegos of the match image are obtained using recognized steganography algorithms. Notice that the match cover image and its corresponding stegos are collectively called match images hereinafter. In the second part, a designed CNN is trained on training set. Then the trained CNN is used to extract deep features from the testing image and its match images. In the third part, we verify whether two CNN-extracted features belong to the same class or not. One feature is extracted from the testing image, the other is extracted from one of the match images. The final decision that which class the testing image belongs to is made via this verification. The overall steganalytic scheme overviewed above is shown in Fig. 4. In Fig. 4 f_t is the feature extracted from the testing image. f_{me}, f_{mA}, f_{mB}…… are features extracted from match images.

4.1 Obtaining match image

Faced with CSM problem, we focus on eliminating the disturbance of image content. A match image is obtained for reference. We generate the match image by performing a low pass

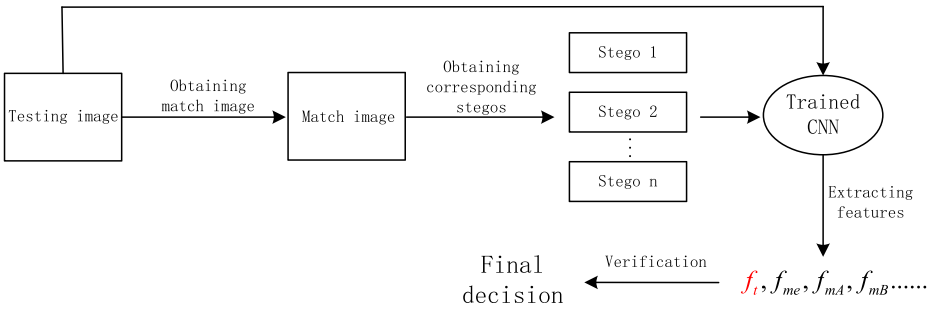

Fig. 4 Framework of proposed scheme

Gaussian filtering on the testing image to remove the possible stego signal. As stated in paper [32], generally, the very weak stego noise added to the cover is a kind of high frequency signal, and thus could be removed by a low pass filter. Here we make a filtering operation with low pass Gaussian filter to generate match image to remove the possible stego signal as completely as possible.

The Gaussian filter is a class of linear smoothing filters that select weights based on the shape of the Gaussian function. The Gaussian smoothing filter is very effective in suppressing noise that obeys normal distribution. For images, the Gaussian filter is a two-dimensional convolution operator using the Gaussian kernel. Equation (13) shows the two-dimensional Gaussian distribution,

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{13}$$

and the convolution operator (size=5 × 5, $\sigma = 1.0$) is shown as (14).

$$K_G = \frac{1}{273}\begin{pmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{pmatrix} \tag{14}$$

We generate the match image using Eq. (15),

$$I_m = conv2(I_t, K_G) \tag{15}$$

Here *conv*2(.) denotes two-dimensional convolutional process with mode *same* in matlab.

As shown in Fig. 5, in order to remove the possible stego signal, Gaussian filtering smooths the image. However, the relative texture complexity between the local regions of the image is not changed, which means embedding on match image is roughly similar to that on the original cover image.

For the match image I_m, we can easily obtain the corresponding stego images $M_I = \{I_m, I_{m,A}, I_{m,B}, \ldots\}_n$ using steganography algorithms $\{A, B, \ldots\}_{n-1}$, where $I_{m,A}$ is the stego image of steganography algorithms A on I_m. In section 4.2, a trained deep CNN is used to extract features from testing image and its corresponding match images.

4.2 CNN-extracted deep features

The feature is the low-dimension representation of image, it should reflect the possible stego signal distinctly. As is stated in instruction, deep CNN is able to learn representation of image

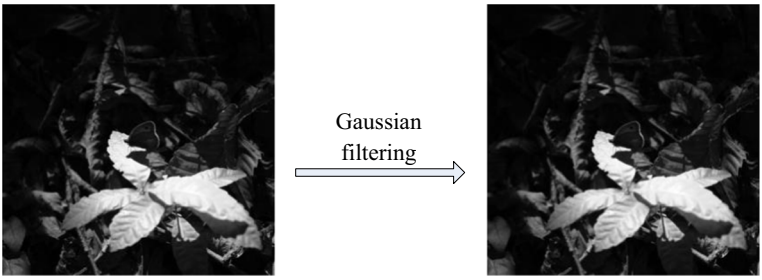

Fig. 5 Obtaining match image from testing image using Gaussian filtering

with multiple levels of abstraction, and has been proved to be effective for steganalysis. We train a deep CNN on a known training image set. Then the trained CNN is used to extract features from images. The trained CNN takes in an image, and after forward pass, the neurons in the penultimate layer are extracted as steganalysis feature. Compared with hand-extracted features, CNN-extracted features are generally high-level ones, and we call these features deep features. Our proposed method uses the structure of CNN structure stated in paper [39].

4.2.1 Structure review

Fig. 6 illustrates the overall architecture of used CNN. An HPF layer whose parameters are not optimized during training, is placed at the very beginning to transform original images to noise residuals to enhance the signal–to-noise/interference ratio. The CNN structure consists of two modules: one is a convolutional module which transforms the images to feature vectors (128-D), the other is a linear classification module which transforms feature vectors to output probabilities for each class. Convolutional layers, pooling layers, BN layers, TanH layers, ReLU layers, and an ABS layer are placed in the convolutional module, while a full-connected layer and a softmax layer are placed in the linear classification module. For more details about the CNN structure, refer to paper [39].

Pay attention that the linear classification module which transforms feature vectors to output probabilities for each class is only used in training phase. In test phase, the 128-D feature output by the last pooling layer is the finally extracted deep feature.

Denote function $F(.)$ as the process of CNN on an input image in test phase. The feature extraction can be described as the following equation:

$$f = F(I) \tag{16}$$

where I is an input image, and f is the extracted feature. By feature extraction from testing image I_t, we obtain the testing feature

$$f_t = F(I_t) = \left(x_1^t, x_2^t, \ldots, x_{128}^t\right) \tag{17}$$

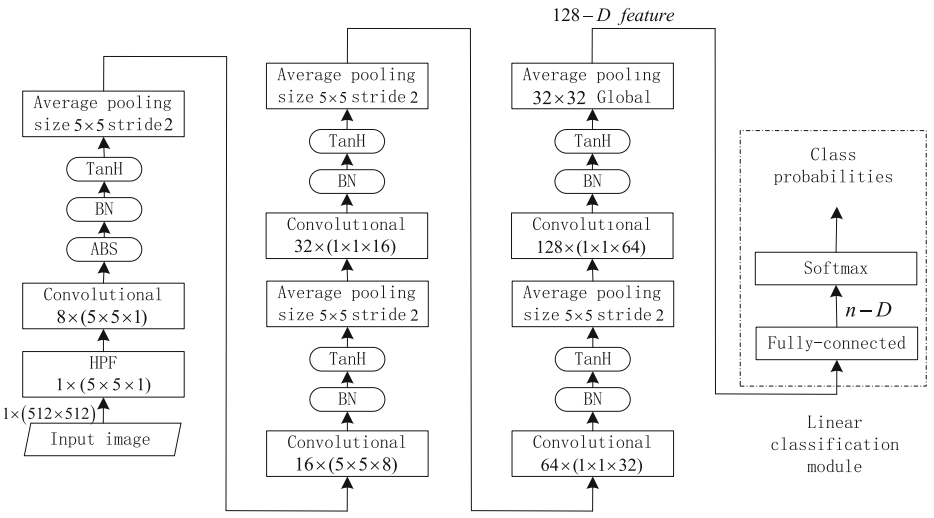

Fig. 6 Structure of used CNN

By feature extraction from match images, we obtain match features

$$M_f = \{f_m, f_{m,A}, f_{m,B}, \ldots\}_n \tag{18}$$

Each element in M_f is a feature extracted from one of match images, for example

$$f_m = F(I_m) = \left(x_1^m, x_2^m, \ldots, x_{128}^m\right) \tag{19}$$

$$f_{m,A} = F\left(I_{m,A}\right) = \left(x_1^{m,A}, x_2^{m,A}, \ldots, x_{128}^{m,A}\right) \tag{20}$$

$$f_{m,B} = F\left(I_{m,B}\right) = \left(x_1^{m,B}, x_2^{m,B}, \ldots, x_{128}^{m,B}\right) \tag{21}$$

Based on the obtained deep features (testing feature f_t and match features in M_f), we will measure the similarity between testing feature f_t and each match feature in M_f in section 4.3 in order to reach the final decision.

4.2.2 Performance analysis

As well known, deep CNN is able to learn representations of data with multiple levels of abstraction. Whether the CNN stated in 3.2.1 can expose stego signal layer by layer or not directly determine the performance of the proposed steganalytic scheme. To this end, we let a cover image and one of its corresponding stego image (embedded using *S-UNIWARD*, payload = 0.4bpp) go through the trained CNN in forward propagation. After the forward propagation, we take out the activated feature maps in each convolutional module. Values in each feature map are normalized and scaled to range of 0 to 255, then displayed Fig. 6. In Fig. 6, we also show the difference between cover and stego of feature maps in each convolutional module. These difference maps are obtained by feature maps subtraction, and are also normalized and scaled to range of 0 to 255.

As shown in Fig. 7, though we can hardly see any distinct texture in feature map of conv1, the texture is becoming highlighted from conv2 to conv5. Obviously, the textures in feature maps of conv2 to conv5 are similar to the texture in original image to some degree. That means the CNN has taken notice of regions with complicated texture, where pixels are more likely to be modified. Moreover, pay attention to the difference map from conv1 to conv5. Spots that are different from the background gray-scale are becoming highlighted, that means some signal difference in cover and stego is becoming more and more distinct in deeper layers. The analysis above explains the trained CNN does have the ability to expose stego signal. Yet residual image content in feature maps of deep layers could still disturb classification. Steganalytic performance could be improved by eliminating influence of image content further.

4.3 Verification

The CSM problem is an objective phenomenon in the detection of steganography. The cause of this phenomenon is likely to be the interference of the image content. Methods of computing the similarity between testing feature and match features can effectively eliminate the interference of image content, and thus alleviate CSM problem. Since the match image is generated via a low pass Gaussian filter, the match image has similar image content with the testing

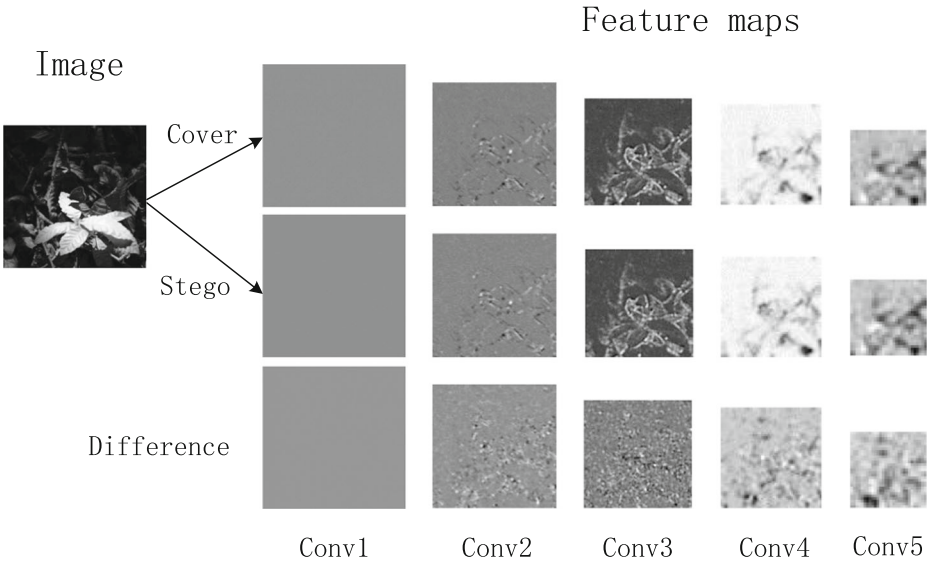

Fig. 7 Feature maps in each convolutional layer their difference between cover and stego

image. CNN-extracted testing feature and match feature should also share similar image content signal. Computing the similarity between testing feature and match features is to computing the similarity of the possible stego signals as shown in Fig. 8. The interfering image content signal is offset in the verification.

4.3.1 Feature normalization

After the process stated in subsection 4.1 and section 4.2, we have got testing feature f_t which is the feature of the testing image, and a match features set M_f, which consists of features of match images. Before verification, we normalize these features to be between zero and one: Each component of the feature vector is divided by its largest value across the training set. This is then followed by L2-normalization:

$$f_i = \frac{f_i}{\max(f_{i\max}, \varepsilon)} \quad , \quad f_i \in f \tag{22}$$

$$f = \frac{f}{\|f\|_2} \tag{23}$$

where f is a feature vector extracted by trained CNN, f_i is the ith element in f and $f_{i\max}$ is the largest value of ith element across the training set. ε is set to 0.01 in order to avoid division by a small number. The similarity between normalized features is obtained via optional two ways respectively stated in 4.3.2 and 4.3.3.

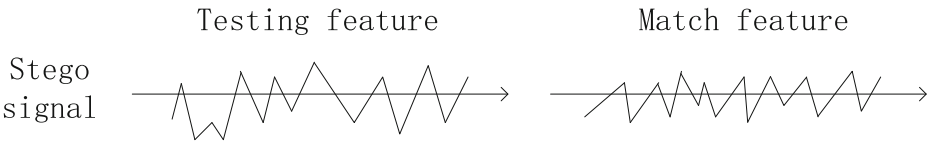

Fig. 8 Comparison between stego signals in testing feature and match feature

4.3.2 Weighted-χ^2 similarity

The weighted-χ^2 similarity is a kind of supervised method of computing similarity. The weighted-χ^2 similarity can be defined as

$$s_{m,A} = \chi^2_{\omega}\left(f_t, f_{m,A}\right) = \sum_{i=1}^{128} \omega_i \frac{\left(x_i^t - x_i^{m,A}\right)^2}{x_i^t + x_i^{m,A}} \tag{24}$$

to measure the similarity between normalized f_t and each normalized feature in M_f. $s_{m,A}$ is the measure of similarity between f_t and $f_{m,A}$. The weight parameters are learned using a linear FLD (Fisher Linear Discriminant) [7], which is also trained on the training image set.

4.3.3 Inner production similarity

The weighted-χ^2 similarity is a kind of unsupervised method of computing similarity. The inner product between normalized f_t and each normalized feature in M_f is computed in order to measure the similarity. The higher the inner product value, the higher the similarity. Here $\cdot$ means inner production operation.

$$s_{m,A} = f_t \cdot f_{m,A} \tag{25}$$

After going through all features in M_f, we obtain an ordered set

$$Sim = \left\{s_m, s_{m,A}, s_{m,B}, \ldots\right\}_n \tag{26}$$

Each element in *Sim* is the similarity between the testing feature f_t and each corresponding match feature in M_f. The class of max value in *Sim* is the final decision.

5 Experiments

Steganography algorithms used in experiments are firstly introduced in section 5.1. In section 5.2, we introduce the details of training deep CNN. In section 5.3, we perform experiments to test the performance on recognition of three steganography algorithms: *S-UNIWARD*, *LSB matching* [27] and *empty* (no embedding). In section 5.4, we analyze the effect of parameter σ in Gaussian filter on detection. In order to further explore the impact of number of classes on classification performance, we also add several steganographic algorithms in experiments, and show the performance in section 5.5. In section 5.6, we explore the performance of proposed method on detecting steganography algorithms unknown in training set.

All of the experiments were performed on a modified version of Caffe toolbox [14], Matlab platform and a workstation with Nvidia Tesla K40 GPU.

5.1 Brief introduction of steganography algorithms

Here we briefly introduce the experimental steganography algorithm, in no particular order.

5.1.1 HUGO_BD

HUGO was proposed in paper [31]. The main design principle of *HUGO* is to minimize a suitably-defined distortion by means of efficient coding algorithm. The distortion is defined as a weighted difference of extended state-of-the-art feature vectors already used in steganalysis. This allows Steganographier to "preserve" the model used by steganalyst and thus be undetectable even for large payloads. *HUGO BD* is steganography based on *HUGO* implemented via the bounding distortion [8].

5.1.2 S-UNIWARD

S-UNIWARD was proposed in paper [13]. The proposed design called universal wavelet relative distortion (*UNIWARD*) can be applied for embedding in spatial domain. The embedding distortion is computed as a sum of relative changes of coefficients in a directional filter bank decomposition of the cover image. The directionality forces the embedding changes to such parts of the cover object that are difficult to model in multiple directions, such as textures or noisy regions, while avoiding smooth regions or clean edges.

5.1.3 WOW

WOW was proposed in paper [11]. The proposed approach defined an additive steganographic distortion in the spatial domain. The change in the output of directional high-pass filters after changing one pixel is weighted and then aggregated using the reciprocal Hölder norm to define the individual pixel costs. In contrast to other adaptive embedding schemes, the aggregation rule is designed to force the embedding changes to highly textured or noisy regions and to avoid clean edges.

5.1.4 MG

MG was proposed in paper [34]. The method allows the embedding changes in highly textured areas to have larger amplitude and thus embedding there a larger payload. The approach is entirely model driven in the sense that the probabilities with which the cover pixels should be changed by a certain amount are derived from the cover model to minimize the power of an optimal statistical test. The embedding consists of two steps. First, the sender estimates the cover model parameters, the pixel variances, when modeling the pixels as a sequence of independent but not identically distributed generalized Gaussian random variables. Then, the embedding change probabilities for changing each pixel by 1 or 2, which can be transformed to costs for practical embedding using syndrome-trellis codes, are computed by solving a pair of non-linear algebraic equations.

5.1.5 MiPOD

MiPOD was proposed in paper [35]. The approach is based on a locally-estimated multivariate Gaussian cover image model that is sufficiently simple to derive a closed-form expression for the power of the most powerful detector of content-adaptive *LSB matching* but, at the same time, complex enough to capture the non-stationary character of natural images. The closed-form expression for detectability within the chosen model is used to obtain new fundamental insight regarding the performance limits of empirical steganalysis detectors built as classifiers.

5.1.6 Pentary MVG

Pentary MVG was proposed in paper [10]. The strategy is that the cover is modeled as a sequence of independent but not necessarily identically distributed quantized Gaussians and the embedding change probabilities are derived to minimize the total KL divergence within the chosen model for a given embedding operation and payload.

5.1.7 LSB matching

As stated in paper [27], *LSB replacement* embeds a message into the cover image by replacing the LSBs of the cover image with message bits to arrive at the stego image. The method increases even pixel values either by one or leaves them unmodified, while odd values are left unchanged or decreased by one. As a result, there exists an imbalance in the embedding distortion in the stego image. Due to this imbalance, *LSB replacement* can be easily detected.

LSB matching also modifies the LSBs of the cover image for message embedding. *LSB matching* does not simply replace the LSBs of the cover image as *LSB replacement* does. Instead, if the message bit does not match the LSB of the cover image, then one is randomly either added or subtracted from the value of the cover pixel.

5.2 Training deep CNN

A deep CNN whose structure is stated in section 4.2 is trained on 5000 cover images in BOSSbase v1.01 [1] and their corresponding stegos. The CNN is validated on other 2000 images in BOSSbase and their corresponding stegos during training. The remaining 3000 images are left for testing.

Like many conventional CNNs, Mini-batch GD is used in the training phase of the used CNN, and the batch size for each iteration of training is fixed to 32. The momentum is fixed to 0.9. The learning rate is initialized to 0.001 and schedule to decrease 0.1 every 5000 iterations. Parameters in convolution kernels and full-connected layer are initialized by random numbers generated from zero-mean Gaussian distribution with standard deviation of 0.01; bias learning is disabled in convolutional layers and inner product layers. The weight decay in convolutional layers and inner product layers is set to 1(disabled).

Taking the training on samples of *S-UNIWARD*, *LSB matching* and *empty* as an example, the relationship between validation accuracy and iterations is shown in Fig. 9.

We choose the CNN models that achieve best accuracy performance on validation set, as shown in Table 1. These CNN models are used in test phase.

5.3 Detecting S-UNIWARD, LSB matching and empty

In test phase, using the trained CNN models, we test the performance of proposed steganalytic scheme on two different image set. One is the remaining 3000 images in Bossbase 1.01 that are from the same source with the training image set. The other is an image dataset we built by ourselves, and we call the dataset Netbase. The Netbase consists of 500 color images with arbitrary sizes. All images in Netbase are collected from Internet, these images may be pictures taken by different models of cameras or computer-generated cartoon images, and may even be edited with unknown post processing. The generation and processing of these images is unknown, so it conforms to the real steganalytic scenario. All images in Netbase are converted

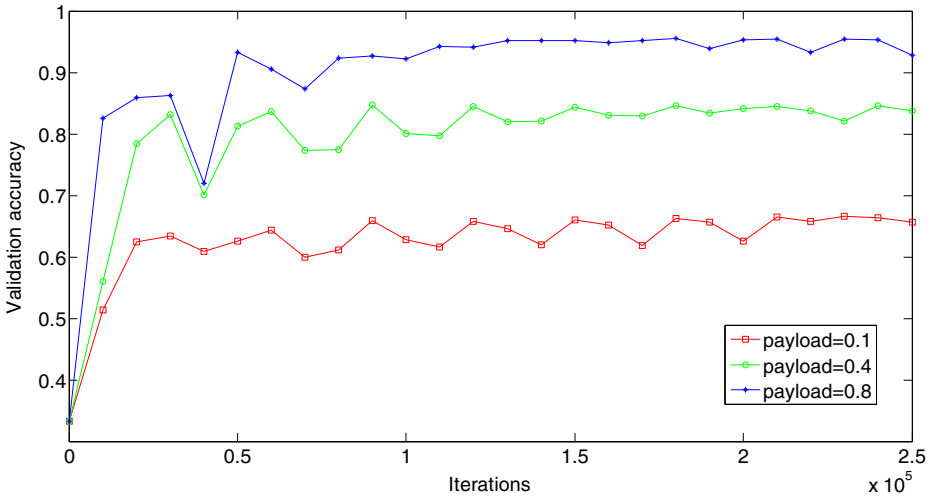

Fig. 9 The relationship between validation accuracy and iterations in training phase

to grey-level and resized to 512 × 512 before experiment. All images of the two dataset are embedding using the recognized steganography algorithms.

We take into account that one of the key purposes of steganography algorithm is to further extract secret information. *S-UNIWARD* and *LSB matching* are representatives of two kinds of algorithms differing on means of secret information extraction. *SUNIWARD* is a representative of steganographic algorithm in which secret information is extract by means of parity-check matrix, and *LSB matching* is a representative of steganographic algorithm in which secret information is extract not using parity-check matrix. The testing accuracy of detecting *S-UNIWARD*, *LSB matching* and *empty* under different methods is shown in Table 2.

The accuracy displayed in Table 2 is the average accuracy of all the three class (*S-UNIWARD*, *LSB matching* and *empty*). The set of used parameters of Gaussian filter is shown in Table 3.

Obviously, though the testing accuracy of proposed method is 0.01–0.02 lower than that of directly used CNN on BOSSbase, the proposed method shows competitive accuracy performance than that of directly used CNN on Netbase. The testing accuracy of proposed method is 0.02–0.12 higher than that of directly used CNN on Netbase.

Table 1 CNN models used in test phase

Payload(bpp)	0.1	0.4	0.8
Iterations	230,000	180,000	180,000
Validation accuracy	0.6663	0.8455	0.9552

Table 2 Testing accuracy under different methods

bpp	CNN directly		Proposed method(w-χ^2)		Proposed method(IP)	
	BOSSbase	Netbase	BOSSbase	Netbase	BOSSbase	Netbase
0.1	0.6579	0.6213	0.6493	0.6420	0.6403	0.6453
0.4	0.8225	0.6740	0.8149	0.7649	0.8045	0.7833
0.8	0.9336	0.8507	0.9226	0.9100	0.9197	0.9253

Table 3 Parameters of Gaussian filter used in proposed method

bpp	Parameters for BOSSbase	Parameters for Netbase
0.1	$\sigma = 0.8$, $size = 3 \times 3$	$\sigma = 1.0$, $size = 3 \times 3$
0.4	$\sigma = 0.8$, $size = 3 \times 3$	$\sigma = 1.1$, $size = 3 \times 3$
0.8	$\sigma = 1.1$, $size = 3 \times 3$	$\sigma = 1.1$, $size = 3 \times 3$

Further observation of testing accuracy of each class on Netbase is shown in Tables 4, 5, and 6. Tables 4, 5, and 6 not only shows the correct testing accuracy each class, but also shows the probability that samples with specific label are classified to other class.

As shown in Table 4, when the CNN model trained on BOSSbase is tested on Netbase, it almost fails to detect *empty* samples especially under low payload. Most of the *empty* samples are identified as *S-UNIWARD*.

When tested on Netbase, the proposed methods with weighted-χ^2 or inner production either shows competitive testing accuracy with directly used CNN, and give relatively balanced testing accuracy on each class.

Table 4 Accuracy performance of directly used CNN on Netbase

bpp	Test label	Testing accuracy		
		empty	S-UNIWARD	LSB matching
0.1	empty	**0.098**	0.782	0.120
	S-UNIWARD	0.078	**0.802**	0.120
	LSB matching	0	0.036	**0.964**
0.4	empty	**0.078**	0.922	0
	S-UNIWARD	0.040	**0.960**	0
	LSB matching	0	0.016	**0.984**
0.8	empty	**0.618**	0.382	0
	S-UNIWARD	0.058	**0.942**	0
	LSB matching	0	0.008	**0.992**

The bold data mean correct testing accuracy

Table 5 Accuracy performance of proposed method with weighted-χ^2 on Netbase

bpp	Test label	Testing accuracy		
		empty	S-UNIWARD	LSB matching
0.1	empty	**0.448**	0.474	0.078
	S-UNIWARD	0.418	**0.494**	0.088
	LSB matching	0.008	0.008	**0.984**
0.4	empty	**0.720**	0.258	0.022
	S-UNIWARD	0.384	**0.598**	0.018
	LSB matching	0.010	0	**0.990**
0.8	empty	**0.870**	0.098	0.032
	S-UNIWARD	0.096	**0.882**	0.022
	LSB matching	0.006	0.016	**0.978**

The bold data mean correct testing accuracy

Table 6 Accuracy performance of proposed method with inner production on Netbase

bpp	Test label	Testing accuracy		
		empty	S-UNIWARD	LSB matching
0.1	empty	**0.450**	0.470	0.080
	S-UNIWARD	0.418	**0.502**	0.080
	LSB matching	0.006	0.010	**0.984**
0.4	empty	**0.744**	0.368	0.020
	S-UNIWARD	0.238	**0.618**	0.016
	LSB matching	0.010	0.008	**0.988**
0.8	empty	**0.944**	0.048	0.008
	S-UNIWARD	0.074	**0.898**	0.028
	LSB matching	0	0.066	**0.934**

The bold data mean correct testing accuracy

5.4 Analysis of parameter σ with inner production method

Based on the experimental results of detecting *S-UNIWARD*, *LSB matching* and *empty* with inner production method, we analyze the effect of parameter σ in Gaussian filter on detection. Under inner production method, using Gaussian filter in size of 3×3, testing accuracy of each class under σ in range of 0.8–1.2 is shown in Tables 7, 8, 9, 10, 11, and 12.

From the experiment results shown in Tables 7, 8, 9, 10, 11, and 12, it is easily observed that, as the value of σ grows, testing accuracy of empty increases while that of S-UNIWARD decreases under payload of 0.1 and 0.4, and this phenomenon not preciously shows under Netbase and payload 0.8. Accuracy of *LSB matching* and average accuracy are not affected noticeably. This means that we can control the accuracy of the detection on *empty* and *S-UNIWARD* by adjusting the value of the parameter σ.

Table 7 Testing accuracy under different σ (Bossbase v1.01, payload = 0.1)

Test label	Testing accuracy under different σ				
	0.8	0.9	1.0	1.1	1.2
empty	0.3833	0.4137	0.4450	0.4565	0.4633
S-UNIWARD	0.5717	0.5412	0.5077	0.4879	0.4697
LSB matching	0.9660	0.9658	0.9627	0.9613	0.9610
average	0.6403	0.6402	0.6384	0.6352	0.6313

Table 8 Testing accuracy under different σ (Bossbase v1.01, payload = 0.4)

Test label	Testing accuracy under different σ				
	0.8	0.9	1.0	1.1	1.2
empty	0.8023	0.8109	0.8173	0.8287	0.8393
S-UNIWARD	0.6220	0.6107	0.5920	0.5853	0.5680
LSB matching	0.9893	0.9890	0.9890	0.9891	0.9883
average	0.8045	0.8035	0.7994	0.8010	0.7985

Table 9 Testing accuracy under different σ (Bossbase v1.01, payload = 0.8)

Test label	Testing accuracy under different σ				
	0.8	0.9	1.0	1.1	1.2
empty	0.7960	0.7998	0.8030	0.8065	0.8097
S-UNIWARD	0.9663	0.9632	0.9607	0.9576	0.9534
LSB matching	0.9953	0.9951	0.9947	0.9950	0.9943
average	0.9192	0.9194	0.9195	0.9197	0.9191

Table 10 Testing accuracy under different σ (Netbase, payload = 0.1)

Test label	Testing accuracy under different σ				
	0.8	0.9	1.0	1.1	1.2
empty	0.204	0.326	0.450	0.458	0.498
S-UNIWARD	0.664	0.584	0.502	0.428	0.367
LSB matching	0.982	0.986	0.984	0.978	0.980
average	0.6233	0.6320	0.6453	0.6213	0.6150

5.5 Detecting more steganography algorithms

Firstly, we perform the proposed scheme on detecting four algorithms: *HUGO BD*, *S-UNIWARD*, *WOW* and *empty*. Pay attention that the CNN model is trained again on the four algorithms. Testing accuracy of detecting four algorithms under different methods are shown in Table 13. More specifically, testing accuracy of each class on Netbase is shown in Table 14.

As shown in Tables 13 and 14, using CNN directly, the testing accuracy dropped significantly when testing set turns to Netbase. Also the testing accuracy on each class of is irregular

Table 11 Testing accuracy under different σ (Netbase, payload = 0.4)

Test label	Testing accuracy under different σ				
	0.8	0.9	1.0	1.1	1.2
empty	0.466	0.506	0.614	0.744	0.868
S-UNIWARD	0.840	0.742	0.738	0.618	0.492
LSB matching	0.988	0.986	0.990	0.988	0.982
average	0.7646	0.7446	0.7806	0.7833	0.7807

Table 12 Testing accuracy under different σ (Netbase, payload = 0.8)

Test label	Testing accuracy under different σ				
	0.8	0.9	1.0	1.1	1.2
empty	0.816	0.852	0.902	0.944	0.940
S-UNIWARD	0.922	0.908	0.876	0.898	0.922
LSB matching	0.982	0.980	0.956	0.934	0.910
average	0.9067	0.9133	0.9113	0.9253	0.9240

Table 13 Testing accuracy of detecting of four algorithms under different methods

bpp	CNN directly		Proposed method (w-χ^2)		Proposed method (IP)	
	BOSSbase	Netbase	BOSSbase	Netbase	BOSSbase	Netbase
0.1	0.2551	0.2495	0.2533	0.2520	0.2515	0.2525
0.4	0.4593	0.2605	0.4467	0.3915	0.4039	0.4125
0.8	0.6339	0.3110	0.6326	0.5585	0.6244	0.5945

Table 14 Testing accuracy for each class under different methods on Netbase

Method	Test label	Payload		
		0.1	0.4	0.8
CNN directly	empty	0.382	0.024	0.206
	HUGO BD	0.022	0.440	0.116
	S-UNIWARD	0.196	0.002	0.040
	WOW	0.398	0.576	0.882
Proposed method (w-χ^2)	empty	0.312	0.406	0.522
	HUGO BD	0.248	0.382	0.568
	S-UNIWARD	0.234	0.366	0.604
	WOW	0.214	0.412	0.540
Proposed method (IP)	empty	0.298	0.440	0.644
	HUGO BD	0.204	0.414	0.608
	S-UNIWARD	0.242	0.362	0.562
	WOW	0.266	0.434	0.564

when using CNN directly on Netbase. The proposed scheme alleviates this phenomenon. Though the testing accuracy of the two proposed method is 0.01–0.04 lower than that of directly used CNN on BOSSbase, the two proposed method shows competitive accuracy performance than that of directly used CNN on Netbase. The testing accuracy of proposed methods is 0.01–0.28 higher than that of directly used CNN on Netbase.

Secondly, we perform the proposed scheme on detecting seven algorithms: *HUGO_BD*, *S-UNIWARD*, *WOW*, *MG*, *MVG*, *MiPOD* and *empty*. Pay attention that the CNN model is trained again on the seven algorithms. Testing accuracy of detecting seven algorithms under different methods are shown in Table 15. More specifically, testing accuracy of each class on Netbase is shown in Table 16. Experiment results showed in Tables 15 and 16 display a similar phenomenon as in Tables 13 and 14.

From these experimental results, we can see that testing accuracy drop distinctly when detecting more steganography algorithms. We consider the reason is that the number of class increases and several algorithms with high security are added in the experiment.

Table 15 Testing accuracy of detecting of seven algorithms under different methods

bpp	CNN directly		Proposed method(w-χ^2)		Proposed method(IP)	
	BOSSbase	Netbase	BOSSbase	Netbase	BOSSbase	Netbase
0.1	0.1431	0.1429	0.1424	0.1426	0.1433	0.1426
0.4	0.2951	0.1486	0.2884	0.2237	0.2653	0.2543
0.8	0.4336	0.2257	0.4157	0.3566	0.3928	0.3894

Table 16 Testing accuracy for each class under different methods on Netbase

Method	Test label	Payload		
		0.1	0.4	0.8
CNN directly	empty	0.082	0.060	0.202
	HUGO BD	0.002	0.002	0.066
	S-UNIWARD	0.026	0.002	0.122
	WOW	0.002	0.044	0.018
	MG	0.848	0.926	0.894
	MVG	0.002	0.004	0.144
	MiPOD	0.038	0.002	0.134
Proposed method (w-χ^2)	empty	0.104	0.208	0.366
	HUGO BD	0.078	0.236	0.302
	S-UNIWARD	0.206	0.192	0.354
	WOW	0.144	0.222	0.348
	MG	0.322	0.388	0.332
	MVG	0.102	0.126	0.398
	MiPOD	0.042	0.194	0. 396
Proposed method (IP)	empty	0.116	0.288	0.380
	HUGO BD	0.184	0.206	0.366
	S-UNIWARD	0.202	0.302	0.342
	WOW	0.066	0.234	0.424
	MG	0.118	0.206	0.420
	MVG	0.134	0.262	0.398
	MiPOD	0.178	0.282	0.396

5.6 Detecting steganography algorithms unknown in training set

At present, there are many steganography algorithms. Also new steganography algorithms are constantly proposed. One reality is that when training a detector model, it is impossible for analysts to take into account all possible steganography algorithms that may appear in the test. How does the proposed method perform when detecting steganography algorithms unknown in training set?

Denote model 1 as a CNN models training on samples of *empty*, *S-UNIWARD* and *LSB matching*; Denote model 2 as a CNN models training on samples of *empty*, *HUGO BD*, *S-UNIWARD* and *WOW*; Denote model 3 as a CNN models training on samples of *empty*, *HUGO BD*, *S-UNIWARD*, *WOW*, *MG*, *MVG* and *MiPOD*. Experimental results of detecting steganography algorithms unknown in training set are shown in Tables 17, 18, 19, 20, 21, and 22.

Table 17 Testing accuracy of algorithms unknown in training set under model 1 on BOSSbase

Method	bpp	Testing label				
		HUGO BD	WOW	MG	MVG	MiPOD
Proposed method (w-χ^2)	0.1	0.4637	0.4107	0.5380	0.3303	0.4011
	0.4	0..4817	0.4357	0.6020	0.3577	0.4610
	0.8	0.5483	0.4500	0.6483	0.3817	0.4830
Proposed method (IP)	0.1	0.4500	0.4000	0.5580	0.3263	0.3877
	0.4	0.4897	0.4057	0.5890	0.3470	0.4397
	0.8	0.5253	0.4170	0.6103	0.3513	0.4497

Table 18 Testing accuracy of algorithms unknown in training set under model 1 on Netbase

Method	bpp	Testing label				
		HUGO BD	WOW	MG	MVG	MiPOD
Proposed method (w-χ^2)	0.1	0.350	0.328	0.488	0.288	0.366
	0.4	0.408	0.342	0.506	0.308	0.420
	0.8	0.442	0.384	0.524	0.330	0.442
Proposed method (IP)	0.1	0.366	0.344	0.490	0.302	0.362
	0.4	0.432	0.372	0.536	0.322	0.434
	0.8	0.468	0.374	0.558	0.340	0.448

Table 19 Testing accuracy of algorithms unknown in training set under model 2 on BOSSbase

Method	bpp	Testing label			
		LSB matching	MG	MVG	MiPOD
Proposed method (w-χ^2)	0.1	0.9147	0.4503	0.3500	0.4117
	0.4	0.9653	0.4987	0.4187	0.4450
	0.8	0.9697	0.5377	0.4360	0.4827
Proposed method (IP)	0.1	0.9183	0.4437	0.3507	0.3923
	0.4	0.9425	0.4750	0.3857	0.4323
	0.8	0.9467	0.5100	0.4227	0.4610

Faced with steganography algorithms unknown in training set, CNN can't be used directly in detection since its structure has been fixed. The proposed scheme still works on this point.

5.7 Detecting the presence of secrete information

The task of detecting the presence of secrete information can be considered as a special case of steganography algorithms recognition that the number of algorithms is 2 (cover and stego). We test the performance of proposed steganalytic scheme on detecting *S-UNIWARD* under payload of 0.4bpp. The corresponding performance achieved by method in paper [33] is used as references. The used CNN model of proposed method is trained again on the cover and S-UNIWARD stegos. The testing accuracies are shown in Table 23.

Although the testing accuracy of proposed method is inferior to that of method in paper [33] on BOSSbase, the proposed method shows competitive result on Netbase compared with method in [33].

Table 20 Testing accuracy of algorithms unknown in training set under model 2 on Netbase

Method	bpp	Testing label			
		LSB matching	MG	MVG	MiPOD
Proposed method (w-χ^2)	0.1	0.936	0.366	0.344	0.358
	0.4	0.944	0.404	0.352	0.406
	0.8	0.952	0.412	0.360	0.424
Proposed method (IP)	0.1	0.942	0.388	0.358	0.374
	0.4	0.968	0.416	0.382	0.418
	0.8	0.960	0.450	0.402	0.442

Table 21 Testing accuracy of algorithms unknown in training set under model 3 on BOSSbase

Method	bpp	Testing label LSB matching
Proposed method (w-χ^2)	0.1	0.8813
	0.4	0.8923
	0.8	0.9340
Proposed method (IP)	0.1	0.8677
	0.4	0.8543
	0.8	0.8860

Table 22 Testing accuracy of algorithms unknown in training set under model 3 on Netbase

Method	bpp	Testing label LSB matching
Proposed method (w-χ^2)	0.1	0.816
	0.4	0.850
	0.8	0.864
Proposed method (IP)	0.1	0.840
	0.4	0.846
	0.8	0.828

Table 23 Testing accuracy of detecting of seven algorithms under different methods

Method in [33]		Proposed method (w-χ^2)		Proposed method (IP)	
BOSSbase	Netbase	BOSSbase	Netbase	BOSSbase	Netbase
0.8094	0.7060	0.7723	0.7310	0.7515	0.7440

6 Conclusions

This paper proposes a steganalytic method for steganography algorithms recognition, and achieves competitive performance when detecting images from unseen source. The proposed scheme can also detect steganography algorithms unknown in training set. Currently, the computational complexity of proposed method is higher than that of conventional methods.

In future, the authors would like to focus on the three following works: firstly, exploring more effect approaches to separate image content signal and the possible stego signal in order to generate more practical match image; secondly, exploring more accurate similarity measures between features; thirdly, trying to reduce the computational complexity of proposed method.

References

1. Bas P, Filler T, Pevný T (2011) Break Our steganographic system: The ins and outs of organizing BOSS [C]//international workshop on information hiding. Springer, Berlin Heidelberg, pp 59–70
2. Boroumand M, Fridrich J. (2016) Boosting Steganalysis with explicit feature maps [C]//proceedings of the 4th ACM workshop on information hiding and multimedia security. ACM: 149–157
3. Cho S, Cha BH, Gawecki M, Jay Kuo CC (2013) Block-based image steganalysis: algorithm and performance evaluation [J]. J Vis Commun Image Represent 24(7):846–856
4. Denemark T, Sedighi V, Holub V et al. (2014) Selection-channel-aware rich model for *s*teganalysis of digital images [C]//information forensics and security (WIFS), 2014 I.E. international workshop on. IEEE : 48–53

5. Dong J, Wang W, Tan T (2009) Multi-class blind steganalysis based on image run-length analysis [C]// international workshop on digital watermarking. Springer, Berlin Heidelberg, pp 199–210
6. Duda RO, Hart PE, Stork DG (1973) Pattern classification [M]. Wiley, New York
7. Duda RO, Hart PE, Stork DG (2012) Pattern classification [M]. John Wiley & Sons
8. Filler T, Fridrich J (2010) Gibbs construction in steganography [J]. IEEE Trans Inform Forensics Sec 5(4): 705–720
9. Fridrich J, Kodovsky J (2012) Rich models for steganalysis of digital images [J]. IEEE Trans Inform Forensics Sec 7(3):868–882
10. Fridrich JJ, Kodovský J (2013) Multivariate gaussian model for designing additive distortion for steganography [C]//ICASSP: 2949–2953
11. Holub V, Fridrich J (2012) Designing steganographic distortion using directional filters [C]//information forensics and security (WIFS), 2012 I.E. international workshop on. IEEE: 234–239
12. Holub V, Fridrich J (2013) Random projections of residuals for digital image steganalysis [J]. IEEE Trans Inform Forensics Sec 8(12):1996–2006
13. Holub V, Fridrich J (2013) Digital image steganography using universal distortion [C]//. Proceedings of the First ACM Workshop on Information Hiding and Multimedia Security. ACM: 59–68
14. Jia Y, Shelhamer E, Donahue J, et al. (2014) Caffe: convolutional architecture for fast feature embedding [C]//. Proc 22nd ACM Int Conf Multimed ACM: 675–678
15. Katzenbeisser S, Petitcolas F (2000) Information hiding techniques for steganography and digital watermarking [M]. Artech house
16. Ker AD, Böhme R (2008) Revisiting weighted stego-image steganalysis [C]//Electronic Imaging 2008. International Society for Optics and Photonics: 681905–681905-17
17. Kodovský J, Fridrich J (2009) Calibration revisited [C]//. Proc 11th ACM Workshop Multimed Sec. ACM: 63–74
18. Kodovsky J, Fridrich J, Holub V (2012) Ensemble classifiers for steganalysis of digital media [J]. IEEE Trans Inform Forensics Sec 7(2):432–444
19. Kodovský J, Sedighi V, Fridrich J (2014) Study of cover source mismatch in steganalysis and ways to mitigate its impact [C]//IS&T/SPIE Electronic Imaging. International Society for Optics and Photonics, : 90280J–90280J-12
20. Kong X, Feng C, Li M, Guo Y (2016) Iterative multi-order feature alignment for JPEG mismatched steganalysis [J]. Neurocomputing 214:458–470
21. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks [C]//. Adv Neural Inform Process Syst: 1097–1105
22. LeCun Y, Bengio Y, Hinton G (2015) Deep learning [J]. Nature 521(7553):436–444
23. Li X (2014) Research on JPEG images mismatched Steganalysis [D]. Dalian University of Technology
24. Li B, Wang M, Huang J, et al. (2014) A new cost function for spatial image steganography [C]//image processing (ICIP), 2014 I.E. international conference on. IEEE: 4206–4210
25. Lu J (2014) Image Steganalytic feature analysis and Steganographic algorithm recognition [D]. PLA Information Engineering University
26. Lubenko I, Ker AD (2012) Steganalysis with mismatched covers: do simple classifiers help? [C]//. Proc Multimed Sec. ACM: 11–18
27. Mielikainen J (2006) LSB matching revisited [J]. IEEE signal processing letters 13(5):285–287
28. Pevny T, Fridrich J (2007) Merging Markov and DCT features for multi-class JPEG steganalysis [C]// Electronic Imaging 2007. Int Soc Optics Photonics: 650503–650503-13
29. Pevny T, Fridrich J (2008) Multiclass detector of current steganographic methods for JPEG format [J]. IEEE Trans Inform Forensics Sec 3(4):635–650
30. Pevny T, Bas P, Fridrich J (2010) Steganalysis by subtractive pixel adjacency matrix [J]. IEEE Trans Inform Forensics Sec 5(2):215–224
31. Pevný T, Filler T, Bas P (2010) Using high-dimensional image models to perform highly undetectable steganography [C]//international workshop on information hiding. Springer, Berlin, Heidelberg, pp 161–177
32. Qian Y, Dong J, Wang W, et al. (2015) Deep learning for steganalysis via convolutional neural networks [C]//. SPIE/IS&T Electronic Imaging. International Society for Optics Photonics: 94090J–94090J-10
33. Qian Y, Dong J, Wang W, et al. (2016) Learning and transferring representations for image steganalysis using convolutional neural network [C]//. Image Process (ICIP), 2016 I.E. Int Conf. IEEE: 2752–2756
34. Sedighi V, Fridrich JJ, Cogranne R (2015) Content-adaptive pentary steganography using the multivariate generalized Gaussian cover model [C]//Media Watermarking, Security, and Forensics: 94090H
35. Sedighi V, Cogranne R, Fridrich J (2016) Content-adaptive steganography by minimizing statistical detectability [J]. IEEE Trans Inform Forensics Sec 11(2):221–234
36. Taigman Y, Yang M, Ranzato M A, et al. (2014) Deepface: Closing the gap to human-level performance in face verification [C]//. Proc IEEE Conf Comput Vision Pattern Recogn: 1701-1708

37. Tan S, Li B (2014) Stacked convolutional auto-encoders for steganalysis of digital images [C]//. Asia-Pacific signal and information processing association, 2014 annual summit and conference (APSIPA). IEEE: 1–4
38. Wang P, Liu F, Wang G, et al. (2008) Multi-class steganalysis for Jpeg stego algorithms [C]//. Image Process, 2008. ICIP 2008. 15th IEEE International Conference on. IEEE: 2076–2079
39. Xu G, Wu HZ, Shi YQ (2016) Structural design of convolutional neural networks for steganalysis [J]. IEEE Sign Process Lett 23(5):708–712
40. Xu G, Wu HZ, Shi YQ (2016) Ensemble of CNNs for steganalysis: an empirical study [C]//. Proc 4th ACM Workshop Inform Hiding Multimed Sec. ACM: 103–107
41. Xu X, Sun Y, Tang G et al (2016) Deep learning on spatial rich model for Steganalysis [C]//international workshop on digital watermarking. Springer, Cham, pp 564–577

Xiaoyu Xu received the B.S. degree in electronic science and technology from ZhengZhou information science and technology institute, Henan, China, in 2015. He is pursuing the M.S. degree in information security at Zhengzhou information science and technology institute. His research interests include deep learning and steganalysis.

Yifeng Sun received the B.S., M.S., ph.D degrees in information security from Zhengzhou information science and technology institute, Henan, China, in 1999, 2002 and 2010 respectively. He is now an associate professor at the Department of Information Security, Zhengzhou information science and technology institute. His research interests include information hiding and artificial intelligence. He has published 30 research articles in these areas.

Jiang Wu is pursuing the B.S. degree in electronic science and technology at Zhengzhou information science and technology institute, Henan, China. His research interests include deep learning and computer vision.

Yi Sun received the B.S. degree in electronic science and technology from ZhengZhou information science and technology institute, Henan, China, in 2015. She is pursuing the M.S. degree in information security at Zhengzhou information science and technology institute. Her research interests include information hiding and information security.