Forensic Science International: Reports 2 (2020) 100112

Contents lists available at ScienceDirect

Forensic Science International: Reports

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

Digital Forensics

Robust forgery detection for compressed images using CNN supervision

Boubacar Diallo *, Thierry Urruty, Pascal Bourdon, Christine Fernandez-Maloigne

Universite de Poitiers, CNRS, XLIM, UMR 7252, F-86000 Poitiers, France

ARTICLE INFO

Keywords:
Forgery image detection
Compression
Camera model identification
Convolutional neural networks supervision

ABSTRACT

Images available on online sharing platforms have a high probability of being modified, with additional global transformations such as compression, resizing or filtering covering the possible alteration. Such manipulations impose many constraints on forgery detection algorithms. This article presents a framework improving robustness for image forgery detection. The most important step of our framework is to take into account the image quality corresponding to the chosen application. Therefore, we relied on a camera identification model based on convolutional neural networks. Lossy compression such as JPEG being considered as the most common type of intentional or inadvertent concealment of image forgery, that leads us to experiment our proposal on this manipulation. Thus, our trained CNN is fed with a mixture of different qualities of compressed and uncompressed images. Experimental results showed the importance of this step to improve the effectiveness of our approach against recent literature approaches. To better interpret our trained CNN, we proposed an in-depth supervision by first a visualization of the layer and an experimental analysis of the influence of the learned features. This analysis led us to a more robust and accurate framework. Finally, we applied this improved system on an image forgery detection application and showed some promising results.

1. Introduction

Today, images have become a pervasive part of our life with the common usage of smart acquisition devices such as cameras and smartphones, and the ease of sharing over the Internet. In parallel to this fact, the number of image-processing software increases. They have become accessible, such that anyone may easily modify and share online images. In parallel, tampering images has become means to harm significantly our society. Several techniques have emerged, among them: splicing, copy move, and removal are the most commonly used manipulations (see Fig. 1 for examples).

1 **Splicing:** This is a manipulation technique that copies one or many regions of an image and pastes them onto another image. It can be used for the purpose of adding an additional element to a scene. The first column of Fig. 1 shows a splicing example[1] : original image on top and the manipulated image below with another person.
2 **Copy-move:** It can be used for the purpose of adding false information or hiding information. The second column of Fig. 1 shows an original image on top and the manipulated image below with extra fountains[2].
3 **Removal:** This is a manipulation that replaces specific parts of an image by, for example, using inpainting approaches to fill the missing parts. It can be used for the purpose of removing objects with the intention of hiding information. The third column of Fig. 1 shows a removal example[2] : the original image on top and a missing fisherman on the manipulated image below.

Some of these manipulations are difficult to detect for non-expert user. Moreover, some manipulated images aim at delivering misleading information which might be a threat to society (e.g. mass manipulation, cyber-criminality, tampering or removing of judicial evidence, etc.). Therefore, in the last decade, the forensic researcher community focused on developing tools which validate the integrity of an image. Many approaches detecting the image authenticity and assessing the integrity of the images have been proposed [1,2]. Some of them have focused their interest on forensic issues, and among them, few have dedicated to camera model identification [3–5].

Images are exploited in a large range of use cases where determining their integrity and origin may have high consequences. For example, it is critical in criminal investigations or for news covering. Thus, confirming the image source and its authenticity is one of the most important tasks of the image forensic community. The extracted information from an image and the other multimedia content could show some inconsistencies which give proof of a possible forgery for the image or the whole document. State-of-the-art methodologies which have focused on identifying the camera model or the camera manufacturer, have mainly used the

* Corresponding author at: XLIM Laboratory, 11 Boulevard Marie et Pierre Curie, 86360 Chasseneuil-du-Poitou, France.
E-mail address: boubacar.diallo@univ-poitiers.fr (B. Diallo).

[1] Source: https://www.factcheck.org/2015/07/obama-rouhani-photo-is-not-real
[2] Images source: https://www5.cs.fau.de/research/data/image-manipulation

http://doi.org/10.1016/j.fsir.2020.100112
Received 30 September 2019; Received in revised form 1 May 2020; Accepted 24 May 2020
Available online 13 June 2020

Fig. 1. Tampered image examples: from left to right are the examples showing manipulations of Splicing (*Switching a person*), Copy-move (*Duplicating the fountain*) and Removal (*Removing a person*).

signature left on the image by the camera component workflow which includes physical and digital process traces. Indeed, specific operations are performed by each camera model while acquiring an image (e.g. various JPEG compression algorithms [6], proprietary Colour Filter Array (CFA) interpolation [7–9], sensor noise and other noise statistics [10,11], and prediction residuals [12]).

All those footprints left on the image during the acquisition process characterize the camera model and are widely used as descriptors or combined to forge handcraft features by numerous approaches. For example, in [13,14], researchers have used those pixel descriptors linked to co-occurrence statistics as input of supervised learning methodologies. In the last decade, the use of convolutional neural networks (CNN) has spread in the image forensic community [14,5,15–19]. These algorithms have focused on training the CNN to determine the best features to classify camera models. One advantage of using CNN is the features are extracted directly from the image dataset. The principal advantage of those CNN based approaches is they are capable of learning classification features directly from image data, however, interpreting the extracted camera identification model is very difficult. Another drawback of using CNN is the model is dedicated to a specific training dataset. However, image modifications like geometric distortions (e.g. resizing, compression, filtering) or content manipulations are unpredictable. Moreover, new image forgeries appear every day. Thus, it is necessary for researchers to improve the robustness of their proposed model with respect to all known manipulations.

This article is an extension of our previous work presented in [20] in which we developed a framework for improving the robustness of image forgery detection. As compression is one of the most common and used type of image processing, we focus on improving the robustness against compression manipulation, however, a similar approach could be applied to any other manipulation. The first important step of our framework is to consider the quality of given image data corresponding to the chosen application. To do so, our CNN is fed by a mixture of different qualities of compressed and uncompressed images. Training the CNN with various image compression qualities increases its robustness. In the second phase, we focused on the camera identification model (CMI) based on convolutional neural networks for classification.

An added contribution of this paper is the qualitative and statistical visualization analysis to improve the low interpretability of the CNN network of the first approach proposed [20]. This analysis allows to translate the extracted entities into visually perceptible patterns by known tools such as principal component analysis (PCA) and the stochastic t-distributed projection to the neighbourhood (t-SNE) of the data. Another contribution consists in studying the correlation between pretrained models by exhaustive experiments. Indeed, we made experiments by freezing the first layers of pre-training models before training anew. This set of experiments has highlighted the existence of correlation between the different networks that we studied.

The remainder of this paper is structured as follows: first, we present a brief overview of the literature of image forgery detection approaches focusing on camera model identification (CMI) and convolutional neural networks in Section 2. Then in Section 3, we present our global framework consisting of 4 parts: image pre-processing, camera model identification, CNN supervision analysis, and forgery image detection. Section 4 details our experiments, we discuss the importance of taking into account the compression manipulation for CMI then we analyse our proposed model for a better understanding of CNN. Finally, we study the influence of generalization for the network layers. In Section 5, we highlight the robustness of our proposal for the forgery images detection. We conclude and present some perspectives of this work in Section 6.

2. Related work

2.1. Camera model identification (CMI)

The process to determine which camera brand or model has been used to capture an image is called camera model identification (CMI). This process of CMI has received a large interest as it may be used as proof in specific legal issues. Researchers from the image forensic community have proposed different approaches to identify a camera model. Mainly, they have proposed to extract the image fingerprints left by the camera digital processes during the acquisition [1]. This process is divided in several steps inside the camera device leaving specific features that can be exploited during the identification process. Those fingerprint features when gathered are unique and can be considered as the signature of a specific camera model. Thus, it allows to identify some camera metadata like its origin, the possible processes applied and the integrity of the original images.

Kharrazi et al. [3] have proposed to use statistical metrics as features to determine the camera signature. Among the selected features, they pointed out the importance of colour features, Image Quality Metrics (IQM), and wavelet domain statistics. Celiktutan et al. [21] have shown

the effectiveness of a subset of the features determined by [3] to improve the identification of smartphone cameras. Some researchers have used fingerprints left by physical components of the camera. In [22], they have used the noise of camera sensors for identification.

Other works relied on digital components. In [6], they have used the information given by the JPEG compression process and also fingerprints left by demosaicing [7–9]. As the image acquisition process pipeline is difficult to model, other CMI approaches made use of features which combine image statistic properties and supervised machine learning techniques. Researchers in [23] have presented a local binary pattern methodology that locally captures neighbouring pixel relations. A couple of research works [13,14] were based on the Colour Filter Array (CFA). The process of interpolation used for image classification is a correlation structure existing in each RGB colour band of the image. The principal assumption of the authors is the information gain given by using CFA interpolation algorithm. Their results showed the discriminative power of CFA interpolation to determine the camera model of the images.

Some researchers focus on digital videos. However, as this paper [24] has shown, applying a CMI system trained on images produces low performance when applied on video frames from the same camera. It means that forensic traces left by the camera are different with respect to the capture mode used.

All the above-mentioned approaches have shown very promising results, in particular with high-resolution images providing plentiful pixel statistics. Even if some recent research works still make use of locally hand-crafted features [12], most image forensic researchers rely on convolutional neural networks to identify new camera features for camera identification [14,5,15–19]. This phenomenon is closely linked to the recent development of deep learning techniques, and the astonishing results obtained in the multimedia retrieval and computer vision fields [25–28]. To perform source camera model identification for digital videos, some works as [29,24] have also proposed deep learning based systems with good performance. Numerous CNN based approaches from these papers have shown the importance of training on a large amount of data to improve the effectiveness of the classification tasks.

2.2. Convolutional neural networks (CNN)

CNN have become a widespread tool to address forgery image detection tasks. They have achieved state-of-the-art performance in a large variety of computer vision and image processing applications to solve detection and classification problems.

In the late 1980s, LeCun et al. [30] are the first to train a CNN to recognize handwritten letters. This CNN ancestor became mature in 1998 and its superior classification power has been demonstrated over the MNIST handwritten number database [31]. Afterwards, CNN have been used in many applications during the 90s with good performance on fingerprint recognition in 1993s by Baldi et al. [32]. But soon they quickly became obsolete with the emergence of new methods such as Support Vector Machines (SVMs) and Bayesian Networks (BNs) [33].

In 2012, CNN made a come back and became a reference in the literature of image classification techniques. This phenomenon was possible due to two main factors. First, high-performance computing systems including GPUs became affordable and available for all. Thus, training CNN with a small dataset was now possible even on a laptop computer. The second important factor of this success was the huge number of annotated images available on the internet and in particular with the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) won by AlexNet [34]. Since then, this ImageNet challenge became a showdown benchmark for big companies like Google, Microsoft, IBM, and many deep learning models like GoogLeNet [35] or ResNet [36] are now available and are commonly used to extract deep features feeding an application-specific classification layer.

In the last few years, Convolutional neural networks have also become a very widespread tool to address forgery image detection tasks [14,12,5,37,38,16]. Bayar et al. [39] have proposed a new form of convolutional layer. This layer has been specifically designed to detect the features that have undergone some manipulations. Their proposed framework is trained on a large dataset of unaltered images plus datasets with different manipulations like noising, blurring or resizing. Bulk et al. [40] have shown the detection tampered regions could be made by combining features provided by CNN and LSTM networks (Long Short Term memory). Their approach did not take into account the possibility of possible manipulations which may happen after the tampering. Bayar et al. [41] have studied the effect of different CNN architectures for forensic purpose. They demonstrate the importance of constraining some convolutional layers (e.g. with a high-pass filter) at the beginning of the CNN model. Bondi et al. [5,37] have proposed two techniques which combine image forgery detection and localization. First, the CNN is used to extract patch features. Then, those features are clustered to detect and localize the tampering regions. Once again, possible post-processing manipulations have not been taken into account in their approach. Alotaib et al. [42] have presented an effective and non-invasive technique to prevent face-spoofing attacks. They have proposed a specialized CNN extracting discriminative deep features to recognize a fake face to a real one.

Other recent approaches have focused their effort on specific image forgery issues, e.g. detecting tampering cues like double-JPEG compression [15,16], re-sampling and contrast enhancement [43]. Detecting and localizing face modification using CNN has been proposed by [44]. Huh et al. [45] have developed a model capable of detecting image manipulations. Their model is specifically trained to detect and localize splicing attacks.

More recently in [46], a new fingerprint called Noiseprint has been proposed. It uses a Siamese CNN framework to link different modalities extracted from the camera model artifacts. It can be combined with photo response non-uniformity scenario with much fewer data for robust and efficient device source identification [47].

In many of the recent proposed paper, CNN based models have achieved state-of-the-art performance. However, their principal drawback for expert user is their interpretation. How can we interpret a feature extracted from a CNN model? What can we learn from it? How to interpret the millions of parameters that have been tuned by training? Several approaches for understanding and visualizing have been developed in the literature. In [48], the authors investigated the effect of some manipulation including compression on the CNN. The authors have demonstrated that meaningful interpretation could be given for very few categories which are easily differentiated by particular filters on first convolutional layers. Qin et al. [49] have provided a comprehensive survey on different CNN visualization approaches, including Activation Maximization, Network Inversion, Deconvolutional Neural Networks (DeconvNet), and Network Dissection based visualization. Please note that CNN tend to learn only features related to its visual content. It is therefore important to take into account the input image quality in regard to the application.

3. Proposed method

In this section, we detail our proposed approach. Its principal objective is to provide a robust and effective framework for camera model identification (CMI) and image forgery detection. Fig. 2 shows the global framework of our proposal. This framework contains four specific parts. First part concerns all the image pre-processing steps. In this part, we also show the importance of taking into account the quality of the input data to strengthen the robustness. Then, we explain the classification approach using Convolutional Neural Networks to identify camera models CMI. Next part highlights an in-depth analysis of our CNN which allows us to better understand and improve our framework. Finally, we test our proposed framework on a forgery detection application.

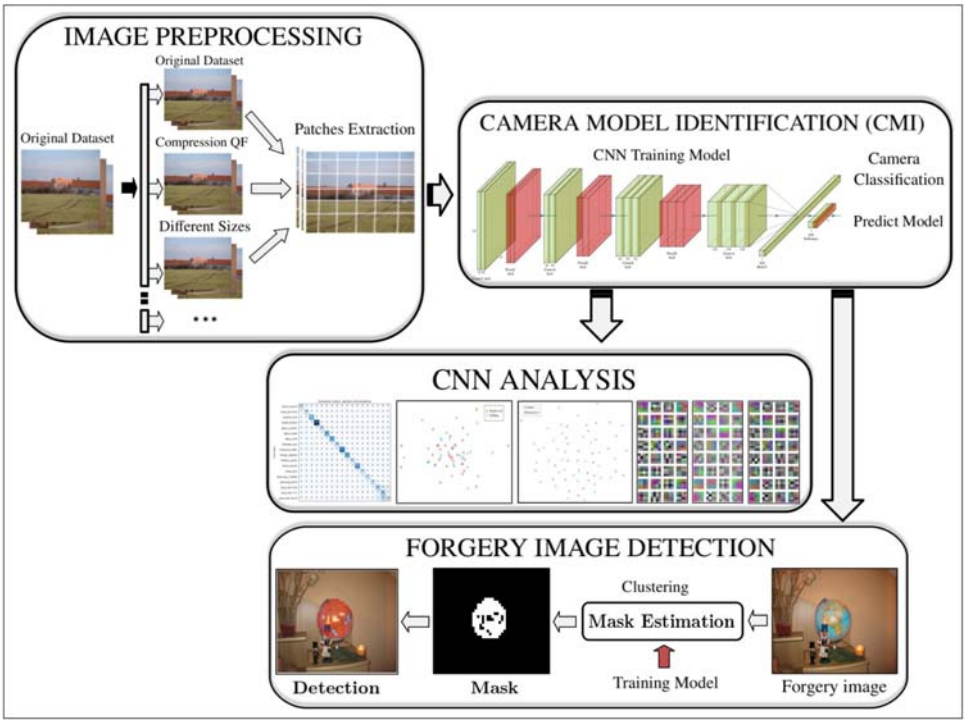


Fig. 2. Our proposed framework.

3.1. Image pre-processing

In this part, we detail the selected pre-processing steps needed in our framework (see Fig. 2). CNN require large amounts of training data. Pre-processing techniques augment the existing dataset with transformed versions of the existing images. This is done to expose the neural network to a wide variety of data. Scaling and compression are transformation examples among others. In the next subsections, we give details about each component used for image pre-processing.

Image transformations: The quality and quantity of the training dataset is a crucial step to achieve high accuracy when using a deep learning model in the framework. One of our research objectives is to apply our image forgery detection approach on online shared images. Thus, our training set has to go through transformations that should be similar to the common ones used on the Internet, e.g. filtering, resizing or compression. Consequently, all images of the original dataset are duplicated and undergo transformations of themselves to form other datasets. For example, for compression manipulation, we need to build new datasets of different qualities of compressed images which are added to the original dataset. Experiments demonstrate the great importance of this step to improve the robustness and accuracy of the framework.

Patch extraction: The second step of our framework divides the images of the dataset into 64×64 pixel patches. Indeed, the choice of using small images instead of full-resolution images can be explained by the following arguments: (i) it reduces the space size to improve the camera model representation; (ii) the efficiency of algorithms used in the framework and (iii) the data quantity to better train the deep learning model. Moreover, state-of-the-art approaches for camera model classification have shown better performance using small image patches according to [5]. As the quality of the input data is important to train the model, we filter the patches which contains overly dark or saturated regions. Indeed, the number of those specific patches is too high and disturb the training process. It also increases the efficiency of the training phase as we reduce the number of patches feeding the neural network. The resulting patches are linked to their camera model label before being distributed into training, validation and test sets used to feed the CNN.

3.2. Camera model identification (CMI)

In this section, we detail our CMI approach which is the second contribution of this article (please refer to Fig. 2). As we already mentioned, detecting the origin of a specific image or picture can be

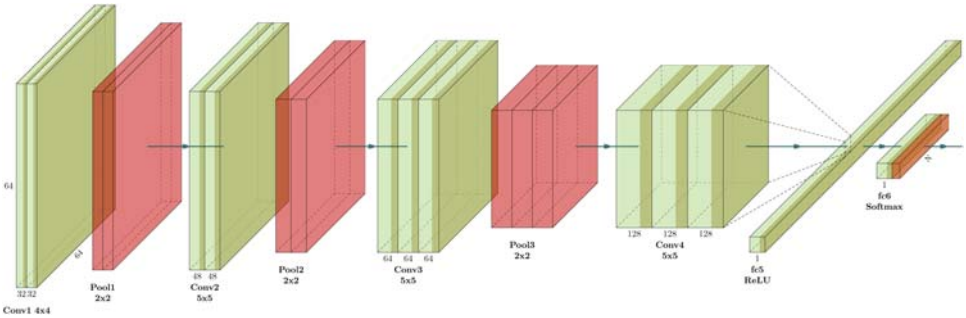


Fig. 3. CNN architecture [5]. *Conv* denotes a convolutional layer, *Pool*, a max-pooling layer, and *Fc* a fully-connected layer.

crucial and in particular for forensic applications as legal proceedings or criminal investigations. This identifying information can be used to detect image inconsistencies and be considered as first hints of an image forgery. Deep learning based approaches are now considered as a baseline for CMI. Significant research has been done in this direction and allows to extract discriminative features from the images of interest to better train the classification model. In the following, we focus the CNN architecture and classification procedure.

Convolutional neural networks for camera model identification: Given its great potential, deep learning has become inevitable for camera model identification. In this work, we exploit convolutional neural networks (CNN) to extract camera model features from image patches. The first CNN architecture specifically dedicated to camera model identification has been proposed in [5]. In our work, we used a similar network. This choice is motivated with the aim to achieve a high camera model attribution accuracy with a fairly small network architecture. The used network contains 9 layers namely 4 convolutional layers, 3 max-pooling layers, 2 fully connected layers (1 ReLU layer and 1 Softmax layer). Image patches are fed into the CNN through an input layer, also known as the data layer. The structure of the CNN architecture is described below and in Fig. 3:

Convolutional Layer: The Convolutional layer is the cornerstone of a CNN, as it performs most of the heavy computational operations. It consists of a set of filters where each filter is spatially defined with a width and a height and is applied with an overlapping distance called 'stride' to all the local regions of the given input. Input can be an image of dimension $w \times l \times c$ where w and l denote respectively the width and the height of the image and c represents its RGB channel values. The input image is convolved with a defined number of kernels of dimension $s \times s \times c$ where s denotes the filter size.

Our model contains 4 convolutional layers we denote Conv1 to Conv4.

- Convolutional layer (Conv1) has a kernel size of $4 \times 4 \times 3$ with 32 feature maps as output. Input is a set of patches of size $64 \times 64 \times 3$. The convolutional filter is applied with a stride of 1.
- Convolutional layer (Conv2) contains 48 filters of size $5 \times 5 \times 32$ (stride = 1). It generates, as output, $28 \times 28 \times 48$ feature maps.
- Convolutional layer (Conv3) contains 64 filters of size $5 \times 5 \times 48$ (stride = 1). It generates, as output, $10 \times 10 \times 64$ feature maps.
- Convolutional layer (Conv4) contains 128 filters of size $5 \times 5 \times 64$ (stride = 1). It generates, as output, a vector of 128 feature maps.

Max-Pooling:It is common to insert a pooling layer in-between successive convolutional layers in a deep learning architecture. The main objective of the pooling layer is to reduce the feature maps created by the convolutional layers. It progressively reduces the number of parameters and the computational cost of training the network. The second goal is to control an overfitting training. The most common down sampling operation is Max Pooling. This method consists of selecting the maximum value in each neighbourhood.Our model uses a 3 max-pooling layers with a kernel size of 2 x 2 and a stride of 2 respectively called Pool1, Pool2 and Pool3. The resulting of the $63 \times 63 \times 32$ maps in Conv1 is aggregated with a Max-Pooling layer producing $32 \times 32 \times 32$ maps as output. The size of the maps in Conv2 is reduced from $28 \times 28 \times 48$ to $14 \times 14 \times 48$ maps as output after the second Max-Pooling layer. The output of the third Max-Pooling layer is maps of size $5 \times 5 \times 64$, reduced from the Conv3 maps of size $10 \times 10 \times 64$.Fully ConnectedLayer:The fully connected (FC) layers connect the convolutional blocks (convolution and pooling layers) to the bottleneck of the CNN, which represents the deep feature of the input image. In this work, we use the RELU activation function as it has shown good performance with the non-saturation of the gradient and its computational efficiency. Indeed it has been shown CNN with ReLU activation train several times faster than other activation functions.In our model, 2 fully connected layers are used. The first layer (Fc5) with a ReLU activation produces a 128 element feature vector as output. The second (Fc6) with a softmax activation produces a feature vector that corresponds to the probability of a patch to belong to one of the camera models used for training.

Classification Procedure: Once the model is trained, the classification procedure can be applied to any new image input. The new image is divided into a set of patches as described in Section 3.1. The softmax layer attributes a prediction label to each patch. The final camera model prediction is obtained using a majority voting on the predicted patch labels.

3.3. CNN analysis

The exceptional ability of CNN to create a convenient feature representation has made them a popular tool achieving high performance in many application domains. However, experts from these domains require more interpretability from the "black box" CNN. Several approaches for understanding and visualizing Convolutional Networks have been developed in the literature [48,49] as a response to the common criticism that the learned features in a Neural Network are not interpretable. Those learned features are indeed hard to interpret from a human expert point of view. This is mainly due to a lack of understanding of the working mechanism of deep learning model, closely linked to the large number of parameters used in these networks. To improve this low CNN interpretability of our CNN architecture, we propose a qualitative and statistical visualization analysis. It converts specific layer features into visually perceptible patterns. To do so, we used the Principal component analysis (PCA) and stochastic t-distributed projection to the neighbourhood (t-SNE) of the data. This analysis detailed in Section 4.3 allows us to improve the robustness and accuracy of our overall framework. Moreover, we detailed our exhaustive experiments in Section 4.4 showing the generalization of the first CNN layers. That is to say by freezing the first layers of a pre-trained model before training anew this model. The main idea of testing the generalization is to know the possibility to mix the training sets into only one for first layers and in a second step, fine tune only with a specific dataset the last layers of the network. This scheme allows to reduce the training time and to improve the robustness of the framework.

3.4. Forgery image detection

This section briefly presents the proposed approach for image forgery detection. We focus our detection on image forged from different camera models. For reproduction purpose, we construct our scenario on the one given by Bondi et al. [37]. We consider that pristine images come directly from a unique camera.

Forged images are created from patches of pristine images of different camera models and pasting them together. The objective of our forgery detection approach is to estimate if the image is fully composed of one camera model patches which means the image is pristine or if some regions of the image are from different camera models, i.e. the image has been forged. In the latter case, determining the forged region in the input image is also part of the performance evaluation of our approach.

Fig. 2 presents the different steps of the forgery detection algorithm. First, the image is divided into non-overlapping patches. The trained model extracts a feature vector of N elements representing the number of camera models. A clustering algorithm uses this information to estimate a binary tampering mask. In this mask, black regions show patches belonging to the pristine region and white ones indicate forged patches. The image is considered as pristine if there is no white region.

4. Experiments

This section presents our exhaustive experiment results. After detailing the experiment setup including chosen datasets and evaluation criteria, we will propose a preliminary study highlighting the importance of compression as an image manipulation technique. It is a summary of our work done in [20]. Then, we will detail the performance of our framework for camera model identification. We will also analyse our proposed model for a better understanding of CNN and finally, we will study the influence of generalization for the network layers.

4.1. Experiment setup

- Datasets description:

  –**CMI:** Dresden dataset [50] is a publicly available dataset suitable for image source attribution problems. Dresden contains more than 13,000 images of 18 different camera models. For each precise scene, several cameras were arranged in approximately the same position and took almost the same picture. This means that each scene is represented by several (or all) existing cameras. The database is composed of color images of very high quality pictures (around 4000×3000 pixels). Note that we selected only natural JPEG photos from camera models with more than one instance. For the purposes of learning, we divided the dataset into separate training, validation, and evaluation sets denoted DT, DV, and DE respectively. This then results in 7938 images in the training set, 1353 images for validation and 5400 images in the evaluation dataset.
  By extracting 32 patches by image, a dataset of more than 254,000 patches is obtained and used as the training dataset. Each patch is labeled with the true camera model. This process is repeated on two other datasets: a validation dataset (DV) containing more than 43,000 patches and an evaluation dataset (DE) with more than 170,000 patches.
  Note that the number of images per camera class is not equal. For example, there is a minimum of 580 images and a maximum of 1200 images per camera class, resulting in an unbalanced classification problem.
  –**Forgery detection:** To evaluate the forgery detection algorithm, we used 2 image datasets. These two separate sets of altered data represent a set of ”Known” data from DE dataset and an ”Unknown” dataset which contains images from another 8 camera models not included in the CNN training phase. The objective is to study the differences in performance when using ”Known” and ”Unknown” camera models. Both sets each contain 500 pristine images and 500 tampered images, making a total of 1000 images per set, generated following the process given in [37].

- Evaluation criteria:

  –**CMI:** To evaluate the camera model identification performance, we use the average accuracy obtained with a majority voting.
  –**Forgery Detection:** We evaluate detection performance on both ”known” and ”unknown” datasets in terms of accuracy, receiver operating characteristic (ROC) curves and Area Under the ROC Curve (AUC). These statistics are commonly known and used, they identify clearly the difference between the performance of studied approaches.

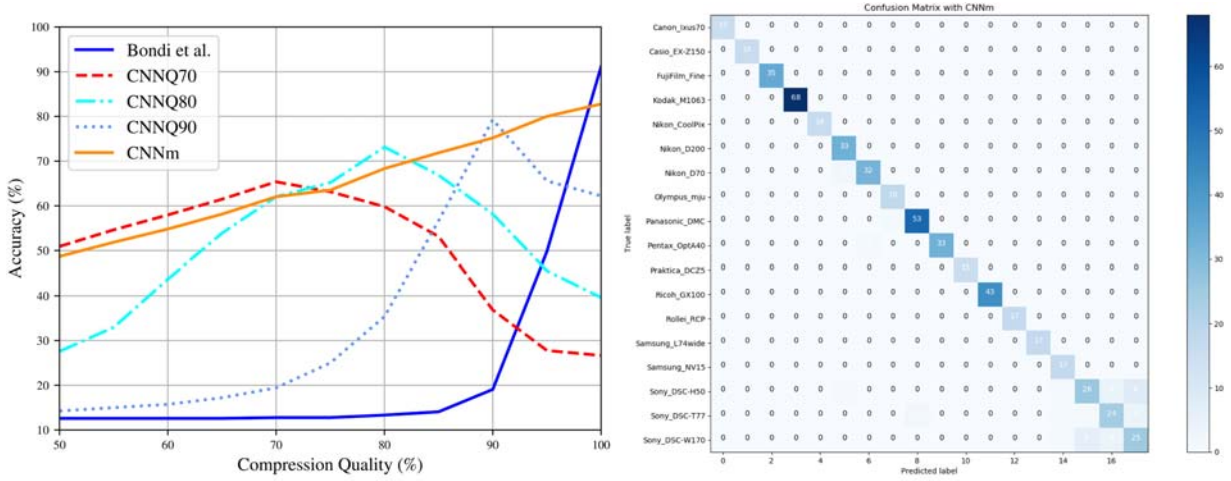
4.2. Influence of compression on camera model identification

In this section, we propose our preliminary study that highlights the importance of manipulation process on the camera model identification (CMI) accuracy of our framework denoted CNNm compared to the one proposed by Bondi et al. [37]. To make this robustness assessment, we train and test all trained CNN previously details with the four datasets of different quality factors. Note that *Bondi et al.* is the CNN model trained on “*Original*” *images*. *CNN70*, *CNN80*, and *CNN90* are the CNN trained on compressed images with respective quality factor *70*, *80*, and *90*. *CNNm* is the CNN trained on *mixed uncompressed and compressed images*.

Parameters training:

The training architecture with 340, 642 parameters in total is learned in batches of 128 patches, using the optimization algorithm called Stochastic Gradient Descending (SGD). The other hyperparameters are momentum equal to 0.9 and decay weight equal to $7.5e^{-3}$. When training the CNN, we select the best model that gives the minimum loss on validation patches within the 50 first epochs. Training is performed on Caffe1.0 on an Nvidia Quadro M 1000M GPU.

The curve of Fig. 4(a) shows the influence of the JPEG compression on the CMI accuracy. On the ”Original images” dataset, the model developed by Bondi et al. shows higher performance [37]. This high performance of their model is only accurate for one dataset as shown by the important accuracy drop obtained on a close quality factor ($QF = 90\%$). Our proposed CNNm model performance presents stable and high accuracy performance with a reasonable quality compression factor. This result demonstrates the importance of the compression manipulation in the robustness of CMI framework. One possible explanation behind this result



(a) Accuracy curves of CMI (b) Confusion Matrix CNNm Model on DE

Fig. 4. (a) CMI accuracy performance [Bondi et al. is the CNN trained only on ”Original” images. CNN70, CNN80, and CNN90 are the CNN trained on compressed images with respective quality factor 70, 80, and 90. CNNm is the CNN trained on mixed uncompressed and compressed images] and (b) Confusion Matrix CNNm Model on Evaluation Dataset.

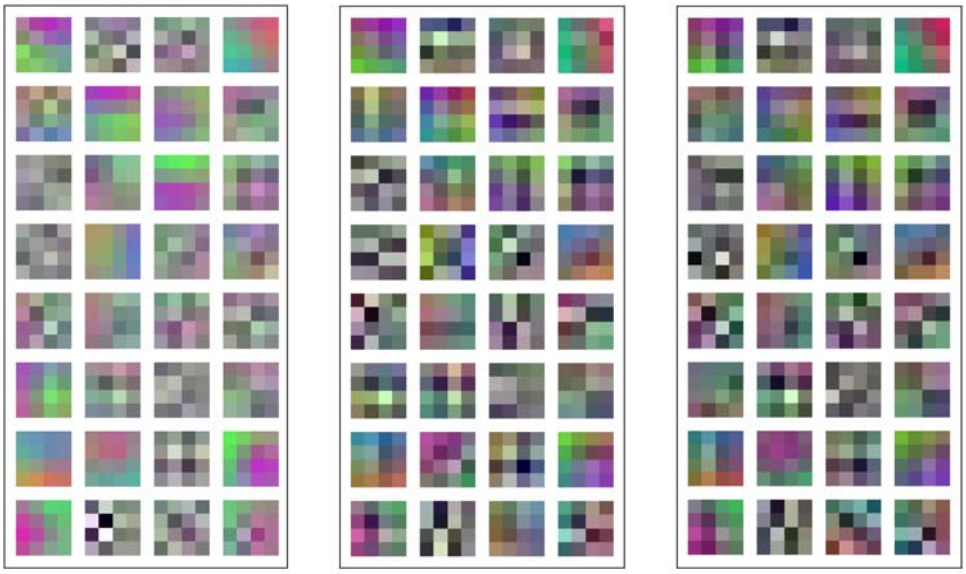


Fig. 5. Visualization of first convolutional layer weights.

is JPEG compression reduces discriminative anomalies between block pairs, deleting camera fingerprints for CNN patch-based approaches.

Those results confirm training a CNN with a specific compression quality factor provides better performance but only for images compressed with this specific *QF*. Consequently, the performance of our proposed approach is globally higher on the mixed test image dataset (with the CNNm Model).

Fig. 4(b) shows the confusion matrix of this CNNm model on the uncompressed test dataset (DE). This matrix shows a good classification of the models of cameras with small errors on the models of a same brand. Indeed, models of a same brand may contain similar components and processing tools which result in similar model fingerprints.

To conclude on this first set of experiments, we obtain a better accuracy from the CNNm model. However, under a specific quality factor threshold, results worsen for all CNN architecture. However, the image quality under this threshold is too low to be considered as a forgery.

4.3. Understanding and visualizing convolutional network

In this section, the addressed strategy is to visualize the feature maps after CNN training. In a network, the filter weights are usually more interpretable for the first convolutional layer which is applied directly at the raw pixels, but it is possible to also analyse them deeper in the network. In our case, we visualize the first layer of the network which has a dimension of 32 filters of size 4x4x3. From this, we try to compare the model learned using original data with the one learned using compressed images. In Fig. 5, we have (a) the filter weights of first convolutional layer of Bondi et al. model [37], (b) those of CNN80 model trained with D_80 and (c) those of CNNm trained from the mixed dataset D_{Mix} (c). Those three models have visually very similar filters, in terms of colour or texture. We observe that the filters of Bondi et al. model seem to be slightly smoother than both CNN80 and CNNm models.

Inspired by the work on network visualization [51], we propose an alternative to the visualization approach, a statistical representation view of the CNN first layer filters. To do so, we use the PCA and t-SNE projections of the data. Principal component analysis (PCA) is a popular technique to construct a representation of the data that captures maximally variant dimensions of the data. It computes a representation with a set of basis vectors said dominant eigenvectors of the covariance matrix generated by the data. Stochastic t-distributed projection to the neighbourhood (t-SNE) is a nonlinear dimensionality reduction algorithm allowing to visualize high-dimensional data by giving each data point a location in a two or three-dimensional map.

We apply those algorithms to the weights of the first convolutional layer of the networks. For comparison purpose clarity, we only show in Fig. 6 the weights from two models, Bondi et al. and CNNm. PCA as well as

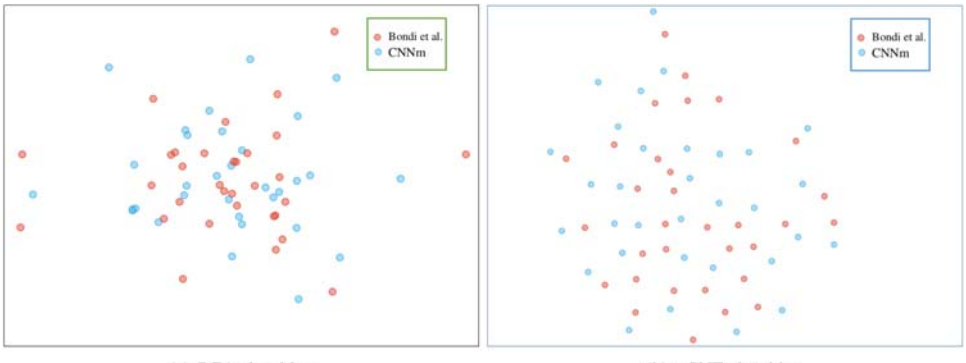


Fig. 6. (a) PCA and (b) t-SNE algorithms applied to the weights of the first convolution layer. Red circles are convolution kernels of Bondi et al. model, and blue circles are convolution kernels of CNNm model.

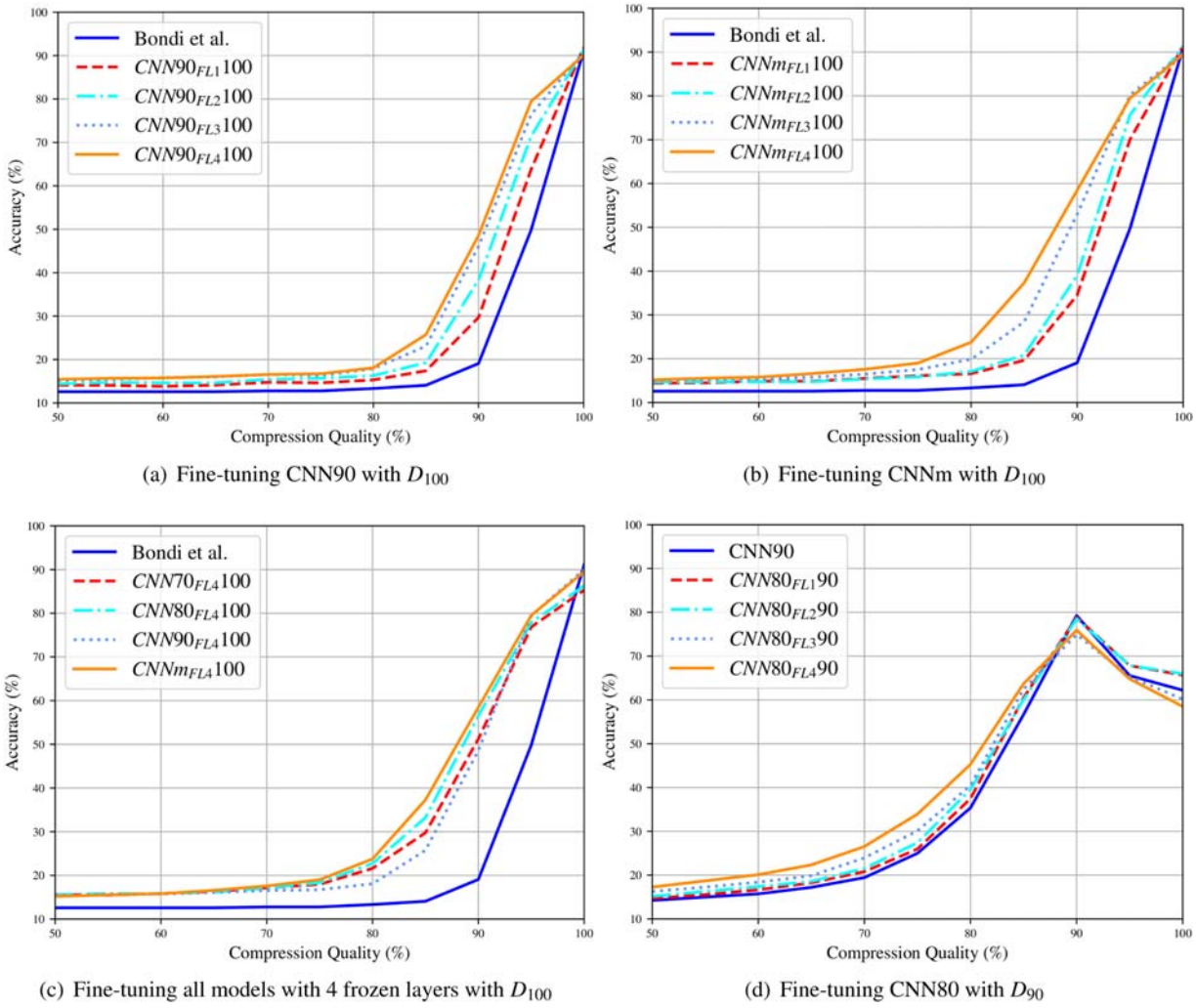


Fig. 7. Influence of fine-tuning models.

t-SNE algorithms show no linear or basic geometric separation able to divide the obtained representation into 2 clusters. We observe similar results with other network representations. This proximity between the first layer of all networks in terms of visualization and statistics leads a network generalization hypothesis. Thus, we propose to fine-tune the different networks to prove this hypothesis which is well known from the literature [52,53].

4.4. Network layer generalization

In this section, we try to determine the existence of a network layer generalization between the different networks we used. To do so, we build exhaustive experiments which consist of using the trained models we already have, and fine-tuning them with another dataset. This is done by freezing first layers of a trained model and restart the training from this model with another quality factor compression dataset. We test this scheme for the first 4 convolutional layers, freezing between one and four first convolutional layers, named *FL*1 to *FL*4 respectively. We denoted for example CNN80$_{FL2}$100 the new trained model from the initial CNN80 model with the 2 first layers frozen and fine-tuned with the dataset D_{100}.

We first observe on all figures of Fig. 7 that highest accuracies of froze models are very close but under the non-frozen model (Bondi et al. model). For example, on the sub-figure (a), for a *QF* of 100%, the fully trained model on this *QF* dataset is equal or higher than other CNN90 fine-tuned models with an accuracy difference below 2%. Similar results may be observed in the sub-figure (b) where the mixed model CNNm has been fine-tuned with D_{100}.

However, when we test those fine-tuned models on other *QF* datasets, we clearly see higher results meaning that freezing first layers has a small effect for a specific dataset if fine-tuned on it but has more effect on other *QF* datasets. Another observation is that the more frozen layers we have, the better are the performance on the other dataset which is logical as more trained parameters come from the other *QF* datasets. For example, in the sub-figure (b), CNNm$_{FL4}$100 obtains an accuracy of 60% on D_{90} compared to the 20% of Bondi et al. model (or CNN100). As those observations have been observed on all our experiments, we present in sub-figure (c) the results of freezing 4 layers from all models before fine-tuning them. These results show once again the robustness of the mixed model CNNm. As shown in sub-figure (d), logical results are also obtained when fine-tuning CNN80 model with D_{90}. For smaller *QF*, the fine-tuned

Table 1
Comparison with other models

Dataset	Models			
	CNNm	ResNet-152 [36]	DenseNet-201 [54]	VGG-19 [55]
Training	**97.20%**	*83.22%*	*66.77%*	*41.68%*
Test	**90.55%**	*30.66%*	*30.22%*	*35.20%*

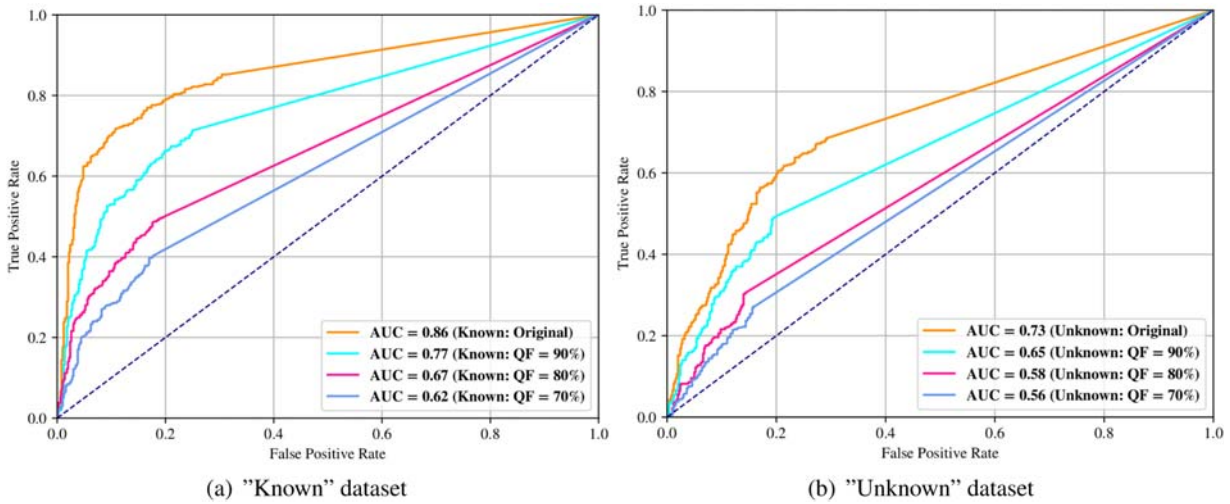


Fig. 8. ROC curves of forgery detection algorithm with CNNm Model tested on (a) "known" and (b) "unknown" datasets.

models perform better, but for higher *QF*, they degrade. It shows the influence of the initial trained model.

In conclusion, results obtained from our exhaustive experiments are mainly positive for the mixed model CNNm. All other fine-tuned models (CNN70, CNN80, CNN90) also yield better classification scores on compressed images. However, the original model still performs a bit better on uncompressed images. Those results demonstrate the advantages of using a generalization approach to improve the robustness of the framework while reducing the computational cost of the training steps of the CNN: first layers are trained only once, and last layers, with fewer parameters, are trained on specific datasets.

4.5. Further analysis with state-of-the-art models

For comparison purpose, we have decided to include recent and weel known deep learning models such as ResNet-152 [36], DenseNet [54], and VGG19 [55]. These recent networks make it possible to train up to hundreds or even thousands of layers and still have allowed boosting the performance of many computer vision applications and classification. The first version of those models have been published between 2014 and 2017. However, those models are updated every year with more layers and new pretrained parameters. Note that, for this experiment, we only use the original dataset without compression factor.

The promising results of the pre-trained convolutional networks led us to apply it to the above-mentioned networks to adapt to our camera classification application in order to be able to make this comparison. The pre-trained models of ResNet, DenseNet, and VGG-19 have millions of parameters trained from the ImageNet dataset with high accuracies. VGG19 model proposed in 2014, contains 19 trainable layers. ResNet model contains 152 layers and DenseNet contains 201 layers. We use the last trained convolution layer with a personalized classification part including a two layers fully-connected classifier adapted to our application.

Table 1 shows the performance of our model compared to transfer learning based approach from very well known models. The low performance (35% test accuracy) we obtained with object classification-targeted models (Resnet, DenseNet and VGG-19) on Dresden ground truth highlight the fact that its classes are unlikely to be identified by specific object features (as opposed to camera model features). In parallel, we have tested our framework for a simple four category scene classification. We obtain a high training accuracy (over 90%) with a low test performance (around 40%). This result also shows that our framework is dedicated to a very specific task, and can not be used for other applications.

5. Image forgery detection

In this section, we analyse the JPEG compression effect on the performance of an image forgery detection approach. Note that, in this section, we have not taken into account the pre-trained models used in previous section as their performance was not good enough.

As previously described, the "Original" datasets have undergone a compression manipulation with increasing quality factors (*QF* of 70%, 80% and 90%). The ROC and AUC for our framework only are shown in Fig. 8. Those figures present the evolution of the mixed model CNNm with respect to different *QF* and they clearly show for forgery detection too, the quality of the image has a great impact on the accuracy with the camera model "Known" or "Unknown" datasets.

Table 2 shows results of forgery detection on camera model "Known" and "Unknown" datasets. Once more, we may notice similarities with previous results. Our proposal performance is close to results obtain by Bondi for the unaltered images (uncompressed). Nonetheless, our results in terms of accuracy is better for the tampered and compressed images of CNNm model, demonstrating the robustness of our framework CNNm. This performance was expected as forgery detection is based on the CNN trained for CMI. The third column shows the results of the fine-tuned model with the four convolutional layers from CNNm model frozen. Observing the results of this column leads to the conclusion that fine-tuning models offers a compromise between fully trained model on one *QF* and training with all *QF*.

Table 2
Results of forgery detection for uncompressed and compressed images

Dataset	Compression	Accuracy		
		Bondi et al. [37]	CNNm	$CNNm_{FL4}100$
Known	*Original*	**0.84**	0.77	**0.79**
	90%	0.56	**0.72**	0.63
	80%	0.52	**0.65**	0.58
	70%	0.52	**0.61**	0.58
Unknown	*Original*	**0.79**	0.7	**0.75**
	90%	0.56	**0.63**	0.58
	80%	0.52	**0.57**	0.56
	70%	0.51	**0.56**	0.54

Fig. 9. Examples of forgery detection: (a) the original image (f) the ground truth with tampered localization (b, c, d, e) results of the detection using Bondi et al. model on an uncompressed image (b), and on compressed images of $QF = 90$ (c), $QF = 80$ (d) and $QF = 70$ (e); (g, h, i, j) results of the detection using our CNNm model on uncompressed on compressed images similarly.

Fig. 9 gives two examples of forgery detection that compares Bondi et al. detection framework [37] to our proposal with respect to different quality factors of compression. As observed previously, a model dedicated the uncompressed images is very accurate for this kind of image, but is ineffective when a *QF* is applied to those same images. Our model detects the tampered region on all images, with a degradation of the accuracy closely linked to the low compression quality factors.

6. Conclusion

In this article, we have proposed a robust framework for camera identification model and image forgery detection. The proposed approach may include any kind of common manipulations frequently used when sharing images on the internet. We were interested in compression, which is one of the most common manipulations on images. First results on different compression quality factors have shown that training with compressed and uncompressed images is of great importance to get better performances than recent literature approaches. Results have proven the performance of the latter strategy which is a compromise between a fully trained model on a specific dataset and on a mixture of different quality factor datasets. Then, to improve the low CNN interpretability (mainly due to a large number of parameters), we have proposed a qualitative and statistical visualization analysis of the internal features. This analysis concludes on the great similarity between first convolutional layer weights of different models leading to the generalization of first layers of those networks with a fine-tuning strategy to a specific dataset. We have also tested a transfer learning-based approach from very well known models (ResNet, VGG-19, and DenseNet). We have highlighted that those models do not allow us to identify the signature of the camera on the images. Finally, we have studied the influence of our different proposal on an image forgery detection application. We have observed promising results showing the robustness of our framework. The perspective of this work includes data augmentation using generative adversarial networks to improve the overall performance over multiple image manipulations.

Declaration of competing interests

The authors have no competing interests to declare

Appendix A. Supplementary Data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.fsir.2020.100112.

References

[1] M.C. Stamm, M. Wu, K.R. Liu, Information forensics: an overview of the first decade, IEEE Access 1 (2013) 167–200.
[2] H. Farid, Photo Forensics, MIT Press, 2016.
[3] M. Kharrazi, H.T. Sencar, N. Memon, Blind source camera identification, Image Processing, 2004. ICIP'04. 2004 International Conference on, Vol. 1, IEEE, 2004, pp. 709–712.
[4] M. Kirchner, T. Gloe, Forensic camera model identification, Handbook of Digital Forensics of Multimedia Data and Devices, (2015) , pp. 329–374.
[5] L. Bondi, L. Baroffio, D. G“uera, P. Bestagini, E.J. Delp, S. Tubaro, First steps toward camera model identification with convolutional neural networks, IEEE Signal Processing Letters 24 (3) (2017) 259–263.
[6] E. Kee, M.K. Johnson, H. Farid, Digital image authentication from jpeg headers, IEEE Trans. Inform. Forensics and Security 6 (3–2) (2011) 1066–1075.
[7] A. Swaminathan, M. Wu, K.R. Liu, Nonintrusive component forensics of visual sensors using output images, IEEE Trans. Inform. Forensics and Security 2 (1) (2007) 91–106.
[8] H. Cao, A.C. Kot, Accurate detection of demosaicing regularity for digital image forensics, IEEE Trans. Inform. Forensics and Security 4 (4) (2009) 899–910.

[9] C. Chen, X. Zhao, M.C. Stamm, Detecting anti-forensic attacks on demosaicing-based camera model identification, Image Processing (ICIP), 2017 IEEE International Conference on, IEEE, 2017, pp. 1512–1516.
[10] T. Filler, J. Fridrich, M. Goljan, Using sensor pattern noise for camera model identification, Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on, IEEE, 2008, pp. 1296–1299.
[11] T.H. Thai, R. Cogranne, F. Retraint, Camera model identification based on the heteroscedastic noise model, IEEE Trans. Image Process. 23 (1) (2013) 250–263.
[12] F. Marra, G. Poggi, C. Sansone, L. Verdoliva, A study of co-occurrence based local features for camera model identification, Multimedia Tools Appl. 76 (4) (2017) 4765–4781.
[13] F. Marra, G. Poggi, C. Sansone, L. Verdoliva, Evaluation of residual-based local features for camera model identification, International Conference on Image Analysis and Processing, Springer, 2015, pp. 11–18.
[14] A. Tuama, F. Comby, M. Chaumont, Camera model identification with the use of deep convolutional neural networks, Information Forensics and Security (WIFS), 2016 IEEE International Workshop on, IEEE, 2016, pp. 1–6.
[15] 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. Visual Commun. Image Representation 49 (2017) 153–163.
[16] I. Amerini, T. Uricchio, L. Ballan, R. Caldelli, Localization of jpeg double compression through multi-domain convolutional neural networks, in: Proc. of IEEE CVPR Workshop on Media Forensics, Vol. 3 (2017) .
[17] M.H. Al Banna, M.A. Haider, M.J. Al Nahian, M.M. Islam, K.A. Taher, M.S. Kaiser, Camera model identification using deep cnn and transfer learning approach, 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), IEEE, 2019, pp. 626–630.
[18] A. Kuzin, A. Fattakhov, I. Kibardin, V.I. Iglovikov, R. Dautov, Camera model identification using convolutional neural networks, 2018 IEEE International Conference on Big Data (Big Data), IEEE, 2018, pp. 3107–3110.
[19] B. Wang, J. Yin, S. Tan, Y. Li, M. Li, Source camera model identification based on convolutional neural networks with local binary patterns coding, Signal Processing: Image Commun. 68 (2018) 162–168.
[20] B. Diallo, T. Urruty, P. Bourdon, C. Fernandez-Maloigne, Improving Robustness of Image Tampering Detection for Compression, (2019) , pp. 387–398.
[21] O. Çeliktutan, B. Sankur, I. Avcibas, Blind identification of source cell-phone model, IEEE Trans. Inform. Forensics and Security 3 (3) (2008) 553–566.
[22] T.H. Thai, R. Cogranne, F. Retraint, Camera model identification based on the heteroscedastic noise model, IEEE Trans. Image Process. 23 (1) (2014) 250–263.
[23] G. Xu, Y.Q. Shi, Camera model identification using local binary patterns, Multimedia and Expo (ICME), 2012 IEEE International Conference on, IEEE, 2012, pp. 392–397.
[24] B. Hosler, O. Mayer, B. Bayar, X. Zhao, C. Chen, J. Shackleford, M. Stamm, A video camera model identification system using deep learning and fusion, ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019, pp. 8271–8275.
[25] Y. Bengio, et al., Learning deep architectures for AI, Foundations and Trends® in Machine Learning 2 (1) (2009) 1–127.
[26] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.
[27] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.
[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Advances in neural information processing systems (2014) 2672–2680.
[29] M. Ye, J. Li, A.J. Ma, L. Zheng, P.C. Yuen, Dynamic graph co-matching for unsupervised video-based person re-identification, IEEE Trans. Image Process. 28 (6) (2019) 2976–2990.
[30] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.
[31] Y. LeCun, The mnist database of handwritten digits, (2018) 1998 http://yann. lecun. com/exdb/mnist.
[32] P. Baldi, Y. Chauvin, Neural networks for fingerprint recognition, Neural Comput. 5 (3) (1993) 402–418.
[33] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, D.J. Inman, 1d convolutional neural networks and applications: A survey, arXiv preprint arXiv:1905.03554.
[34] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems (2012) 1097–1105.
[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, Proceedings of the IEEE conference on computer vision and pattern recognition (2015) 1–9.
[36] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proceedings of the IEEE conference on computer vision and pattern recognition (2016) 770–778.
[37] L. Bondi, S. Lameri, D. G”uera, P. Bestagini, E.J. Delp, S. Tubaro, Tampering detection and localization through clustering of camera-based cnn features, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (2017) 1855–1864.
[38] B. Bayar, M.C. Stamm, Towards open set camera model identification using a deep learning framework, The 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2018.
[39] B. Bayar, M.C. Stamm, A deep learning approach to universal image manipulation detection using a new convolutional layer, Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, ACM, 2016, pp. 5–10.
[40] J. Bunk, J.H. Bappy, T.M. Mohammed, L. Nataraj, A. Flenner, B. Manjunath, S. Chandrasekaran, A.K. Roy-Chowdhury, L. Peterson, Detection and localization of image forgeries using resampling features and deep learning, Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on, IEEE, 2017, pp. 1881–1889.
[41] B. Bayar, M.C. Stamm, Design principles of convolutional neural networks for multimedia forensics, Electron. Imaging 2017 (7) (2017) 77–86.
[42] A. Alotaibi, A. Mahmood, Deep face liveness detection based on nonlinear diffusion using convolution neural network, Signal, Image Video Process. 11 (4) (2017) 713–720.
[43] L. Wen, H. Qi, S. Lyu, Contrast enhancement estimation for digital image forensics, ACM Trans. Multimedia Comput. Commun. Appl. (TOMM) 14 (2) (2018) 49.
[44] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, M. Nießner, Faceforensics: A large-scale video dataset for forgery detection in human faces, arXiv preprint arXiv:1803.09179.
[45] M. Huh, A. Liu, A. Owens, A.A. Efros, Fighting fake news: Image splice detection via learned self-consistency, arXiv preprint arXiv:1805.04096.
[46] D. Cozzolino, L. Verdoliva, Noiseprint: a cnn-based camera model fingerprint, IEEE Trans. Inform. Forensics Security 15 (2019) 144–159.
[47] D. Cozzolino, F. Marra, D. Gragnaniello, G. Poggi, L. Verdoliva, Combining prnu and noiseprint for robust and efficient device source identification, (2020) - arXiv preprint arXiv:2001.06440.
[48] R. Abbasi-Asl, B. Yu, Interpreting convolutional neural networks through compression, arXiv preprint arXiv:1711.02329.
[49] Z. Qin, F. Yu, C. Liu, X. Chen, How convolutional neural network see the world-a survey of convolutional neural network visualization methods, arXiv preprint arXiv:1804.11191.
[50] T. Gloe, R. B”ohme, The'dresden image database'for benchmarking digital image forensics, Proceedings of the 2010 ACM Symposium on Applied Computing, ACM, 2010 pp. http://forensics.inf.tu-dresden.de/ddimgdb/.
[51] L.v.d. Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (Nov) (2008) 2579–2605.
[52] A.K. Reyes, J.C. Caicedo, J.E. Camargo, Fine-tuning deep convolutional networks for plant recognition., CLEF (Working Notes) 1391.
[53] E. Al Hadhrami, M. Al Mufti, B. Taha, N. Werghi, Transfer learning with convolutional neural networks for moving target classification with micro-doppler radar spectrograms, 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), IEEE, 2018, pp. 148–154.
[54] J. Yang, Y.-Q. Shi, E.K. Wong, X. Kang, Jpeg steganalysis based on densenet, arXiv preprint arXiv:1711.09335.
[55] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556.