Signal Processing: Image Communication 67 (2018) 90–99

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

A deep learning approach to patch-based image inpainting forensics

Xinshan Zhu [a,b], Yongjun Qian [a], Xianfeng Zhao [b], Biao Sun [a,*], Ya Sun [a]

[a] School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China
[b] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

ARTICLE INFO

Keywords:
Inpainting
Forensics
Convolutional neural network
Loss function
Compression

ABSTRACT

Although image inpainting is now an effective image editing technique, limited work has been done for inpainting forensics. The main drawbacks of the conventional inpainting forensics methods lie in the difficulties on inpainting feature extraction and the very high computational cost. In this paper, we propose a novel approach based on a convolutional neural network (CNN) to detect patch-based inpainting operation. Specifically, the CNN is built following the encoder–decoder network structure, which allows us to predict the inpainting probability for each pixel in an image. To guide the CNN to automatically learn the inpainting features, a label matrix is generated for the CNN training by assigning a class label for each pixel of an image, and the designed weighted cross-entropy serves as the loss function. They further help to strongly supervise the CNN to capture the manipulation information rather than the image content features. By the established CNN, inpainting forensics does not need to consider feature extraction and classifier design, and use any postprocessing as in conventional forensics methods. They are combined into the unique framework and optimized simultaneously. Experimental results show that the proposed method achieves superior performance in terms of true positive rate, false positive rate and the running time, as compared with state-of-the-art methods for inpainting forensics, and is very robust against JPEG compression and scaling manipulations.

1. Introduction

With the rapid development of computer and Internet technologies, digital images have been widely used in many aspects of social life, such as news, advertisement, publications, etc. However, due to the availability of powerful image processing software, the images can be accessed, copied and edited more easily than ever before. The reliability and authenticity of image information have been a main concern in the digital multimedia era. As such a promising technique, digital image forensics has attracted considerable attention and seen numerous applications recently. The basic idea is to extract the unique artifacts introduced by an image operation to decide whether an image once underwent such an operation. A great effort has been spent by researchers in developing various forensics techniques on JPEG compression [1,2], histogram equalization [3], median filtering [4], resizing [5], copy–move forgery [6,7] and so on [8].

As a new and effective image editing technique, image inpainting aims to effectively repair damaged or removed image regions in a visually plausible manner using the known image information. It is an important topic in the field of computer vision and image processing, and has made great progress in the past few years. Applications of image inpainting include image restoration (e.g., scratch or text removal), image coding and transmission (recovery of missing blocks), photo-editing (object removal), virtual restoration of digitized paintings (crack removal), etc. [9].

However, inpainting could be also exploited to modify image content with malicious motives, which brings the crisis of confidence in image content. Consequently, the inpainting forensics, i.e., the detection of inpainting operation, is proposed to deal with this problem. In a practical scenario for inpainting forensics, a host image is first altered by a forger with the inpainting operation. For instance, in Fig. 1, an image is modified by removing an object in it using inpainting operation. Then, the inpainted image could undergo other image processing operations before forensics, e.g., JPEG compression, which inevitably cause the inpainting information loss. Last, an investigator receives the possibly distorted inpainted image, and detects the existence of inpainting by forensic techniques. This process is depicted in Fig. 2. Generally, there are two major forensic tasks to be considered: is the input image inpainted by a forger, and if yes, where is the inpainted region? By the inpainting region information, the investigator knows exactly what happened, e.g., scratch, text and object removal, etc., and sometime even infers the missing image information, like text and object shape.

* Corresponding author.
E-mail addresses: xszhu@tju.edu.cn (X. Zhu), sunbiao@tju.edu.cn (B. Sun).

https://doi.org/10.1016/j.image.2018.05.015
Received 6 November 2017; Received in revised form 11 April 2018; Accepted 28 May 2018
Available online 13 June 2018

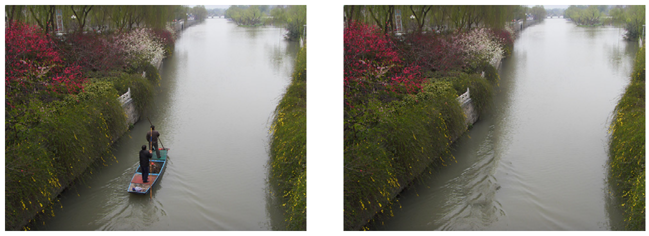

Fig. 1. An illustrative example for object removal by image inpainting: original image (left) and inpainted image (right).

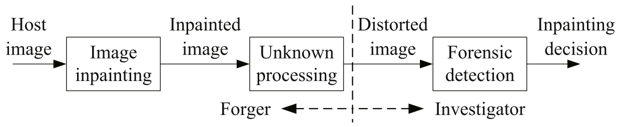

Fig. 2. Basic process that an image may go through in a practical scenario for inpainting forensics.

To accomplish the forensic tasks, the image pixels need to be divided into two classes: inpainting and uninpainting. Usually, the classification problem is considered as consisting of two parts: feature extraction and actual classification. Since no obvious trace is left by inpainting operation, the features with high discrimination are hardly obtained. Conventional inpainting forensics methods mainly depend on similarity features among image patches to differ inpainted patches from uninpainted ones [10–14]. They has some common drawbacks, such as high computational cost for feature extraction, high false alarm performance in uniform image areas, etc. In this paper, we propose to detect the inpainting manipulation and identify the inpainting localization by employing a convolutional neural network (CNN). To the best of our knowledge, this is the first attempt to investigate CNNs in forensically determining the presence of image inpainting operation.

The main contribution of this paper is as follows. First, CNNs tend to learn features to represent the image content rather than the manipulation information. To solve this problem, we construct a class label matrix for an image instead of a single label to provide more supervisory information for the training process. Second, the inpainted pixels are usually much less than the uninpainted ones, causing the imbalance between the positive and negative sample data. For this problem, we design the weighted cross-entropy as the loss function for the training. Last, the CNN is built following the encoder–decoder network, which allows to predict the inpainting probability for each pixel in an image. Thanks to the perfect feature representation ability, the proposed method not only achieves the superior forensics performance, but also reduces the computation costs largely.

The remainder of this paper is structured as follows. Section 2 briefly introduces the related works on inpainting forensics. Next, in Section 3, the problem model for inpainting forensics is developed, and the basic patch-based inpainting process and a conventional inpainting forensics framework are also formulated. Section 4 presents the details of the CNN architecture for inpainting forensics with strengthening supervisory strategies considered. A series of tests are done to evaluate the presented method in Section 5. Finally, Section 6 concludes this paper.

2. Related works

There has been limited research on inpainting forensics until now. Early, the inpainted region was recognized by a fuzzy membership function of patch similarity measured by zero-connectivity length (ZCL) [10]. This method is developed based on the fact that the inpainted patches are generated with some pixels in their reference patches, causing zero connectivity paths in their difference arrays. The drawback is that the suspicious regions require to be manually selected in advance. A similar forensics method was presented in [11] for video inpainting. Then, the skipping patch matching was proposed by Bacchuwar et al. [12], which reduces the amount of the computational cost for forensics. This method suffers from the same drawback, and gets a high false-alarm rate in uniform areas of an image, such as sky and grass.

To overcome the above drawbacks, in [13], a two-stage searching method based on weight transformation was proposed to accelerate the search of suspicious patches, and the multi-region relations were utilized to filter the false-alarm patches. Although the method achieves the improved false-alarm performance and the computational efficiency, the detection accuracy is limited due to the approximately similar patch search. The work was further improved in [14] by exploiting the central pixel mapping method for the patch searching and fragment splicing detection for the removal of false-alarm patches. However, the method is very sensitive to the common image processing operations.

Besides, a few research has been done for the detection of the inpainting forgery in the presence of other image processing manipulations. The inpainting operation for a compressed inpainted image was recognized by computing and segmenting the averaged sum of absolute difference images between the target image and a resaved JPEG compressed image at different quality factors [15]. However, this is only valid for the forensics with JPEG compression. In [16], the marginal density and neighboring joint density features were constructed to deal with the forensics with more combinations of inpainting, compression, filtering and resampling considered. This scheme suffers from the drawback similar to [15].

Recently, as a common deep learning network, convolutional neural network (CNN) has made great success in many fields, such as image segmentation, image fusion, image classification, target recognition, location and so on [17]. This encourages researchers to develop the CNN-based forensics methods to improve the forensics performance. The related works mainly include median filtering detection [18], camera model identification [19,20], copy–move forensics [21], JPEG compression forensics [22–24], universal image manipulation detection [25], counter-anti-forensics [26], recapture image forensics [27], as well as image steganalysis [28,29]. They demonstrated the significant performance advantage of CNNs on learning the characteristics of image manipulations and optimizing the classification rules. Inspired by these works, we consider to develop an effective inpainting forensics method by CNNs with deep model and avoid the difficulties that conventional inpainting forensics methods face.

3. Problem model

To better understand our work, image inpainting technology is first briefly introduced in this section. On the basis, the forensic problems for inpainting operation are carefully described, and a general process is also summarized for the conventional inpainting forensics schemes.

3.1. Patch-based inpainting

Image inpainting is a technique attempting to effectively repair damaged or removed image regions in a visually plausible manner. Existing approaches can be roughly divided into two main categories: the diffusion-based and the patch-based approaches. The former diffuses image information from known regions into unknown regions by solving partial differential equations or similar diffusion systems [30–32]. They manifest good performance when inpainting long thin regions, but experience difficulties in replicating texture and recovering large missing regions [9]. The patch-based approaches inwardly propagate the image patches into the unknown region patch by patch from the known regions. They typically produce better inpainting results, especially when inpainting large unknown regions. Therefore, our attention is concentrated on patch-based image inpainting forensics.

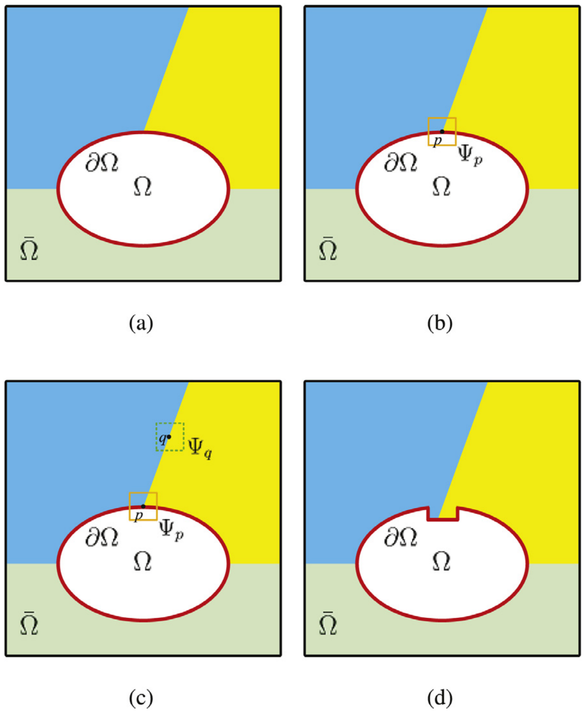

Fig. 3. Patch-based image inpainting: (a) original image with a missing region Ω, (b) selecting a target patch Ψ_p with the largest priority, (c) searching the best-matching patch Ψ_q for Ψ_q in the known region $\bar{\Omega}$, (d) inpainting the patch Ψ_q with Ψ_q and updating the interior $\partial\Omega$.

Taking Criminisi's method [33] as an example, the basic process for patch-based inpainting is described as follows. Given an image with an unknown region Ω, shown in Fig. 3(a), the inpainting process is briefly explained as follows.

Step 1: By computing the priority for each point on the interior $\partial\Omega$ of the unknown region Ω, we determine the position for the point with the largest priority denoted by p. The image patch Ψ_p centered at p is chosen as the current patch to be inpainted, as shown in Fig. 3(b).

Step 2: According to a certain patch matching rule, we search the whole known region $\bar{\Omega}$ and find out the most similar patch Ψ_q of Ψ_p, as shown in Fig. 3(c).

Step 3: The missing region in Ψ_p is filled by using the corresponding pixels in Ψ_q, and the interior $\partial\Omega$, the known region $\bar{\Omega}$ and the unknown region Ω are all updated in Fig. 3(d).

Step 4: By performing Step 1 to Step 3 iteratively, the unknown region can be entirely inpainted, producing the inpainted image.

The structure and texture information are taken into account in the computation of patch priority and the design of the patch matching rule. This brings about the performance advantage for Criminisi's method. Many later inpainting methods [34,35] are developed based on the main idea of Criminisi's method. With advances of inpainting technology, the inpainting forensics has become increasingly important in the authentication of multimedia contents.

3.2. Forensic problems

In this study, we are only interested in the blind inpainting forensics problems. That is, the forensics is performed on a given image in the case that we do not have the original image and any information of the inpainted region, e.g., location, size and shape, etc., and we do not know any specific inpainting parameter values about Criminisi's method. In other words, the decision about the presence of inpainting and the inpainted region is made only according to the given image itself.

To describe the forensics problem, we use a matrix $\boldsymbol{X} \in \mathbb{R}^{W\times H}$ to denote a given image. For the forensic tasks, we need to divide the pixels of $\boldsymbol{X}$ into two classes: inpainting and uninpainting. Thus, we take the matrix $\boldsymbol{Y}$ of the same size as $\boldsymbol{X}$ to represent the probabilities that all pixels of $\boldsymbol{X}$ are inpainted, where each element Y_{ij} of $\boldsymbol{Y}$ takes a value within $[0,1]$. Correspondingly, $1-\boldsymbol{Y}$ is the matrix of probabilities that all pixels of $\boldsymbol{X}$ are uninpainted. Considering that the inpainting forensics is blind, the inpainting probability matrix (IPM) $\boldsymbol{Y}$ only depends on the input image $\boldsymbol{X}$, and we can write

$$\boldsymbol{Y} = \mathcal{F}(\boldsymbol{X}), \tag{1}$$

where the function $\mathcal{F}(\cdot)$ is unknown and will be learned from the training data.

By the use of IPM $\boldsymbol{Y}$, we may separate the input image $\boldsymbol{X}$ into two regions: inpainted and uninpainted regions, denoted by the sets Ω^F and $\bar{\Omega}^F$ respectively. They can be expressed as $\Omega^F = \{(i,j) \mid Y_{ij} \geq 0.5, 1 \leq i \leq H, 1 \leq j \leq W, i,j \in \mathbb{Z}\}$ and $\bar{\Omega}^F = C - \Omega^F$, where (i,j) stands for the coordinate of a pixel in the image $\boldsymbol{X}$ and the set C contains the coordinates of all pixels in $\boldsymbol{X}$. Further, we may make a decision whether or not the whole image $\boldsymbol{X}$ is inpainted according to the cardinality of Ω^F, i.e., the size of the recognized inpainted region, denoted by $|\Omega^F|$.

The true inpainted and uninpainted regions are respectively represented by Ω and $\bar{\Omega}$ as in the inpainting process, which may be different from the forensics results. Using the above notations, the performance of an inpainting forensics method can be assessed by true positive rate (TPR) and false positive rate (FPR), which are defined as $\mathrm{TPR} = |\Omega^F \cap \Omega|/|\Omega|$ and $\mathrm{FPR} = (|\Omega^F \cap \bar{\Omega}|)/|\bar{\Omega}|$, respectively. In addition, for the classification problem, the average precision (AP) is often used for the performance evaluation, which is defined as $\mathrm{AP} = (|\Omega^F \cap \Omega| + |\bar{\Omega}^F \cap \bar{\Omega}|)/|C|$. Actually, the three metrics have the relation $\mathrm{AP} = |\Omega|/|C|\mathrm{TPR} + \bar{\Omega}/|C|(1-\mathrm{FPR})$.

3.3. Conventional inpainting forensics

To our knowledge, only a few works [10,13,14,16] have been proposed for the inpainting forensics problem, and the main idea and basic framework are similar. In principle, the conventional inpainting forensics process may be decomposed into four main steps:

Step 1: For each patch of the input image, the best-matching patch is found out by searching the whole image according to a certain patch matching rule, e.g., Euclidean distance.

Step 2: Between the patch pair, an inpainting feature is extracted to reflect the inpainting information, e.g., ZCL.

Step 3: By applying a classification rule to the extracted feature, the pixels of the investigated patch are categorized into the inpainting set and the uninpainted set, e.g., a fuzzy membership function.

Step 4: To remove false-alarm pixels in the inpainting set, a post-processing operation combining the observed major propositions for false-alarm regions is performed after the classes of all image pixels are obtained.

The conventional methods were developed based on the fact that the generated inpainted patches are very similar to some those reference ones. They have several obvious drawbacks. First, the patch searching process is very time-consuming especially for a large image, because the similarity between any two patches of the whole image needs to be measured. Second, the inpainting features only depend on the patch similarity, however, there exist some very similar patches in a nature image like blue sky. Thus, it is difficult to identify the class labels for the pixels in such a kind of patches by the similarity feature. Third, the adopted decision rules are built neither by using the stochastic models of the features nor by supervised learning technologies. So the classification performance is undesirable. Last, the post-processing operation does not work steadily. The reason is that the operation rules are obtained by observing the propositions for false-alarm regions, however, the situations in practice might be very rich and diverse.

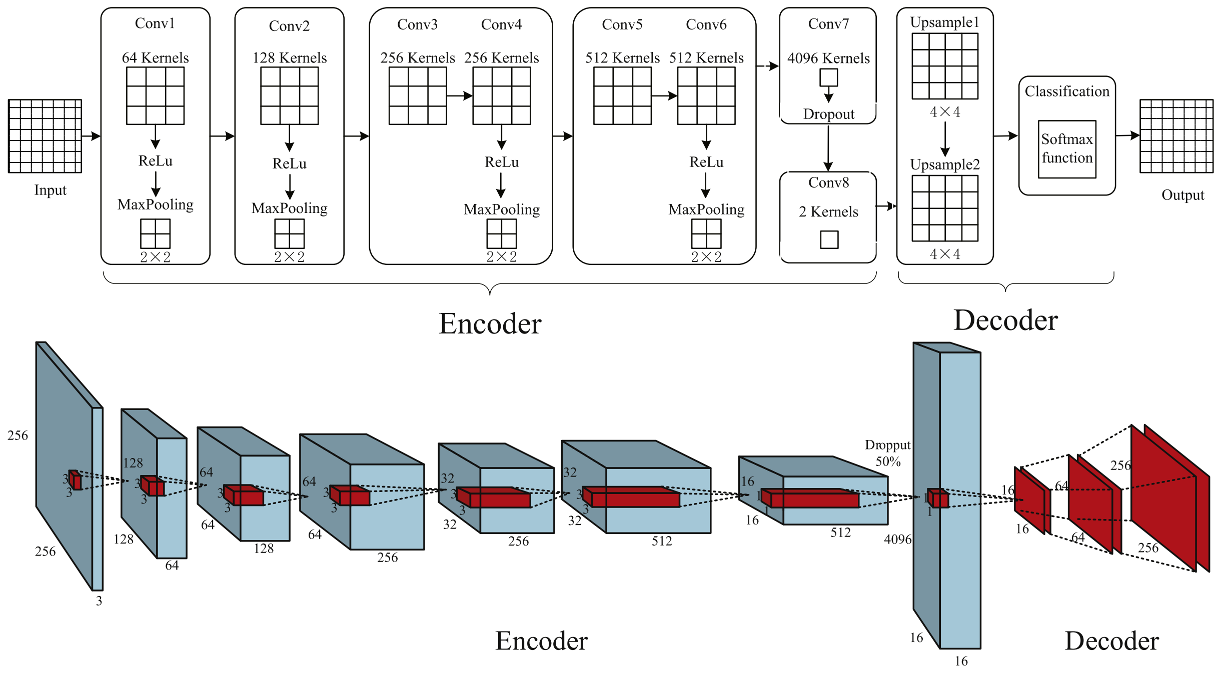

Fig. 4. The layout, architecture and parameter settings of the CNN for inpainting forensics.

As a supervised learning method, CNN manifests the excellent performance while solving the classification related problems. Although the training of CNN costs much time due to the training database size, it generally runs very fast on the test data. More importantly, CNN extracts features automatically and learning the hierarchical representations through multiple layers of nonlinear processing. In this way, a CNN-based detector for inpainting could automatically combine feature extraction, classification steps and the post-processing operation in the unique framework, and this “fully automatic” tool would be optimized simultaneously. These considerations motivate us to design the CNN-based inpainting forensics scheme.

4. CNN-based forensics for inpainting

In this section, image inpainting is detected by a deep learning approach. We present strongly supervised strategies for CNN to learn the inpainting features, and all details of our best CNN architecture for the forensics purpose.

4.1. *Feature learning with more supervisory information*

The general architecture of a CNN is made up of layers composed of neurons, where a neuron is a computing unit which evaluates a set of input values, coming from the neurons of the previous layer or input units, and gives a set of output values to neurons of the next layer or output units. The performance of a CNN depends on the parameters of all neurons, which are obtained by automatically learning from a lot of sample data. To learn inpainting feature effectively, we build the CNN by mainly considering three aspects: sample data, network architecture, and loss function.

4.1.1. *Sample data*

A straightforward way to construct a dataset for training is to collect a number of samples of $(\boldsymbol{X}, z)$, where the image $\boldsymbol{X}$ serves as the input of a CNN, and $z \in \{0,1\}$ is a scalar label to indicate whether or not $\boldsymbol{X}$ is inpainted. However, the CNN trained on the dataset in our preliminary study did not perform very well, suggesting the important inpainting features can be hardly captured. This is because inpainting causes the manipulation feature that each inpainted patch has relatively high similarity with its reference one while CNNs tend to learn features that represent an image’s content. The similar effects have been observed when training CNNs to detect median filtering, Gaussian blurring, etc. in [18,25]. To deal with the problem, they proposed to add a new filter layer before the input layer of a CNN to preprocess the input images so that the CNNs are made to learn local structural relationships between pixels rather than the content features.

Unfortunately, the above strategy is not suitable for inpainting forensics. This is mainly because image inpainting is a global operation highly depending on image content, and in an inpainted image the distance between an inpainted patch and the reference one might be very different for different inpainted patches. As a result, the trace left by inpainting cannot be successfully exposed by local filtering processes as done in [18,25].

Instead, we form a label matrix $\boldsymbol{Z}$ for the input image $\boldsymbol{X}$ by assigning a class label to each pixel of $\boldsymbol{X}$ as

$$Z_{ij} = \begin{cases} 1, & (i,j) \in \Omega \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Then, we use the dataset of samples of $(\boldsymbol{X}, \boldsymbol{Z})$ to train a CNN, and the CNN outputs the prediction $\hat{\boldsymbol{Z}}$ of the label matrix $\boldsymbol{Z}$. As a result, the CNN obtains more supervisory information, and is guided to learn the manipulation features of inpainting rather than content features.

4.1.2. *Framework overview*

Since our CNN is used to predict the label matrix $\boldsymbol{Z}$, we build it following the encoder–decoder network structure [36,37]. The encoder consists of convolutional layers, pooling layers, shown in Fig. 4.

In a convolutional layer, the convolution and activation operations are performed by neurons. Let $\boldsymbol{X}_j^{(l)}$ represent the output of the channel j in the layer l, $\boldsymbol{W}_{ij}^{(l)}$ refer to the ith channel of the jth convolutional filter in the layer l, and $\boldsymbol{B}_j^{(l)}$ be the corresponding training bias term in the layer l. The mathematical operations in the layer l are formulated as

$$\boldsymbol{X}_j^{(l)} = \eta\left(\sum_{i=1}^{c} \boldsymbol{W}_{ij}^{(l)} * \boldsymbol{X}_i^{(l-1)} + \boldsymbol{B}_j^{(l)}\right), \tag{3}$$

where $*$ is the convolution operator, $\eta(\cdot)$ denotes the activation function and c stands for the channel number.

The output of a convolution layer can be considered as a particular feature representation of the image $\boldsymbol{X}$, and is thus also called a feature map. The feature map $\boldsymbol{X}_j^{(l)}$ is completely characterized by the filter parameters $\boldsymbol{W}_{ij}^{(l)}$ and $\boldsymbol{B}_j^{(l)}$, which are obtained by learning from training data. In practice, a feature map is made to be connected to only a subset of the previous feature maps, called receptive field, and each neuron in a filter in the same layer has the same convolution filter parameters, called the share of weights. They reduce the number of the training parameters, hence improves the generalization performance of a CNN [38].

The activation function $\eta(\cdot)$ is an element-wise non-linear operation. It has several alternative forms, such as, sigmoid, tanh, etc. In this study, the rectified linear units (ReLUs) is used, which is expressed as

$$\eta(x) = \max(0; x), \tag{4}$$

because it can lead to fast convergence in training large models on large datasets [39].

A pooling layer is usually inserted between two successive convolutional layers. Through this layer, the feature map of the previous convolutional layer is divided into small windows, and a single statistical feature value (e.g., mean value or max value) is retained in per window. The max-pooling is adopted in the constructed CNN. The pooling operation reduces the spatial size of a feature map and the amount of parameters to be trained. This causes the decrease of the training time and the chances of over-fitting.

The decoder contains upsampling layers and the classification layer. An upsampling layer is used to acquire the high-resolution results by upsampling the low-resolution feature maps. The upsampling operation is performed by the bilinear interpolation in our upsampling layers. After the upsampling layers, we obtain the output $\boldsymbol{X}^{(L-1)}$ with L being the number of network layers in the CNN. According to the prediction target $\boldsymbol{Z}$, $\boldsymbol{X}^{(L-1)}$ is required to be of the same size as the input image $\boldsymbol{X}$ and have two channels.

Finally, in the classification layer, the estimation $\hat{\boldsymbol{Y}}$ of IPM $\boldsymbol{Y}$ is calculated by applying Softmax function to $\boldsymbol{X}^{(L-1)}$, i.e. ,

$$\hat{Y}_{ij} = \frac{e^{X_{1ij}^{(L-1)}}}{e^{X_{1ij}^{(L-1)}} + e^{X_{2ij}^{(L-1)}}}, \tag{5}$$

where X_{kij} denotes the element at the coordinate (i, j) in the kth channel of $\boldsymbol{X}^{(L-1)}$. Then, the estimated label matrix $\hat{\boldsymbol{Z}}$ is output according to the rule that the element Z_{ij} equals 1 if $\hat{Y}_{ij} \geq 0.5$ and 0 otherwise.

4.1.3. Loss function

We actually want to approximate the mapping function $\mathcal{F}(\cdot)$ between the input image $\boldsymbol{X}$ and IPM $\boldsymbol{Y}$ in Eq. (1) by our CNN. A loss function is needed to measure the performance of the CNN based on how well the predictions given by the CNN agree with the ground truth labels in the training data.

A commonly used loss is called the cross-entropy loss. Using the previous notations, the loss function for the kth training sample data $(\boldsymbol{X}_k, \boldsymbol{Z}_k)$ is expressed as

$$E_k = \frac{-1}{W \cdot H} \sum_{j=1}^{W} \sum_{i=1}^{H} \log \left(\hat{Y}_{k,ij}^{Z_{k,ij}} \left(1 - \hat{Y}_{k,ij}\right)^{1 - Z_{k,ij}} \right), \tag{6}$$

where $\hat{Y}_{k,ij}$ denotes the (i, j)th element of the IPM estimation $\hat{\boldsymbol{Y}}_k$ for the kth training image $\boldsymbol{X}_k$, $Z_{k,ij}$ is the corresponding element of the label matrix $\boldsymbol{Z}_k$ for $\boldsymbol{X}_k$, and E_k denotes the cross-entropy between $\boldsymbol{Z}_k$ and $\hat{\boldsymbol{Y}}_k$. In (6), the assumption $\boldsymbol{Y}_k = \boldsymbol{Z}_k$ is applied, where $\boldsymbol{Y}_k$ denotes IPM for the image $\boldsymbol{X}_k$.

The expression (6) implies that the number of the inpainted pixels should be (or approximately) equal to that of the uninpainted ones in an image. However, in practice, the inpainted region is of random size since it is produced by a forger depending on the forgery purpose and the image content. In addition, the inpainted region is usually smaller than the known region, otherwise, the inpainting performance degrades largely because less information from the uninpainted region can be used for the generation of inpainted regions. Due to the imbalance between the positive and negative sample data, the use of (6) for training our CNN will inevitably cause the decrease of TPR.

To overcome the problem, we calculate the weighted cross-entropy loss as

$$E_k = \frac{-1}{W \cdot H} \sum_{j=1}^{W} \sum_{i=1}^{H} \log \left(\hat{Y}_{k,ij}^{\alpha_{k,ij} Z_{k,ij}} \left(1 - \hat{Y}_{k,ij}\right)^{\beta_{k,ij}(1 - Z_{k,ij})} \right), \tag{7}$$

where $\alpha_{k,ij}$ and $\beta_{k,ij}$ are the weights, satisfying $\alpha_{k,ij} \geq 0$, $\beta_{k,ij} \geq 0$ and $\alpha_{k,ij} + \beta_{k,ij} = 1$. Notice that the traditional cross-entropy loss (6) is just a special case of (7) when $\alpha_{k,ij} = Z_{k,ij}$. Obviously, by adjusting $\alpha_{k,ij}$ and $\beta_{k,ij}$, we can change the importance of the loss from miss detection or false alarm (corresponding to $Z_{k,ij} = 1$ and $Z_{k,ij} = 0$ respectively). For our CNN, we set $\alpha_{k,ij} = |\bar{\Omega}_k|/(W \cdot H)$ and $\beta_{k,ij} = |\Omega_k|/(W \cdot H)$ in order to highlight the importance of the loss from miss detection according to the ratio $|\bar{\Omega}_k|/|\Omega_k|$, where Ω_k and $\bar{\Omega}_k$ represent the inpainted region and the uninpainted region in the image $\boldsymbol{X}_k$ respectively. Therefore, the TPR performance can be enhanced.

The proposed CNN is trained in order to minimize the average loss E over the training dataset, which is written as $E = \sum_{k=1}^{N} E_k/N$ with N being the number of the training samples. To this goal, the stochastic gradient descent (SGD) is adopted to iteratively update the CNN parameters during the training process.

4.2. Architecture of CNN

The proposed CNN contains a total of $L = 15$ layers: 8 cascaded convolutional layers, 4 max-pooling layers, 2 upsampling layers and a classification layer with a 2-way Softmax classifier. Color images of size 256×256 (3 color channels) are fed into the CNN, and binary images of the same size are output. The CNN architecture as well as specific information about the size of each layer are shown in Fig. 4, and described in details as follows.

The first layer is a convolutional layer called Conv1 that has 64 kernels of size $3 \times 3 \times 3$. The small kernel has less parameters to be trained, which decreases the chance of overfitting and is in favor of the learning efficiency. Conv1 outputs a total of 64 feature maps of size 256×256. After Conv1, an overlapping max-pooling layer is arranged, where the pooling with window size 2×2 and step size 2 is applied. Through this layer, the number of the input feature maps remains the same but their spatial resolution decreases to one fourth of the original value, i.e., 128×128.

The 3rd, 5th, 6th, 8th, 9th, 11th, and 12th layers are convolutional layers for Conv2 to Conv8. They have 128, 256, 256, 512, 512, 4096 and 2 kernels respectively. And the corresponding kernel sizes are given as follows: $3 \times 3 \times 64$ for Conv2, $3 \times 3 \times 128$ for Conv3, $3 \times 3 \times 256$ for Conv4 and Conv5, $3 \times 3 \times 512$ for Conv6, $1 \times 1 \times 512$ for Conv7, and $1 \times 1 \times 4096$ for Conv8. Particularly in Conv7 and Conv8, the dropout technique is applied [40], which sets to zero the output of each hidden neuron with probability 0.5. By this way, the complex co-adaptations of neurons are reduced, and they are guided to learn more robust features for inpainting forensics. In addition, Conv7 and Conv8 have the functions similar to fully-connected layers but no restriction on the size of the input feature maps. According to the investigation in [41], such a convolutional layer as Conv7 or Conv8, called "Network in Network", can be used to effectively learn the relationships among different feature maps by doing dot product in three dimensions.

The 4th, 7th, and 10th layers are three pooling layers taking the same parameter settings as the first pooling layer in the CNN. Thus, the 12th layer outputs two feature maps of size 16×16, which corresponds to the classes inpainting and uninpainting respectively.

The latter two layers are the upsampling layers, each of which has the same sampling rate 1/16. Through them, two feature maps are yielded with the same spatial resolution as the input image of the CNN. They are finally fed to the classification layer, where the Softmax function is performed. The last layer outputs the IPM prediction for the input image, and thus the corresponding label matrix is yielded.

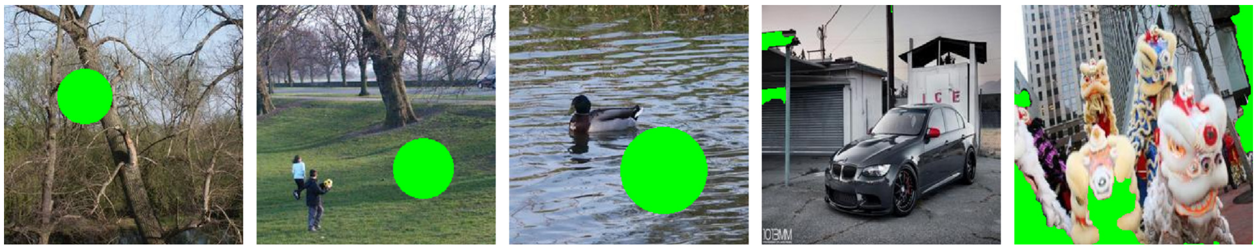

Fig. 5. The sample images with the missing regions (marked in green) in the regular and irregular shapes and the removing ratio being $\{5\%, 10\%, 20\%, 5\%, 20\%\}$ (from left to right), respectively.

Table 1
Comparing TPR (%), FPR (%) and AP (%) of Refs. [13,14] and the proposed method.

Region shape	Tampered ratio (%)	Ref. [13]			Ref. [14]			Proposed		
		TPR	FPR	AP	TPR	FPR	AP	TPR	FPR	AP
Regular	5	85.1	2.9	96.8	94.5	2.8	97.1	89.8	1.4	98.3
	10	91.3	7.3	92.7	96.0	6.0	94.1	87.9	1.4	97.8
	20	91.7	15.9	85.1	96.8	10.8	90.1	85.3	1.5	97.1
Irregular	[0, 10)	78.5	5.2	94.1	92.1	4.3	95.6	89.1	0.5	98.9
	[10, 30)	88.4	14.5	85.9	94.4	11.7	89.2	93.7	1.2	97.9
	[30, 50]	89.1	42.5	69.1	94.7	29.7	79.6	96.5	1.8	97.6

5. Experimental results

In order to validate the CNN-based forensics approach, we train and test the established CNN on two typical image databases. The performance is assessed by TPR, FPR, AP as well as the computational cost per image, and compared with state-of-the-art inpainting forensics methods in [13,14].

5.1. Experimental setup

5.1.1. Training and validation datasets

In MIT Place dataset [42], we randomly select 10^4 color images of size 256×256, and apply the image rotation and mirror operations to them, yielding a total of 5×10^4 images. To create the inpainted image data, these images are then inpainted by Criminisi's method in [33] with different tampered regions in size and shape. Specifically, on one third images, the tampered regions are rectangular with the tampering ratios (i.e., $|\Omega|/|C|$) being 5%, 10% and 20%, but the tampered location is randomly selected. On another one third images, the round tampered regions are utilized while the tampering ratios and the tampered locations are generated in the same way as the previous case. On the remaining one third images, the tampered regions are of irregular shape with the tampering ratios varying from 1% to 50%. Several sample images are shown in Fig. 5 with the masked regions in green to be tampered, which are produced following the above design rules. From the utilized tampered regions, the ground truth label matrices are formed for the inpainted images according to (2). Last, the inpainted images with the associated label matrices are divided into two subsets: one contains 80% samples as training data; the left samples serve as validation data.

5.1.2. Testing dataset

For better experimental validation of the proposed approach, a total of 2000 color images are chosen from UCID [43], which is commonly used in image forensics literature for testing process. These images are cropped to 256×256 pixels from the center. Ninety percent images are inpainted as done for training data while the remaining images are not tampered. Their associated label matrices are still acquired by (2). These sample data serve as the testing dataset.

5.1.3. Implementation of the CNN

The CNN for inpainting forensics is implemented with the Keras deep learning framework [44]. The training and testing are carried out using one Nvidia GeForce GTX 980 GPU with 4 GB RAM. The training parameters of the stochastic gradient descent are set as follows. The momentum value is fixed to 0.95. The weight decay is 10^{-4}. The learning rate is initialized to 4×10^{-3}. And the batch size is set to 32 images.

5.2. Forensics on images without other distortions

The performance of the trained CNN is first tested on the original inpainted images, and compared with that of two related methods [13,14], which use the default parameter values provided in their papers.

Several inpainted sample images are shown in Fig. 6 together with the tampered regions recognized by all the tested methods. We can see, the true missing regions (illustrated in Fig. 5) can be accurately determined in location and shape by the proposed method (shown in the last row of Fig. 6) for various tampered region settings. Moreover, false-alarm pixels often locate at uniform areas, since these areas share similar color distributions, which inevitably causes the increase of FPR. Comparing to [13] and [14], the CNN-based method not only outputs less false-alarm pixels on each tested images, but also gets the detected inpainting regions better fitting with the ground truth masks shown in Fig. 5.

The TPR, FPR and AP averaged over the testing data are summarized in Table 1 for all the tested methods. It is shown that, on the tested images with the irregular tampered regions, the proposed method gets the highest TPR of 96.5% and the FPR as low as 1.8% while the tampered ratio is larger than 30%. With the decrease of the tampered ratio, the forensics performance degrades very slowly. In the extreme case that the images are uninpainted, our method still provides the FPR less than 1.5%. This means that the inpainting features are definitely captured by the CNN. Similar observations can be made for the round or rectangular tampered regions, but the CNN manifests a little worse performance for the latter case. This might be due to the fact that the CNN is driven to learn the regular shape feature of the tampered region, which brings out the negative influence. For all types of the tampered regions to be considered, the CNN performs significantly better than other methods [13,14] in terms of FPR and AP.

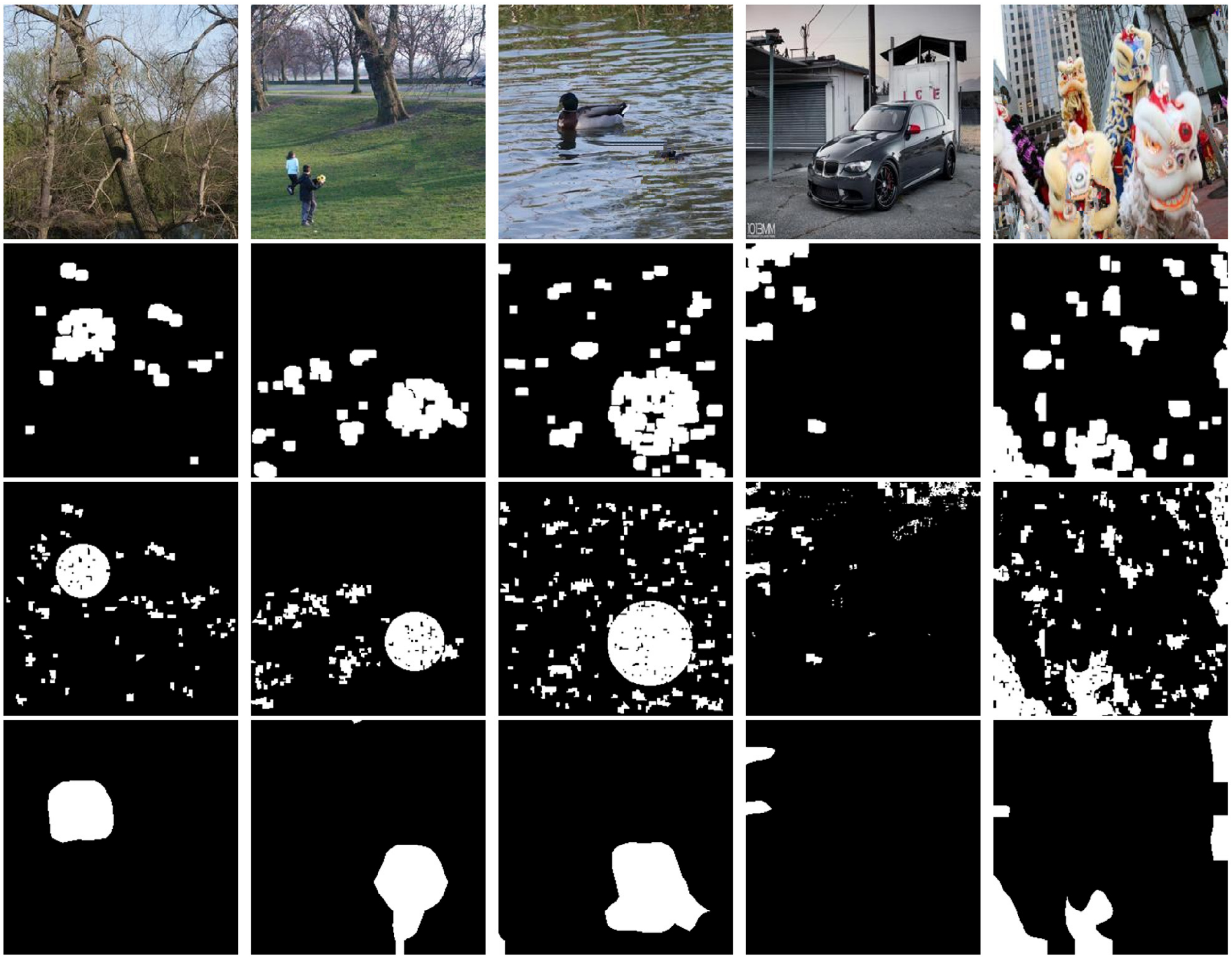

Fig. 6. The original inpainted images (1st row) and the corresponding tampered region detection results obtained by the methods proposed in [13] (2nd row) and [14] (3rd row), and our CNN (4th row).

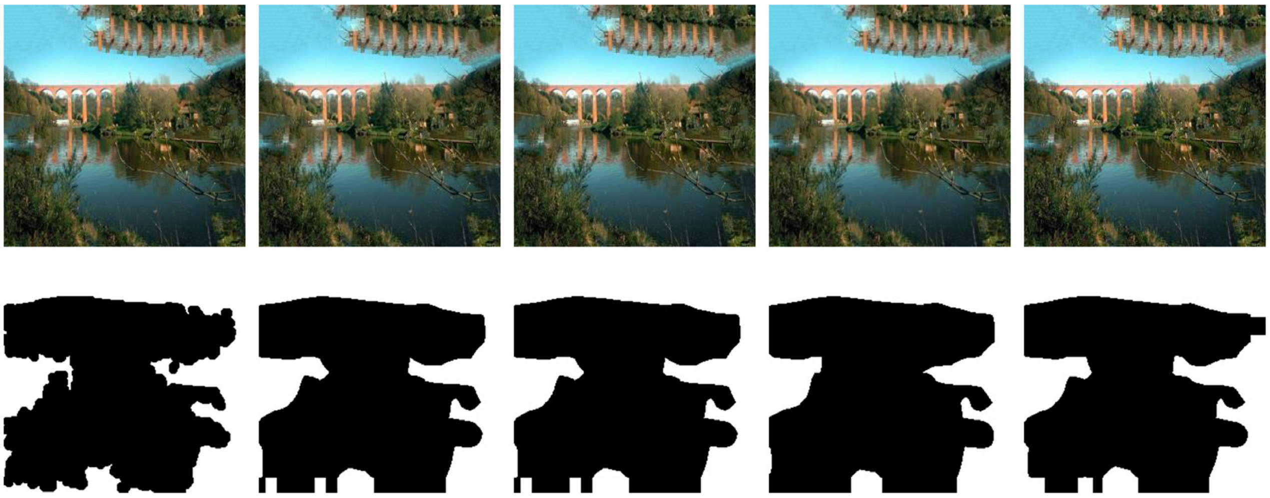

Fig. 7. An inpainted image without distortions (upper left) and the utilized ground truth mask (bottom left), as well as the compressed inpainted images (1st row) and the corresponding tampered region detection results (2nd row) with QF being 95, 90, 80, 70, respectively (2nd to 5th column).

5.3. Effects of compression and scaling manipulations

Further, we investigate the robustness of the CNN-based method with respect to image compression and scaling. The two manipulations are considered because they are often performed in many applications.

In this test, we randomly select 1600 images in UCID to produce the inpainted images with the irregular tampered regions and the tampering ratios larger than 30%. First, to obtain the inpainted images in JPEG format, these images are compressed with quality factors (QF) of $\{70\%, 80\%, 90\%, 95\%\}$, respectively. The proposed method is

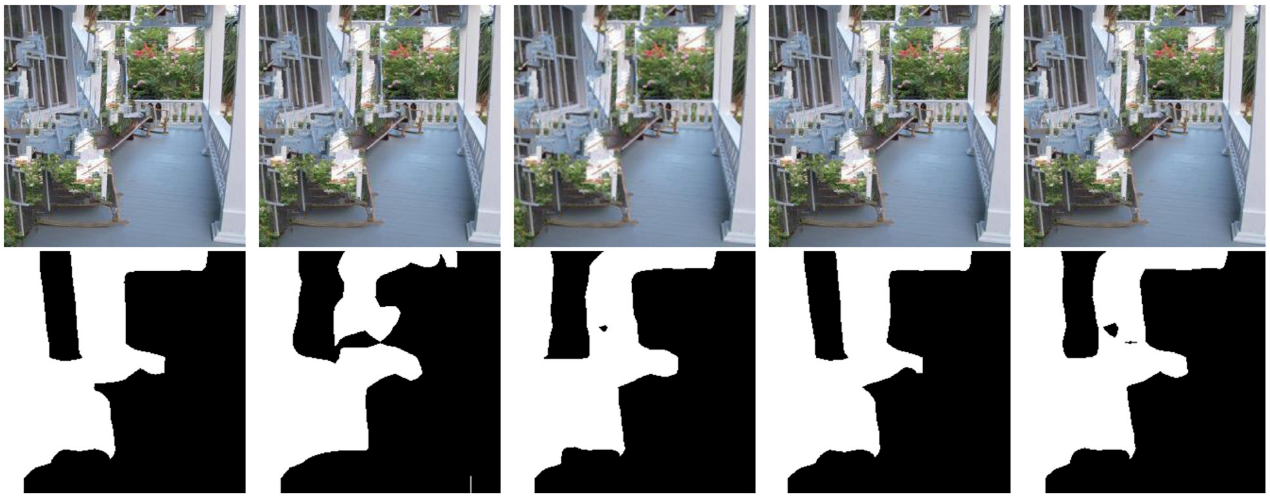

Fig. 8. An inpainted image without distortions (upper left) and the utilized ground truth mask (bottom left), as well as the scaled inpainted images (1st row) and the corresponding tampered region detection results (2nd row) with SF being $0.5, 0.75, 1.5, 2.0$, respectively (2nd to 5th column).

Table 2
Effects of JPEG compression on the performance of the proposed CNN: the inpainted images in the test are generated with the irregular tampered regions and the tampered ratio larger than 30%.

QF (%)	TPR (%)	FPR (%)	AP (%)
95	96.7	2.1	97.5
90	96.6	2.2	97.4
80	94.5	1.5	97.0
70	96.7	2.6	97.2

Table 3
Effects of image scaling on the performance of the proposed CNN: the inpainted images in the test are generated with the irregular tampered regions and the tampered ratio larger than 30%.

SF (%)	TPR (%)	FPR (%)	AP (%)
0.5	90.3	1.3	91.5
0.75	90.4	1.0	95.8
1.5	90.7	1.5	95.6
2	90.7	1.5	93.1

then performed on the compressed images. As an example, Fig. 7 illustrates the uncompressed inpainted image, the compressed ones and the corresponding forensics results for one sample image. Clearly, the minor change for the identified tampered region is caused in size and shape even if QF decreases to 70%. The obtained average TPR, FPR and AP for the values of QF tested are listed in Table 2. It is observed that, the TPRs above 93% are received under the attacks, which are slightly lower than 96.5% for the case that the tested images do not undergo JPEG compression. Meanwhile, the corresponding FPR becomes a little larger except the case of QF = 80%. Similar observations can be made for the AP values. These results demonstrate the proposed approach is very robust to JPEG compression.

The selected images are then scaled by the scaling factors (SFs) within $0.5, 0.75, 1.5, 2$ respectively, and scaled back to the image size 256×256 before forensics. As an example, Fig. 8 depicts the inpainted image without distortions, the scaled ones and the corresponding forensics results for one sample image. It is clear that all the identified tampered regions after the attacks are approximately the same as that obtained on the original inpainted image. Table 3 summarizes the obtained average TPR, FPR and AP for the values of SF tested. We can see that, TPR decreases very slowly as SF goes far from 1, and remains to be larger than 90% even for SF $= 0.5$. Moreover, FPR and AP are affected more weakly. Relatively speaking, the scaling with SF < 1 causes more information loss than that with SF > 1 for the inpainting operation. Thus, the proposed method is insensitive to scaling operations.

Table 4
Comparing TPR (%), FPR (%) and AP (%) of the CNN with the loss function being WCEL and ACEL respectively.

Types of distortions	WCEL			ACEL		
	TPR	FPR	AP	TPR	FPR	AP
No distortions	92.9	3.5	95.2	90.3	2.3	95.0
JPEG 80	78.2	5.9	88.1	77.6	6.3	88.0
JPEG 70	71.1	6.1	85.4	70.2	5.6	85.3
scaling 0.75	89.7	4.5	93.4	89.0	1.5	93.3
scaling 2	91.9	4.0	94.5	90.8	3.7	94.3

Table 5
Comparing running time per image of Refs. [13,14] and our method.

Methods	Image size	Time (s)
Ref. [13]	256×256	195.6
Ref. [14]	256×256	30.7
Proposed	256×256	1.9

5.4. Effects of the weighted cross-entropy loss

In the test, we compare the performance of the CNN with the loss function being the weighted cross-entropy loss (WCEL) and the average cross-entropy loss (ACEL) respectively. To be fair, they are both trained on the established training dataset by taking the training parameters given before. The trained CNNs are tested on the dataset used previously for the robustness evaluation. The obtained results are summarized in Table 4, where the inpainted images without distortions as well as the distorted ones by using JPEG compression with QF = 70 and 80 (i.e., JPEG 70 and JPEG 80), scaling manipulations with SF = 0.75 and 2 (i.e., scaling 0.75 and scaling 2) are considered.

It can be observed that, with the use of WCEL, the CNN gets a little larger TPR on the undistorted inpainted images. However, the FPR increases at the same time. Similar observation can be made on the compressed or scaled images. This behavior can be explained as follows. The loss from miss detections is weighted by a weight proportional to the area of the uninpainted region, which is usually larger than that for the loss from false alarms. As a result, the CNN with WCEL is optimized to be incline to minimize miss detection errors rather than false alarm errors during the training process. In particular, the overall performance in terms of AP can be improved by replacing ACEL with WCEL. Notice that the CNN with WCEL manifests worse performance than that in Tables 1 to 3 for less epochs are used in the training process to save the training time.

5.5. Running time

The computational cost for a forensics approach is a main concern in real-time applications. We carry out the proposed method on all the training images and compute the average running time for each detection. The result is given in Table 5 together with the computational costs for methods [13] and [14], which are respectively obtained by averaging the results presented in [13] and [14].

It is shown that, our method runs very fast, which takes approximately 2 s to detect the inpainting manipulation on an image of size 256 × 256. This is very desirable for a practical application. However, other two methods for comparison perform significantly worse in this aspect. On an image of the same size, they cost us about 195 s and 30 s in average respectively. The high computational complexity mainly results from the fact that the inpainting feature extraction in methods [13] and [14] requires to search the whole image for any investigated patch. The CNN-based method largely reduces the testing time but at the expense of the training time. Generally, we are only concerned with the computational cost in the testing phase since the training phase is usually operated off-line.

6. Conclusion

In this paper, a novel blind image inpainting forensics method based on deep learning has been presented. In the method, we constructed the class label matrix for all the pixels of an image during training the CNN, and designed the weighted cross-entropy as the loss function. The two strengthening supervisory strategies were taken to guide the CNN to automatically learn the inpainting features rather than the image content features. Moreover, our CNN was built following the encoder–decoder network structure in order to predict the inpainting probability for each pixel in an image.

The proposed method has been extensively tested on various images, and compared with state-of-the-art inpainting forensics methods. Experimental results demonstrated that the CNN-based method can automatically learn how to detect image inpainting manipulation without considering feature extraction and classifier design, and using any postprocessing as in conventional forensics methods. Comparing with forensics methods in [13,14], the CNN possesses significantly better forensics performance. Moreover, it yields very good robustness against JPEG compression and scaling manipulations, and offers the apparent advantage on running time.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (Grant No. 60803122, 61401303, and 61573253), by the China Scholarship Fund (Grant No. 201506255067), by the Opening Project of State Key Laboratory of Information Security (Grant No. 2017-MS-11), and by the Seed Foundation of Tianjin University (Grant No. 2017XZY-0090 and 2018XZC-0033).

References

[1] T. Pevny, J. Fridrich, Detection of double-compression in JPEG images for applications in steganography, IEEE Trans. Inf. Forensics Secur. 3 (2) (2008) 247–258.
[2] W. Luo, J. Huang, G. Qiu, JPEG error analysis and its applications to digital image forensics, IEEE Trans. Inf. Forensics Secur. 5 (3) (2010) 480–491.
[3] M.C. Stamm, K.J.R. Liu, Forensic detection of image manipulation using statistical intrinsic fingerprints, IEEE Trans. Inf. Forensics Secur. 5 (3) (2010) 492–506.
[4] X. Kang, M.C. Stamm, A. Peng, K.J.R. Liu, Robust median filtering forensics using an autoregressive model, IEEE Trans. Inf. Forensics Secur. 8 (9) (2013) 1456–1468.
[5] M. Kirchner, Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue, in: ACM workshop on Multimedia and Security, 2008, pp. 11–20.
[6] J. Fridrich, D. Soukal, J. Lukas, Detection of copy-move forgery in digital images, in: Proceedings of Digital Forensic Research Workshop, 2003, pp. 55–61.
[7] I. Amerini, L. Ballan, R. Caldelli, A.D. Bimbo, G. Serra, A sift-based forensic method for copy-move attack detection and transformation recovery, IEEE Trans. Inf. Forensics Secur. 6 (3) (2011) 1099–1110.
[8] M.C. Stamm, M. Wu, K.J.R. Liu, Information forensics: An overview of the first decade, IEEE Access 1 (2013) 167–200.
[9] T. Ružić, A. Pižurica, Context-aware patch-based image inpainting using markov random field modeling, IEEE Trans. Image Process. 24 (1) (2015) 444–456.
[10] Q. Wu, S. Sun, W. Zhu, G.-H. Li, D. Tu, Detection of digital doctoring in exemplar-based inpainted images, in: Proc. IEEE Int. Conf. Mach. Learn. Cybern., Vol. 3, 2008, pp. 1222–1226.
[11] S. Das, G.D. Shreyas, L.D. Devan, Blind detection method for video inpainting forgery, Int. J. Comput. Appl. 60 (11) (2012) 33–37.
[12] K.S. Bacchuwar, K.R. Ramakrishnan, A jump patch-block match algorithm for multiple forgery detection, in: Proc. of IEEE International Multi-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013, pp. 723–728.
[13] I. Chang, J. Yu, C. Chang, A forgery detection algorithm for exemplar-based inpainting images using multi-region relation, Image Vis. Comput. 31 (1) (2013) 57–71.
[14] Z. Liang, G. Yang, X. Ding, L. Li, An efficient forgery detection algorithm for object removal by exemplar-based image inpainting, J. Vis. Commun. Image Represent. 30 (2015) (2015) 75–85.
[15] Y.Q. Zhao, M. Liao, F.Y. Shih, Y.Q. Shi, Tampered region detection of inpainting JPEG images, Optik-Int. J. Light Electron Opt. 124 (16) (2013) 2487–2492.
[16] Q. Liu, B. Zhou, A.H. Sung, M. Qiao, Exposing inpainting forgery in JPEG images under recompression attacks, in: IEEE International Conference on Machine Learning and Applications, 2016, pp. 164–169.
[17] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Netw. 61 (2014) 85–117.
[18] J. Chen, X. Kang, Y. Liu, Z.J. Wang, Median filtering forensics based on convolutional neural networks, IEEE Signal Process. Lett. 22 (11) (2015) 1849–1853.
[19] A. Tuama, F. Comby, M. Chaumont, Camera model identification with the use of deep convolutional neural networks, in: IEEE International Workshop on Information Forensics and Security, 2016, pp. 1–6.
[20] L. Bondi, L. Baroffio, D. Güera, P. Bestagini, E.J. Delp, S. Tubaro, First steps towards camera model identification camera identification with deep convolutional networks, IEEE Signal Process. Lett. 24 (3) (2017) 259–263.
[21] Y. Rao, J. Ni, A deep learning approach to detection of splicing and copy-move forgeries in images, in: IEEE International Workshop on Information Forensics and Security, 2016, pp. 1–6.
[22] Q. Wang, R. Zhang, Double JPEG compression forensics based on a convolutional neural network, EURASIP J. Inf. Secur. 23 (1) (2017) 1–12.
[23] I. Amerini, T. Uricchio, L. Ballan, R. Caldelli, Localization of JPEG double compression through multi-domain convolutional neural networks, in: IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), Workshop on Media Forensics, 2017, pp. 53–59.
[24] M. Barni, L. Bondi, N. Bonettini, P. Bestagini, A. Costanzo, M. Maggini, B. Tondi, S. Tubaro, Aligned and non-aligned double JPEG detection using convolutional neural networks, J. Vis. Commun. Image Represent. 49 (2017) 153–163.
[25] B. Bayar, M.C. Stamm, A deep learning approach to universal image manipulation detection using a new convolutional layer, in: ACM Workshop on Information Hiding and Multimedia Security, 2016, pp. 5–10.
[26] J. Yu, Y. Zhan, J. Yang, X. Kang, A multi-purpose image counter-anti-forensic method using convolutional neural networks, in: Y.Q. Shi, et al. (Eds.), 15th International Workshop on Digital Forensics and Watermarking, in: Lecture Notes in Computer Science (LNCS), vol. 10082, 2016, pp. 3–15.
[27] P.P. Yang, R.R. Ni, Y. Zhao, Recapture image forensics based on laplacian convolutional neural networks, in: Y.Q. Shi, et al. (Eds.), 15th International Workshop on Digital Forensics and Watermarking, in: Lecture Notes in Computer Science (LNCS), vol. 10082, 2016, pp. 119–128.
[28] Y. Qian, J. Dong, W. Wang, T. Tan, Learning and transferring representations for image steganalysis using convolutional neural network, in: IEEE International Conference on Image Processing, 2016, pp. 2752–2756.
[29] G. Xu, H.Z. Wu, Y.Q. Shi, Structural design of convolutional neural networks for steganalysis, IEEE Signal Process. Lett. 23 (5) (2016) 708–712.
[30] M. Bertalmio, G. Sapiro, V. Caselles, C. Ballester, Image inpainting, in: Proc. 27th Annu. Conf. Comput. Graph. Interact. Tech., 2000, pp. 417–424.
[31] D. Tschumperlé, R. Deriche, Vector-valued image regularization with pdes: A common framework for different applications, IEEE Trans. Pattern Anal. Mach. Intell. 27 (4) (2005) 506–517.
[32] T. Chan, J. Shen, Variational restoration of non-flat image features: Models and algorithms, SIAM J. Appl. Math. 61 (4) (2001) 1338–1361.
[33] A. Criminisi, P. Perez, K. Toyama, Region filling and object removal by exemplar-based image inpainting, IEEE Trans. Image Process. 13 (9) (2004) 1200–1212.
[34] Z. Xu, J. Sun, Image inpainting by patch propagation using patch sparsity, IEEE Trans. Image Process. 19 (5) (2010) 1153–1165.
[35] Z. Li, H. He, H.-M. Tai, Z. Yin, F. Chen, Color-direction patch-sparsity-based image inpainting using multidirection features, IEEE Trans. Image Process. 24 (3) (2015) 1138–1152.

[36] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, A. Efros, Context encoders: Feature learning by inpainting, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2536–2544.

[37] E. Shelhamer, J. Long, T. Darrell, Fully convolutional networks for semantic segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (4) (2017) 640–651.

[38] M. Browne, S.S. Ghidary, Convolutional neural networks for image processing: an application in robot vision, in: AI 2003: Advances in Articial Intelligence, in: LNCS, vol. 2903, 2003, pp. 641–652.

[39] A. Krizhevsky, I. Sutskever, G. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[40] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, ResearchGate 3 (4) (2012) 212–223.

[41] M. Lin, Q. Chen, S. Yan, Network in network, in: International Conference on Learning Representations, 2014, pp. 1–10.

[42] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, A. Oliva, Learning deep features for scene recognition using places database, in: Advances in Neural Information Processing Systems (NIPS 2014), Vol. 27, 2014, pp. 487–495.

[43] G. Schaefer, M. Stich, Ucid-an uncompressed color image database, in: SPIE Storage and Retrieval Methods and Applications for Multimedia, 2004, pp. 472–480.

[44] F. Chollet, F. Keras, Deep learning library for theano and tensorflow, in: https://github.com/fchollet/keras, 2015.