Camera model identification based on forensic traces extracted from homogeneous patches

Bennabhaktula, Guru Swaroop; Alegre, Enrique; Karastoyanova, Dimka; Azzopardi, George

Published in:
Expert systems with applications

DOI:
10.1016/j.eswa.2022.117769

Publication date:
2022

Expert Systems With Applications 206 (2022) 117769

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Camera model identification based on forensic traces extracted from homogeneous patches

Guru Swaroop Bennabhaktula [a,b,c,*], Enrique Alegre [b,c], Dimka Karastoyanova [a], George Azzopardi [a]

[a] Information Systems Group, Bernoulli Institute for Mathematics, Computer Science, and Artificial Intelligence, University of Groningen, The Netherlands
[b] Group for Vision and Intelligent Systems at the Department of Electrical, Systems, and Automation, University of León, Spain
[c] Researcher at INCIBE (Spanish National Cybersecurity Institute), León, Spain

ARTICLE INFO

Keywords:
Sensor pattern noise
Camera model identification
Digital image forensics

ABSTRACT

A crucial challenge in digital image forensics is to identify the source camera model used to generate given images. This is of prime importance, especially for Law Enforcement Agencies in their investigations of Child Sexual Abuse Material found in darknets or seized storage devices.

In this work, we address this challenge by proposing a solution that is characterized by two main contributions. It relies on the extraction of rather small homogeneous regions that we extract very efficiently from the integral image, and on a hierarchical classification approach with convolutional neural networks as the underlying models. We rely on homogeneous regions as they contain camera traces that are less distorted than regions with high-level scene content. The hierarchical approach that we propose is important for scaling up and making minimal modifications when new cameras are added. Furthermore, this scheme performs better than the traditional single classifier approach. By means of thorough experimentation on the publicly available Dresden data set, we achieve an accuracy of 99.01% with 5-fold cross-validation on the 'natural' subset of this data set. To the best of our knowledge, this is the best result ever reported for Dresden data set.

1. Introduction

An intriguing question in digital image forensics is that given an image, would it be possible to identify the source camera which was used to capture it? This question is of prime importance to Law Enforcement Agencies (LEAs) when investigating digital image and video information. The last two decades have seen rapid growth in the usage of the Internet, and unfortunately, this has also led to an increase in the circulation of illicit content of minors especially in darknets. As part of their investigations, LEAs aim to identify the source of such content because knowing the source camera used to capture such content will help them gain additional intelligence in building stronger cases against suspected offenders. Source Camera Identification (SCI) is an important low-level problem in the field of computer vision and plays a crucial role in the forensic investigations of digital images. The methods of SCI can also be used in a few related applications such as image forgery detection (Bondi et al., 2017; Cozzolino & Verdoliva, 2019; Li et al., 2014) for fighting fake news, and image integrity verification (Li et al., 2009) for verifying digital evidence presented to the courts of law, among others. Our work is part of the EU-funded 4NSEEK project to develop forensic tools for LEAs that will help them further to fight against child sexual abuse.

In this work, we aim to identify the source camera model of an image by examining only the pixel values. Clues for identifying the source camera can also be gathered from the metadata that comes in the format of the Exchangeable Image File (EXIF) header. However, this information can be modified when the image is re-saved in a different format or when it is re-compressed. Moreover, any tampering with the information in the EXIF headers cannot be detected. This issue makes the information in the EXIF headers unreliable for the task of SCI, and we, therefore, avoid using them. In contrast, when the pixel values are altered in an image, research has shown that it is possible to detect such image tampering. For example, it is possible to identify some common image processing operations such as median filtering (Kang et al., 2013; Kirchner & Fridrich, 2010), Gaussian filtering (Fan et al., 2015; Kang & Wei, 2008), and JPEG image compression (Kang & Wei, 2008; Luo et al., 2010; Wang & Zhang, 2016), among others. Therefore, relying on the pixel values makes SCI more robust when compared to a system that depends on image meta-data.

* Correspondence to: 351, Escuela de Ingenierías (Industrial, Informática, Aeronáutica), Campus de Vegazana, Universidad de León, León, 24071, Spain.
E-mail addresses: g.s.bennabhaktula@rug.nl, gben@unileon.es (G.S. Bennabhaktula), enrique.alegre@unileon.es (E. Alegre), d.karastoyanova@rug.nl (D. Karastoyanova), g.azzopardi@rug.nl (G. Azzopardi).

https://doi.org/10.1016/j.eswa.2022.117769
Received 29 January 2022; Received in revised form 11 May 2022; Accepted 3 June 2022
Available online 10 June 2022

Another approach for SCI involves the examination of digital watermarks. While this is a very effective approach, in practice, such watermarks are not embedded into images by most consumer-grade cameras. Digital watermarking is mainly used for identifying copyright infringements in commercial digital media, but it has limited applicability for SCI.

With smartphones becoming a common commodity, and image editing tools gaining more popularity, it is getting even more challenging to identify cameras from images. Operations such as image resizing, filtering, and compression, among others, not only tamper with the sensor pattern noise but also leave behind operation-specific traces that are embedded into the image. While the presence of these additional traces makes SCI more difficult, it is still possible to detect their presence (Fan et al., 2015; Kang et al., 2013; Kang & Wei, 2008; Kirchner & Fridrich, 2010; Luo et al., 2010; Wang & Zhang, 2016). Detection of image forgery is an important forensic task, but falls beyond the scope of this work, as we consider only unedited images for this study.

It is necessary here to clarify what is meant by camera-model and camera-device identification. These are two main approaches for SCI. In model identification, the task is to identify the specific camera model that was used to capture an image (for example, an iPhone 10, an iPhone 11, etc.). We refer to the manufactured instances of the same camera model as devices. Device identification is more challenging and is also of high interest to forensic investigators. A prerequisite for an approach with good performance for device identification is a method that performs very well first and foremost for model identification. In this work, we focus on camera model identification.

The key contributions of our work are threefold and can be summarized as follows. Firstly, we observed that during the extraction of camera features from an image, it is important to ensure that the features correspond to the processing noise and not the scene details. This is more essential when employing neural networks for feature extraction. Therefore, in order to prevent any bias due to the involved scenes, we decided to work only with homogeneous patches. Most of the images contain such regions, and we propose a method to identify and extract homogeneous patches from an image. The proposed method for patch selection ensures that the learning process is not influenced by the scene content in the images. Secondly, we propose a systematic hierarchical scheme for data balancing which is more suited to the problem of camera identification. Finally, we propose a hierarchical approach for classification, where we perform brand classification in the first level followed by the classification of camera models in the second level (refer to Fig. 1). We empirically show that this hierarchical scheme is more effective than non-hierarchical classification for SCI, among other advantages. We also share the source code[1] for further dissemination of our approach and experiments.

The rest of the paper is organized as follows: The following section describes related methods and summarizes the state-of-the-art for SCI. The proposed methodology is described in Section 3. Section 4 elucidates the systematic experiments performed along with the obtained results. The discussion on our experiments is presented in Section 5. Finally, the main conclusions are drawn in Section 6.

2. Related works

Ever since digital cameras gained popularity, SCI has become an important research problem for forensic analysis of digital media. Camera identification from images becomes possible due to artefacts introduced during the generation of a digital image. Fig. 2 captures a high-level pipeline for image generation inside a digital camera, which involves a sequence of hardware and software processing steps. The light rays from the scene enter the camera through a set of lenses, followed by an anti-aliasing filter, colour filter arrays (CFA), and finally the imaging sensor. The sensor converts the analog signal to digital and produces a RAW image. This image is further enhanced and typically involves processing steps such as demosaicing, gamma correction, compression, and other operations before the generation of the final image.

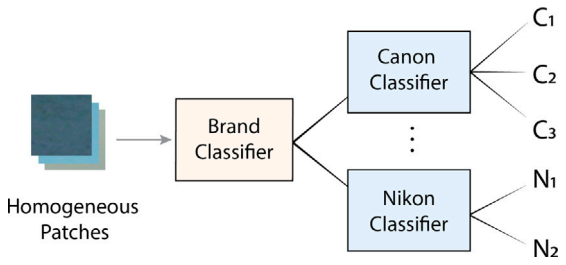

Fig. 1. The proposed hierarchical classification of homogeneous patches extracted from a single image. At the first level, the patches are classified to determine the camera brand. A majority voting scheme determines the specific model classifier as elucidated in Section 2. The model-level classifier is then used to classify all the homogeneous patches to determine the specific camera model. For example, the Canon model classifies each patch to either of the three camera models (C_1, C_2, or C_3). A majority voting scheme is once again employed to determine the source camera model for the given image.

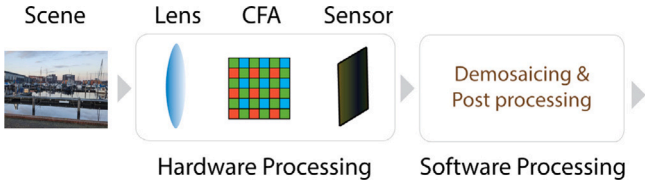

Fig. 2. A typical image generation pipeline inside a digital camera. It consists of hardware followed by software processing steps. The light rays from the scene enter the camera through a set of lenses, followed by anti-aliasing filters, colour filter arrays (CFA), and finally the imaging sensor. Once the sensor converts the analog signal to digital the software processing steps are performed. This generally involves demosaicing, image compression, and other post-processing steps before the generation of the final output image.

The exact implementation of these processing steps varies for each camera model and leaves behind a unique processing trace during the generation of a digital image. Furthermore, two camera devices of the same camera model also generate a slightly different processing noise. This intra-model variance is due to the unique random noise generated by each imaging sensor in every device. Though challenging to extract, the noise generated by the sensor makes camera device identification possible. Note that, we interchangeably use the terms camera traces, artefacts, and processing noise as they refer to the same thing.

The final image produced can therefore be considered as a combination of scene details and the in-camera processing noise. An overview of the classification of this noise is shown in Fig. 3. The camera noise consists of sensor noise, CFA artefacts, compression artefacts, and image post-processing artefacts, among others. Besides sensor noise, all other components are deterministic and when combined can identify the specific camera model. Lukas et al. (2006) further classified the sensor noise into shot noise and pattern noise. Shot noise is an additive random component and can be removed by frame averaging. The pattern noise is a multiplicative deterministic component that is unique to each camera device. The pattern noise further consists of fixed pattern noise (FPN) and Photoresponse non-uniform noise (PRNU). FPN is an additive component caused due to dark currents, i.e. when the sensor is not exposed to any light. PRNU is a multiplicative noise caused due to the different sensitivities of pixels when illumined with a light source. This component is primarily responsible for device identification. For an elaborate classification of sensor noise we refer the reader to Lukas et al. (2006).

One of the earliest experiments was conducted by Kurosawa et al. (1999), where the FPN generated by the dark frames were studied. The

[1] https://github.com/bgswaroop/scd-images

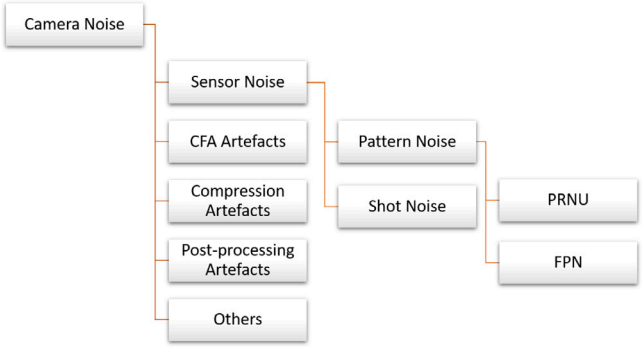

Fig. 3. Categorization of camera processing noise which is embedded in every image. There are several sources of camera noise, namely sensor noise, CFA artefacts, compression artefacts, and post-processing artefacts, among others. The sensor noise can be further classified into shot noise, which is an additive random component, and pattern noise, which is a multiplicative deterministic component. The pattern noise is further classified into FPN and PRNU based on sensor response to light.

authors examined 9 camera devices from 4 models and showed that the FPN is unique to every camera device. This uniqueness is attributed to the imperfections in the silicon wafer fabrication process. As the generation of the dark frame requires access to the physical camera device, Lukas et al. (2006) proposed a method to directly extract the PRNU noise from natural images. In their work, a high-pass filter was used to extract the camera fingerprints, which were in turn compared using correlation measures to determine their source identity. Further methods based on PRNU were presented by Chen et al. (2008), Li (2010), Lin and Li (2015), Rosenfeld and Sencar (2009).

Methods for camera identification were also proposed that aimed to extract the artefacts introduced by a specific set of processing operations during the image generation inside a camera. Kharrazi et al. (2004) extracted 34 handcrafted features and used an SVM for the classification of images. These handcrafted features were designed to target a specific set of forensic traces. Several approaches were also proposed that specifically target the demosaicing artefacts, which are related to the colour-filter arrays (Bayram et al., 2005; Cao & Kot, 2009; Chen & Stamm, 2015; Swaminathan et al., 2007). Some approaches use non-trainable features (Xu & Shi, 2012) which cannot be adapted for newer camera models, while other approaches consider a particular camera noise model (Thai et al., 2013). All these methods aim to extract features by assuming that the artefacts were mainly introduced by a subset of processing steps, and such methods are referred to as the ones based on a closed set of forensic traces. Such methods require expertise and effort to design feature descriptors for a specific processing step. The drawback of these approaches is that they might miss out on some non-intuitive forensic traces, for example, those that arise because of a combination of such processing steps. In contrast, we do not make any assumption on the specific source of artefacts and therefore consider all possible forensic traces. Methods that do not make any assumptions about the source of forensic traces, like ours, are referred to as the ones based on the open set of forensic traces. These methods are more promising in extracting a wide variety of camera fingerprints.

Recently, methods based on deep learning were presented to address a variety of image forensic tasks. These methods help in overcoming the problems associated with feature engineering, which is more suited for the extraction of open set of forensic traces. Some image forensic tasks that were recently approached using deep learning include the detection of image in-painting (Wang et al., 2019; Zhu et al., 2018), median filtering (Chen et al., 2015; Tang et al., 2018), resizing (Bunk et al., 2017), and JPEG compression (Barni et al., 2017, 2016), and suchlike. Though these methods are based on deep learning they are characterized by a closed set of forensic traces. We are, however, interested in methods using deep learning and based on an open set of forensic traces (Bayar & Stamm, 2016; Mayer et al., 2018).

Bondi et al. (2016) showed that Convolutional Neural Networks (ConvNets) can be used to extract forensic traces from images to identify source camera models. They performed experiments on the publicly available Dresden data set (Gloe & Böhme, 2010) and showed that ConvNet based methods achieve state-of-the-art performance when compared to other approaches. A similar study using ConvNets was also performed by Tuama et al. (2016). Bayar and Stamm (2018b) proposed a system based on an open set of forensic traces, which indicates whether an image was captured by a camera model used during training, or from the collection of camera models outside the training set.

Every image generated by a digital camera consists of scene details along with camera processing noise, which is embedded into the image. This raises an important question. Can we extract camera traces from an image ignoring the scene details? This is a challenge, especially in the context of ConvNets, where we allow the model to learn its own feature extractor. In order to prevent the ConvNet from learning high-level features, some methods have been proposed (Bayar & Stamm, 2018a; Timmerman et al., 2020) to suppress the scene content as a pre-processing step. These methods, however, do not fully prevent the ConvNet from learning the high-level scene details. We address this issue by choosing to extract patches from image regions that are homogeneous. Further details of our patch filtering and selection strategy are presented in Section 3.

We further note that training a hierarchy of classifiers is more suited than training a single classifier for SCI. In machine learning, training a highly effective single classifier becomes a challenge when there are multiple camera devices. To overcome these challenges, we propose a hierarchical classification scheme as shown in Fig. 1. A more detailed account of this scheme is presented in the following section.

3. Methodology

This section describes our proposed approach, where we use hierarchical classification based on deep learning for source camera model identification. We also describe in detail the data preparation steps necessary for our pipeline.

3.1. Overview

A high-level overview of the proposed methodology is presented in Fig. 4. The pipeline highlights three major steps. Firstly, we determine the homogeneous regions in an image. This is done by dividing the input image into overlapping blocks of size 128 × 128 pixels. These blocks are then subjected to the proposed homogeneity criteria, in order to filter out non-homogeneous patches. In the second step, a pre-trained ConvNet is used to determine the camera brand for each of the homogeneous patches. These predictions are then combined using a majority vote to determine the camera brand. Finally, based on the determined brand the corresponding model-level classifier is used to determine the specific camera model for each of the homogeneous patches. The patch-level predictions are then combined by performing a majority vote which determines the source camera model for the given input image. We begin by describing the methodology for patch selection.

3.2. Homogeneous patch selection and extraction

As described earlier, every image consists of both scene details and camera processing noise. Image regions can be classified into regions with high-level scene details (that is, regions containing edges, high texture details, etc.) and low-level scene details (that is, regions that are homogeneous, where a group of neighbouring pixels have almost similar intensity values). The latter type of region contains camera processing noise that is least distorted by high-level scene details. We

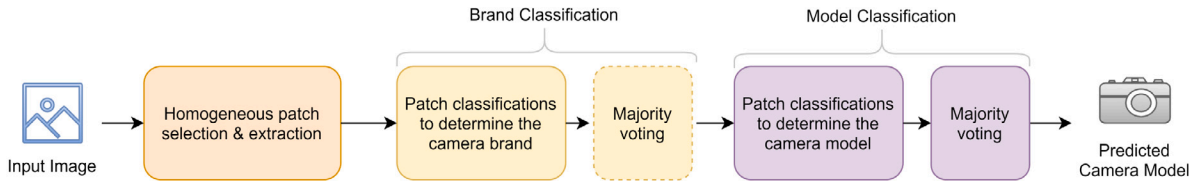

Fig. 4. The flow chart depicts a high-level pipeline of the proposed methodology, highlighting the three major steps. Firstly, the input image is divided into overlapping blocks of size 128 × 128 pixels, which are further filtered based on our proposed homogeneity criteria, resulting in homogeneous patches. Secondly, a ConvNet is used to determine the camera brand for each patch, which is followed by a majority vote on the predictions. Finally, based on the brand-level prediction the corresponding model-level classifier is used to determine the camera model for each homogeneous patch. The model-level predictions are then combined by performing a majority vote to determine the source camera model for the given input image.

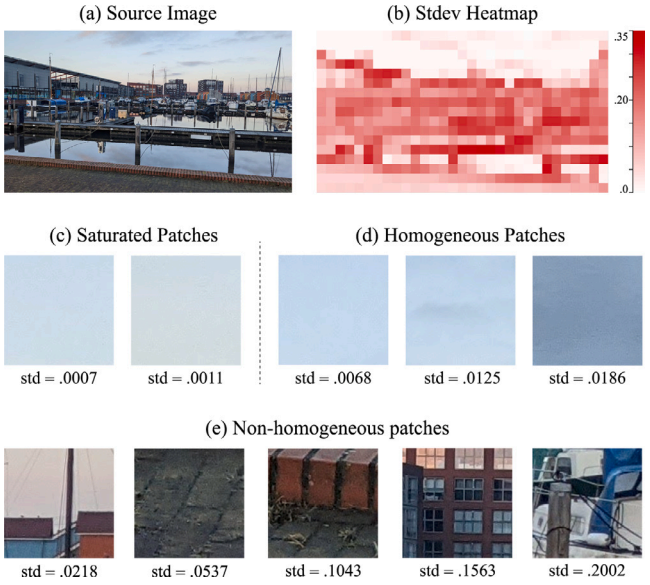

Fig. 5. An illustration of patch selection using standard deviation. (a) A sample image of size 2176 × 3968 pixels. (b) The heat map of standard deviations for non-overlapping patches of size 128 × 128 pixels from the red colour channel. Examples of (c) saturated patches (std < 0.005), (d) homogeneous patches (0.005 ≤ std ≤ 0.02), and (e) non-homogeneous patches (std > 0.02). Note that, for the sake of visual clarity, in (b) we only show non-overlapping tiles of size 128 × 128 pixels. In our experiments, however, we use a stride of 0.25 times the block size to sample image patches. Furthermore, this is done for all three colour channels and the std criterion in (c), (d), and (e) must hold for all three colour channels separately.

now describe the methodology to extract homogeneous patches from such image regions.

In order to extract patches from an image we tile the input image with blocks of size 128 × 128 pixels. Using a larger block size, such as 256 × 256 pixels, would result in fewer blocks, and the chances of a block being homogeneous are reduced. Using a smaller block size, such as 32 × 32 pixels, or 64 × 64 pixels, would result in more image patches that are homogeneous, however, extracting camera-specific features from smaller image patches is more difficult. To account for this trade-off, we sample the input image using a tile of size 128 × 128 pixels, and with a stride of 0.25 times the block size. This effectively increases the number of extracted image regions substantially, while retaining a sufficiently large image block.

Each block is then examined for homogeneity by determining the standard deviation of its pixel values. Since there are three colour channels, we determine three standard deviations, one for each channel. Fig. 5 depicts a sample input image in part (a) and the corresponding standard deviations of each image block are shown in part (b). Note that for the sake of clarity, we show the standard deviations for the red colour channel and use non-overlapping patches with a stride of 1 block. The standard deviation values of the homogeneous regions are smaller than the others. We subjectively determined a high threshold with a value of 0.02 to exclude non-homogeneous patches with significant scene level details. This threshold was determined after manually examining hundreds of extracted patches. We also set a low threshold of 0.005 to eliminate saturated patches. A saturated patch is one that has true loss of data due to having many pixel values clipped to the maximum intensity (i.e. 255), thereby overriding the camera noise. This occurs when there is very high incoming light intensity to the sensor or when certain operations (e.g. denoising) are used by editing software. In the latter case, such operations may result in a few regions where the difference between the neighbouring pixels is close to zero. Finally, for a patch to be considered homogeneous, all three standard deviations must independently adhere to the threshold limits defined above. Sample image patches for saturated, homogeneous, and non-homogeneous are shown in Fig. 5(c–e).

In our experiments, described in Section 4, we determine the number of homogeneous patches to be extracted from each image as follows. Suppose a given number of p patches need to be extracted. If the number of homogeneous patches for an image is more than p, then we uniformly sample p patches so that they are evenly distributed among the homogeneous regions across the whole image. On the other hand, when there are fewer than p patches per image, we choose all the homogeneous patches and choose the remaining patches from the saturated and non-homogeneous patches with the lowest standard deviations. Thus, we ensure that homogeneous regions are given priority during the patch extraction. Finally, we subtract the per-colour-channel mean from the respective colour channel of the input image. This is done to minimize the colour information being inadvertently learnt for classification. Fig. 6 illustrates this pre-processing step and demonstrates that the effect of brightness in the input patches is reduced by this operation. Thereby, the classifier focuses on the noise, which is our matter of interest. Fig. 7 depicts the ratio of the average number of homogeneous to non-homogeneous to saturated patches for each camera model in the Dresden data set. Further details concerning the camera models are presented in Table 1.

The patch selection approach that we propose enable us to process images of different dimensions. In contrast, methods that use a full image (Bennabhaktula et al., 2020) may have to do resizing first, which can lead to unintended modifications to the forensic traces. Moreover, our approach of choosing patches gives the ability to choose regions that are less affected by the scene content. Next, we describe the details of the ConvNet architecture used for patch classification.

3.3. Patch classification

For each patch to be classified into its source camera brand and source camera model, we use a ConvNet for feature extraction that gives input to a fully connected neural network for classification. The ConvNet architecture that we use is inspired by the MISLNet, which was proposed by Bayar and Stamm (2018a). The overall design of the proposed architecture is shown in Fig. 8. The ConvNet architecture is divided into seven blocks. The dimensions of the input layer, 128 × 128 × 3, correspond to the patch size used in our experiments. This is followed by four convolutional blocks and three fully connected

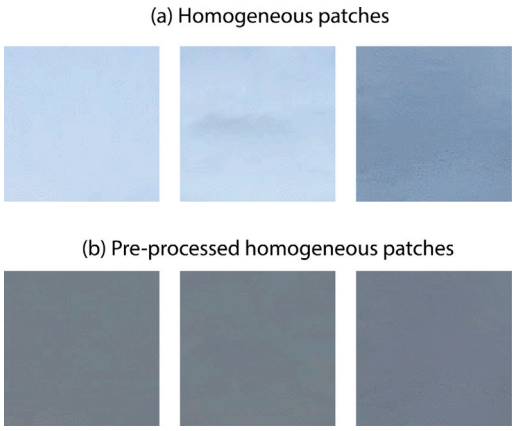

Fig. 6. An illustration of the pre-processing step. (a) Examples of homogeneous patches and (b) the corresponding pre-processed patches after per-channel-mean subtraction. The resulting images are in the range $[-1, +1]$. For visualization purposes the resulting images are shifted to the range $[0, 1]$ by adding 1 and dividing by 2.

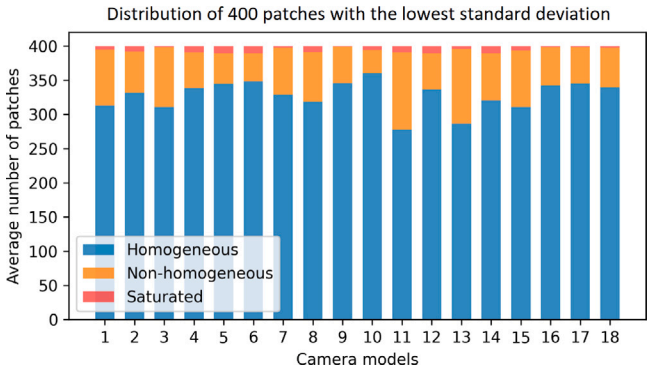

Fig. 7. A depiction of the ratio between the average number of homogeneous, non-homogeneous, and saturated patches for the 18 camera models in the Dresden data set. This plot is generated by considering the 400 patches (of size 128 × 128 pixels) with the lowest standard deviations from each image.

Table 1
List of camera models considered for our experiments from the Dresden data set.

Sr. no.	Camera model name	# Devices	Total # images
1	Canon_Ixus70	3	522
2	Casio_EX-Z150	5	850
3	FujiFilm_FinePixJ50	3	630
4	Kodak_M1063	5	2314
5	Nikon_CoolPixS710	5	846
6	Nikon_D200	2	673
7	Nikon_D70	4	676
8	Olympus_mju_1050SW	5	965
9	Panasonic_DMC-FZ50	3	931
10	Pentax_OptioA40	4	638
11	Praktica_DCZ5.9	5	942
12	Ricoh_GX100	5	854
13	Rollei_RCP-7325XS	3	544
14	Samsung_L74wide	3	641
15	Samsung_NV15	3	599
16	Sony_DSC-H50	2	541
17	Sony_DSC-T77	4	906
18	Sony_DSC-W170	2	405

blocks. Bayar and Stamm (2018a) used a constrained convolutional layer in the design of the MISLNet to suppress the high-level scene details. As our approach relies on the homogeneous regions in the images, we skip the constrained convolutional layer. Note that we make use of all three input colour channels (RGB) to extract better forensic traces instead of relying on monochrome images as done by Bayar and Stamm (2018a).

As shown in Fig. 8, the first block consists of 96 convolutional filters each of size $7 \times 7 \times 3$, which are configured to use a stride of 2×2 and a valid zero padding. This convolutional layer is followed by a batch normalization layer (Ioffe & Szegedy, 2015), which in turn is followed by a Rectified Linear Unit (ReLU) activation to introduce non-linearity.

Unlike MISLNet, we use ReLU activations for the convolutional layers instead of the tanh activations. The activation layer is followed by a max-pooling layer of size 2×2, which effectively reduces the spatial dimensions by a factor of 4. We follow a similar pattern for blocks 1–4, where each block consists of a convolution layer, batch normalization, and ReLU transformation followed by a max-pooling layer. The exact details of these blocks are given in Fig. 8. In the proposed ConvNet, blocks 1–4 constitute the feature extraction part of the network, while blocks 5–7 represent the classifier.

The output of the feature extractor block is flattened to obtain a 2048−element feature vector. We use a fully connected neural network with three layers to classify these features into the desired camera class. In the case of brand classification, the output of the classifier is the camera brand (for example Sony, Canon, Nikon, and so on), while in the case of model classification the number of units in the output layer is set to match the number of camera models (for example Nikon_CoolPixS710, Nikon_D200, or Nikon_D70, among others). In order to avoid overfitting and achieve regularization, we use a dropout layer with a dropout factor of 0.3.

The learning parameters, loss function, and data set used for the experiments are described in Section 4.

3.4. Majority voting

An image may contain many homogeneous regions. Based on the patch selection scheme numerous homogeneous patches are extracted from the image. For each patch in a given image, we apply the concerned ConvNet model which gives us a label. All such patch-level predictions are then combined by taking a majority vote to determine the label for the given test image. As demonstrated in Section 4 this scheme of majority voting significantly decreases the error rate for image-level prediction in comparison to patch-level predictions.

3.5. Hierarchical classification scheme

To classify an input image to its source camera model, we propose to follow a two-level hierarchical classification approach as shown in Fig. 1. The first level is concerned with brand classification while the second one is for model classification. The training of these classifiers at both brand and model level are performed independently of each other. The trained classifiers can then be used in a hierarchical fashion to perform predictions on images. In order to predict the source camera model for a single image, firstly, homogeneous patches are extracted. Secondly, the brand level classifier determines the brand for each homogeneous patch. In case the majority vote on the predictions corresponds to only one camera model, the brand classification trivially determines the corresponding model-level classification. On the other hand, if the predicted brand corresponds to multiple camera models, then a second-level classifier is used to determine the class labels for the concerned patches. This step is followed by a majority vote to determine the predicted camera model for the given image.

The hierarchical scheme takes the advantage of training with fewer camera devices for each classifier. Since each classifier in the hierarchical scheme accounts for only a limited set of camera models, the training time to learn each classifier is reduced considerably. This also makes it possible to independently train each classifier in parallel. Moreover, with this modular approach, issues with any brand-specific classifier do not impact other brand-specific classifiers, and the classification scheme can be extended to include additional camera devices without having to retrain all classifiers.

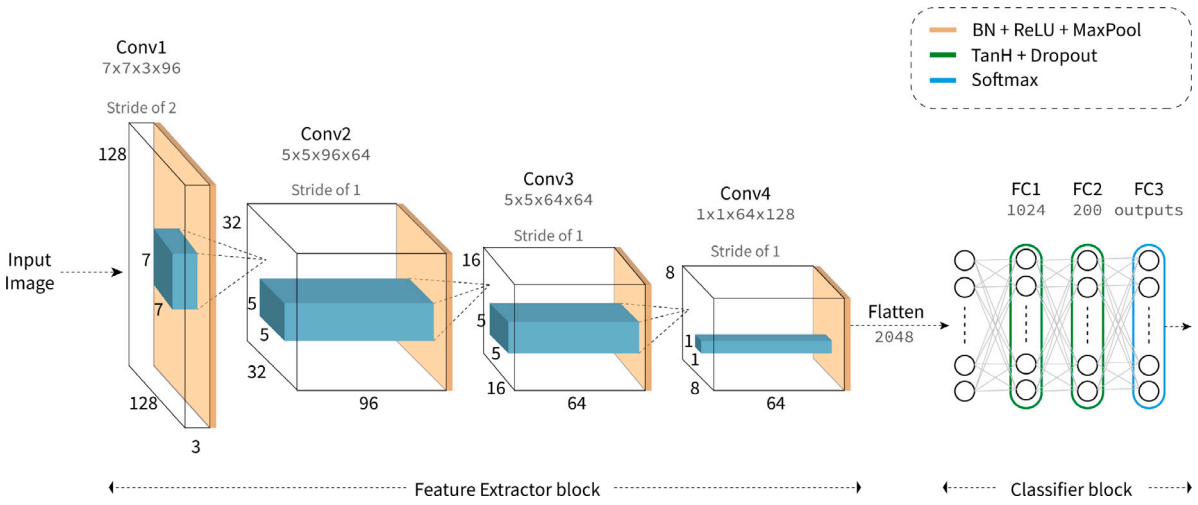

Fig. 8. The proposed ConvNet architecture consisting of convolutional (Conv) and fully connected (FC) layers. The first four Conv blocks represent the feature extraction part of the network. Each Conv layer is followed by a batch normalization (BN), ReLU activations and a max-pooling (MaxPool) layer with a filter of size 2×2 pixels along with a stride of 2×2. The convolutional layers transform the input image of size $128 \times 128 \times 3$ pixels into a feature vector of size 2048. The final three fully connected layers classify the extracted feature vector into a source camera class label. The number of outputs in the FC3 layer is determined upon the sub-problem (brand or model classification) and the type of available devices. The blocks are not drawn to scale and, therefore, the labelled dimensions must be used for the exact details.

4. Experiments

This section describes the data set used, along with the experiments performed to train and test the proposed hierarchical approach.

4.1. Data set

To conduct our experiments we used the publicly available benchmark Dresden data set (Gloe & Böhme, 2010). It consists of more than 14,000 images captured by 66 camera devices, 18 camera models, and 13 major camera brands. The images are divided into several subsets, namely 'JPEG', 'natural', 'flat-field', and 'dark-field', among others. We conducted our experiments on the 'natural' subset of the data set, which contains 74 camera devices that were used to capture 84 different indoor and outdoor scenes. This high number of diverse scenes makes the 'natural' subset the most challenging and realistic one, unlike the other subsets which contain images from at most two different scenes. The methods that we compare our results with also use the same 'natural' subset of Dresden. Of the 74 camera devices, 8 devices were present with a single instance of a camera model. In order to perform a fair evaluation for camera model identification, we decided to consider devices that have at least two instances from the same camera model. This requirement allows us to keep one instance aside for testing while using the rest for training. We, therefore, conduct experiments on the remaining 66 devices. Together, they represent 18 camera models, whose distribution is listed in Table 1. As suggested by Kirchner and Gloe (2015), we combined the models Nikon_D70 and Nikon_D70s into a single camera model (Nikon_D70) as they correspond to the same model with a different lens.

The natural images can be further classified into 84 sub-categories, based on the scene content. In order to perform a reliable evaluation, Bondi et al. (2016) and Rafi, Wu, and Hasan (2020) used a subset of scenes for testing, while the rest were used for training and validation. That approach ensured that the model's accuracy is scene independent. In our experiments we do not need this setting as the selection of homogeneous patches ensures that the scene-specific details are omitted.

4.2. Data balancing

In order to train a robust machine learning model it is necessary to ensure that the data imbalance is appropriately handled during the training phase. This becomes essential for our study as the distribution of the number of images per camera model varies significantly between classes. We refer the reader to Table 1 for the exact details of image distribution per camera model. Traditional data balancing techniques such as over-sampling and under-sampling of images from camera models may not result in a representative data set. More specifically, over-sampling causes repetition in the training data which can lead to model overfitting. Random under-sampling of data, on the other hand, can lead to missing out on potentially useful data and, therefore, we propose a systematic top-down scheme for patch balancing.

Consider the data imbalance problem for brand classification. Let the total number of image patches that we would like to train be denoted by k. The goal is to distribute these k patches evenly across the training data set. The value of k is an estimate that will drive the rest of the patch balancing algorithm. Suppose that brand classification is an n_b-class classification problem, where n_b represents the total number of camera brands. Without loss of generality, let k_b represents the number of patches to be sampled from any specific brand b. In order to construct a balanced data set, k_b can be determined as:

$$k_b = \left\lfloor \frac{k}{n_b} \right\rceil \tag{1}$$

where k, k_b, $n_b \in \mathbb{N}$, and $\lfloor \cdot \rceil$ denotes the standard rounding function in Python.[2] Let n_m denotes the number of models representing the brand b. Without loss of generality, let k_m denotes the number of patches to be sampled from a particular model m. In order to keep the number of patches the same across different models of the same brand, k_m can be set to:

$$k_m = \left\lfloor \frac{k}{n_m \cdot n_b} \right\rceil \tag{2}$$

where k_m, $n_m \in \mathbb{N}$. We continue this process to determine the number of patches to be sampled at the device-level k_d, for all the devices n_d of model m. The value of k_d is given by:

$$k_d = \left\lfloor \frac{k}{n_d \cdot n_m \cdot n_b} \right\rceil \tag{3}$$

[2] The in-built Python function *round*() rounds to the nearest even number for half-integer values.

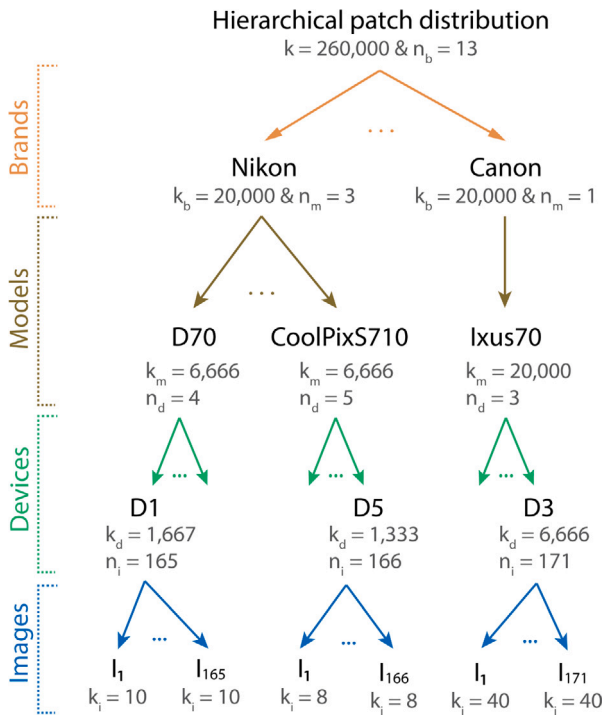

Fig. 9. An illustration of the patch balancing algorithm, where the patches are balanced at each level of the hierarchy. At the first level, an initial estimate of $k = 260,000$ patches is distributed equally among the 13 camera brands, resulting in $k_b = 20,000$ patches per brand. Thereafter, in the second level, 20,000 patches are evenly distributed among the 3 Nikon models, resulting in $k_m = 6666$ patches per model, and likewise for Canon models. In the same fashion, patches are equally distributed at device and image levels.

where k_d, $n_d \in \mathbb{N}$. Finally, we determine the number of patches to be sampled from each image i, captured by the device d as:

$$k_i = \left\lfloor \frac{k}{n_i \cdot n_d \cdot n_m \cdot n_b} \right\rfloor \tag{4}$$

where n_i represents the number of images in the data set belonging to the device d. Thus, for an initial estimate of k, the number of patches that need to be extracted from an image i is determined by k_i, where d, m, and b correspond to the device, model, and brand of the image, respectively. It should be noted that these k_i patches are chosen by following the patch selection algorithm presented in Section 3.2.

The proposed method for patch selection ensures that the patches are evenly distributed at all levels of the hierarchy. Fig. 9 illustrates the patch balancing algorithm. Based on the choice of k, there might be very minor differences in the number of patches between different classes which is caused due to the rounding function. For our setting, this minor difference in class distribution is acceptable and is not considered a data imbalance.

4.3. Data set split

In line with Kirchner and Gloe (2015) we perform a 5-fold cross-validation, where we leave out one device per model for the test and use the remaining for training. For each fold, the device to be considered as a test device is determined in a round-robin fashion. As shown in Table 1, the number of camera devices per model is not the same. The maximum number of devices per model is 5, and for such models, each device appears exactly once in the test set across the 5 folds. For models where there are 4 or fewer devices, we repeat the devices in the test set following a round-robin approach across the 5 folds. This strategy of cross-validation accounts for the data set bias and provides us with reliable results. We report the results for all the folds, along with the global average.

Table 2
Classification accuracy for the proposed approach on classifying 200 homogeneous patches from each image in the test set.

Classification	fold 1	fold 2	fold 3	fold 4	fold 5	Average
Brands	0.995	0.992	0.994	0.994	0.995	0.9941 ± 0.0010
Nikon	0.997	0.997	0.999	0.997	0.997	0.9973 ± 0.0006
Samsung	1.000	0.995	0.998	1.000	0.995	0.9976 ± 0.0022
Sony	0.970	0.989	0.965	0.980	0.972	0.9751 ± 0.0085
Hierarchical	0.991	0.991	0.989	0.990	0.989	0.9899 ± 0.0007

4.4. Brand classification

As a first step, we perform camera brand classification. The data set that we consider consists of $n_c = 13$ camera brands. Firstly, we balance the patches by setting $\hat{k} = 260,000$, which gives us 20,000 patches per class.

The training loss is determined by computing the categorical cross-entropy between the target and the predicted output of the network. Let $f(\mathbf{x_i}; \mathbf{w})$ represents the network with weights $\mathbf{w}$, input example $\mathbf{x_i}$, and the corresponding softmax output as $\mathbf{y_i}$. The categorical cross-entropy is a multi-class logistic loss function which is defined as:

$$\mathcal{L}(f(\mathbf{x_i}; \mathbf{w}); \mathbf{t}_i) = -\log(\mathbf{y}_i) \cdot \mathbf{t}_i \tag{5}$$

$$\mathcal{J}(\mathbf{w}) = \frac{1}{s} \sum_{i=1}^{s} l_i \tag{6}$$

where $\mathcal{L}$ denotes the loss function, $\mathbf{t_i} \in \{0, 1\}^{n_c}$ denotes the one-hot encoded target vector for the input $\mathbf{x_i}$, and $\mathcal{J}$ is the empirical loss for the mini-batch of size s. In our experiments, we set the batch size $s = 512$.

We use the technique of stochastic gradient descent (SGD) for the optimization of the model parameters with a learning rate of 0.1 and momentum of 0.8. We were able to set such a high initial learning rate because of the batch normalization layers in the ConvNet. Furthermore, an exponential learning rate decay is used with a multiplicative factor of 0.9 for achieving model convergence. The learning rate is decayed at the end of every epoch. In order to avoid overfitting, we used l2-regularization by setting the weight decay factor to 0.0005. With these settings, the model was trained until convergence. We perform early stopping when the loss converges, and it does not fluctuate more than 0.02 for 5 consecutive epochs. Fig. 10 shows the convergence in the loss that was achieved with these hyper-parameters. The best epoch is then determined with the maximum validation accuracy for each fold and used during the evaluation phase of the brand classifier. The proposed ConvNet model consists of $2,585,149$ trainable parameters (for $n_c = 13$).

Fig. 11 shows the confusion matrix for the classification of 13 camera brands. We show the confusion matrix only for the first fold (refer to Fig. 11) and report a summary of the remaining folds in the first row of Table 2. Note that the results in Table 2 correspond to image-level classification accuracy and not at the level of patches. During the training phase, the training data set is balanced in a hierarchical fashion as shown in Fig. 9. This is, however, not the case during model evaluation. For a given test image, a set of 200 homogeneous patches is extracted. The trained model is then used to predict class labels for all 200 homogeneous patches. Finally, a majority vote is taken to determine the predicted class label for the given image. Taking the majority vote on patch-level predictions decreased the average error rate by almost 50 percent for image-level predictions in comparison to individual patch-level predictions. Since the test data is skewed between classes, in addition to accuracy we also report the macro F1 score. It is defined as the empirical average of class-wise F1 scores, with the F1 score for each class defined as:

$$\text{F1} = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}} \tag{7}$$

where TP, FP, and FN stand for true positives, false positives, and false negatives, respectively. The accuracy values reported in our experiments have been determined using the following equation:

$$\text{Accuracy} = \frac{\#\text{ correct predictions}}{\text{total }\#\text{ predictions}} \tag{8}$$

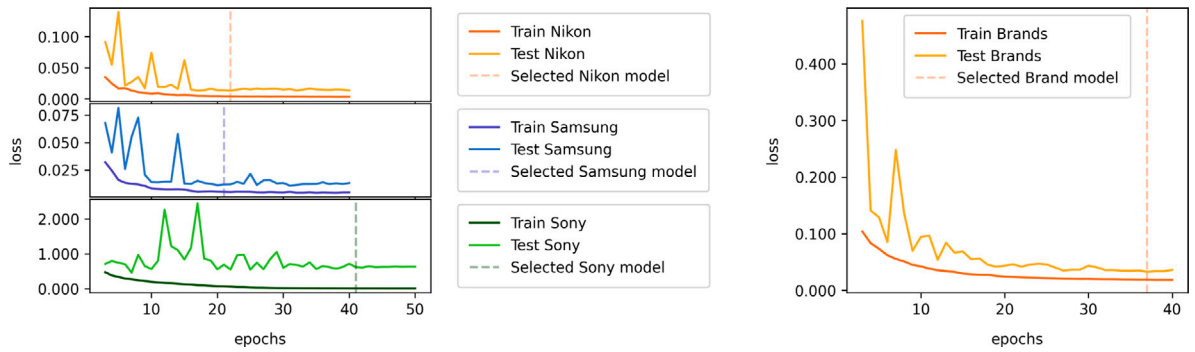

Fig. 10. The plots depict the convergence of the loss for fold 1 (out of 5). On the left are three stacked plots each representing the respective training loss plot for Nikon, Samsung, and Sony classifiers. The loss plot on the right corresponds to the training of the brand-level classifier. In all plots the dashed vertical lines indicate the epoch where the highest validation accuracy was achieved. The trained model for each classifier was chosen based on that epoch.

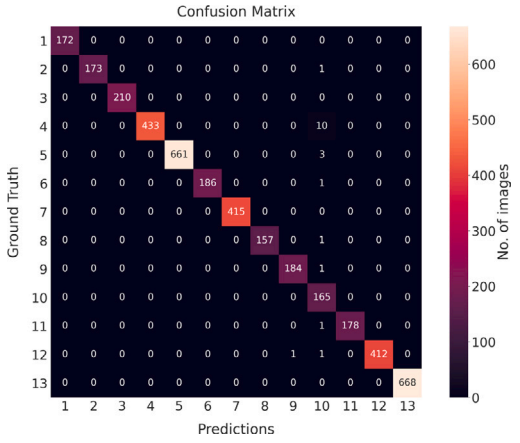

Fig. 11. The confusion matrix of the results (fold 1) for the classification of 13 camera brands. The 13 camera brands are 1. Canon, 2. Casio, 3. FujiFilm, 4. Kodak, 5. Nikon, 6. Olympus, 7. Panasonic, 8. Pentax, 9. Pratica, 10. Ricoh, 11. Rollei, 12. Samsung, and 13. Sony. Note that each brand in the test set has a different number of images, which were all used during the evaluation.

We achieved an average accuracy of 0.9941 ± 0.0010 percent and an average macro F1 score of 0.992 ± 0.001 for brand-level classification. Having trained and tested the brand classifier, we now discuss similar aspects for model-level classifiers.

4.5. Model classification

In order to train model-level classifiers, we consider only those camera brands which have multiple camera models. Of the 13 camera brands, only 3 brands have multiple camera models, namely Nikon, Samsung, and Sony. We use the same ConvNet architecture for training model-level classifiers as described in Section 4.4. In order to create a level-balanced training data set for model classification, we need to determine the number of patches to be extracted from each image. This can be determined using Eq. (4) and by setting $n_b = 1$ (since in model-level classification we are dealing with 1 brand per model). Therefore, for model-level classification, the number of patches to be extracted from each image i is determined by:

$$k_i = \left\lfloor \frac{k}{n_m \cdot n_d \cdot n_i} \right\rfloor \tag{9}$$

For the training of the Nikon, Samsung, and Sony classifiers, we choose k = 60,000, 40,000, and 60,000 patches, respectively. We set these values with the idea of extracting 20,000 patches from each camera model. Based on the mentioned values for k, the data sets are prepared for training the respective classifiers. We continue to perform 5-fold cross-validation and ensure that the distribution of camera devices remains consistent between the folds of the brand and model-level classification.

Moreover, the training of each model-level and the brand-level classifiers can be performed in parallel, as there is no dependency during the training phase. Using the same hyperparameters, which were used to train the brand classifier, we train each model-level classifier for about 40 epochs. The model-level classifiers that achieve the highest validation accuracy rates are chosen as the final ones.

Fig. 12 shows the confusion matrix for each model-level classifier. They are generated on the test data for the first fold. The Nikon classifier achieves an average accuracy of 0.9973 ± 0.0006 percent across all 5 folds. Similarly, the Samsung and Sony classifiers achieve an average accuracy of 0.9976 ± 0.0022 and 0.9751 ± 0.0085 percent, respectively. As in the case of brand classification, taking a majority vote on patch-level predictions to determine the image-level prediction reduces the error rate by almost 60 percent. For complete details regarding the reduction in error rates, refer to Table 4. Finally, we determine the macro F1 score to account for the imbalance in the test data set. The Nikon, Samsung, and Sony classifiers achieve an average F1 score of 0.997 ± 0.001, 0.998 ± 0.002, and 0.976 ± 0.008, respectively. We refer the reader to Table 3 for complete details concerning the macro F1 scores. The end-to-end hierarchical evaluation pipeline is described in the following sub-section.

4.6. Hierarchical classification

Having trained the brand-level and model-level classifiers, we now use them in a hierarchical fashion to predict the source camera model for a given image. We begin by extracting 200 homogeneous patches from a given image. At the top of the hierarchy is the brand-level classifier, which selects the brand for each of the 200 patches. A majority vote is performed to determine the source camera brand. In the trivial case, when there is only one camera model for a particular brand, the brand classification directly determines the model classification. Otherwise, based on the predicted brand, the corresponding model-level classifier is used to determine the source camera model. In our case, we trained three different model-level classifiers, one each for the Nikon, Samsung, and Sony brands. A majority vote is once again performed on all 200 patch predictions that determine the source camera model for the given image.

The first fold results of the hierarchical evaluation are presented as a confusion matrix in Fig. 13. We achieve an overall classification accuracy of 0.9899 ± 0.0007 and an overall macro F1 score of 0.990 ± 0.001 across all five folds. Tables 2 and 3 summarize the accuracy and the macro F1 score achieved for each fold, respectively.

In order to compare our method with the state-of-the-art, we choose methods that have been evaluated on the same 'natural' subset of the Dresden data set. The summary of this comparison is presented in

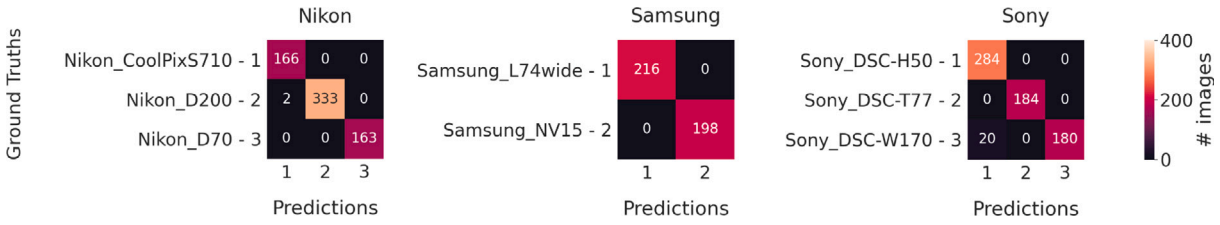

Fig. 12. Confusion matrices achieved by the Nikon, Samsung, and Sony model-level classifiers. They are generated by evaluating the test data in fold 1 (out of 5).

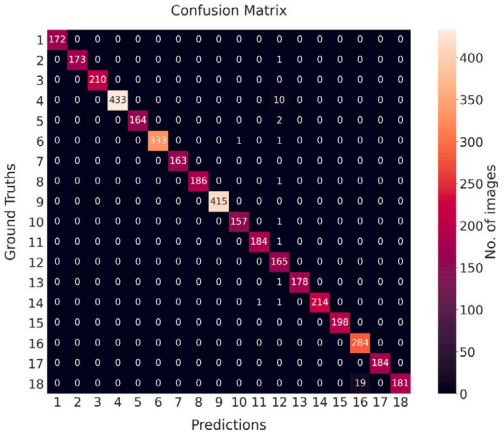

Fig. 13. Confusion matrix for classifying 18 camera models, based on the hierarchical pipeline. The result shown here is generated on the test data for fold 1 of 5. The 18 camera models correspond to the list of camera models in Table 1. Note that each brand in the test set has a different number of images and all were used during the evaluation.

Table 3
Macro F1 scores obtained on classifying 200 homogeneous patches from each image in the test set.

Classification	fold 1	fold 2	fold 3	fold 4	fold 5	Average
Brands	0.993	0.992	0.992	0.992	0.993	0.992 ± 0.001
Nikon	0.997	0.997	0.999	0.997	0.997	0.997 ± 0.001
Samsung	1.000	0.995	0.998	1.000	0.995	0.998 ± 0.002
Sony	0.971	0.989	0.967	0.981	0.973	0.976 ± 0.008
Hierarchical	0.989	0.991	0.988	0.990	0.990	0.990 ± 0.001

Table 4
Reduction in error rate (percent) with respect to the accuracy for image-level predictions (majority vote of 200 homogeneous patches) in comparison to patch-level predictions.

Classification	fold 1	fold 2	fold 3	fold 4	fold 5	Average (%)
Brands	49.2	40.8	47.1	50.9	52.7	48.17 ± 03.75
Nikon	30.3	77.1	65.7	70.2	35.2	55.69 ± 17.47
Samsung	100	29.4	25.5	100	31.1	57.20 ± 31.95
Sony	68.0	88.5	66.6	79.5	69.4	74.43 ± 07.66

Table 5. Similar to the works in Bondi et al. (2016), Rafi, Tonmoy, et al. (2020), Rafi, Wu, and Hasan (2020) we follow the leave-one-device-out strategy for cross-validation, and show the results in Table 5. Our strategy of homogeneous patch selection ensures to ignore scene details from the image, which allows us to avoid inadvertent classification of the scene content. It was, therefore, not necessary to employ the scene-independent test set strategy proposed by Bondi et al. (2016). Marra et al. (2017) performed experiments on 25 camera models of which 7 models have only one device per model in the Dresden data set. In their setting, it is not possible to test the generalizability of the trained classifier for those 7 devices.

Among all the proposed methods that use 18 camera models on the Dresden data set, we achieve the best classification accuracy and reduce

Table 5
Comparison with the state-of-the-art on the "natural" subset of the Dresden image data set for camera model identification.

Method	Accuracy	No. of models	Evaluation scheme
Tuama et al. (2016)	0.9709	14	5-fold CV
Marra et al. (2017)	0.9627	25	20-fold CV with 7 devices used for both train and test sets
Marra et al. (2017)	0.9872	25	20-fold CV with 7 devices used for both train and test sets
Bondi et al. (2016)	~ 0.965	18	Scene independent test set
Rafi, Tonmoy, et al. (2020)	0.9703	18	Scene independent test set
Rafi, Wu, and Hasan (2020)	0.9815	18	Scene independent test set
Ours - **200** patches	**0.9899**	**18**	Leave-one-device-out **5**-fold CV
Ours - **400** patches	**0.9901**	**18**	Leave-one-device-out **5**-fold CV

the classification error rate by 46.49 percent compared to the previous best result achieved by Rafi, Wu, and Hasan (2020).

5. Discussion

This section describes further experiments that were conducted to support our hypothesis for SCI.

5.1. Number of patches

The results reported thus far were achieved by using 200 homogeneous patches per image during the test phase. In order to understand the number of patches required for evaluation, we conducted a series of experiments which are summarized in Fig. 14. As can be seen, certain brands can work with a few patches while others require more patches to obtain high accuracy. This behaviour is evident with the Sony classifier, which significantly improves in performance with an increasing number of patches. Interestingly, the performance of the Samsung classifier slightly decreases when the number of patches is increased from 1 to 10. This minor deviation from the expected trend is, however, so small that it is probably due to chance. In general, classification becomes progressively robust with increasing number of randomly selected homogeneous patches, up to a certain point.

The extraction of too many patches would lead us into the regime of non-homogeneous patches. Therefore, for the Dresden data set, we recommend performing prediction using 200 patches. Based on the homogeneous patch distribution shown in Fig. 7 we can conclude that most of the images (from all camera models) contain at least 200 homogeneous patches of size 128×128 pixels. In fact, in Fig. 14 one can observe that the classification accuracy of the Sony classifier decreases on considering 400 patches when compared to using only 200 patches. This decrease in accuracy may be due to the inclusion of non-homogeneous patches when we extract 400 patches (Fig. 7). As the Dresden data set constitutes images belonging to diverse scenes, it is reasonable to assume that most natural images contain about 200 homogeneous patches.

It is also important to account for the time taken to extract such patches. In our experiments, the extraction of patches from an image

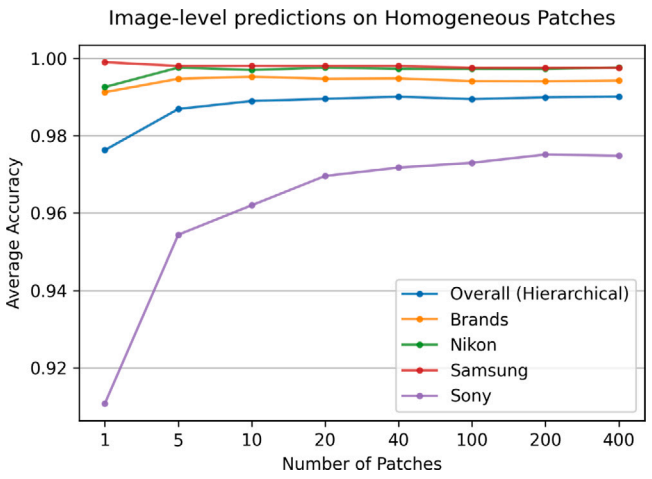

Fig. 14. The image-level accuracy achieved with varying numbers of homogeneous patches. Each accuracy point corresponds to the average across the 5-folds.

Table 6
Comparison of the classification accuracy between the hierarchical and the flat approaches. The results are generated by considering 200 homogeneous patches per image.

Classification	fold 1	fold 2	fold 3	fold 4	fold 5	Average
Flat	0.986	0.987	0.988	0.979	0.986	0.9851 ± 0.0033
Hierarchical	0.990	0.991	0.989	0.990	0.989	0.9899 ± 0.0007

of size 3072×2304 pixels with patches of size 128×128 pixels took approximately 2.16 seconds when measured on a single core of Intel Xeon E5-2680 v3 CPU (2.5 GHz). Because we use the integral image for patch selection and extraction the execution time primarily depends on the image size and not on the number of homogeneous patches.

5.2. Hierarchical vs flat approach

One of our hypotheses was that a hierarchical approach is better than using a single classifier (flat approach). In order to evaluate this hypothesis, we conducted experiments by training all the 18 camera models using a single classifier. For a fair comparison, the ConvNet parameters were kept unchanged and were set to match our earlier experiments. We evaluated the flat ConvNet by using 200 homogeneous patches and performed a 5-fold cross-validation. The results of these experiments are summarized in Table 6. We obtained an average classification accuracy of 0.9851 ± 0.0033. In comparison to the hierarchical approach, the average error rate increases by more than 50 percent for the flat approach.

The flat classifiers also took more epochs to converge when compared to their hierarchical counterparts. This difficulty in model learning is due to the mixing up of different camera models and brands into a single classifier. When using the hierarchical approach the brand classifier accounts for inter-brand noise variation, while the model-level classifiers account for the intra-brand noise variations. This separation of noise variations allows the hierarchical classifiers to learn better features for classification. Notable is also the fact that the results obtained by the flat approach are also better than existing works (Table 5).

The proposed hierarchical approach has other advantages. It trivializes parallelization, as each classifier in the hierarchy can be trained independently of others. Moreover, this approach is robust in handling the addition of new camera models. In such cases, only the specific model-level classifier needs to be modified without having to update other classifiers. This not only saves computation time during the initial training but also during future updates.

Certain brands can achieve high accuracy with fewer patches. For instance, referring to Fig. 14 we could use only 10 patches per image to determine the camera brand with around 99.5 percent accuracy. This is also the case with Nikon and Samsung classifiers. The Sony classifier, however, achieves the highest classification accuracy with 200 patches. One may, therefore, exploit this result to use the optimal number of patches per model in order to also maximize the speed of the evaluation process, something that would not be possible with a single-classifier approach.

5.3. Future work

One direction for future research is the evaluation of the proposed approach to varying levels of image quality. In such evaluation one may consider no-reference image quality measures (Gu et al., 2017, 2016; Mittal et al., 2012), which can be used to determine the visual quality of images, among others (Gu, Li, et al., 2017; Gu, Zhou, et al., 2017). Using this information, the relationship between the input image quality and the resulting performance for SCI can be determined. This information could be used to assess whether a given image has sufficient quality for a reliable determination of its camera model.

Another direction would be to extend this work for device-level identification, which presents further challenges than those of the model-level identification at hand. Such an approach would be particularly useful when LEAs need to discriminate between two devices of the same camera model.

6. Conclusion

We propose a new approach for camera model identification that leverages the homogeneous regions in given images, which are less distorted by the scene content, for reliable extraction of forensic traces. We showed that when such input data is trained in a hierarchical fashion, it results in a classifier that is computationally efficient, modular and more effective than a flat (single classifier) approach. Modular design allows the addition of other camera brands without having to retrain the model classifiers of the already known brands. The accuracy rate of 99.01% that we achieve is the best ever reported for the 'natural' subset of the benchmark Dresden data set of 18 camera models.

CRediT authorship contribution statement

Guru Swaroop Bennabhaktula: Methodology, Validation, Software, Investigation, Resources, Writing – original draft, Writing – review & editing. **Enrique Alegre:** Conceptualization, Methodology, Writing – review & editing, Validation, Investigation, Supervision, Project administration. **Dimka Karastoyanova:** Writing – review & editing. **George Azzopardi:** Conceptualization, Methodology, Writing – original draft, Validation, Investigation, Writing – review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the framework agreement between the University of León and INCIBE (Spanish National Cybersecurity Institute), Spain under Addendum 01. This research has been partly funded with support from the European Commission under the 4NSEEK project with Grant Agreement 821966. This publication reflects the views only of the authors, and the European Commission cannot be held responsible for any use which may be made of the information contained therein. We thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Peregrine high-performance computing cluster.

References

Barni, M., Bondi, L., Bonettini, N., Bestagini, P., Costanzo, A., Maggini, M., Tondi, B., & Tubaro, S. (2017). Aligned and non-aligned double JPEG detection using convolutional neural networks. *Journal of Visual Communication and Image Representation*, *49*, 153–163.

Barni, M., Chen, Z., & Tondi, B. (2016). Adversary-aware, data-driven detection of double JPEG compression: How to make counter-forensics harder. In *2016 IEEE international workshop on information forensics and security* (pp. 1–6). IEEE.

Bayar, B., & Stamm, M. C. (2016). A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of the 4th ACM workshop on information hiding and multimedia security* (pp. 5–10).

Bayar, B., & Stamm, M. (2018a). Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection. *IEEE Transactions on Information Forensics and Security*, *13*(11), 2691–2706.

Bayar, B., & Stamm, M. C. (2018b). Towards open set camera model identification using a deep learning framework. In *2018 IEEE international conference on acoustics, speech and signal processing* (pp. 2007–2011). IEEE.

Bayram, S., Sencar, H., Memon, N., & Avcibas, I. (2005). Source camera identification based on CFA interpolation. In *IEEE international conference on image processing 2005 (vol. 3)* (pp. III–69). IEEE.

Bennabhaktula, G. S., Alegre, E., Karastoyanova, D., & Azzopardi, G. (2020). Device-based image matching with similarity learning by convolutional neural networks that exploit the underlying camera sensor pattern noise. In *Proceedings of the 9th international conference on pattern recognition applications and methods (vol. 1)* (pp. 578–584). http://dx.doi.org/10.5220/0009155505780584.

Bondi, L., Baroffio, L., Güera, D., Bestagini, P., Delp, E. J., & Tubaro, S. (2016). First steps toward camera model identification with convolutional neural networks. *IEEE Signal Processing Letters*, *24*(3), 259–263.

Bondi, L., Lameri, S., Güera, D., Bestagini, P., Delp, E. J., & Tubaro, S. (2017). Tampering detection and localization through clustering of camera-based CNN features. In *2017 IEEE conference on computer vision and pattern recognition workshops* (pp. 1855–1864). IEEE.

Bunk, J., Bappy, J. H., Mohammed, T. M., Nataraj, L., Flenner, A., Manjunath, B., Chandrasekaran, S., Roy-Chowdhury, A. K., & Peterson, L. (2017). Detection and localization of image forgeries using resampling features and deep learning. In *2017 IEEE conference on computer vision and pattern recognition workshops* (pp. 1881–1889). IEEE.

Cao, H., & Kot, A. C. (2009). Accurate detection of demosaicing regularity for digital image forensics. *IEEE Transactions on Information Forensics and Security*, *4*(4), 899–910.

Chen, M., Fridrich, J., Goljan, M., & Lukás, J. (2008). Determining image origin and integrity using sensor noise. *IEEE Transactions on Information Forensics and Security*, *3*(1), 74–90.

Chen, J., Kang, X., Liu, Y., & Wang, Z. J. (2015). Median filtering forensics based on convolutional neural networks. *IEEE Signal Processing Letters*, *22*(11), 1849–1853.

Chen, C., & Stamm, M. C. (2015). Camera model identification framework using an ensemble of demosaicing features. In *2015 IEEE international workshop on information forensics and security* (pp. 1–6). IEEE.

Cozzolino, D., & Verdoliva, L. (2019). Noiseprint: a CNN-based camera model fingerprint. *IEEE Transactions on Information Forensics and Security*, *15*, 144–159.

Fan, W., Wang, K., & Cayre, F. (2015). General-purpose image forensics using patch likelihood under image statistical models. In *2015 IEEE international workshop on information forensics and security* (pp. 1–6). IEEE.

Gloe, T., & Böhme, R. (2010). The 'Dresden Image Database' for benchmarking digital image forensics. In *Proceedings of the 2010 ACM symposium on applied computing* (pp. 1584–1590).

Gu, K., Jakhetiya, V., Qiao, J.-F., Li, X., Lin, W., & Thalmann, D. (2017). Model-based referenceless quality metric of 3D synthesized images using local image description. *IEEE Transactions on Image Processing*, *27*(1), 394–405.

Gu, K., Li, L., Lu, H., Min, X., & Lin, W. (2017). A fast reliable image quality predictor by fusing micro-and macro-structures. *IEEE Transactions on Industrial Electronics*, *64*(5), 3903–3912.

Gu, K., Lin, W., Zhai, G., Yang, X., Zhang, W., & Chen, C. W. (2016). No-reference quality metric of contrast-distorted images based on information maximization. *IEEE Transactions on Cybernetics*, *47*(12), 4559–4565.

Gu, K., Zhou, J., Qiao, J.-F., Zhai, G., Lin, W., & Bovik, A. C. (2017). No-reference quality assessment of screen content pictures. *IEEE Transactions on Image Processing*, *26*(8), 4005–4018.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448–456). PMLR.

Kang, X., Stamm, M. C., Peng, A., & Liu, K. R. (2013). Robust median filtering forensics using an autoregressive model. *IEEE Transactions on Information Forensics and Security*, *8*(9), 1456–1468.

Kang, X., & Wei, S. (2008). Identifying tampered regions using singular value decomposition in digital image forensics. *3*, In *2008 International conference on computer science and software engineering* (pp. 926–930). IEEE.

Kharrazi, M., Sencar, H. T., & Memon, N. (2004). Blind source camera identification. In *2004 International conference on image processing, 2004 (vol. 1)* (pp. 709–712). IEEE.

Kirchner, M., & Fridrich, J. (2010). On detection of median filtering in digital images. In *Media forensics and security II (vol. 7541)*. International Society for Optics and Photonics, Article 754110.

Kirchner, M., & Gloe, T. (2015). Forensic camera model identification. In *Handbook of digital forensics of multimedia data and devices* (pp. 329–374). Wiley Online Library.

Kurosawa, K., Kuroki, K., & Saitoh, N. (1999). CCD fingerprint method-identification of a video camera from videotaped images. In *Proceedings 1999 international conference on image processing (cat. 99CH36348) (vol. 3)* (pp. 537–540). IEEE.

Li, C.-T. (2010). Source camera identification using enhanced sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, *5*(2), 280–287.

Li, C.-T., Chang, C.-Y., & Li, Y. (2009). On the repudiability of device identification and image integrity verification using sensor pattern noise. In *International conference on information security and digital forensics* (pp. 19–25). Springer.

Li, J., Li, X., Yang, B., & Sun, X. (2014). Segmentation-based image copy-move forgery detection scheme. *IEEE Transactions on Information Forensics and Security*, *10*(3), 507–518.

Lin, X., & Li, C.-T. (2015). Preprocessing reference sensor pattern noise via spectrum equalization. *IEEE Transactions on Information Forensics and Security*, *11*(1), 126–140.

Lukas, J., Fridrich, J., & Goljan, M. (2006). Digital camera identification from sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, *1*(2), 205–214.

Luo, W., Huang, J., & Qiu, G. (2010). JPEG error analysis and its applications to digital image forensics. *IEEE Transactions on Information Forensics and Security*, *5*(3), 480–491.

Marra, F., Poggi, G., Sansone, C., & Verdoliva, L. (2017). A study of co-occurrence based local features for camera model identification. *Multimedia Tools and Applications*, *76*(4), 4765–4781.

Mayer, O., Bayar, B., & Stamm, M. C. (2018). Learning unified deep-features for multiple forensic tasks. In *Proceedings of the 6th ACM workshop on information hiding and multimedia security* (pp. 79–84).

Mittal, A., Moorthy, A. K., & Bovik, A. C. (2012). No-reference image quality assessment in the spatial domain. *IEEE Transactions on Image Processing*, *21*(12), 4695–4708.

Rafi, A. M., Tonmoy, T. I., Kamal, U., Wu, Q. J., & Hasan, M. K. (2020). RemNet: remnant convolutional neural network for camera model identification. *Neural Computing and Applications*, 1–16.

Rafi, A. M., Wu, J., & Hasan, M. K. (2020). L2-constrained RemNet for camera model identification and image manipulation detection. In *European conference on computer vision* (pp. 267–282). Springer.

Rosenfeld, K., & Sencar, H. T. (2009). A study of the robustness of PRNU-based camera identification. In *Media forensics and security (vol. 7254)*. International Society for Optics and Photonics, 72540M.

Swaminathan, A., Wu, M., & Liu, K. R. (2007). Nonintrusive component forensics of visual sensors using output images. *IEEE Transactions on Information Forensics and Security*, *2*(1), 91–106.

Tang, H., Ni, R., Zhao, Y., & Li, X. (2018). Median filtering detection of small-size image based on CNN. *Journal of Visual Communication and Image Representation*, *51*, 162–168.

Thai, T. H., Cogranne, R., & Retraint, F. (2013). Camera model identification based on the heteroscedastic noise model. *IEEE Transactions on Image Processing*, *23*(1), 250–263.

Timmerman, D., Bennabhaktula, G. S., Alegre, E., & Azzopardi, G. (2020). Video camera identification from sensor pattern noise with a constrained ConvNet. In *ICPRAM 2021 proceedings*. SciTePress, 10th International Conference on Pattern Recognition Applications and Methods ICPRAM 2021.

Tuama, A., Comby, F., & Chaumont, M. (2016). Camera model identification with the use of deep convolutional neural networks. In *2016 IEEE international workshop on information forensics and security* (pp. 1–6). IEEE.

Wang, X., Wang, H., & Niu, S. (2019). An image forensic method for AI inpainting using faster R-CNN. In *International conference on artificial intelligence and security* (pp. 476–487). Springer.

Wang, Q., & Zhang, R. (2016). Double JPEG compression forensics based on a convolutional neural network. *EURASIP Journal on Information Security*, *2016*(1), 1–12.

Xu, G., & Shi, Y. Q. (2012). Camera model identification using local binary patterns. In *2012 IEEE international conference on multimedia and expo* (pp. 392–397). IEEE.

Zhu, X., Qian, Y., Zhao, X., Sun, B., & Sun, Y. (2018). A deep learning approach to patch-based image inpainting forensics. *Signal Processing: Image Communication*, *67*, 90–99.