Detection of Double JPEG Compression with the Same Quantization Matrix Based on Convolutional Neural Networks

Peng Peng *, Tanfeng Sun *, Xinghao Jiang *, Ke Xu *, Bin Li † and Yunqing Shi ‡
* School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China
† Shenzhen Key Laboratory of Media Security, Shenzhen, China
‡ Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, USA
E-mail: {1099516307, tfsun, xhjiang, l13025816}@sjtu.edu.cn; libin@szu.edu.cn; shi@njit.edu

***Abstract*—The detection of double JPEG compression with the same quantization matrix is a challenging problem in image forensics. In this paper, a CNN framework is proposed to solve this problem. This framework contains a preprocessing layer and a well-designed CNN. In the preprocessing layer, the rounding and truncation error images are extracted from continuous recompressed input samples and then fed into the following CNN. In the design of the CNN architecture, several advanced techniques are carefully considered to prevent overfitting, such as 1×1 convolutional kernel and global average pooling layer. The performance of proposed framework is evaluated on the public available image dataset (BOSSbase) with various quality factors (QF). Experimental results have shown the proposed CNN framework performs better than the state-of-the-art method based on hand-crafted features.**

***Index Terms*—Image Forensics, Double JPEG Compression, the Same Quantization Matrix, Convolutional Neural Network.**

I. Introduction

Nowadays, digital images have become an integral part of our daily lives. However, with the aid of sophisticated editing software such as Photoshop, digital images can be tampered easily. The reliability and integrity of digital images have been threatened, especially when digital images are used as evidence in a court of law. Hence, image forensics has become a popular research topic attracting more and more attention of researchers in multimedia security. One of the most significant techniques in image forensics is double JPEG compression detection, since most of digital images are stored in JPEG format and the tampering operation usually involves the recompression process. Several methods have been proposed to detect double JPEG compression successfully. For detection of double JPEG compression with different quantization matrices, existing methods applied statistical models to extract distinguishing features, such as the distributions of discrete cosine transform (DCT) coefficients [1], the first digit distributions of non-zero AC coefficients [2], [3] and Markov statistics [4], [5]. For detection of double JPEG compression with the same quantization matrix, Huang et.al. [6] proposed a novel detection scheme based on the observation that the number of altered DCT coefficients between the n^{th} and the $(n{+}1)^{\text{th}}$ recompressed image monotonically decreases with the value of n increasing. Yang et.al. [7] utilized the statistical differences of rounding and truncation error blocks between single and double JPEG compressed images to achieve more robust detection performance. However, methods mentioned above based on hand crafted features requires complicatedly and arduously feature design process with professional forensics knowledge.

In recent years, deep neural networks such as CNN have been widely utilized in computer vision tasks [8], [9] and achieved overwhelming superiority compared with traditional methods. More recently, researchers in image forensics have successfully explored the powerful representation learning capability of deep learning to solve forensics tasks [10], [11], [12], [13], [14], [15]. In [14], Wang et.al. proposed a CNN-based method to detect double JPEG compression with different quantization matrices, which extracts histograms of DCT coefficients as the input. Inspired by [14], Amerini et.al. [15] proposed a hybrid architecture combing CNNs in the spatial domain and the frequency domain for double JPEG compression detection. However, to the best of our knowledge, there is not any deep learning-based method for the detection of double JPEG compression with the same quantization matrix in the literature.

In this paper, we proposed a novel CNN-based detection pipeline for double JPEG compression with the same quantization matrix. Note that, for abbreviation, double JPEG compression refers to double JPEG compression with the same quantization matrix hereinafter, unless otherwise specified. The main contributions of this work are listed as follows:

1) In the preprocessing layer, error images [7] of input samples caused by rounding and truncation operations are extracted to enhance the slight traces left by double compression and reduce the influence of various image contents.
2) To prevent overfitting and extract more latent concept for double JPEG compression detection, 1×1 convolution filters are employed in deeper layers and a global pooling structure is adopted before the fully connected layer.
3) The performance of the proposed framework is evaluated

on the public available image dataset (BOSSbase) [16] with various quality factors (QF). Besides, the influence of different network structures is analyzed in this paper. Experimental results have shown better performance of the proposed CNN framework than the state-of-the-art method [7] based on hand-crafted features.

The rest of this letter is organized as follows. Section II introduces the error image and its statistical properties. In Section III, a novel framework to detect double JPEG compression with the same quantization matrix is proposed based on CNN. Section IV describes the details of our experiments and the results of proposed method. In the end, a conclusion is drawn in Section V.

II. Error Image in the Decompression Process

JPEG is a widely used lossy compression standard for digital images. There are three types of errors during the JPEG compression and decompression process. The first one is quantization error, which is caused by rounding the float value of quantized DCT coefficients to their nearest integer value. The second and third errors are rounding and truncation errors, which occur in the JPEG decompression process. The rounding error is defined as the difference between the float reconstructed pixel within [0, 255] and its rounded version. The truncation error is defined as the difference between the float reconstructed pixel out of [0, 255] and its truncated integer. In this work, only rounding and truncation errors are applied to detect double JPEG compression.

Mathematically, the $(n+1)^{\text{th}}$ encoding process can be expressed as follows:

$$\mathbf{D}_{n+1} = [DCT(RT(IDCT(\mathbf{D}_n \times \mathbf{Q})))/\mathbf{Q}] \quad (1)$$

where the subscript n denotes the times of JPEG recompression; $\mathbf{D}_n$ is the 8×8 quantized DCT coefficients matrix; $DCT(\cdot)$ and $IDCT(\cdot)$ represent 8×8 discrete cosine transform and inverse discrete cosine transform separately; $\mathbf{Q}$ is the quantization matrix; $RT(\cdot)$ denotes rounding and truncation operations; $[\cdot]$ denotes the rounding operation; $\times$ and $/$ are component-wise operations. The rounding and truncation error block E_n can be defined as:

$$\mathbf{E}_n = RT(IDCT(\mathbf{D}_n \times \mathbf{Q})) - IDCT(\mathbf{D}_n \times \mathbf{Q}) \quad (2)$$

To analyze the effect of rounding and truncation error, Eq. 1 can be rewritten as follows:

$$\begin{aligned} \mathbf{D}_{n+1} &= [DCT(RT(IDCT(\mathbf{D}_n \times \mathbf{Q})))/\mathbf{Q}] \\ &= \mathbf{D}_n + [DCT(\mathbf{E}_n/\mathbf{Q})] \\ &= \mathbf{D}_n + \mathbf{R}_n \end{aligned} \quad (3)$$

where $\mathbf{R}_n$ denotes quantized DCT coefficients of $\mathbf{E}_n$. It can be found that if $\mathbf{R}_n$ is a zero matrix, $\mathbf{D}_{n+1}$ will be equal to $\mathbf{D}_n$, which means $\mathbf{E}_{n+1} = \mathbf{E}_n$. The authors in Ref. [7] call this kind of blocks ($\mathbf{R}_n = \mathbf{0}$) as stable block. Other blocks are treated as unstable blocks. Furthermore, it can be observed that if $\mathbf{R}_n = \mathbf{0}$, then $\mathbf{R}_{n+1} = \mathbf{0}$. It infers that stable blocks will remain stable after more recompression processes. Meanwhile, $\mathbf{R}_n$ is more likely to be a zero matrix with the number of recompression operations increasing. It means the numbers of unstable blocks in single compressed and double compressed images perform different convergent speed after recompressions. To illustrate this phenomenon more intuitively, we calculated the average number of unstable blocks after recompressions for different QFs using BOSSbase[16] image dataset. In Fig. 1, each point denotes the number of unstable blocks in the n^{th} decompression process. It can be observed that the number of unstable blocks in double compressed images ($n=2$ to $n=5$) converges more quickly to zero than that in single compressed images ($n=1$ to $n=4$).

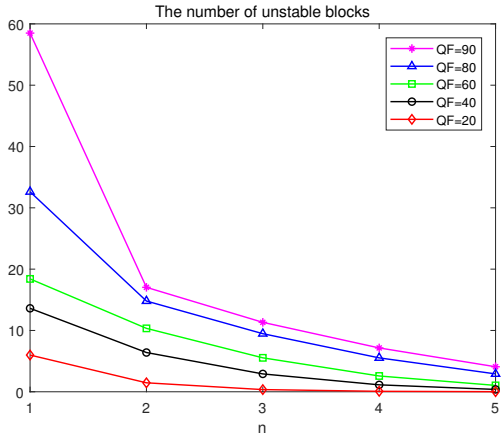

Fig. 1. Variation tendency of the number of unstable blocks.

III. The Proposed Framework

A. Overall Detection Framework

The overall detection framework of the proposed method is shown in Fig. 2. It consists of a preprocessing layer and a well-designed convolutional neural network. The preprocessing layer first extracts K rounding/truncation error images from the input sample. As shown in Fig. 1, the number of unstable blocks at the 4^{th} decompression becomes relatively small for different QFs. To balance detection capability and computational complexity, K is set to 3 in our work. The details of rounding/truncation error image extraction are illustrated in Section III-B. Then, three error images are stacked as three input feature maps ($3 \times H \times W$) for the following CNN, where $H \times W = 256 \times 256$ (spatial resolution).

B. The Details of Preprocessing Layer

To reduce the impact of various image contents and enhance traces left by double compression, a preprocessing layer is proposed in the detection pipeline as shown in Fig. 2. For the sake of simplicity, the input sample of our method is a grayscale image. For color image, only luminance component is considered. The extraction process of error images is illustrated as follows:

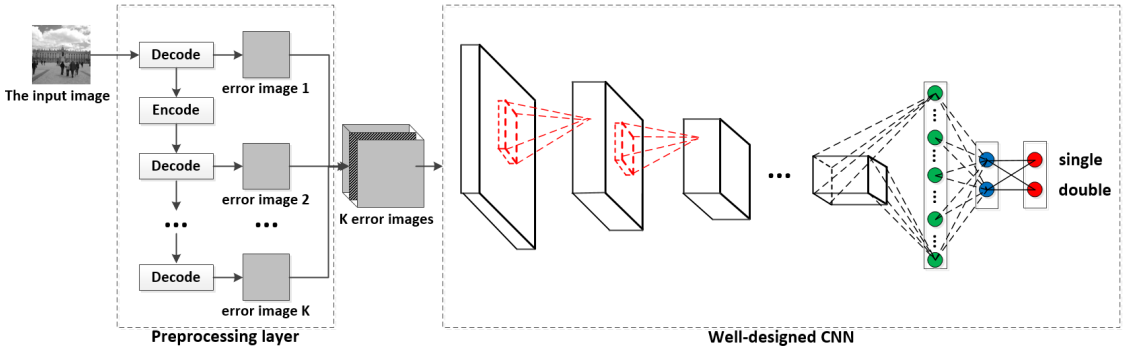

Fig. 2. The framework of the proposed detection method.

1) Decode the input image I_1 to PGM file P_1, and extract the error image E_1 according to Eq. 2.
2) Recompress the PGM file P_n to JPEG image I_{n+1} with the same quantization matrix ($n = 1, 2, \cdots$). Then decode I_{n+1} to PGM file P_{n+1} and extract its error image E_{n+1} at the same time.
3) Repeat step 1 to 2 until K error images are obtained ($K = 3$). Then construct the input data ($E = \{E_1, E_2, E_3\}$) for the CNN.

C. Network Architecture

Table I illustrates the whole architecture of the proposed CNN. The details are listed in Table I, such as kernel sizes and steps of convolutional operations. In this work, the dimension of input data to CNN is 3×256×256 while the output is a 2-D vector. The output vector represents the probabilities for each class ($\mathbf{p} = [p(\mathbf{H_0}|\mathbf{E}), p(\mathbf{H_1}|\mathbf{E})]$, where $\mathbf{E}$ denotes the input sample; $\mathbf{H_0}$ denotes the input image is single compression image and $\mathbf{H_1}$ denotes the input image is double compression image). Finally, the $p(\mathbf{H_1}|\mathbf{E})$ is compared with a pre-defined threshold (0.5) and obtain the detection result.

As shown in Table I, this architecture contains six layers. The first to the fourth layers are convolutional and pooling layers. The last two are fully-connected layer and softmax layer respectively. Similar to traditional CNN architecture, the kernel size of the first two convolutional layers is set to 5×5 in order to explore relations between neighboring elements and learn low-level local patterns. According to [17], we use 1×1 convolution kernel in the deeper layers to avoid fitting image content information in training set too much well. Each convolutional layer is followed by a pooling layer. The first three pooling layers are max pooling with window size 5×5 and stride size 2×2. After Conv4, a global average pooling layer is employed to replace the traditional fully-connected layer. This structure has been proved to be effective to prevent overfitting and improve the network generalization ability in [18]. The batch normalization (BN) [19] and rectified linear unit (ReLU) activations are applied to the output of every convolutional layer. Finally, a 128-D feature vector is obtained after the global average pooling layer. The fully-connected layer transforms the 128-D feature to a 2-D feature and feed it to a two-way softmax. Cross-entropy loss is selected to train the CNN.

TABLE I
THE OVERALL ARCHITECTURE OF CNN

	LAYER	SIZE[1,2]	STRIDE	OUTPUT
L1	Conv1	$16\times5\times5$	1×1	$16\times256\times256$
	BN+ReLu	-	-	$16\times256\times256$
	Pooling1	5×5	2×2	$16\times128\times128$
L2	Conv2	$32\times5\times5$	1×1	$32\times128\times128$
	BN+ReLU	-	-	$32\times128\times128$
	Pooling2	5×5	2×2	$32\times64\times64$
L3	Conv3	$64\times1\times1$	1×1	$64\times64\times64$
	BN+ReLU	-	-	$64\times64\times64$
	Pooling3	5×5	2×2	$64\times32\times32$
L4	Conv4	$128\times1\times1$	1×1	$128\times32\times32$
	BN+ReLU	-	-	$128\times32\times32$
	Pooling4	32×32	32×32	$128\times1\times1$
L5-L6	FC1	128×2	-	2
	Softmax	2×2	-	2

[1] For Convs (nOutput×kW×kH), "nOutput" denotes the number of output feature maps; kW×kH denotes the spatial size of convolutional kernels.
[2] For Poolings (kW×kH), kW×kH denotes the spatial window size.

IV. Experiments

In this section, several experiments are conducted to demonstrate the superiority of the proposed method for double JPEG detection. The significance of the preprocessing layer and the CNN structures of our method is investigated. Furthermore, the proposed method is compared with the state-of-the-art method [7] referred to as Yang's method.

A. Experimental Setup

To evaluate the performance of the proposed method, the image dataset BOSSbase v1.01 [16] is used in our experiments. BOSSbase v1.01 dataset contains 10000 uncompressed images with the size 512×512. To fit the structure of the

proposed CNN, images are first rescaled to 256×256. Please note, there is no lossy compression trace introduced to these uncompressed images in the rescaling process. Then, libjpeg tool [20] is employed to encode and decode images. Then several subsets $S_{i,q}$ are constructed, where i denotes the number of compression operations ($i \in \{1, 2\}$) and q denotes the quality factors ($q \in \{95, 90, 85, 80, 75, 70, 60, 40, 20\}$). For each QF, 10000 pairs of JPEG images are randomly divided into three parts with the ratio 8:1:1, namely the training set, the validation set and the testing set. All experiments are conducted using Tensorflow toolbox.

The detection accuracy rate (AR) is calculated as follows:

$$\begin{aligned} TPR &= \frac{TP}{P} \times 100\% \\ TNR &= \frac{TN}{N} \times 100\% \\ AR &= \frac{TP + TN}{P + N} \times 100\% \end{aligned} \tag{4}$$

where P denotes the total number of positive samples, N denotes the total number of negative samples, TP denotes the number of correctly classified positive samples and TN denotes the number of correctly classified negative samples.

B. Hyperparameters

Weights in convolution kernels and fully-connected layers are initialized with random numbers generated from the zero-mean Gaussian distribution with standard deviation of 0.01. The value of bias in each layer is initialized as zero. Mini-batch gradient descent is selected to train the CNN model in our experiments. The size of a mini-batch is set to 64 (32 positive samples and 32 negative samples) for each iteration. The momentum is fixed to 0.9. The initial learning rate is set to 0.001 and decrease 10% for every 20 epochs. The training phase of CNN is stopped at 100th epoch. The "dropout"[21] is used after the global average pooling.

C. The Influence of the Preprocessing Layer

In this experiment, the significance of the preprocessing layer is investigated. In the preprocessing layer, K error images are extracted from continuous recompressed JPEG images to reduce the disturbance of various image contents. According to the analysis in Section III-A, K is set to 3 in this work. We investigated the performance of the proposed CNN without the preprocessing layer. Hyper-parameters for optimizing the proposed CNN without the preprocessing layer are carefully adjusted. However, for all selected QFs, the accuracy of the proposed CNN without the preprocessing layer is about 50%. It infers the CNN taking images as inputs directly is hard to learn the distinguishing representations for double JPEG compression detection. In other words, the preprocessing layer is significant in our detection pipeline.

Besides, the importance of the correlation information between error images after different times of recompressions is also studied. The performance of taking 3 error images as input sample and 1 error image as input sample are evaluated. As shown in Table II, the average AR of taking 1 error image as input sample is only 83.64%, which is lower than the "3 error images" version (87.71%) by a distinct margin (4.07%). It illustrates the correlation information between error images undergoing different times of recompressions is helpful for detection.

TABLE II
EVALUATION ON THE INFLUENCE OF THE NUMBER OF ERROR IMAGES (%)

Quality Factor	3 error images as input sample	1 error images as input sample
95	**100**	**100**
90	**98.55**	95.20
85	**94.60**	91.15
80	**94.00**	90.85
75	**83.25**	78.20
70	**82.35**	77.65
60	**82.00**	76.50
40	**80.25**	74.45
20	**74.35**	68.75
Average	**87.71**	83.64

D. Analysis of Different CNN Structures

How to determine the optimized structure of the CNN is the significant issue. In Table III, we record the average detection accuracies evaluated on the QFs in Section IV-A and the corresponding model size with different modified CNN structures. The number of weights and biases in the CNN is considered to calculate the model size. These results verified the proposed CNN in Section III-C, which includes 1×1 convolution kernels and the global average pooling layer, has more efficient detection capability and distinctively smaller model size.

TABLE III
EVALUATION ON THE INFLUENCE OF DIFFERENT CNN STRUCTURES

	Proposed	3×3[1]	5×5[2]	FC[3]
Average AR (%)	**87.71**	87.41	86.83	86.95
Model Size	**25K**	107K	271K	17M

[1] 3×3 = replace all 1×1 convolution kernels with 3×3 kernels.
[2] 5×5 = replace all 1×1 convolution kernels with 5×5 kernels.
[3] FC = replace the global average pooling with fully-connected layer.

E. Performance Evaluation on Double Compression Detection

In this experiment, the detection performance is evaluated for different QFs, where QF is selected from the set $\{95, 90, 85, 80, 75, 70, 60, 40, 20\}$. The overall results achieved by our method and Yang's method are shown in Table IV. It can be observed that the proposed CNN-based method perform better than Yang's method in all cases. The average AR of our method is 87.71%, which is 3.01% higher than 84.70% achieved by Yang's method. Even when the value of QF is very low (e.g. 20), the proposed method can still achieve 74.35% detection accuracy. It can be concluded the proposed method

TABLE IV
THE DETECTION ACCURACIES (%) ACHIEVED BY THE PROPOSED METHOD AND YANG'S METHOD

Quality Factor	The proposed CNN	Yang's method
95	**100**	**100**
90	**98.55**	95.25
85	**94.60**	92.30
80	**94.00**	89.60
75	**83.25**	78.03
70	**82.35**	79.50
60	**82.00**	79.45
40	**80.25**	76.65
20	**74.35**	71.50
Average	**87.71**	84.70

has more robust detection capability than the state-of-the-art method distinctively for different image qualities, which is important in real forensics applications.

F. CNN Generalization Capability

In this experiment, UCID [22] dataset (1338 images of size 384×512) is used to test the generalization capability of the proposed CNN. The images in UCID dataset are first resized to 256×256. Then, we used the pre-trained CNN models to test the UCID dataset. The results are shown in Table V. It can be seen that the performance of the proposed CNN on UCID dataset is still good. It shows that our method has good generalization ability.

TABLE V
THE DETECTION ACCURACIES (%) ON UCID DATASET BY OUR METHOD

QF	***90***	***80***	***70***	***60***	***40***	***20***
AR	96.89	93.27	86.28	85.46	83.27	80.38

V. CONCLUSION

In this paper, a novel CNN-based method is proposed. In the preprocessing layer, error images are extracted by continuously recompressing input samples to reduce the disturbance of various image contents and enhance the traces of double compression. The architecture of the CNN is carefully designed to prevent overfitting by considering several advanced structures such as 1×1 convolution kernels and the global average pooling layer . The well-known BOSSbase dataset is used to conduct experiments. Experimental results demonstrated that the proposed method outperform the state-of-the-art method on with different quality factors.

VI. ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (No. 61572320, 61572321). The corresponding author is Dr. Xinghao Jiang.

REFERENCES

[1] A. C. Popescu and H. Farid, "Statistical tools for digital forensics," in *Proceedings of the 6th International Conference on Information Hiding*, 2004, pp. 128–147.
[2] D. Fu, Y. Q. Shi, and W. Su, "A generalized benford's law for jpeg coefficients and its applications in image forensics," *Proc. SPIE*, vol. 6505, p. 65051L, 2007.
[3] B. Li, Y. Q. Shi, and J. Huang, "Detecting doubly compressed jpeg images by using mode based first digit features," in *Multimedia Signal Processing, 2008 IEEE Workshop on*, 2008, pp. 730–735.
[4] S. Shang, Y. Zhao, and R. Ni, "Double jpeg detection using high order statistic features," in *IEEE International Conference on Digital Signal Processing*, 2017, pp. 550–554.
[5] Z. F. Wang, L. Zhu, Q. S. Min, and C. Y. Zeng, "Double compression detection based on feature fusion," in *2017 International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 2, July 2017, pp. 379–84.
[6] F. Huang, J. Huang, and Y. Q. Shi, "Detecting double jpeg compression with the same quantization matrix," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 848–856, Dec 2010.
[7] J. Yang, J. Xie, G. Zhu, S. Kwong, and Y. Q. Shi, "An effective method for detecting double jpeg compression with the same quantization matrix," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 11, pp. 1933–1942, Nov 2014.
[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.
[9] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1725–1732.
[10] J. Chen, X. Kang, Y. Liu, and Z. J. Wang, "Median filtering forensics based on convolutional neural networks," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1849–1853, Nov 2015.
[11] L. Bondi, L. Baroffio, D. Gera, P. Bestagini, E. J. Delp, and S. Tubaro, "First steps toward camera model identification with convolutional neural networks," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 259–263, March 2017.
[12] Y. Rao and J. Ni, "A deep learning approach to detection of splicing and copy-move forgeries in images," in *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*, Dec 2016, pp. 1–6.
[13] P. Rota, E. Sangineto, V. Conotter, and C. Pramerdorfer, "Bad teacher or unruly student: Can deep learning say something in image forensics analysis?" in *2016 23rd International Conference on Pattern Recognition (ICPR)*, Dec 2016, pp. 2503–2508.
[14] Q. Wang and R. Zhang, "Double jpeg compression forensics based on a convolutional neural network," *EURASIP Journal on Information Security*, vol. 2016, no. 1, p. 23, Oct 2016.
[15] I. Amerini, T. Uricchio, L. Ballan, and R. Caldelli, "Localization of jpeg double compression through multi-domain convolutional neural networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, July 2017, pp. 1865–1871.
[16] P. Bas, T. Filler, and T. Pevn, "Break our steganographic system: The ins and outs of organizing boss," *Journal of the American Statistical Association*, vol. 96, no. 454, pp. 488–499, 2011.
[17] G. Xu, H. Z. Wu, and Y. Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 708–712, May 2016.
[18] M. Lin, Q. Chen, and S. Yan, "Network in network," *Computing Research Repository (CoRR)*, vol. abs/1312.4400, 2013.
[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Computing Research Repository (CoRR)*, vol. abs/1502.03167, 2015.
[20] T. Lane, "Libjpeg," http://libjpeg.sourceforge.net/.
[21] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan 2014.
[22] G. Schaefer and M. Stich, "UCID - an uncompressed colour image database," *Storage and Retrieval Methods and Applications for Multimedia*, vol. 5307, pp. 472–480, 2004.