Image Steganalysis via Multi-Column Convolutional Neural Network

Qi Ke
Guangzhou University
Guangzhou, China
e-mail: qikersa@163.com

Liu Dongming
Guangzhou University
Guangzhou, China
e-mail: liudm@163.com

ABSTRACT

Deep learning that jointly studies and extracts features is very promising for steganalysis. In this article, we design a simple but effective Multi-column Convolutional Neural Network (MCNN) based on steganalysis architecture for images. The proposed MCNN architecture allows the input image to be of arbitrary size or resolution. In particular, by utilizing filters with receptive fields of different sizes, the features learned by each column CNN are adaptive to variations in payloads. Comprehensive experiments on standard dataset show that MCNN model can detect the state of arts steganographic algorithms at a high accuracy. It also outperforms several recently proposed CNN-based steganalyzers in conditions of the same embedding key stego and cover-source mismatch scenarios.

Keywords

image steganalysis, MCNN, the same key stego, cover-source mismatch.

1. INTRODUCTION

The most current steganalysis frameworks involve a large number of techniques for feature extraction and classification. The objective of the feature extraction is to capture as much stego information as possible. Many image features have been proposed such as Rich Models (RM) for the spatial domain [1] and JPEG one [2].

As a state-of-the-art classification method, deep learning has been receiving a continuously increasing attention in the past years. The main advantage of deep learning is the automatic extraction of the most relevant high-level features of the input data to improve the learning of the targeted task [3].

In fact, a CNN-based steganalyzer makes it possible to automatically unify feature extraction and classification steps in one unique architecture without a prior feature selection. The first CNN-based steganalysis experiment [4], which applied the stacked convolutional auto-encoders, have not reached accuracy level similar to the one given by SRM steganalysis. Further woks [5, 6] succeeded to improve the performance in the context of the “same embedding key” scenario, which resulted in weakening of security for embedding several messages with the same key. Even if these successive works have shown the advantages of CNN-based steganalysis, some limitations have still to be overcome: processing varisized images and with different payload values (smaller ones).

In this paper, a Multi-Column Convolutional Neural Network (MCNN) based steganalyzer is designed to improve the steganlysis performance. Moreover, our proposal aims at being more general and at overcoming the limitations noted previously. More specifically, contributions of this paper are summarized as follows:

Firstly, the key idea of the proposed MCNN framework is the use of three columns filters with receptive fields of different sizes (large, medium, small) so that the overall MCNN framework is robust to different stego field.

Secondly, a convolution layer the filter size of which is 1×1 is used before the fully connected layer. Therefore, the input image can be of arbitrary size so that the proposed method is adaptive to large variation in stego image size.

2. RELATED WORKS

The deep learning-based steganalyzer has become a breakthrough technology which outperforms conventional steganalysis methods since the first stacked convolutional auto-encoders structure steganalyer [4].

In 2015, Qian et al. [5] proposed a CNN architecture consisting of 5 convolutional layers followed by three fully connected layers: two hidden layers of 128 ReLU neurons each and one output layer of 2 softmax neurons. Rather than directly process the input image, this CNN works on a 252×252 high-pass filtered image issued by a 5×5 kernel. The experiments showed that the designed CNN structure was only slightly outperformed by state-of-the-art SRM-based steganalyzers.

In 2016, Pibre et al. [6] investigated Qian’s work and improved the detection performance in the scenario of reusing the same embedding key for different images. They designed a CNN with fewer but larger layers including 2 layers with 64 convolution kernels in the first layer and 16 kernels in the seconde one. The experiments showed that the proposed method was able to reduce the detection error by more than 16% in comparison with the SRM-based steganalyzers in case of S-UNIWARD at 0.4bpp embedding rate, but gained bad results when considering a different key for each embedding.

In 2016, Couchot et al. [7] also designed a CNN-based steganalyzer for stego images with a unique embedding key. The proposed structure embeds less convolutions but with much larger filters in the final convolutional layer, which can deal with larger images and lower payloads. The architecture took 512×512 image as input image, and the input image was firstly filtered by a single kernel of size 3×3, then followed by a layer of 64 filters as large as possible with zero padding. The experiments showed that the proposal outperformed the existing CNN-based steganalyzer and defeated many state-of-the-art image steganography schemes in case of the “same embedding key”.

3. MCNN FOR IMAGE STEGANALYSIS

3.1 CNN architecture overview

A convolutional neural network (CNN) is designed to learn how to extract sets of smaller feature maps with different size kernels, which usually consists of several kinds of layers, namely convolutional layer, fully connected layer.

The convolutional layer is the most important layer in CNN, which usually produces feature maps by a successive three-step process. The first step is a convolutional step, which performs a filtering process using K kernels resulting in K new feature maps. Then, the second step is the application of an activation step, which adds some nonlinearity to feature maps. Finally, the third step is a pooling step, which is applied to reduce each feature map by a pooling operation, typically by computing the mean or the max over p×p regions.

As a classification network, the last convolution layer is usually connected to a fully connected layer. Then, a softmax function is connected to the outputs of the last layer in order to normalize the outputs delivered by the network between [0, 1]. The predicted probability of belonging to which class is given by the softmax function. Thus, the network delivers the probability values as an output, each presents the probability of classifying into the corresponding class. The classification decision is finally obtained by returning the class with the highest probability.

3.2Architecture of the proposed MCNN

The design of the proposed MCNN is driven by the following considerations.

Firstly, steganography embeds the secret information by modifying pixels that are randomly widespread throughout the whole input image. Consequently, it is better to use large convolution filters to build features to capture the slight modifications performed by a steganographic algorithm. Various filter sizes had been proposed such as 3×3, 5×6, to 12×12 or 15×15. Overall, the larger the filters are, the more complex stego information the feature map can contain.

Secondly, due to the random embedding position and different embedding rate, the patch of images usually contain stego pixels of very different sizes, hence, filters with receptive fields of the same size are unlikely to capture characteristics of stego pixels at different scales. Therefore, it is more natural to use filters with different sizes of local receptive field to capture the features from the raw patches to the stego patches.

Motivated by the success of Multi-column Deep Neural Networks (MDNNs) [8], and after some preliminary experiments, we proposed to use a Multi-column CNN (MCNN) to capture the stego features. In our MCNN, for each column, the different sizes of filters are used to model the stego patches corresponding to stego pixels of different scales. For instance, filters with larger receptive fields are more useful for modeling the stego patches corresponding to more stego pixels.

The structure of the proposed MCNN is illustrated in Figure 1. It contains three parallel CNNs the filters of which are with local receptive fields of different sizes. All columns have the same network structures except for the sizes and numbers of filters. Note that the pooling operation is dropped in all layers, and the rectified linear unit (ReLU) is adopted as an activation function in each convolution layer because of its good performance for CNNs [9]. To reduce the computational complexity, less number of filters for CNNs with larger filters is used. We stack the output feature maps of all CNNs and adopt a convolution layer the filter size of which is 1×1 to map the stacked feature maps to the stego map before fully connected layer, which makes the input image be of arbitrary size so that the proposed method is adaptive to large variation in stego image size. The fully connected part is a classical neural network in its simplest form: a single output layer with two softmax neurons.

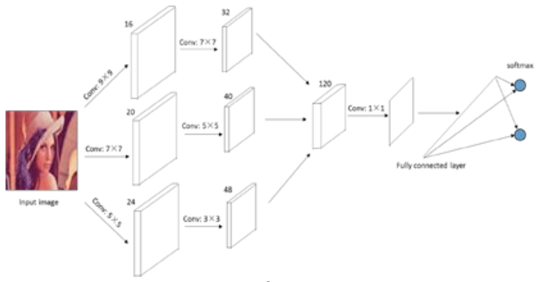

Fig.1 The structure of the proposed MCNN

The benefits of the designed MCNN structure include: 1) the input images of the CNN are usually normalized to the same size. Here the input images with their original sizes without resizing are preferred because resizing images to the same size will introduce additional distortion in the stego image that is difficult to estimate. 2) Besides the fact that we use three columns filters with different sizes, another difference between our MCNN and conventional MDNNs is that the output of all CNNs is combined with learnable weights by 1×1 filters, while the outputs are simply averaged in MDNNs which are not suitable for steganalysis.

3.3 Optimization of MCNN training

The output layer of softmax function can be optimized via batch-based stochastic gradient descent and backprogagation, typical for training stage. However, as the number of training samples are very limited, it is not easy to learn all the parameters simultaneously. Motivated by the success of pre-training, CNN in each single column is pre-trained separately by directly mapping the outputs of the last convolutional layer to the stego map. Then three pre-trained CNNs are used to initialize CNNs in all columns and fine-tune all the parameters simultaneously.

4. EXPERIMENTAL RESULTS

4.1 Databases and parameters used

We evaluated the proposed MCNN-based steganalysis on two image cover databases, BOSS database [10] and RAISE database [11]. For the experiments, the BOSS database consists of 10000 grey-level images of size 512×512, and each image is divided into four parts to obtain 40000 images of size 256×256. The RAISE database includes 8156 high-resolution raw images, in which all photos are stored in uncompressed formats, in high quality (3008 × 2000, 4288 × 2848 and 4928 × 3264 pixels), and each image is also randomly divided to obtain 20000 images of size 512×512 and 20000 images of size 256×256. Note that we used the same script as that used for the BOSS in order to transform the RAW full resolution color images into grey-level images.

In our evaluation, we embedded the messages using two steganographic tools of HUGO[12] and S-UNIWARD [13] with two embedding payload values: 0.4 and 0.1 bpp, using the C++ implementations available from DDE Lab Binghamton web site. After embedding, the database obtained from BOSS consisted of 150000 images (50000 covers and 100000 stegos), and the database from RAISE consisted of 120000 images (40000 covers and 80000 stegos). We limited the experiments to these two payload sizes due to the high number of images and computations.

From the computational complexity point of view, the numerous parameters such as the number of input images, the number of iterations should be optimized in order for the CNN to converge. In our experiments, the training parameters were set as follows: the “mini-batch” size was 100, the “moment” was 0.01, the “learning coefficient” was 0.0001 for weights and 0.0002 for bias, the “weight decay” was 0.002 for convolutions layers and 0.001 for the fully connected network, learning method was set as “SGD”, the “drop out” and “pooling” is not activated, the max training epochs were set as 200. With a learning database consisting of 200000 blended images of sizes 256×256 and 512×512, It took about five days of computation in order to find the “best” network with the most efficient double precision GPU card like the Nvidia Tesla K40. Note that the implementation of the proposed network and its training are based on Caffe framework.

4.2 Clairvoyant scenarios

In this scenario, the cover and stego images come from the same database, the embedding algorithm and the payload size are known by the steganalyst. That is to say, the steganalyst knows all the public parameters, but does not know the private parameters such as the embedding secret key. In order to compare with other CNN-based steganalyzer, we also additionally add the condition that stego images are obtained by applying steganography using the same secret embedding key. Note that the embedding has been done with the simulator using always the same key.

Our experiments were carried out on the BOSS database, which consists of 10000 grey-level images on 8bits with size of 512×512 and 10000 grey-level images on 8 bits with size of 256×256. We embedded the messages using HUGO and S-UNIWARD with two payload values: 0.4 bpp and 0.1 bpp. The obtained set of images is, thus, made of 250000 images (20000 covers and 80000 stegos).

Three CNN-based steganalyzer methods were evaluated. The first steganalyzer was done using the CNN presented by Pibre et al. [6], which is denoted as Pibre-CNN. The second steganalyzer was done using the CNN presented by Couchot et al. [7], which is denoted as Couchot-CNN. The third steganalyzer was done using the MCNN we built (see Figure.1), which is denoted as MCNN.

For each payload size (0.4 bpp and 0.1 bpp) and mixed payload size, 10 tests were carried out where, for each test, the learning was done on 40000 images randomly taken from the obtained BOSS database. The tests were performed on 20000 images (10000 covers and 10000 associated stegos) randomly taken from the remaining BOSS database.

For both Pibre-CNN, Couchot-CNN, and MCNN, detection accuracy was computed and averaged over the 10 tests. The results are reported in Table 1. It can be noticed that the CNN network is usually converged before the end of the training iteration except the mixed steganographic algorithm and payload.

Table 1. Results under Clairvoyant scenarios

Stego algorithm	Payload	Iter number	Detection accuracy		
			Pibre-CNN	Couchot-CNN	MCNN
HUGO	0.4	52	93.03%	93.76%	94.83%
HUGO	0.1	168	72.17%	75.59%	77.07%
S-UNI	0.4	61	91.47%	93.15%	95.13%
S-UNI	0.1	142	70.35%	72.94%	74.25%
Mixed	0.1&0.4	200	79.49%	80.14%	83.68%

Whatever the stegnanographic algorithm or payloads are chosen, the proposed MCNN has at least 1% improvement in detection accuracy compared with Pibre-CNN and Couchot-CNN. Note that there is at least 3.5% improvement in the situation of mixed steganographic algorithm and mixed payload, which is an impressive outperformance considering the difficulty to improve percentages on the accuracy of steganalysis.

Our experiment shows that our proposed MCNN is trained well. We conjecture a few reasons:

i) There are three parallel CNN columns with different numbers of filters and different sizes of receptive fields. The total high number of filters (60 filters for the first layer and 120 filters for the second layer) enriches the diversity of features extraction. The three different receptive fields have multi-scale analysis functions and can extract multi-scale features which improves the ability of characterizing local features.

ii) The pooling layers are eliminated because they were acting as a down-sampling, and, thus, leading to information loss.

4.3Cover-source mismatch scenarios

In this scenario, the trained networks in the previous section are chosen and we applied them on the testing dataset built from 20000 images (10000 covers and 10000 stegos) randomly taken from RAISE database. Note that the cover-source mismatch was present since the RAISE database is rather different from the BOSS one.

The detection accuracy is reported in Table 2. We also evaluate the same three CNN-based steganalyzer methods as in the previous section. We can see that all three CNN-based steganalyzers with a small payload can detect stego images with a higher payload. To sum up, for a payload of 0.4bpp the proposed MCNN can detect stego images with an accuracy higher than 90%, while it falls to at least 65% for the payload of 0.1bpp and at least 81% for the unmatched training and testing payload. In comparison with the Pibre-CNN and Couchot-CNN, the proposed MCNN has about 2.5% improvement in detection accuracy.

Table 2. Results under Cover-source mismatch scenarios

Stego algorithm & payload		Iter number	Detection accuracy		
Training BOSS	Testing RAISE		Pibre-CNN	Couchot-CNN	MCNN
HUGO 0.4	HUGO 0.4	52	89.72%	88.03%	92.83%
HUGO 0.1	HUGO 0.1	168	65.46%	64.31%	68.87%
HUGO 0.1	HUGO 0.4	52	79.87%	81.34%	84.56%
S-UNI 0.4	S-UNI 0.4	61	87.02%	87.97%	90.45%
S-UNI 0.1	S-UNI 0.1	142	61.45%	62.34%	65.05%
S-UNI 0.1	S-UNI 0.4	61	74.76%	78.21%	81.04%
Mixed	Mixed	200	70.49%	72.14%	74.73%

5. CONCLUSION

The increasing attention gained by deep learning has raised the interest in whether such a method is relevant for the design of steganalyzer. In this paper, we put forward a MCNN-based steganalyzer. The designed MCNN structure consists of three parallel CNNs with different receptive fields, and each CNN consists only of two convolutional layers. We use a more classical ReLU activation function and suppress the pooling step that was acting as a down-sampling.

We evaluated the detection ability of the MCNN against two steganographiers of HUGO, and S-UNIWARD with payload of 0.1 bpp and 0.4 bpp in two standard databases of BOSS and Raise. The obtained results show the high performance of the proposed MCNN-based steganalyzer. More precisely, compared to the state-of-the-art approach, i.e., the Ensemble Classifier with SRM features, MCNN reduces the classification error by 10 percent or more. Also, compared to the previous CNN-based proposals for steganalysis, i.e., Pibre et al. [6] and Couchot et al. [7], MCNN improves the classification accuracy about 3 percent.

Since most of stegnographic softwares in Internet adopt the steganographic methods using the same secret embedding key, in which the embedding path is always the same and the embedding software always uses the same pseudo-random number sequence for generating the change probabilities, the proposed MCNN steganalysis is suitable for detecting internet stegnographic softwares.

In future work, we will concentrate on enlarging the set of steganography algorithms considered during both the training and the testing stages, and further experiments on different payload sizes. Moreover, our intension is to optimize some of the network parameters such as the shape of the filters, the columns of the CNNs, and to gain insight into the relationship between the CNNs and the chosen steganographiers.

6. ACKNOWLEDGMENT

This research was supported by scientific and technological projects of Guangzhou (201707010283), State Scholarship Fund (201408440338), scientific research project of Guangzhou municipal universities (2017A056) and national natural science foundation of China (Grant No.61572160).

REFERENCES

[1] Vojtech Holub, Jessica J Fridrich, “Random projections of residuals for digital image steganalysis”, IEEE Trans. information Forensics and Security, 8(12):1996-2006, 2013.

[2] V. Holub, J. Fridrich, “Low-complexity features for jpeg steganalysis using undecimated dct”, IEEE Transactions on Information Forensics and Security, 10(2):219-228, Feb 2015.

[3] Y. Bengio, A. Courville, P. Vincent, “Representation learning: A review and new perspectives”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2013,35(8):1798-1828.

[4] Tan S Q, Li B, “Stacked convolutional auto-encoders for steganalysis of digital images”, In Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA), pp. 1-4, IEEE, 2014.

[5] Qian Y L, Dong J, Wang W, Tan T N, “Deep learning for steganalysis via convolutional neural networks”, In IS&T/SPIE Electronic Imaging, pages 94090J-94090J. International Society for Optics and Photonics, 2015.

[6] Pibre L, Jerome P, Ienco D, Chaumont M, “Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch”, In EI: Electronic Imaging, 2016.

[7] Couchot J F, Couturier R, Guyeux C, Salomon M, “Steganalysis via a convolutional neural network using large convolution filters for embedding process with same stego key”, arXiv:1605.07946v3

[8] Ciregan D, Meier U, Schmidhuber J, “Multi-column deep neural networks for image classification”, Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012: 3642-3649.

[9] Zeiler M D, Ranzato M, Monga R, et al, “On rectified linear units for speech processing”, Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013: 3517-3521.

[10] Pevný T, Filler T, Bas P, “Using high-dimensional image models to perform highly undetectable Steganography”, International Workshop on Information Hiding. Springer Berlin Heidelberg, pp. 161-177, 2010.

[11] Dang-Nguyen D T, Pasquini C, Conotter V, et al, “RAISE: a raw images dataset for digital image forensics”, Proceedings of the 6th ACM Multimedia Systems Conference. ACM, pp. 219-224, 2015.

[12] Pevný T, Filler T, Bas P, “Using high-dimensional image models to perform highly undetectable steganography”, International Workshop on Information Hiding. Springer Berlin Heidelberg, pp. 161-177, 2010.

[13] Holub V, Fridrich J, Denemark T, “Universal distortion function for steganography in an arbitrary domain”, EURASIP Journal on Information Security, 2014(1):1, 2014