Distinguishing Between Natural and Computer-Generated Images Using Convolutional Neural Networks

Weize Quan, Kai Wang, Dong-Ming Yan, and Xiaopeng Zhang

***Abstract*— Distinguishing between natural images (NIs) and computer-generated (CG) images by naked human eyes is difficult. In this paper, we propose an effective method based on a convolutional neural network (CNN) for this fundamental image forensic problem. Having observed the rather limited performance of training existing CCNs from scratch or fine-tuning pre-trained network, we design and implement a new and appropriate network with two cascaded convolutional layers at the bottom of a CNN. Our network can be easily adjusted to accommodate different sizes of input image patches while maintaining a fixed depth, a stable structure of CNN, and a good forensic performance. Considering the complexity of training CNNs and the specific requirement of image forensics, we introduce the so-called *local-to-global* strategy in our proposed network. Our CNN derives a forensic decision on local patches, and a global decision on a full-sized image can be easily obtained via simple majority voting. This strategy can also be used to improve the performance of existing methods that are based on hand-crafted features. Experimental results show that our method outperforms existing methods, especially in a challenging forensic scenario with NIs and CG images of heterogeneous origins. Our method also has good robustness against typical post-processing operations, such as resizing and JPEG compression. Unlike previous attempts to use CNNs for image forensics, we try to understand what our CNN has learned about the differences between NIs and CG images with the aid of adequate and advanced visualization tools.**

***Index Terms*— Image forensics, natural image, computer-generated image, convolutional neural network, robustness, local-to-global strategy, visualization.**

Manuscript received September 19, 2017; revised January 19, 2018, March 2, 2018, and April 10, 2018; accepted April 18, 2018. Date of publication May 7, 2018; date of current version May 22, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61772523, Grant 61620106003, and Grant 61331018 and in part by the French National Agency for Research through PERSYVAL-lab under Grant ANR-11-LABX-0025-01 and through DEFALS under Grant ANR-16-DEFA-0003. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Siwei Lyu. *(Weize Quan and Kai Wang are co-first authors.) (Corresponding author: Dong-Ming Yan.)*

W. Quan is with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, also with the University of Chinese Academy of Sciences, Beijing 100049, China, and also with University Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France (e-mail: qweizework@gmail.com).

K. Wang is with University Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France (e-mail: kai.wang@gipsa-lab.grenoble-inp.fr).

D.-M. Yan and X. Zhang are with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: yandongming@gmail.com; xiaopeng.zhang@ia.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2018.2834147

I. Introduction

NATURAL images (NIs) reflect real-world scenes, and computer graphics tools can now generate virtual but visually plausible images. Consequently, distinguishing between NIs and computer-generated (CG) images has attracted increasing attention and has become an important research problem in image forensics. However, this problem is difficult to solve because the ultimate goal of computer graphics is to equip CG images with the same photorealism as NIs possess. Fig. 1 shows a pair of images. At first glance, people may have the impression that the left image is a real photo acquired by a digital camera, and the right image is generated from computer graphics. The truth is contrary to this impression.

Two research directions, *i.e.*, *subjective* and *objective*, mainly exist with regard to the problem of distinguishing between natural and CG images or videos. The former usually involves psychophysical experiments, and the latter is often based on the statistical or intrinsic properties of the two classes. A simple and effective objective method that is particularly suitable for CG character identification is to design a category-distinctive scalar feature and select an appropriate threshold to separate the two classes. Prevalent methods for general cases (*i.e.*, distinguishing between NIs and CG images with various scenes and contents) follow the classical pipeline of machine learning, which consists of two separate phases: (1) designing sophisticated, discriminative and hand-crafted features (almost always multidimensional features); (2) training classifiers (*e.g.*, support vector machine (SVM), ensemble classifier). This pipeline usually performs well in relatively simple datasets, such as those in which NIs are acquired by only one or two digital cameras. However, such methods often exhibit limited performance (to be shown later) in complex datasets comprising images of heterogeneous origins. An example is the challenging setting of the Columbia dataset [1], in which we want to differentiate between NIs collected from Google Image Search (Google) and photorealistic computer graphics (PRCG) images downloaded from various websites of CG image collections. In general and as argued by other image forensic researchers [2], [3], discriminative hand-crafted features are tedious to design, and the designed features are not necessarily the most adequate for a given forensic problem, especially in complex and challenging datasets.

Fig. 1. Pair of images. On the left is a CG image, and on the right is an NI. Both images are from the Columbia dataset [1].

Convolutional neural network (CNN) has recently obtained notable success in computer vision and pattern recognition [4], [5]. An important reason is that CNN attempts to *automatically* learn multiple levels of an appropriate representation for a given task in an "*end-to-end*" manner and from available data. This automatic feature learning and abstraction from data makes CNN more suitable for complex datasets (*e.g.*, with heterogeneous origins) than conventional hand-crafted features that are based on a set of prior knowledge and assumptions. Inspired by the recent success of CNN, several researchers have introduced CNN into multimedia security research [2], [3], [6]–[8]. In our work, we propose a new data-driven, CNN-based framework to distinguish between NIs and CG images. The proposed framework is different from the traditional pipeline of almost all existing methods with two separate steps of feature extraction and classifier training. The proposed framework is "end-to-end" and does not require designing features by hand.

Our work provides several contributions:

- We introduce a generic framework that uses CNN to identify NIs and CG images. This framework can be easily adjusted to handle different sizes of input image patches.
- We fine tune a pre-trained CNN and subsequently design and implement a new and improved CNN for this forensic problem. Both CNN-based solutions outperform state-of-the-art methods that combine hand-crafted feature extraction and classifier training.
- Our method exhibits good forensic performance in the challenging dataset of Google versus PRCG comprising images of heterogeneous origins and is thus close to the real-world application. Our method also demonstrates strong robustness against several post-processing operations, including resizing and JPEG compression.
- Unlike previous attempts to use CNNs for other image forensic problems, we attempt to understand what our CNN has learned about the differences between NIs and CG images by using advanced visualization tools, which provide interesting observations and insights for future studies.

The remainder of this paper is organized as follows. Section II presents relevant existing work and delineates how our work differs from the related work. Section III describes the dataset used for validating our method. Section IV discusses the motivation of every step of our work, important technical points, and the proposed network. Section V presents the performance evaluations for our method and detailed comparisons with existing methods. Section VI provides interesting observations from our CNN obtained by using a set of advanced visualization tools. Section VII shows the conclusions.

II. Related Work

A. Distinguishing Between NIs and CG Images

For the discrimination of natural versus CG images and videos, two lines of research, namely, (1) subjective, perceptual studies and (2) objective studies, mainly exist.

Subjective studies involve performing a series of psychophysical experiments to study the effects of image properties and cognitive characteristics of human observers on the discrimination between photorealistic and photographic images. Farid and Bravo [9] designed and conducted a perceptual study, in which human observers were asked to identify CG and photographic images of different scenes and contents. They reported that human observers possess a reliable capability of distinguishing NIs from CG images. On the basis of this first study, the same authors [10] focused on CG versus photographic faces and studied the effects of image properties, *i.e.*, resolution, JPEG compression and color versus grayscale. Fan *et al.* [11] studied the impacts of cognitive characteristics of viewers and image properties (*i.e.*, color and shading). Experimental results showed that experts outperform laypersons but only for grayscale images. Holmes *et al.* [12] found that human observers have a certain degree of bias in identifying photographic and CG portraits, and viewers are more likely to select the former than the latter. Such bias, however, could be significantly reduced by a small amount of training before the main experiment. Recently, Mader *et al.* [13] further enhanced this effectiveness by introducing feedback and incentives.

Objective studies usually depend on the statistical or intrinsic properties of natural and CG images or videos and design efficient algorithms to separate them. For CG character identification, a simple and popular strategy is to find a class-sensitive quantity and select an appropriate threshold for classification. For example, Dang-Nguyen *et al.* [14] distinguished CG characters from real ones by analyzing variations in facial expressions. Conotter *et al.* [15] identified CG faces in videos by detecting a physiological signal resulting from human pulse, which was absent in videos of CG faces. The features extracted by such methods can also be combined with machine learning. Dang-Nguyen *et al.* [16] proposed an asymmetry-information-based method to discriminate between natural and CG human faces with a threshold, and this feature can be added to other feature sets to improve their performance by using SVM binary classification. To distinguish between natural and CG faces in videos, Dang-Nguyen *et al.* [17] developed an SVM-based method by examining the spatial-temporal variation of their 3D models. The underlying idea is that the variations in real faces are more complex than those in CG faces. The latter often follows repetitive or fixed patterns.

For general cases, in which we wish to distinguish between NIs and CG images of various scenes that are not limited to those depicting human beings, the most common strategy is to use machine-learning-based methods with multidimensional feature extraction and classifier training. Inspired by the generation process of natural and computer-rendered images, specially object model, light transport, and acquisition differences, Ng *et al.* [18] proposed geometry-based features aided by fractal and differential geometry. They created an open dataset [1] to assess this approach. Lyu and Farid [19] proposed a feature combining first-order and higher-order wavelet statistics for the identification of photographic and photorealistic images. On the basis of the fact that the hue, saturation, and value (HSV) color model is motivated by the human visual system, Chen *et al.* [20] adopted the feature of the statistical moments of wavelet characteristic function in the HSV space to separate CG images from NIs. Gallagher and Chen [21] detected traces of demosaicing of original camera images to distinguish camera images from computer graphics and reported a good forensic performance. However, this method may be sensitive to post-processing operations, such as resizing, which can remove the demosaicing interpolation structure [22]. On the basis of several previous studies, Sankar *et al.* [23] proposed a set of combined features, including periodic-correlation-based feature [24], color histogram feature [25], moment-based statistical feature in the YCbCr color space [20], and local patch statistics [18]. Zhang *et al.* [26] proposed a method that analyzed the statistical property of local image edge patches. First, a visual vocabulary on local image edges was constructed with the aid of Voronoi cells. Second, a feature vector was formed with a binned histogram of visual words. Finally, an SVM classifier was trained for image classification. Recently, Peng *et al.* [27] used a linear regression model to extract the residual of a Gaussian low-pass-filtered image and combined the histogram statistics and multi-fractal spectrum of the residual image with the fitness of the regression model as a feature to discriminate between NIs and CG images. Several other methods that borrow features from steganalysis have been proposed [28]–[30]. In this study, we consider a recent and state-of-the-art steganalytic feature called subtractive pixel adjacency matrix (SPAM) [31] to classify NIs and CG images.

All of the methods above manually design discriminative features in either the spatial domain or a transformed domain. By contrast, our method automatically learns discriminative information from the data. For algorithm validation, many existing methods construct their own datasets, which often include images acquired by one or two digital cameras with a similar resolution and visual quality. This forensic scenario is somehow less challenging than the difficult setting of the Columbia dataset, *i.e.*, Google versus PRCG, which comprises NIs and CG images of heterogeneous origins. We mainly consider this challenging setting but also present results for simple datasets. Robustness against post-processing on NIs and CG images is another important factor for forensic methods to be useful in real-world applications. Among previous studies, Zhang *et al.* [26] studied the impact of JPEG compression post-processing and found that this process leads to the loss of image edge information. In Section V-D, we present a series of experiments performed for robustness evaluation against typical post-processing operations and provide the comparison results with existing methods (including [26]).

B. CNN for Computer Vision and Multimedia Security

LeCun *et al.* [32] accomplished groundbreaking work of using CNN aided by the back propagation algorithm [33] to identify handwritten postal codes, and they obtained high accuracy. With the improvement in machine performance, especially graphics processing units, Krizhevsky *et al.* [4] introduced a complex and deep CNN architecture, *i.e.*, AlexNet, and won the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012). Jia *et al.* [34] slightly adjusted the training sets and network architecture of AlexNet, and provided a pre-trained CNN (CaffeNet) based on Caffe [35] as the reference model. A popular technique based on transfer learning [36], *i.e.*, fine-tuning, has further increased the power of deep models [37], [38]. Fine-tuning is an efficient solution to reuse the weights of specific layers of a pre-trained base network and carry out training on a new task.

In the multimedia security community, a number of approaches use CNN for steganalysis [6]–[8], [39] and image forensics [2], [3], [40]–[42]. Qian *et al.* [6] proposed a deep model based on CNN for steganalysis and reported encouraging results. Later, Pibre *et al.* [7] studied the "shape" of CNN and identified the best CNN model after numerous experiments. Their CNN obtained a significant performance improvement over previous steganalytic methods in the specific setting of reused secret key for information embedding. However, the performance of their model in the realistic setting of using a random secret key for each embedding was weaker than that of conventional methods based on hand-crafted features. Chen *et al.* [39] proposed a CNN-based deep model for steganalysis of JPEG-domain steganography that is different from the spatial-domain steganalysis [8]. They introduced the concept of JPEG-phase awareness into the CNN architecture. For source camera identification, Tuama *et al.* [40] combined a high-pass filter with CNN to automatically and simultaneously extract features and learn to classify. Recently, Bondi *et al.* [41] utilized CNN to extract characteristic camera model features in an automated manner and trained an SVM for classification. This method outperforms previous methods in small color image patches, and its features have a good generalization capability. Using these deep features, Bondi *et al.* [42] introduced an iterative clustering algorithm to efficiently solve image tampering detection and localization.

In many previous CNN-based methods for multimedia security [2], [3], [6], [8], including recent methods [43], [44], a rather ad-hoc layer (fixed or constrained) exists in the beginning of the deep model. This layer is often composed of one or several high-pass filters and is fixed and thus not trainable during the training process. For example, Qian *et al.* [6] and Xu *et al.* [8] used a high-pass filter to extract the noise residual of an input image at the bottom of their network.

Similarly, Chen *et al.*'s CNN [2] also had a first fixed filter layer that accepts an image as the input and outputs its median filtering residual. This essentially performs nonlinear high-pass filtering. Inspired by such empirical technical choices, Bayar and Stamm [3] introduced a constrained filter layer for universal image manipulation detection, that is, for this layer, the center of filter kernel is -1, and the sum of remaining elements is $+1$. They forced this first layer to learn a set of high-pass, prediction error filters with such a constraint. The core idea of this constrained filter layer is to extract manipulation traces (assumed to be of high frequency) while suppressing the semantic content (assumed mainly to be of low frequency) of the input image.

In our work, we use a 3D filter group in the first layer instead of 2D linear convolutional kernels because our input is RGB images. The 3D convolutional filters are trainable without any constraint and are thus flexible. We show through experiments and visualization that these filters can, to some extent, automatically extract useful and discriminative features from available data for our classification task.

During the revision stage of this paper, we become aware of a recent parallel work for distinguishing CG images from NIs that also uses CNN [45]; hereafter, we refer to it as StatsNet. In Section V-H, we provide detailed qualitative and quantitative comparisons and show the advantages of our method over StatsNet in terms of network design, architecture, and experimental forensic performance, especially in the challenging Columbia dataset [1].

III. Dataset

All our experiments are conducted on the Columbia Photographic Images and PRCG Dataset [1]. Our experiments consider three sets of images from the Columbia dataset: (1) 800 PRCGs from 40 3D graphic websites (PRCG), (2) 800 NIs from the authors' personal collections (Personal), and (3) 800 photographic images from Google Image Search (Google). We remove five images with incorrect labels from the Google set after discussing via email with the first author of the dataset and obtaining his approval. The final number of images in the Google set is 795. Previous studies have included several common dataset settings, such as Personal+Google versus PRCG [18], [20], Personal versus PRCG [21], and authors' own datasets (mostly not publicly available) [19], [27], [46], [47], which were sometimes combined with the Columbia dataset. The NIs in the authors' own datasets are often acquired by a small number of digital cameras; this is similar to the configuration of Columbia's Personal set and thus appears to be less challenging. To the best of our knowledge, no previous method has been tested under the challenging setting of Google versus PRCG. This setting is difficult because NIs in Google and CG images in PRCG have heterogeneous origins [21]. We focus on this most challenging setting, *i.e.*, Google versus PRCG, which comprises images that we typically encounter in a real-world forensic scenario. We also test our method on two other settings: Personal versus PRCG and Personal+Google versus PRCG.

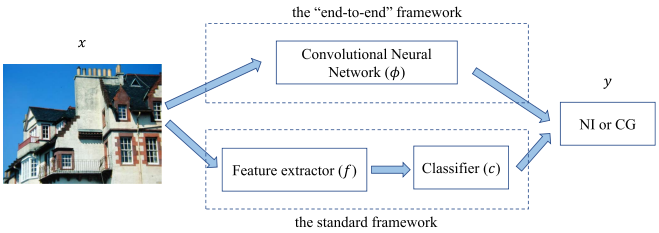


Fig. 2. Two different frameworks for the image forensic problem.

IV. Proposed Framework

A. Motivation

Distinguishing between NIs and CG images can be regarded as a binary classification problem. Given the set of training data $\{(x^1, y^1), \cdots, (x^N, y^N)\}$ of N samples, where x stands for the image and y corresponds to its label (0: CG image and 1: NI), our goal is to find a good mapping function $\phi : y = \phi(x)$ using the given training samples.

For this problem, the standard framework (bottom of Fig. 2) is to find a mapping $y = c(f(x))$, where f is a well-designed feature extractor and c stands for a classifier, such as SVM, or in a larger sense, it can also stand for thresholding for scalar features. This framework is a two-stage model, and its core is the feature extractor. However, these features are often time-consuming and tedious to design and not necessarily the most adequate ones, especially for complex data with heterogeneous origins. A generic "end-to-end" framework (top of Fig. 2), such as CNN, may be a better option. Given a testing image, a well-trained CNN can directly and accurately predict its label. To this end, we need to introduce a suitable CNN model for our framework. We consider three different methodologies in our approach: (1) following the existing network architecture and training it from scratch, (2) fine-tuning an "off-the-shelf" network that has been pre-trained on another dataset and/or for another task, and (3) designing a new network and training it from scratch. Before providing details on these methodologies, we present our general strategy adopted when using CNN for classifying NIs and CG images.

B. Local-to-Global Strategy

In view of computational cost, diversity of image size, and specific requirement of image forensics, we adopt the *local-to-global* strategy of training in small patches and classifying full-sized images using the simple majority voting rule. This strategy is partly based on the concept of data augmentation, which is a commonly used technology to expand training data, especially for deep learning [4], [48]. Krizhevsky *et al.* [4] randomly altered the intensities of the RGB channels of each training image using principal component analysis. The motivation behind this scheme is that object identification in digital images should be invariant to changes in the pixel intensity and color of the illumination. Simonyan and Zisserman [48] resized each training image, with the length of its shorter edge as an integer randomly sampled from the range of $[256, 512]$ for scale augmentation.

For our classification problem, on the one hand, local decisions, *i.e.*, high accuracies in small image patches, are important and generally desirable in many image forensic applications. On the other hand, a small patch cropped from a CG image is still CG, and this is also true for NI. Therefore, we apply *patch augmentation*, that is, we crop a certain number of image patches of a fixed size from each training image to augment the training dataset and try to obtain an accuracy as high as possible on patches.

This strategy is flexible for local and global forensic decisions. The direct result of such a strategy is high classification accuracy on patches, and the strategy can thus be used for the case of local decisions without any modification. For global decisions, merely conducting majority voting of multiple local decisions can lead to good performance, which is a natural result of the high accuracy on patches. In practice, we randomly crop a fixed number of patches from each training image using Maximal Poisson-disk Sampling (MPS) [49] to construct the training set. Unlike random sampling, cropping with MPS can completely cover the entire image and thus retains the original information as much as possible. In the testing phase, we crop a certain number of patches from each testing image and take the label (0: CG image and 1: NI) of patches with a higher number as the prediction result of this image. As shown later, this strategy can also enhance the performance of existing approaches that are based on manually designed features.

C. Fine-Tuning

Fine-tuning, a technique based on the concept of transfer learning, is pervasive in the field of deep learning. Yosinski *et al.* [38] analyzed the transferability of neurons in each layer of a deep CNN. For similar datasets, they found that initializing the weights of almost any number of layers from a well-trained network in an original dataset can improve the generalization performance after fine-tuning to the new dataset. Such transferability generally declines as the dissimilarity between the source and target task/data increases. However, fine-tuning pre-trained CNN models from computer vision tasks has been omitted by the multimedia security community. We are curious about and want to verify the transferability of such pre-trained models when applied to image forensic problems, although our available data and classification task are somewhat different from those of the pre-trained CNN model.

A well-known reference network for visual recognition is CaffeNet [34], which is trained on 1.3 million images with 1,000 categories. CaffeNet has eight layers (or group of layers): two convolutional groups, each of which includes one convolutional layer, one max-pooling layer and one local response normalization layer; three cascaded convolutional layers, followed by a max-pooling layer; and three fully-connected (FC) layers. In CaffeNet, each convolutional layer consists of linear multidimensional convolutional kernels and rectified linear unit (ReLU) activation [50], [51]. We successively fine-tune the first N layers, where $N = 1, 2, \cdots, 7$. We always need to change the number of neurons in the last output layer from 1,000 (for the 1,000 classes of ImageNet [52]) to 2 (binary classification of NIs and CG images). The detailed results are reported in Section V-B, where we show that fine-tuning CaffeNet leads to satisfactory results that are better than those of state-of-the-art methods.

Next, we decide to design and implement our own CNN that can cope even better with the classification of NIs and CG images. We present its architecture and energy function in the next two subsections. The design of the new CNN is motivated by the following observations and intuitions. First, our task and the corresponding dataset are more or less dissimilar to those of CaffeNet (in particular, no CG image is used during the training of CaffeNet); therefore, transferability might not be optimal. Second, CaffeNet is a relatively complex CNN designed for advanced and complicated computer vision tasks, but our task on hand is a less complicated two-class classification problem. Thus, a less deep and less complex network would suffice to solve our problem. Third, the fixed architecture of CaffeNet prevents us from easily adapting the network to accommodate different sizes of input patches.

D. Our Network – Architecture

The input of our network is an image patch, and the output is a binary label. One image patch is abstracted step by step through nonlinear mapping (*i.e.*, linear convolution and nonlinear activation) and down-sampling. A powerful high-level reasoning is applied. The informative and highly abstracted vector is converted into the probability vector of the label. Fig. 3 illustrates the architecture of our network. The entire network is made up of the so-called *convFilter* layer, three convolutional groups, two FC layers and a softmax layer. Before explaining each layer, we mention one detail about the relationship between the patch and input sizes of CNN. We actually follow the common way in the field of computer vision [4], [53]. The patch size (240×240) is slightly larger than the input size of the network (233×233) shown in Fig. 3, which can increase the space of training samples and is thus useful in suppressing potential over-fitting. During each iteration of network training, every 233×233 training sample is randomly cropped from a 240×240 patch. In the testing stage, the network extracts five patches of 233×233 pixels (the center and four corner patches) from a testing sample, flips these patches in the left-right direction (*i.e.*, horizontal reflection), and finally averages the predictions of total 10 patches as the final result.

The convFilter layer consists of a few 3D convolutional kernels (32 in our network). In multimedia security, such as steganalysis, a common operation is applying a group of filters on an input image/patch prior to the execution of the main algorithm [31], [54]. The convFilter layer cannot be simply regarded as "pre-processing" like fixed and manually designed filters in previous steganalytic methods because our layer is trainable without any constraint. In addition, these kernels are not explicitly required to have high-pass properties, such as in several previous methods that use CNN for steganalysis and forensics [2], [3], [6]. Technically, the convFilter layer maps an RGB image to several feature images filled in with real value elements and co-adapts to the successive convolutional group. In Section V-B, we analyze the classification accuracy

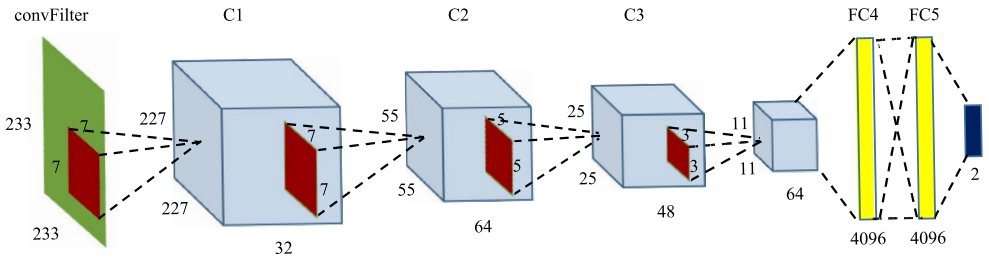


Fig. 3. Architecture of our network. The network input is a 233 × 233 RGB image, which is represented by a green square for simplicity. A red square stands for a convolutional kernel, and the numbers close to it denote the kernel size. For example, the first red square from the left is a 7 × 7 convolutional kernel. The feature maps are represented by shaded cuboids. No padding exists in our network.

of our network with the convFilter layer and compare with several different configurations related to this layer.

In our network, a convolutional group includes convolutional (Conv), batch normalization (BN), ReLU activation and max-pooing layers. The Conv layer conducts multidimensional linear operations and produces multiple feature maps. BN [55] explicitly forces the output of Conv to take on a unit Gaussian distribution. This layer makes network training highly robust to poor initialization. The ReLU activation layer introduces nonlinearity into the network and thus enhances the mapping capacity of the model. Its form is $f(x) = max(0, x)$. Max-pooling is a down-sampling operation, where the maximum value within a local window is taken as the output. On the one hand, this operation reduces the number of parameters to learn by reducing the spatial size of representation and thus decreases computational cost. On the other hand, this operation provides basic translation invariance to internal representation. In our network, all max-pooling layers have the same kernel size of 3 × 3 and a stride of 2.

The two FC layers constitute an FC two-layer neural network, where every single neuron connects to all neurons in the previous layer. The network conducts high-level reasoning. Many parameters of the network are located here. A simple and effective regularization technique, *i.e.*, Dropout [56], is applied to each FC layer to prevent potential over-fitting. In the training stage, each unit in the FC layers is kept active with a probability (default value is 0.5), with the interpretation of sampling the neural network and updating the weights of such sub-networks on the basis of input data. No Dropout is applied in the testing stage.

The softmax layer maps the high-level feature vector (output of FC layers) to the probability vector of class labels. Therefore, the dimension of its output is equal to the number of classes, and the sum of its output is 1.

Our proposed network is flexible and can accommodate multiple input sizes. We do not change the depth of the network and the number of kernels in each layer during the adjustment of network architecture to maintain the structural stability of our CNN. For a small input size, a new network can be rapidly built by simply reducing the kernel size and removing the striding of the first few layers (further details in Section V-C). This minor adjustment also ensures that experimentally the input flow can propagate to the last FC layers with a sufficient amount of useful information.

E. Our Network – Loss Function with Regularization

CNN models are usually trained by minimizing a well-designed loss function with the aid of back propagation. A loss function is often composed of a data loss term and a regularization term. The data loss term evaluates the compatibility between a prediction (*e.g.*, the class scores in a classification problem) and the ground-truth label, and the regularization term on model weights is designed to prevent the over-fitting of trained models. In our method, we use multinomial (binomial in our case) logistic loss (also known as cross-entropy loss) with softmax, that is

$$J(\theta)^{(data)} = -\frac{1}{N}[\sum_{i=1}^{N}\sum_{j=1}^{K}\mathbb{1}\{y^i = j\}log\frac{e^{a_j^i}}{\sum_{j=1}^{K}e^{a_j^i}}], \tag{1}$$

where N is the number of training samples, K is the number of categories, $\mathbb{1}\{\cdot\}$ is the indicator function so that $\mathbb{1}\{True\} = 1$ and $\mathbb{1}\{False\} = 0$, $\frac{e^{a_j^i}}{\sum_{j=1}^{K}e^{a_j^i}}$ is the softmax function (normalized exponential function) that converts the network output into the probability of the class label, $a^i = \phi(x^i, \theta)$ is the 2D output vector of the network parameterized by θ, and $K = 2$ in our case.

We select L_1 regularization to be added to the loss function to reduce the complexity of model and prevent over-fitting. The total loss is

$$J(\theta) = -\frac{1}{N}[\sum_{i=1}^{N}\sum_{j=1}^{K}\mathbb{1}\{y^i = j\}log\frac{e^{a_j^i}}{\sum_{j=1}^{K}e^{a_j^i}}] + \lambda|\theta|, \tag{2}$$

where regularization weight λ balances the data loss and regularization terms.

We also attempt L_2 regularization and find that L_1 regularization yields better results. A possible explanation is that in this work, we consider a binary classification problem, which is not highly complex. From a human cognition point of view, solving such a problem may not require a large amount of brain activity and area to learn and understand. Analogically, our problem may just require a relatively simple model, that is, a model with sparse parameters reflected by the selected L_1 regularization.

TABLE I

IMPACT OF NUMBER OF EXTRACTED PATCHES (IN ROW) ON THE NETWORK'S PERFORMANCE FOR TESTING PATCHES OF 240 × 240 PIXELS ON THE GOOGLE VS. PRCG DATASET

Num.	Accuracy (%)	Standard deviation (%)
100	84.67	0.6203
200	85.15	0.2861
300	85.15	0.2545

TABLE II

IMPACT OF DIFFERENT NUMBERS OF EXTRACTED PATCHES ON THE CLASSIFICATION ACCURACY OF TESTING PATCHES OF 240 × 240 PIXELS AND FULL-SIZED TESTING IMAGES WITH VOTING FOR THE BEST TWO STATE-OF-THE-ART METHODS, NAMELY, GEO [18] AND SPAM [31]. EXPERIMENTS ARE CONDUCTED ON THE GOOGLE VS. PRCG DATASET

Num.	Geo		Spam	
	patch	voting	patch	voting
10	80.65	87.91	76.13	84.63
20	80.76	88.16	76.33	84.89

V. EXPERIMENTAL RESULTS

A. Experimental Setting and Implementation Details

As mentioned in Section III, we perform experimental studies on the Columbia Photographic Images and PRCG Dataset [1] mainly for the challenging setting of Google versus PRCG. Before conducting all the experiments, we resize all images using bicubic interpolation so that the shorter edge of each resized image has 512 pixels. This operation can reduce the impact of scale and thus ensure the consistency of all image patches. For all three settings, namely, Google versus PRCG, Personal versus PRCG and Personal+Google versus PRCG, we use the ratio of 3:1 to randomly split each dataset into training and testing sets. To follow the local-to-global strategy and generate sufficient training data for our CNN model, we randomly crop 200 patches from each training image using MPS [49]. Similarly, the testing set is obtained by cropping 30 patches from each testing image. Every patch is pre-processed by subtracting the per-pixel mean of all training patches. Stochastic gradient descent with a minibatch of 128 patches is used to train CNN models. The base learning rate is initialized to 1e-3 and is divided by 10 every 30K iterations. The training procedure stops after 90K iterations. The default value of regularization weight λ is 1e-4, except for patch sizes of 60×60 and 30×30, whose regularization weights are 5e-5 and 1e-5, respectively. As the patch size decreases, the number of parameters in the corresponding CNN model decreases (additional details of networks for different patch sizes are given in Section V-C). Thus, using a small λ value for regularization is reasonable, and experimentally, this leads to a slightly improved performance.

As described above, we extract 200 patches from each training image. For the patch size of 240×240, we have relatively high overlapping between patches. This does not weaken the performance of our network. Table I shows the median and standard deviation of the results of 7 runs for different amounts of cropped patches (*i.e.*, 100, 200 and 300). Compared with 100 patches, the classification accuracy of 200 patches is increased by 0.48%, and the standard deviation is reduced by 0.3342%. This result means that in this case, doubling the training data can improve the performance and stability of our network. However, when we increase the number of cropped patches from 200 to 300, the network's performance remains nearly the same, but the computational cost increases.

We compare our proposed method with four state-of-the-art methods that are based on hand-crafted features, namely, Spam [31], Geo [18], Mfra [27], and Vlie [26] (the fourth method is mainly for robustness evaluation against JPEG post-processing). These four methods follow the conventional two-stage pipeline of machine learning and use SVM as the classifier. Considering the long training time and high memory footprint of SVM, we randomly crop 10 patches from each training image to construct the corresponding training sets for the first three conventional methods [18], [27], [31] and 15 patches for the Vlie method to compensate for the 100 images of each category that are used to compute the visual vocabulary, similar to the original paper [26]. The number of samples in these training sets ensures reasonable training time and memory consumption and is also sufficient for obtaining stable and near-optimal results for the four methods. As shown in Table II, doubling the training samples exerts a minor impact on classification performance, but the memory footprint considerably increases. For Spam with a high-dimensional feature vector, the memory consumption becomes prohibitive even on a computer equipped with 32GB of RAM. All these methods are evaluated on the same testing set as our proposed method. For SVM training, we use the popular and efficient LS-SVM implementation [57].

B. Fine-Tuning CaffeNet and Analysis of convFilter Layer

We have explored fine-tuning the CaffeNet for this forensic problem, and corresponding experimental results are reported in Table III, in which the accuracy is computed on all testing image patches of 240×240 pixels in the setting of Google versus PRCG. In addition to this selected accuracy, we generally observe the same trend for other metrics, such as the accuracy after voting on full-sized images. We also train CaffeNet on the Google versus PRCG dataset from scratch for comparison. All of the results of fine-tuning are better than the result of the network trained from scratch (the column of "C-S" in Table III), which is consistent with the observation in [38]. Through fine-tuning, we can obtain relatively good classification accuracies that are higher than those of traditional methods based on hand-crafted features. The detailed results of traditional methods can be found in the second last column of Table VII, with the highest attained accuracy being 80.65%, which is lower than any accuracy obtained by fine-tuning (*i.e.*, "C-1" to "C-7" in Table III). A possible explanation is that having a large number of NIs from ImageNet (to our knowledge, no CG image exists in ImageNet) is beneficial for the network during its pre-training, and this helps the network understand the "intrinsic" properties

TABLE III

CLASSIFICATION ACCURACY OF FINE-TUNING DIFFERENT LAYERS OF CAFFENET ON DATASET OF GOOGLE VS. PRCG. ACCURACY IS COMPUTED ON ALL PATCHES OF 240×240 PIXELS IN THE TESTING SET OF GOOGLE VS. PRCG. "C" STANDS FOR CAFFENET. "C-S" MEANS TRAINING CAFFENET FORM SCRATCH ON GOOGLE VS. PRCG. "C-N" MEANS FINE-TUNING THE FIRST N LAYERS OF PRE-TRAINED CAFFENET [34] WITH THE REMAINING LAYERS RETRAINED USING RANDOM WEIGHT INITIALIZATION, WHERE $N = 1, 2, \cdots, 7$

Network	C-S	C-1	C-2	C-3	C-4	C-5	C-6	C-7
Accuracy (%)	76.85	81.22	82.08	82.23	81.17	**82.71**	82.13	82.06

TABLE IV

IMPACTS OF DIFFERENT CONFIGURATIONS RELATED TO CONVFILTER LAYER ON THE CLASSIFICATION ACCURACY OF TESTING PATCHES OF 240×240 PIXELS ON THE GOOGLE VS. PRCG DATASET. 'CF' IS THE ABBREVIATION OF CONVFILTER. WE SHOW THE MEDIAN OF RESULTS OF 7 RUNS, EACH WITH RANDOM INITIALIZATION OF CNN

Network	with cF	without cF	with cF and ReLU	cF with constraint
Accuracy (%)	**85.15**	84.51	83.97	82.35

of NIs, one of the two classes that we want to distinguish in our work.

Our CNN copes better with this forensic problem. Here we first analyze the performance of the convFilter layer of our network. Table IV lists the classification accuracy of four different configurations related to this convFilter layer: (1) our proposed network shown in Fig. 3 with two cascaded convolutional layers at the beginning of network; (2) removing the convFilter layer from the proposed network; (3) inserting an additional ReLU activation layer in our network after the convFilter layer; and (4) adding the high-pass filtering constraint from [3] to the convFilter layer in our network. Our configuration provides the highest accuracy, which demonstrates the utility of convFilter layer. The classification accuracy decreases when the convFilter layer is followed by ReLU activation, which may be an evidence that the relationship of "co-adaptation" between the convFilter layer and the successive convolutional group is weakened by ReLU activation to some extent. Adding a constraint to the convFilter layer, such as in Bayar and Stamm's work [3], also leads to performance degradation, which means that prior knowledge that is useful for image manipulation detection in [3] is not well suited for the task of NI and CG image classification.

C. Classification Performance on Patches of Different Sizes

In this subsection, we compare the classification accuracy of the proposed method with that of state-of-the-art methods on patches of different sizes. As mentioned earlier, our network is slightly modified to accommodate different input sizes. The difference of networks used for different patch sizes is provided in Table V. Furthermore, the max-pooling of the C1 of Net-3 has no stride (*i.e.*, the stride is equal to 1). The network adjustment is simple. For small patches, we only reduce the kernel size and remove the stride to guarantee the structural stability of our CNN with a fixed depth and ensure

TABLE V

DIFFERENCE OF NETWORKS USED FOR DIFFERENT PATCH SIZES. "$7 \times 7(2)$" MEANS THAT THE CONVOLUTIONAL KERNEL SIZE IS 7×7 WITH A STRIDE EQUAL TO 2, AND ALL OTHER STRIDES ARE EQUAL TO 1

	convFilter	C1	C2	C3	F4	F5
Net-1	7×7	$7 \times 7(2)$	5×5	3×3	4096	4096
Net-2	5×5	5×5	3×3	3×3	2048	2048
Net-3	3×3	3×3	3×3	3×3	2048	2048

that the information flow can pass to the FC layers under an appropriately abstracted form. For Net-2 and Net-3, we cut the number of neurons of the FC layers in half to prevent over-fitting. The correspondence between networks and patch sizes is as follows: Net-1 for 240×240 and 180×180; Net-2 for 120×120 and 60×60; and Net-3 for 30×30. The corresponding input sizes of the networks for the five patch sizes are 233×233, 169×169, 107×107, 51×51, 27×27.

Fig. 4 shows a comparison of the classification accuracy of our method and those of three hand-crafted-feature-based methods (Spam [31], Geo [18] and Mfra [27]) under different patch sizes. For each patch size and method, the experiment is repeated seven times with different randomized initialization/parameterization to enhance the statistical significance of the results. We show the median of the results obtained by seven runs. Our method demonstrates the best performance for all patch sizes, followed by Geo and Mfra as the worst. As mentioned earlier, the classification performance of conventional methods based on hand-crafted features mainly depends on the discriminability of features. These features do not appear to be discriminative in this complex and challenging setting of Google versus PRCG. By contrast, our method automatically learns, as much as possible, useful and task-specific information from available data with the aid of the powerful learning capacity of CNN. Such automatic learning and unified "end-to-end" optimization for this classification task is a better choice than previous two-stage solutions. The accuracies of almost all methods decrease with decreasing patch size. This condition is understandable because smaller patches intuitively contain less information. Therefore, correctly classifying them is difficult for computational forensic algorithms and even human beings. The numerical results that correspond to the median accuracies shown in Fig. 4 can be found in the group of columns labeled "Original" in Table VI. The performance improvement of our method compared with the second-best method, Geo [18], varies between 2.48% and 4.50% depending on the patch size.

D. Robustness Against Post-Processing

An effective image forensic algorithm should not only correctly deal with original data, which is Columbia's testing data in our experiments, but should also have a good level of robustness on post-processed data because post-processing is likely to occur either as a routine operation or an intentional attack. To evaluate robustness, we perform tests against two typical post-processing operations of rescaling and JPEG compression. For rescaling, we consider down-scaling and up-scaling. Section V-A indicates that all images

TABLE VI

CLASSIFICATION ACCURACIES (%) FOR DIFFERENT TESTING SETTINGS: ORIGINAL, SCALE300, SCALE1000, JPEG90, AND JPEG80. "-" MEANS THAT IN THIS CASE, THE MFRA METHOD CANNOT SUCCESSFULLY EXTRACT FEATURES. FOR EACH TESTING SETTING, WE SHOW THE ACCURACIES OF FOUR METHODS (IN GROUP OF FOUR COLUMNS) ON PATCHES OF FIVE DIFFERENT SIZES (IN ROW). FOR EVERY TESTING SETTING, THE FIRST COLUMN (OUR METHOD) WITHIN THE GROUP OF FOUR COLUMNS ALWAYS HAS THE HIGHEST ACCURACY

Patch	Original				Scale300				Scale1000				JPEG90				JPEG80			
	Our	Geo	Spam	Mfra	**Our**	Geo	Spam	Mfra	**Our**	Geo	Spam	Mfra	**Our**	Geo	Spam	Mfra	**Our**	Geo	Spam	Mfra
240 × 240	85.15	80.65	76.13	63.04	84.33	76.38	64.09	56.15	85.01	80.04	73.62	61.44	83.52	76.94	65.53	58.35	82.39	75.82	63.47	59.50
180 × 180	83.69	79.32	74.61	60.94	83.63	75.18	63.67	55.94	83.78	78.77	72.04	59.27	81.79	75.58	64.05	56.13	80.92	75.05	62.70	57.26
120 × 120	81.81	79.08	72.70	61.37	80.65	74.82	63.41	57.56	81.72	78.44	71.64	60.04	79.10	74.50	64.21	57.98	77.56	73.82	62.43	58.10
60 × 60	77.03	74.55	69.75	59.81	76.71	72.59	61.13	57.75	76.98	74.12	69.64	-	74.69	71.32	62.80	56.98	73.73	71.51	61.34	56.85
30 × 30	73.45	69.30	66.53	-	72.38	67.96	57.61	-	73.33	69.19	66.16	-	71.05	67.55	60.81	-	69.80	66.61	59.71	-

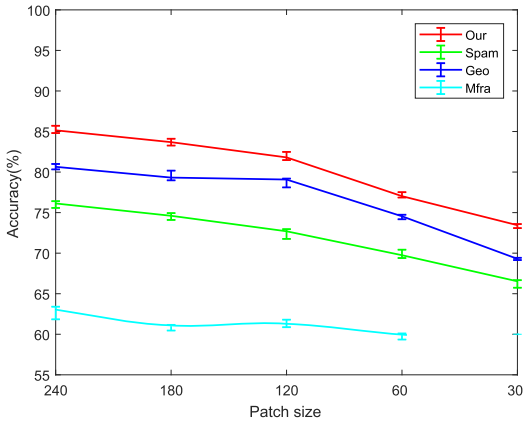


Fig. 4. Comparison of our method with three state-of-the-art methods under different patch sizes. The solid lines show the median of 7 runs, and the error bars illustrate the maximum and minimum. The result of Mfra for 30 × 30 patches is not indicated because Mfra fails to extract features from such small patches.

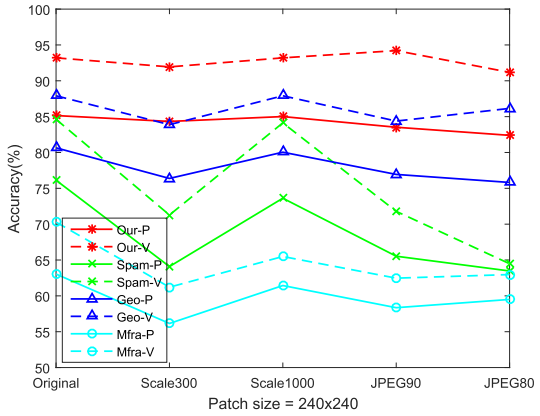


Fig. 5. Classification accuracies of the four methods on five different testing sets. The patch size is 240 × 240. "-P" is the accuracy on patches, and "-V" is the accuracy after voting on full-sized images.

to be classified are resized in a pre-processing step before they are fed into CNN so that the shorter edge of the resized image has 512 pixels. Therefore, we test our trained network on testing sets, including images with a shorter edge rescaled to 300 pixels (simulation of post-processing) and then resized back to 512 pixels by the pre-processing of our method ("Scale300") and images with a shorter edge rescaled to 1000 pixels and then to 512 pixels ("Scale1000"). We compare the results with the baseline setting (denoted by "Original"). We use bicubic interpolation to rescale the image while preserving its aspect ratio, and we intentionally choose 300 and 1000 pixels for rescaling to avoid the potential side effect induced by the divisor and multiple of 512 (*e.g.*, 256 and 1024). As for JPEG compression, in the first place we consider two typical quality factors: 90 ("JPEG90") and 80 ("JPEG80").

For all methods, we select the trained model, which provides the median classification accuracy of seven runs in the "Original" setting, to perform this robustness test. All testing results are reported in Table VI. Mfra fails to extract features from 30 × 30 patches because the patch is too small. For all five testing settings and five patch sizes, the performance of our method is stable and always better than that of the three other methods. As an example, for Spam, the average performance drop of "Scale300" on all patch sizes is 9.962%, whereas the corresponding value of our method is only 0.686%. In Fig. 5, the solid lines show the classification accuracies of the four methods on five different testing sets for 240 × 240 patches. Our method has stronger robustness than the other methods. The two post-processing operations slightly change the correlation of pixels with their local neighborhoods, and conventional methods, especially the Spam method, might be sensitive to this subtle alteration of local statistical property. By contrast, our method is almost insensitive to rescaling and quite robust against JPEG compression. This robustness can be attributed to the diversity of the challenging dataset and the powerful learning capability of CNN.

In addition, we investigate the JPEG robustness of our method, Geo [18], and Vlie [26] for a large range of quality factors, that is, from 100 to 10 with a step of 10. The corresponding results are shown in Fig. 6. Compared with Geo and Vlie, our method always demonstrates the best performance on patches and full-sized images under all considered factors. Although the results of Vlie remain relatively stable, the accuracies on patches and full-sized images are the lowest with or without (corresponding to the "Original" case in Fig. 6) JPEG post-processing. When the quality factor is very low (*e.g.*, 20 and 10, although such factors are rarely used in real-world applications), the performance of our method drops more rapidly than that of Geo but remains the best among the three methods.

E. From Local to Global Decision

The local-to-global strategy, an important component of our framework, is highly flexible and produces local and global

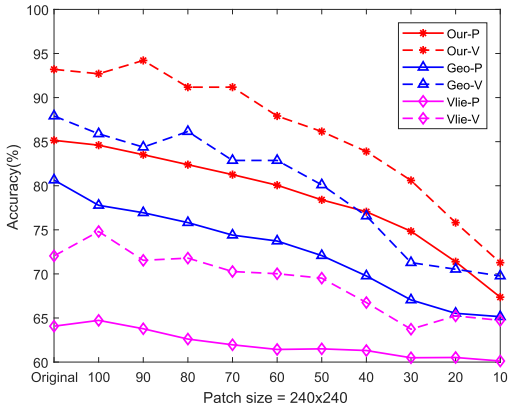


Fig. 6. Classification accuracies of our method, Geo [18], and Vlie [26] for a large range of quality factors, *i.e.*, from 100 to 10 with a step of 10. The patch size is 240 × 240. "-P" is the accuracy on patches, and "-V" is the accuracy after voting on full-sized images.

TABLE VII

EFFECT OF LOCAL-TO-GLOBAL STRATEGY ON THE CLASSIFICATION ACCURACY (%) OF THE SIX METHODS. "OUR-DF" MEANS THAT THE ACTIVATION OUTPUT OF FC5 IS EXTRACTED AS DEEP FEATURE AND AN SVM CLASSIFIER IS TRAINED. THE SECOND COLUMN CORRESPONDS TO THE CASE OF TRAINING ON THE FULL-SIZED IMAGES, AND THE THIRD AND FOURTH COLUMNS CORRESPOND TO THE TWO CASES OF TRAINING ON THE LOCAL PATCHES (240 × 240) THEN TESTING ON EITHER PATCHES OR FULL-SIZED IMAGES WITH VOTING. THE FIRST CASE DOES NOT APPLY TO "OUR" METHOD AND "OUR-DF" METHOD

Method	Full-sized images	Local patches	
	full-size	patch	voting
Our	-	85.15	93.20
Our-DF	-	84.62	92.70
Geo	86.14	80.65	87.91
Spam	81.86	76.13	84.63
Vlie	69.77	64.07	72.04
Mfra	65.49	63.04	70.28

decisions. Such a strategy applies not only to our CNN-based method, which is related to data augmentation, but also to conventional methods based on hand-crafted features, as shown later in this subsection.

The accuracies obtained by our method after voting from patches of different sizes (ranging from 30 × 30 to 240 × 240 pixels) are as follows: 83.63%, 88.41%, 88.66%, 92.70%, and 93.20%. Accuracy after voting refers to the accuracy on full-sized images where the predicted label of each testing image is obtained via majority voting of the predictions of 29 cropped patches (we ignore the last one of the 30 randomly cropped testing patches to avoid tie votes). The voting result is improved when the patch size increases, but we find a very minor performance improvement for patches larger than 240 × 240, which lead to a more costly computation. Therefore, we select the 240 × 240 patch to produce the final voting result. In addition, the voting accuracy, that is, the classification accuracy on full-sized images, of our method is always higher than the corresponding values of existing methods, as can be seen from the last column of Table VII. In particular, a considerable performance improvement of 5.29% is observed for our method compared with the Geo method, which is the best hand-crafted-feature-based method.

For an objective comparison with previous SVM-based methods, we extract CNN deep features (*i.e.*, the activation output of FC5) and train an SVM classifier with the same experimental setting of SVM-based methods (*i.e.*, cropping 10 patches from each training image). The results on local patches and full-sized images are reported in Table VII. Compared with the Geo method (best among all hand-crafted-feature-based methods), our deep-feature-based method ("Our-DF" in Table VII) is improved by 3.97% and 4.79% on local patches and full-sized images, respectively. This improved performance indicates that our deep feature has better discriminative power than traditional hand-crafted features. In addition, our CNN-based method has slightly higher classification accuracies than our deep-feature-based method (compare the second and third rows of Table VII). This finding implies that merging feature extraction and classifier training into a unified "end-to-end" framework brings additional benefits and supports our motivation (see Section IV-A) of developing a CNN-based method.

We then evaluate and verify the robustness of the voting accuracy against the post-processing operations, and the obtained results are shown by dashed lines in Fig. 5. The comparison of the four dashed lines indicates that our method has a stable and consistently better performance than the three state-of-the-art methods.

Next, we validate this local-to-global strategy on hand-crafted-feature-based methods, and the results are shown in Table VII. Each method has three cases: train on full-sized images and test on full-sized images; train on local patches and test on local patches; and train on local patches and test on full-sized images using voting. These three cases correspond to the last three columns of Table VII, respectively. The first case does not apply to our methods. The accuracy of training on local patches and testing on full-sized images with voting is higher than that of directly training and testing on the full-sized images for the four conventional methods, and this can be observed by comparing the second and last columns of Table VII. A possible reason behind this improvement is that the local-to-global strategy increases the diversity of training samples to some extent. In this work, we use the simple majority rule to vote. This point can be further improved in our future work.

F. Further Analysis and Failed Examples

In this subsection, we further analyze the results of our method in terms of two additional measures, namely, the error rate of CG patches misclassified as NI (denoted as CGmcNI) and its counterpart (denoted as NImcCG). The corresponding results are reported in Table VIII. With decreasing patch size, these two measures increase for almost all testing settings, which is consistent with the previous findings (Section V-C). The two error rates are often balanced. However, the NImcCG values of JPEG90 and JPEG80 are clearly higher than CGmcNI (last four columns of Table VIII, particularly for

TABLE VIII

STATISTICS ON MISCLASSIFICATION RATES (%) OF OUR METHOD FOR DIFFERENT TESTING SETTINGS: ORIGINAL, SCALE300, SCALE1000, JPEG90, AND JPEG80. WE CONSIDER THE ERROR RATE OF CG PATCHES MISCLASSIFIED AS NI (CGMCNI) AND ITS COUNTERPART (NIMCCG). FOR EACH TESTING SETTING, WE SHOW THESE TWO ERROR RATES (IN GROUP OF TWO COLUMNS) ON PATCHES OF FIVE DIFFERENT SIZES (IN ROW)

Patch	Original		Scale300		Scale1000		JPEG90		JPEG80	
	CGmcNI	NImcCG	CGmcNI	NImcCG	CGmcNI	NImcCG	CGmcNI	NImcCG	CGmcNI	NImcCG
240 × 240	15.67	14.03	16.65	14.67	15.70	14.26	16.06	16.90	19.13	16.06
180 × 180	15.95	16.67	16.45	16.29	16.20	16.26	14.63	21.84	17.87	20.32
120 × 120	19.20	17.17	22.93	15.70	20.28	16.24	17.55	24.30	21.55	23.33
60 × 60	20.98	25.00	23.77	22.80	22.00	24.06	19.52	31.18	21.15	31.47
30 × 30	25.70	27.41	28.12	27.11	27.02	26.31	24.15	33.82	24.97	35.52

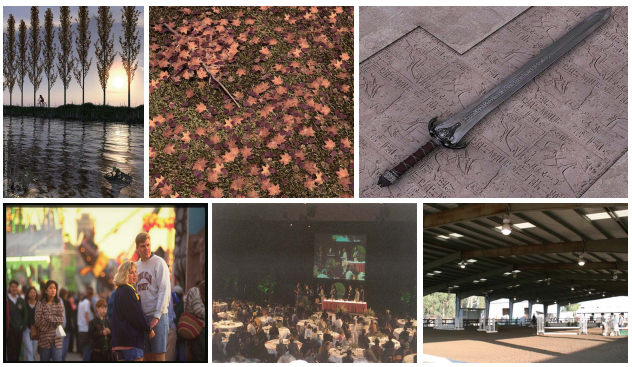


Fig. 7. Failed examples: the top row corresponds to CG images misclassified as NIs, and the bottom row corresponds to NIs misclassified as CG images.

small patches). A possible reason is that the details of natural patches are partially removed by JPEG compression. Thus, the NI patches, especially those of small sizes, become relatively "simple" and appear computer generated.

Fig. 7 shows several failed examples of our method. The light in the first image on the top row has good naturalness, and the color transition and texture of the two other images are rather plausible. Therefore, these CG images are misclassified as NIs by our network. On the contrary, the first two natural images on the bottom row have a certain degree of unnaturalness in light and color, and the last image has a dramatic color transition (*e.g.*, the ceiling and shadow). These clues lead to the wrong classification.

G. Performance Evaluation on Other Datasets

Our method demonstrates good performance in the highly challenging dataset of Google versus PRCG. We also conduct tests of the proposed method on other datasets, namely, Personal versus PRCG and Personal+Google versus PRCG and compare our method's performance with that of state-of-the-art methods. Fig. 8 shows a comparison of patch classification accuracies under the two settings. In these two settings, our method still exhibits the best performance, especially for Personal versus PRCG (Fig. 8(a)), where the classification accuracy remains stable for patches larger than 60×60 pixels. Table IX presents the classification accuracies on full-sized testing images obtained after voting from 240×240 patches. We observe an accuracy improvement of 1.50% and 4.02%

TABLE IX

COMPARISON OF CLASSIFICATION ACCURACY (%) ON FULL-SIZED TESTING IMAGES, OBTAINED AFTER VOTING FROM 240 × 240 PATCHES, ON TWO OTHER DATASETS

Dataset	**Our**	Geo	Spam	Mfra
Personal vs. PRCG	98.50	95.50	97.00	82.75
Personal+Google vs. PRCG	93.13	89.11	88.61	72.19

when we compare the result of our method to that of the second-best method (Spam and Geo, respectively) under the setting of Personal versus PRCG and Personal+Google versus PRCG, respectively. In addition, compared with the setting of Google versus PRCG (the last column in Table VII), a noticeable performance improvement for the setting of Personal versus PRCG (the second row in Table IX) is observed for our method and existing methods. Our explanation is that the Personal set is simpler (*i.e.*, acquired by a small number of digital cameras) than the Google set. Thus, the classification is less difficult, and the result is improved for all methods.

H. Comparison With StatsNet

During the revision stage of this paper, we become aware of a very recent parallel work presented in December 2017 at IEEE WIFS [45]. StatsNet, the CNN-based method proposed in that parallel work, follows the usual three-step procedure: filtering (*i.e.*, two conventional convolutional layers each with Conv and ReLU, different from our cascaded convolutional layers), statistical feature extraction (*i.e.*, mean, variance, maximum, and minimum of filtered images), and classification (*i.e.*, multi-layer perceptron). Several apparent differences exist between [45] and our work. First, we consider the highly challenging setting of Google versus PRCG, whereas [45] considered a relatively simple setting with homogeneous NIs and CG images, which is reflected in the experimental results presented below. Second, our network is deeper and has particularly designed cascaded convolutional layers. Therefore, it has a higher classification accuracy and is thus more suitable for distinguishing between NIs and CG images, as shown later in this subsection. Third, the method in [45] selects only the green channel of images, whereas our method uses all three channels. Lastly, we conduct a more comprehensive study of using CNNs for the discrimination of NIs and CG images; specifically, we consider the fine-tuning of CNN and attempt to understand what our CNN has learned about the difference

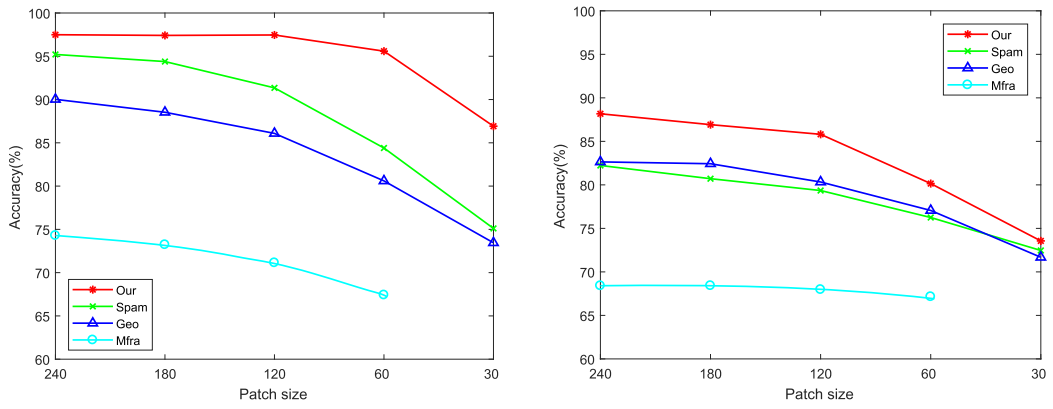


Fig. 8. Comparison of patch classification accuracy on two other datasets: (a) Personal vs. PRCG and (b) Personal+Google vs. PRCG. The results of Mfra on 30×30 patches are not provided because of feature extraction failure on such small patches.

TABLE X

COMPARISON OF CLASSIFICATION ACCURACY (%) OF OUR METHOD WITH THAT OF STATSNET ON FOUR DATASETS: RAISE VS. LEVEL-DESIGN, GOOGLE VS. PRCG, PERSONAL VS. PRCG, AND PERSONAL+GOOGLE VS. PRCG. "STATSNET" REFERS TO THE CASE WHERE THE NUMBER OF TRAINING PATCHES IS ALMOST THE SAME AS (IN FACT SLIGHTLY LARGER THAN) 40,000 USED IN THE ORIGINAL PAPER [45], AND "STATSNET2" REFERS TO THE CASE WHERE MORE TRAINING SAMPLES ARE USED WITH 200 PATCHES CROPPED FROM EACH TRAINING IMAGE (THE SAME AS IN 'OUR' METHOD)

Method	Raise vs. Level-Design		Google vs. PRCG		Personal vs. PRCG		Personal+Google vs. PRCG	
	patch	full-size	patch	full-size	patch	full-size	patch	full-size
Our	94.75	99.58	77.03	88.41	95.60	97.25	80.15	86.43
StatsNet	89.76	99.30	67.50	75.31	63.63	75.75	69.68	75.38
StatsNet2	89.68	99.30	68.27	76.57	65.56	80.00	68.51	69.51

between NIs and CG images by using advanced visualization tools (Section VI).

In the following, we experimentally compare the classification performance of our method with that of StatsNet, not only on their dataset (Raise versus Level-Design comprising 1,800 CG images from the Level-Design Reference Database [58] and 1,800 photographic images randomly selected from RAISE dataset [59]) but also on all the three datasets described in Section III. The results are reported in Table X. For StatsNet, we use the authors' shared implementation [60] and follow the default setting described in [45]. The patch size of StatsNet is 100×100, while we use patches of 60×60 pixels for our method. This is a disadvantageous setting for our method because small patches contain less information. However, although the patch size of our method is nearly a quarter of that of StatsNet, our network performs consistently better on all the four datasets (Table X). Furthermore, instead of using the default number of training samples as described in [45], we increase the amount of training data of StatsNet to match that of our method (*i.e.*, cropping 200 patches from each training image), and this network variant is denoted by StatsNet2. The last two rows in Table X show that in general, no guaranteed performance improvement from StatsNet to StatsNet2 is observed, although a large amount of data is used for training. A possible reason is that StatsNet is a three-layer network with a limited number of parameters; thus, using more training data than necessary would not improve its patch classification accuracy considerably.

I. Summary

Existing methods based on hand-crafted features exhibited rather limited performance, especially in the challenging setting of Google versus PRCG, mainly due to the limited discrimination capability of extracted features and the separate feature extraction and classification stages without strong co-adaptation between them. These shortcomings can be overcome by using a data-driven and "end-to-end" CNN-based solution. In addition, the performance of our network is better than that of StatsNet, which implies that a deeper architecture with cascaded convolutional layers probably has a stronger learning capacity and can cope better with this forensic problem.

VI. VISUALIZATION AND UNDERSTANDING

Our CNN-based method exhibits good performance in terms of classification accuracy and robustness against typical post-processing operations. This characteristic is attributed to a well-designed and implemented CNN model. In this section, we wish to understand knowledge that is hidden in the data and cannot be reflected by the quantitative evaluation metrics used in Section V. Specially, we analyze and understand what our CNN method has learned about the difference between NIs and CG images by using several advanced and appropriate visualization tools that are relevant to CNNs. This point was omitted in previous attempts of using CNNs for image steganalysis and forensics.

We study the filters that CNN has learned at its first layer. The first convolutional layer of CNN directly takes raw pixel

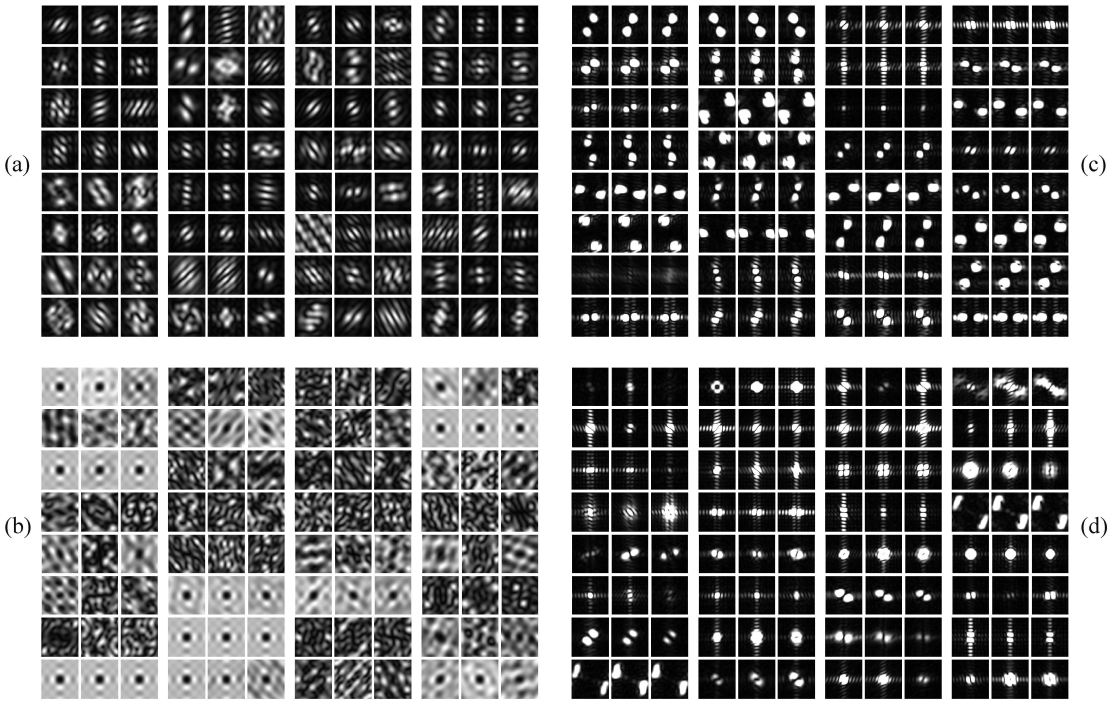


Fig. 9. Visualization of the FFT of the first-layer filters in our method (a), our method with the constraint from [3] (b), and pre-trained CaffeNet [34] (c,d). (c) corresponds to the first 96 kernels of the first layer in CaffeNet and (d) corresponds to the last 96 kernels. The filters are organized in groups of three (in columns) corresponding to the three color channels B, G and R. Brighter pixels mean higher values.

data as an input and is thus more interpretable than other layers in the remaining part of the network [61]. A common phenomenon exists in many CNNs well trained on natural images for computer vision tasks: the kernels they learn in the first layer are similar to Gabor filters and color blobs [38]. From the signal processing perspective, the convolution kernels of the first layer are linear filters. Therefore, a powerful analysis tool, that is, fast Fourier transform (FFT), can be used to analyze the properties of these kernels. Fig. 9 shows the FFT of the kernels in the first layer of our CNN, our CNN with an additional high-pass filtering constraint from [3], and CaffeNet pre-trained on ImageNet. The filters are organized in groups of three (in columns), which correspond to the three color channels B, G and R. Many kernels in the first layer with the constraint of [3] [Fig. 9(b)] have an apparent high-pass response, whereas the convFilter kernels of our method [Fig. 9(a)] mainly capture the band-pass frequency information. However, the high-pass filtering constraint reduces the performance by approximately 3% (see results in Table IV). This performance drop is evidence that the band-pass information in a certain range of frequency may be more useful for identifying NIs from CG images than that at high frequencies. Furthermore, the first group of 96 filters of the first layer in CaffeNet shown in Fig. 9(c) are highly consistent among the three color channels, but this consistency slightly decreases in the last group of 96 filters, as shown in Fig. 9(d). The former collects orientated information, and the latter considers color to some extent. By contrast, almost all the filters in our method show no apparent consistency among the three channels, as illustrated in Fig. 9(a), which implies that this identification task between NIs and CG images is more color-sensitive than conventional computer vision tasks.

To summarize, we obtain the following observations concerning filters in the first layer. Image forensic tasks may need a new set of appropriate filters aside from those tailored for computer vision tasks, but not necessarily high-pass filters as suggested in [3]. Different forensic tasks may require different, adequate filters that can be learned with or without constraint. An appropriate constraint may improve performance as shown in [3], whereas an inappropriate constraint may decrease the performance as shown in our paper. In the latter case, "freely" learning these filters is a better solution. Further studies should examine the interesting research problem of the design and training of CNN and its layers for different forensic problems.

In the following, we continue our analysis of what our CNN has learned and what inspiration we can obtain from the well-trained model. Through two advanced visualization tools, namely, layer-wise relevance propagation (LRP) toolbox [62] and deep visualization toolbox [63], we analyze the trained model from the *data-centric* and *network-centric* point of view, respectively. The network-centric approach only requires the trained network for its analysis, and the data-centric approach additionally requires passing sample data through that network.

LRP [64] is a technique for determining the degree of local contribution in an individual input to the neural network's output. In practice, we can obtain information about which pixels in the input image are relevant to the prediction outcome of the CNN by using the LRP toolbox [62]. The

Fig. 10. Heatmaps of four sample image patches. The top row corresponds to NIs, and the bottom row corresponds to CG images. Each group consists of the input image patch (left) and its heatmap (right). The red color in heatmaps stands for a large value, blue for a small value, and black for an intermediate value.

relevance scores assigned to the pixels can be visualized as an image with the same size as the input image, which is called a *heatmap*. Fig. 10 shows four sample image patches and the corresponding heatmaps on our trained CNN model. Here, the CNN model does not use batch normalization due to the limitation of LRP toolbox. Fig. 10(a) and (c) are NIs from Google while (b) and (d) are CG images from PRCG. We observe several very red pixels (meaning high contributions) in the heatmap of (b) corresponding to bright parts on the forehead, shoulder, and arm in the CG image, which implies that the prediction of CNN is relevant to the unnaturalness of light. The same CNN regards the light in (a) as rather natural, which contributes to the prediction (see red pixels corresponding to the left collar and slightly red pixels on the nose and chin). For (d), high relevance in the bottom of car is observed because the transition of the shadow is unnaturally sharp, but the color transition in (c) is smoother and natural and thus contributes to the prediction. Hence, our CNN model uses the naturalness degree of light and the smoothness of color transition as important clues for NI and CG image classification. This finding also provides insights into possible directions for computer graphics algorithms to further improve the photorealism of rendered images and synthesized images [65].

With the help of the deep visualization toolbox [63], we compute the preferred inputs in the image space for two output units located immediately prior to the final softmax layer, which are shown in Fig. 11. The preferred input refers to the input image that causes the corresponding unit to have high activation. Fig. 11(a) is filled by multiple color blobs, which implies that CG images often contain large color primitives and look relatively "simple". By contrast, the recurrent appearance of "light points" shown in Fig. 11(b) implies that natural images have more variability and look rather "complex". This condition might be one of the main differences between NIs and CG images. Our observation is completely in line with the hypothesis and observation of Dang-Nguyen *et al.* [17], who assumed that synthetic facial animations present a less complex pattern, whereas natural ones have a much more complicated variability. Our observations of static NIs and CG images are similar to those of [17] that examined the difference between natural and CG facial videos. This similarity may imply that CNNs can, to some extent, unconsciously follow a similar idea of the hard intelligent work of researchers when facing similar problems. In the future, we plan to design CNNs for the discrimination of natural and CG videos, and we expect to gain similar understanding and observation as those reported in [17].

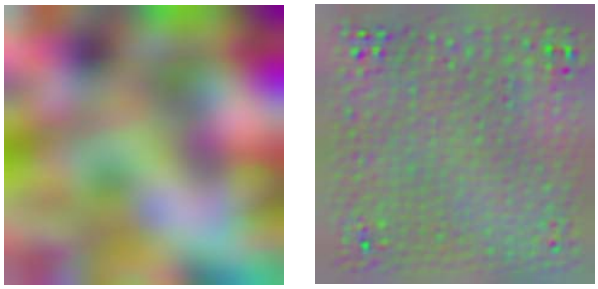


Fig. 11. Visualization of preferred inputs in image space for two output units. The left corresponds to CG image, and the right corresponds to NI.

VII. Concluding Remarks

In this work, we proposed a generic framework based on the convolutional neural network to identify and understand the difference between natural and computer-generated images. The performance of our network is better than that of conventional methods and a very recent parallel CNN-based method not only in the highly challenging Google versus PRCG dataset, but also in relatively simple datasets. Our method also outperforms state-of-the-art methods in terms of performance in small image patches and robustness against typical post-processing operations. These factors are important for a forensic method to be useful in real-world applications.

We attempted to conduct an extensive study on using CNN for distinguishing between NIs and CG images. We considered the fine-tuning, structure, energy function design, flexibility, visualization, and understanding of CNN. To our knowledge, fine-tuning of pre-trained CNN from computer vision tasks and the visualization and understanding of what a CNN has learned are new in image forensics and might be useful and inspiring for other multimedia security tasks. Our source code is available at https://github.com/weizequan/NIvsCG.

In the future, we would like to continue our work on using CNNs for image forensics. First, we will try to improve the performance of our method by using for instance an ensemble of CNNs. It would also be interesting to introduce high-level semantic information to assist in the classification task of NIs and CG images, evaluate and improve (if necessary) the forensic performance against highly malicious attacks (*e.g.*, identifying recaptured NIs and CG images), and extend our method to the classification of natural and CG videos. Furthermore, an exciting research problem is the architectural design and efficient training of CNNs, which are particularly tailored for either universal or targeted image forensics.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers for their helpful comments and suggestions. They would like to thank Dr. T.-T. Ng for the Columbia dataset, shared code of [18] and discussions about Google set, Dr. T. Pevny and Dr. F. Peng for kindly sharing the source code of their algorithms respectively described in [31] and [27], Dr. J.A.K. Suykens for making the LS-SVM toolbox [57] publicly available, N. Rahmouni for the dataset, shared code of [45] and advices for the usage of code, and Dr. S. Lapuschkin for detailed and helpful discussions about the LRP toolbox [64].

REFERENCES

[1] T.-T. Ng, S.-F. Chang, J. Hsu, and M. Pepeljugoski, "Columbia photographic images and photorealistic computer graphics dataset," Columbia Univ., New York, NY, USA, Tech. Rep. #205-2004-5, 2004.

[2] J. Chen, X. Kang, Y. Liu, and Z. J. Wang, "Median filtering forensics based on convolutional neural networks," *IEEE Signal Process. Lett.*, vol. 22, no. 11, pp. 1849–1853, Nov. 2015.

[3] B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, 2016, pp. 5–10.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[6] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," *Proc. SPIE*, vol. 9409, p. 94090J, Mar. 2015.

[7] L. Pibre, J. Pasquet, D. Ienco, and M. Chaumont, "Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch," in *Proc. IS T Electron. Imag.*, 2016, pp. 1–23.

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

[9] H. Farid and M. J. Bravo, "Photorealistic rendering: How realistic is it?" *J. Vis.*, vol. 7, no. 9, p. 766, 2007.

[10] H. Farid and M. J. Bravo, "Perceptual discrimination of computer generated and photographic faces," *Digit. Invest.*, vol. 8, nos. 3–4, pp. 226–235, 2012.

[11] S. Fan, T.-T. Ng, J. S. Herberg, B. L. Koenig, and S. Xin, "*Real or Fake?*: Human judgments about photographs and computer-generated images of faces," in *Proc. SIGGRAPH Asia Tech. Briefs*, 2012, Art. no. 17.

[12] O. Holmes, M. S. Banks, and H. Farid, "Assessing and improving the identification of computer-generated portraits," *ACM Trans. Appl. Perception*, vol. 13, no. 2, 2016, Art. no. 7.

[13] B. Mader, M. S. Banks, and H. Farid, "Identifying computer-generated portraits: The importance of training and incentives," *Perception*, vol. 46, no. 9, pp. 1062–1076, 2017.

[14] D.-T. Dang-Nguyen, G. Boato, and F. G. B. De Natale, "Identify computer generated characters by analysing facial expressions variation," in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, Dec. 2012, pp. 252–257.

[15] V. Conotter, E. Bodnari, G. Boato, and H. Farid, "Physiologically-based detection of computer generated faces in video," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2014, pp. 248–252.

[16] D.-T. Dang-Nguyen, G. Boato, and F. G. B. De Natale, "Discrimination between computer generated and natural human faces based on asymmetry information," in *Proc. 20th Eur. Signal Process. Conf.*, Aug. 2012, pp. 1234–1238.

[17] D.-T. Dang-Nguyen, G. Boato, and F. G. B. De Natale, "3D-model-based video analysis for computer generated faces identification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 8, pp. 1752–1763, Aug. 2015.

[18] T.-T. Ng, S.-F. Chang, J. Hsu, L. Xie, and M.-P. Tsui, "Physics-motivated features for distinguishing photographic images and computer graphics," in *Proc. 13th Annu. ACM Int. Conf. Multimedia*, 2005, pp. 239–248.

[19] S. Lyu and H. Farid, "How realistic is photorealistic?" *IEEE Trans. Signal Process.*, vol. 53, no. 2, pp. 845–850, Feb. 2005.

[20] W. Chen, Y. Q. Shi, and G. Xuan, "Identifying computer graphics using HSV color model and statistical moments of characteristic functions," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2007, pp. 1123–1126.

[21] A. C. Gallagher and T. Chen, "Image authentication by detecting traces of demosaicing," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[22] T.-T. Ng and S.-F. Chang, "Discrimination of computer synthesized or recaptured images from real images," in *Digital Image Forensics*, H. T. Sencar and N. Memon, Eds. New York, NY, USA: Springer, 2013, pp. 275–309.

[23] G. Sankar, V. Zhao, and Y.-H. Yang, "Feature based classification of computer graphics and real images," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Apr. 2009, pp. 1513–1516.

[24] A. C. Popescu and H. Farid, "Exposing digital forgeries by detecting traces of resampling," *IEEE Trans. Signal Process.*, vol. 53, no. 2, pp. 758–767, Feb. 2005.

[25] T. I. Ianeva, A. P. de Vries, and H. Rohrig, "Detecting cartoons: A case study in automatic video-genre classification," in *Proc. IEEE Int. Conf. Multimedia Expo*, vol. 1. Jul. 2003, pp. I-449–I-452.

[26] R. Zhang, R.-D. Wang, and T.-T. Ng, "Distinguishing photographic images and photorealistic computer graphics using visual vocabulary on local image edges," in *Proc. Int. Workshop Digit. Watermarking*, 2012, pp. 292–305.

[27] F. Peng, D.-L. Zhou, M. Long, and X.-M. Sun, "Discrimination of natural images and computer generated graphics based on multi-fractal and regression analysis," *AEU-Int. J. Electron. Commun.*, vol. 71, pp. 72–81, Jan. 2017.

[28] A. Rocha and S. Goldenstein, "Is it fake or real?" in *Proc. Brazilian Symp. Comput. Graph. Image Process.*, 2006, pp. 1–2.

[29] P. Sutthiwan, X. Cai, Y. Q. Shi, and H. Zhang, "Computer graphics classification based on Markov process model and boosting feature selection technique," in *Proc. 16th IEEE Int. Conf. Image Process.*, Nov. 2009, pp. 2913–2916.

[30] P. Sutthiwan, J. Ye, and Y. Q. Shi, "An enhanced statistical approach to identifying photorealistic images," in *Proc. Int. Workshop Digit. Watermarking*, 2009, pp. 323–335.

[31] T. Pevný, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 2, pp. 215–224, Jun. 2010.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[34] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.

[35] Y. Jia. (2013). *Caffe: An Open Source Convolutional Architecture for Fast Feature Embedding*. Accessed: Jan. 15, 2018. [Online]. Available: http://caffe.berkeleyvision.org/

[36] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning," in *Proc. ICML Workshop Unsupervised Transf. Learn.*, 2011, pp. 17–36.

[37] J. Donahue *et al.*, "DeCAF: A deep convolutional activation feature for generic visual recognition," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1–9.

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

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

[40] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification with the use of deep convolutional neural networks," in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, Dec. 2016, pp. 1–6.

[41] L. Bondi, L. Baroffio, D. Güera, P. Bestagini, E. J. Delp, and S. Tubaro, "First steps toward camera model identification with convolutional neural networks," *IEEE Signal Process. Lett.*, vol. 24, no. 3, pp. 259–263, Mar. 2017.

[42] L. Bondi, S. Lameri, D. Güera, P. Bestagini, E. J. Delp, and S. Tubaro, "Tampering detection and localization through clustering of camera-based CNN features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jul. 2017, pp. 1855–1864.

[43] B. Bayar and M. C. Stamm, "A generic approach towards image manipulation parameter estimation using convolutional neural networks," in *Proc. 5th ACM Workshop Inf. Hiding Multimedia Secur.*, 2017, pp. 147–157.

[44] J. Yang, K. Liu, X. Kang, E. Wong, and Y. Shi, "Steganalysis based on awareness of selection-channel and deep learning," in *Proc. Int. Workshop Digit. Watermarking*, 2017, pp. 263–272.

[45] N. Rahmouni, V. Nozick, J. Yamagishi, and I. Echizen, "Distinguishing computer graphics from natural images using convolution neural networks," in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, Dec. 2017, pp. 1–6.

[46] N. Khanna, G. T.-C. Chiu, J. P. Allebach, and E. J. Delp, "Forensic techniques for classifying scanner, computer generated and digital camera images," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar./Apr. 2008, pp. 1653–1656.

[47] Z. Li, J. Ye, and Y. Q. Shi, "Distinguishing computer graphics from photographic images using local binary patterns," in *Proc. Int. Workshop Digit. Forensics Watermarking*, 2013, pp. 228–241.

[48] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–14.

[49] W. Quan, D.-M. Yan, J. Guo, W. Meng, and X. Zhang, "Maximal poisson-disk sampling via sampling radius optimization," in *Proc. SIGGRAPH ASIA Posters*, 2016, Art. no. 22.

[50] R. H. R. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, no. 6789, pp. 947–951, Jun. 2000.

[51] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 1–8.

[52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[54] J. Fridrich and J. Kodovský, "Rich models for steganalysis of digital images," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 868–882, Jun. 2012.

[55] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 1–9.

[56] 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, 2014.

[57] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. Singapore: World Scientific, 2002.

[58] M. Piaskiewicz. (2017). *Level-Disign Reference Database*. Accessed: Jan. 15, 2018. [Online]. Available: http://level-design.org/referencedb/

[59] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, "RAISE: A raw images dataset for digital image forensics," in *Proc. 6th ACM Multimedia Syst. Conf.*, 2015, pp. 219–224.

[60] N. Rahmouni. (2017). *CGvsPhoto*. Accessed: Jan. 15, 2018. [Online]. Available: https://github.com/NicoRahm/CGvsPhoto/

[61] F.-F. Li, J. Johnson, and S. Yeung. (2017). Visualizing what convnets learn. Stanford University. Accessed: Jan. 15, 2018. [Online]. Available: http://cs231n.github.io/understanding-cnn/

[62] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Müller, and W. Samek, "The LRP toolbox for artificial neural networks," *J. Mach. Learn. Res.*, vol. 17, pp. 1–5, Jun. 2016.

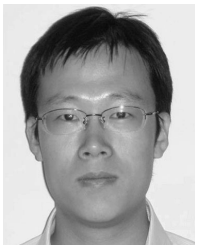
[63] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," in *Proc. ICML Deep Learn. Workshop*, 2015, pp. 1–12.

[64] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, no. 7, p. e0130140, 2015.

[65] C. Barnes and F.-L. Zhang, "A survey of the state-of-the-art in patch-based synthesis," *Comput. Vis. Media*, vol. 3, no. 1, pp. 3–20, Mar. 2017.

Weize Quan received the B.S. degree from the Wuhan University of Technology in 2014. He is currently pursuing the Ph.D. degree with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. His research interests include geometry processing and image forensics.

Kai Wang received the Ph.D. degree in computer science from the University of Lyon, Lyon, France, in 2009. Since 2011, he has been a full-time CNRS Researcher with GIPSA-lab, Grenoble, France. His current research interests include multimedia security and surface analysis.



Dong-Ming Yan received the bachelor's and master's degrees from Tsinghua University, in 2002 and 2005, respectively, and the Ph.D. degree from Hong Kong University in 2010. He is currently an Associate Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. His research interests include computer graphics, geometric processing, and visualization.

Xiaopeng Zhang received the Ph.D. degree in computer science from the Institute of Software, Chinese Academy of Sciences (CAS) in 1999. He is currently a Professor with the National Laboratory of Pattern Recognition, Institute of Automation, CAS. His main research interests include computer graphics and image processing. He received the National Scientific and Technological Progress Prize (second class) in 2004.