Pattern Recognition 124 (2022) 108413

Contents lists available at ScienceDirect

Pattern Recognition

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

PRNU registration under scale and rotation transform based on convolutional neural networks

Marco Fanfani*, Alessandro Piva, Carlo Colombo

Department of Information Engineering (DINFO), University of Florence, Via S. Marta, 3, Firenze 50139, Italy

ARTICLE INFO

Article history:
Received 12 March 2021
Revised 12 August 2021
Accepted 30 October 2021
Available online 3 November 2021

Keywords:
Image forensics
PRNU
Deep learning
CNN
Rotation
Scale

ABSTRACT

Assessing if an image comes from a specific device is fundamental in many application scenarios. The most promising techniques to solve this problem rely on the Photo Response Non Uniformity (PRNU), a unique trace left during image acquisition. A PRNU fingerprint is computed from several images of a given device, then it is compared with the probe residual noise by means of correlation. However, such a comparison requires that PRNUs are synchronized: even small image transformations can spoil this task. Most of the attempts to solve the registration problem rely on time consuming brute-force search, which is prone to missing detections and false positives. In this paper, the problem is addressed from a computer vision perspective, exploiting recent image registration techniques based on deep learning, and focusing on scaling and rotation transformations. Experiments show that the proposed method is both more accurate and faster than state-of-the-art approaches.

1. Introduction

Automatic methods to assess the integrity of digital images are of paramount importance in order to counter the ever-increasing production and spread of fake imagery through the media. Image forensic methods [1] try to solve this problem by observing distinctive traces left during the image acquisition or manipulation. During the years, several methods have been developed exploiting either invisible footprints introduced in the signal statistics or physical inconsistencies left directly into the scene. Invisible footprints include demosaicing artefacts [2], characteristic camera [3] or scanner [4] sensor noise, and compression anomalies [5,6]. Physical inconsistencies encompass shadows [7], light color [8] and direction [9], and scene geometry, like perspective [10], 3D constraints [11], and camera principal point [12,13].

In this paper, we focus on sensor noise analysis to solve the *camera identification* problem, i.e., assessing if a given image was acquired with a specific camera. In particular, we studied the Photo Response Non Uniformity (PRNU) pattern, a device-dependent noise left during image acquisition by the camera sensor [14]. PRNU based camera identification is typically accomplished by evaluating the correlation between the residual noise extracted from the probe image, and a set of reference PRNU fingerprints, obtained from a collection of flat-field images captured with each candidate camera. The Peak-to-Correlation Energy ratio (PCE) is used to measure the similarity between the probe and reference signals. Alternative solutions have been proposed recently, that attempt to boost camera identification performance by exploiting deep learning either to extract better sensor noises [15] or to speed-up and improve PRNU matching [16,17].

PRNU matching is a delicate task, that requires pixel level accuracy. Hence, even slight geometric image transformations can misalign the residual noise of the probe and the fingerprint, thus spoiling the camera identification task. While image translations can easily be recovered as a by-product of PCE computation (the position of the peak obtaining the maximum score also indicates the translation between the two signals), scale and rotation transformations introduce a higher degree of complexity, since they have to be explicitly recovered before evaluating any correlations. To date, brute-force is the most popular approach for PRNU alignment [18]. It tests all possible combinations of scale and rotation, retrieving the one that maximizes the PCE. However, such approach is computationally slow, and can produce false positives or missing detections.

In order to provide a faster and more accurate solution, in this paper we present a method based on deep learning to recover scale and rotation transformations between PRNU signals. We show how it is possible to train a Convolutional Neural Network (CNN) to recover PRNU transformations and use it as a fast pre-processing step before evaluating the PCE.

* Corresponding author.
E-mail addresses: marco.fanfani@unifi.it (M. Fanfani), alessandro.piva@unifi.it (A. Piva), carlo.colombo@unifi.it (C. Colombo).

https://doi.org/10.1016/j.patcog.2021.108413
0031-3203/

The paper is organized as follows: In the next section related works are discussed. Then, in Section 3 the proposed method is outlined, by describing how training data are generated and providing details on the network architecture. Results of a comparative evaluation between the proposed method and state-of-the-art approaches are reported and discussed in Section 4. Finally, in Section 5 conclusions are drawn and directions for future work are outlined.

2. Related works

PRNU is a unique fixed pattern noise generated during the acquisition process by any digital camera and pixel-wise related to the specific device sensor. Therefore, it is best extracted and compared at native camera resolution from unaltered probes [19]. Being compared pixel-wise, PRNU signals become particularly difficult to match when the source images have been warped as result of the acquisition post-processing. Even small geometric transformations, maliciously applied by a forger or introduced directly by the device (e.g., during electronic image stabilization), can spoil the camera identification task, by strongly reducing the PCE.

During the last few years several works addressed PRNU based camera identification for stabilized videos, from preliminary works [20] to more advanced solutions such as brute-force search on video frames [21], hybrid image/video identifications [22,23], optimized search of transformation parameters [24,25], or preliminary camera model characterization [26,27]. However, a similar effort was not devoted to single images, for which information redundancy cannot be exploited in order to ease the identification task. In [3], the only geometric transformation admitted is translation. In [18], image scaling is also considered, thus requiring to remove any amount of zooming before computing the PCE. Scale removal is typically carried out by brute-force search, which is computationally expensive, and not always sufficiently accurate. In order to speed-up PCE computations, in [28] the PRNU is first reduced to a digest. The approach, which can deal with zooming and lens distortion, still requires a full grid-search of the transformation parameters. In [29], a similar PRNU compression is proposed based on Principal Component Analysis. Alternative approaches detect and estimate specific image transformations (e.g., resampling) using deep learning. In [30], a CNN based method to estimate the probe scaling factor directly from natural image patches is presented. In [31], the authors propose a deep learning classifier to deal with scale, compression, blurring and median filtering transformations applied to natural images. In [32], a Bayesian neural network is trained in order to detect resampling manipulations. None of the traditional and deep methods above takes into account rotation transformations, which nevertheless are quite common yet difficult to deal with in a camera identification scenario. Moreover, none of the CNN-based methods above operate directly on PRNU signals. To the best of our knowledge, the method proposed in this paper is the first to address scale and rotation transformations simultaneously on PRNU signals directly extracted from single images.

In order to register PRNU images, our method employs a deep network which is inspired by the deep image homography estimation network initially proposed in [33] for natural images. Differently from natural images, PRNU images are characterized by patterns that neither have a clear structure nor possess characteristic elements such as edges or corners [34]. Moreover, the fingerprint obtained from a collection of flat-field images and the residual noise extracted from a single probe coming from the same device are not identical, since high frequency content unrelated to the PRNU can still be present in the residual noise. These observations suggest that working with PRNU images is more challenging than with natural images, where the network can actually rely on low frequency information that is unavailable in PRNU signals. Nevertheless, in the rest of the paper we show that a suitably trained deep net is actually capable to estimate warping transformations from PRNU details only. Our method specializes on linear homographies, a.k.a. image similarities, which are the warps that occur most commonly in practical applications [27].

3. Proposed method

The alignment between the probe residual noise and the camera PRNU fingerprint is obtained with a deep network trained on similarity transformations characterized by a combination of scaling and rotation. Once the probe is properly registered using the parameters estimated by the network, PRNU matching is achieved with PCE correlation. In order to estimate the scale and rotation parameters on PRNU signals—which are characterized by very different contents and structures w.r.t. natural images—the net presented in [33] was trained from scratch using examples extracted from PRNU images (see Section 3.1): Given the different nature of the inputs, it was impossible to achieve positive results with fine tuning techniques. Also, the net architecture (see Section 3.2) was slightly modified in order to produce in output the parameters of similarity transforms, instead of the those required for homographies.

3.1. Example generation

In order to generate the examples used to train and test the network, patches from the fingerprint and the residual noise were stacked as follows.

Let $\mathcal{F}_d$ be the fingerprint of a device d obtained from flat-field unaltered images, and let $\mathcal{N}_d$ be a residual noise extracted from an unaltered natural image taken with the same device d. Selecting at random the point $\mathbf{p} \in \mathcal{F}_d$ and perturbing its x and y coordinates with a random shift $\Delta = [\delta_x, \delta_y]^\top$, a new point $\mathbf{p}' = [x + \delta_x, y + \delta_y]^\top$ is obtained. The shift vector Δ naturally induces a minimal representation of the random similarity transformation

$$\mathcal{S} = \sigma \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{1}$$

such that $\mathbf{p}' = \mathcal{S}\mathbf{p}$. Indeed, the scale factor σ and the rotation angle θ in Eq. (1) can be obtained as:

$$\sigma = \frac{||\mathbf{p}'||}{||\mathbf{p}||} \tag{2}$$

$$\theta = -\tan^{-1}\left(\frac{y'}{x'}\right) - \tan^{-1}\left(\frac{y}{x}\right) \quad . \tag{3}$$

Once the random similarity map $\mathcal{S}$ is generated as above, its inverse $\mathcal{S}^{-1}$ is used to scale and rotate $\mathcal{N}_d$ and obtain $\hat{\mathcal{N}}_d$, such that a point $\mathbf{p}' \in \mathcal{N}_d$ is coincident with $\mathbf{p} \in \hat{\mathcal{N}}_d$. A patch of 128×128 pixels is then extracted from both $\mathcal{F}_d$ and $\hat{\mathcal{N}}_d$, with its top-left corner in $\mathbf{p}$. The extracted pair of patches is finally stacked along with the channel dimension and given as input to the net, with a label representing the random shift Δ. In our implementation, both δ_x and δ_y are randomly sampled in the interval $[-32, +32]$ pixels. With this sampling interval, we experimentally measured that the covered set for σ and θ are respectively [0.85,1.15], and $[-0.15, 0.15]$ degrees. The choice of parametrizing the similarity transform using the shift vector Δ is inspired by [35], where a 4-point parametrization was used in order to model the 8 degrees of freedom of a general homography. Note that, in this work, a *1-point parametrization* is sufficient to cover the 2 degrees of freedom of $\mathcal{S}$. Fig. 1 depicts the example generation process.

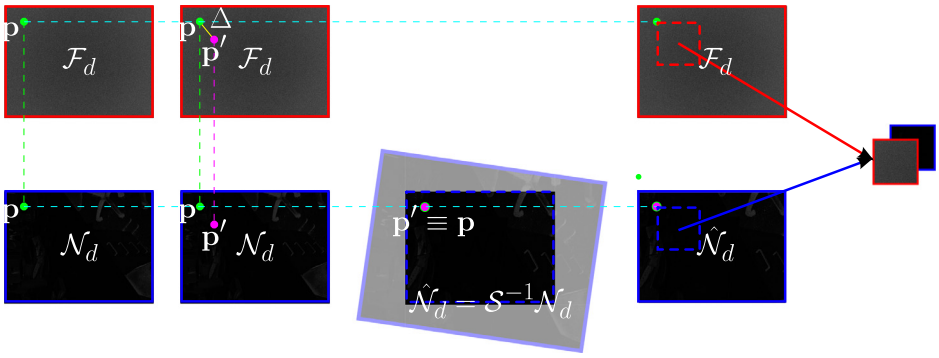


Fig. 1. Patch extraction to create examples. Given a fingerprint $\mathcal{F}_d$ and a residual noise $\mathcal{N}_d$ a random point $\mathbf{p}$ is selected. Then, $\mathbf{p}$ is perturbed with a random shift Δ, so to obtain $\mathbf{p}'$. From $\mathbf{p}$ and $\mathbf{p}'$ a transformation matrix $\mathcal{S}$ can be defined, such that $\mathbf{p}' = \mathcal{S}\mathbf{p}$. The inverse transformation $\mathcal{S}^{-1}$ is used to compute the transformed version of $\mathcal{N}_d$, i.e. $\hat{\mathcal{N}}_d = \mathcal{S}^{-1}\mathcal{N}_d$, such that in $\hat{\mathcal{N}}_d$ the point $\mathbf{p}'$ gets the same coordinate of $\mathbf{p}$ in $\mathcal{N}_d$. Finally, two 128×128 patches are extracted — one from $\mathcal{F}_d$ and the other from $\hat{\mathcal{N}}_d$ — and then stacked together to build an example to be passed to the net, with label Δ. (Best viewed in color and zoomed in).

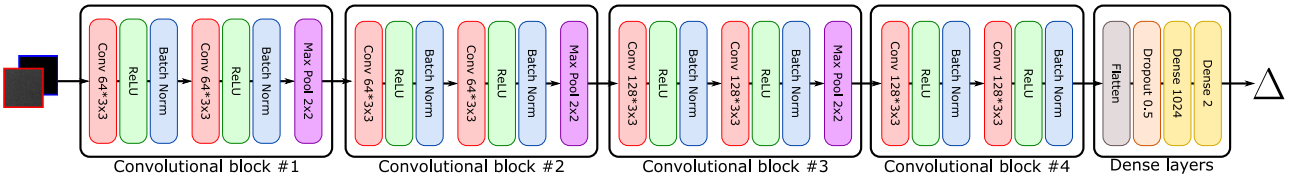


Fig. 2. Convolutional network architecture.

3.2. CNN implementation

In this section implementation details of the proposed approach are given. Before extracting the patches as described above, the PRNU signals are individually normalized to have zero mean and unit variance. Moreover, natural images whose unaltered residual noise obtains a low PCE score are discarded, as they may make less reliable the comparison against the device fingerprint.

The used convolutional network is schematically reported in Fig. 2. It takes as input patches of 128×128×2 pixels and is composed by four convolutional blocks followed by two dense layers. Each of the first two convolutional blocks include two convolutional layers with 64 3×3 filters, with stride 1 and ReLU activations. After each convolutional layer batch normalization is applied, and finally a max pooling operation with 2×2 filters and stride 2 is used. The last two convolutional blocks use layers with 128 3×3 filters. In particular, the last convolutional block does not have a max pooling stage: Its output is first flattened to a 1024 vector, and, after a dropout stage with a probability of 0.5 (only during training), goes through two dense layers, the first with 1024 units and the second and last with only two output units, predicting the 2D shift vector $\Delta = (\delta_x, \delta_y)$.

Although several loss and optimization functions have been applied in CNN architectures in the literature [36–38], they have not been preferred in this work due to their computational complexity. The net was trained using stochastic gradient descent with momentum 0.9 and an initial learning rate of 0.005, that was progressively reduced by 1/10 every four epochs. The loss function was the Euclidean distance between the predicted and the ground truth shift vector Δ, as already done in [33]. Training went on for 15 epochs, evaluating 500,000 examples per epoch fed to the net in batches of 50 examples. At the end of each epoch, 20,000 validation examples were used to control overfitting and possibly perform early-stopping.

3.3. Camera identification

Once trained, the net presented above can be used as a preprocessing step for the camera identification task. Typically, the camera identification based on PRNU analysis is solved by simply evaluating the PCE between the fingerprint signal of a device and the residual noise extracted from a probe. Even a slight de-synchronization of the two signals (due to rotation or scale changes) would dramatically reduce the PCE value under the 'classical' thresholds (i.e., 60 and 100), thus spoiling camera identification. As an example, in Fig. 3a a pristine natural image is shown: the PCE between its residual noise and the device fingerprint reaches a value of 12,836.27. In Fig. 3b the same image is shown after being subjected to a scale factor of 1.01 and a rotation of 0.001 degrees: while the transformation is practically imperceptible, its PCE reaches a value of 36.95 only (with a reduction of more than 99%!), resulting in a false negative camera attribution.

(a) (b)

Fig. 3. In (a) a natural image from device D01 of VISION dataset. The PCE between its residual noise and the D01 fingerprint returns a score of 12836.27. In (b) the same image after been subjected to a scaling ($\sigma = 1.01$), and a rotation ($\theta = 0.001$). The resulting image do not show any particular differences w.r.t. the original, however its PCE score drops to 36.95, spoiling any possible device attribution.

To mitigate the risk of missed attributions, our net can be used to pre-process a given fingerprint-residual noise pair by estimating the scale and rotation parameters applied to the probe image (and consequently to the residual noise). The estimated parameters can then be used to register the residual noise w.r.t. the fingerprint, thus re-synchronizing the two signals and eventually obtaining a reliable value for the PCE (see Fig. 4). As shown in the experimental section, if the fingerprint and the residual noise come from the same device, the PCE value computed after the registration is

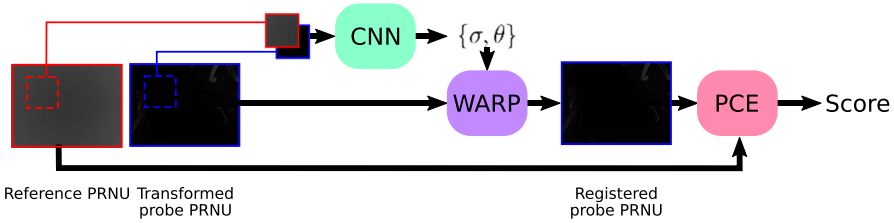


Fig. 4. Proposed camera identification pipeline. A pair of corresponding patches are sampled from both PRNU signals and given as input to the CNN to estimate the probe transformation parameters σ, θ. The probe is resynchronized w.r.t. the fingerprint. Finally, PCE is normally evaluated to decide the probe attribution.

higher than any sensible threshold with a high probability, while if the two PRNU are extracted from different devices, the transformation estimated from the network is typically not sufficient to obtain a high PCE score, thereby avoiding to introduce false positives.

4. Experimental analysis

4.1. Dataset

To train the net and evaluate the proposed approach we used the recently proposed VISION dataset [19]. This dataset includes still images and videos recorded from 35 smartphone devices, covering 29 different models from 11 manufacturers. For each device, flat-field (i.e., planar, monochromatic) and natural images are available, for a total of 4167 flat-field and 7565 natural images.

4.2. Training

In order to compute a strong fingerprint of each device, all the available flat-field images taken with it were used. The natural images taken with the same device were divided into three sets: training (using 70% of the probes), validation (10%), and test (20%). Then, 500,000 examples were generated from the training set, following the method described in Section 3.1, and 20,000 from the validation set. Test examples were instead generated at runtime from the remaining test set[1]. A different net was trained separately for each selected device, evaluating 15 epochs with batches of 50 examples, repeated 10,000 times (so as to observe all 500,000 examples for each epoch). At the end of each epoch, 20,000 validation examples were used to control overfitting and possibly perform early-stopping.

4.3. Results

4.3.1. Performance measure

Since the aim of this method is to solve the camera identification task when the probe has been modified by a scale and rotation transform, the performance of the method was assessed by evaluating the PCE between the fingerprint and the registered residual noise (using the scale and rotation parameter predicted by our net) and then counting how many test cases obtained a PCE score higher than the two thresholds 60 and 100. The higher is the number of passed tests, the more reliable is the net.

4.3.2. Single device attribution

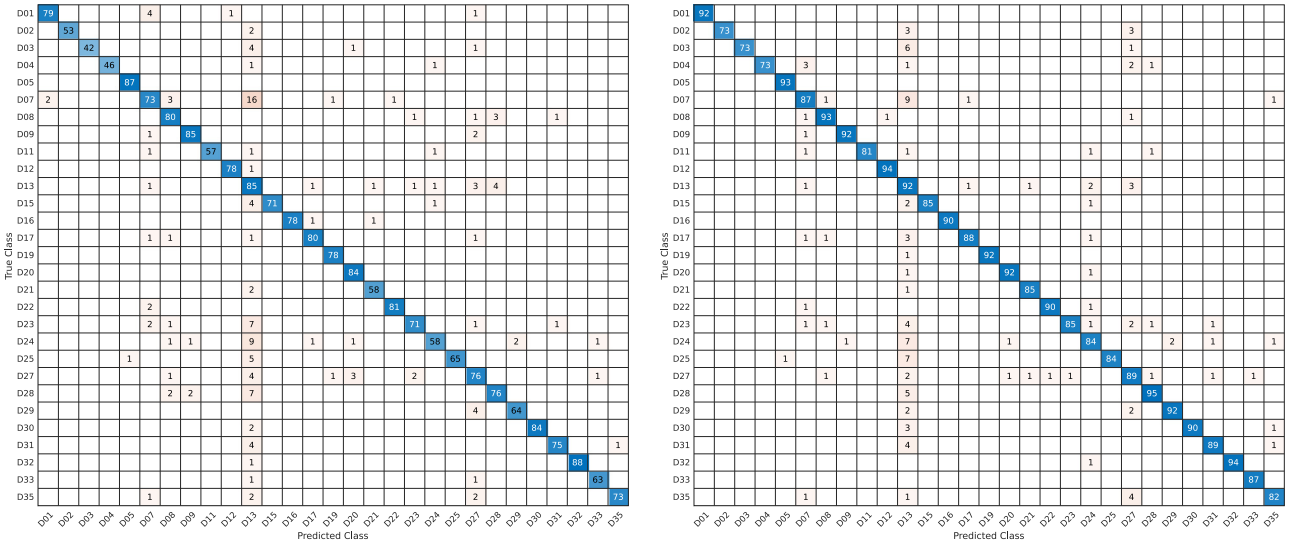
The first test was devoted to assess the ability of the net in recognizing a transformed probe coming from a single device. In practice, after having trained the net on data coming from a single device, for example D01, in the test phase the net is evaluated on data coming only from D01. For each device/net pair, we created 1000 test samples using the same procedure described in Section 3.1 applying different scales and rotations randomly selected at runtime. The predicted Δ was then used to define (using Eqs. (1)–(3)) a similarity transform, which in turn was used to register the residual noise on the fingerprint. Then PCE was evaluated and a positive result was recorded whenever its value was higher than $\tau_{60} \doteq 60$ or $\tau_{100} \doteq 100$. Table 1 shows the results thus obtained: In most of the cases, more than 80% of the test probes were correctly identified, with an average of 84,62% for τ_{60}, and of 81,22% for τ_{100}.

Table 1
Results on single device attribution for 1000 test example, using $\tau_{60} = 60$ and $\tau_{100} = 100$ as thresholds on the PCE score.

Device	τ_{60}	τ_{100}
D01 - Samsung Galaxy S3 Mini	87.6%	84.3%
D02 - Apple iPhone 4s	60.2%	51.9%
D03 - Huawei P9	59.5%	52.3%
D04 - LG D290	71.0%	64.1%
D05 - Apple iPhone 5c	90.9%	88.9%
D07 - Lenovo P70A	91.4%	88.6%
D08 - Samsung Galaxy Tab 3	82.4%	79.0%
D09 - Apple iPhone 4	93.6%	92.2%
D11 - Samsung Galaxy S3	83.6%	80.1%
D12 - Sony Xperia Z1 Compact	86.0%	83.5%
D13 - Apple iPad 2	89.7%	87.8%
D15 - Apple iPhone 6	88.5%	85.1%
D16 - Huawei P9 Lite	86.4%	84.6%
D17 - Microsoft Lumia 640 LTE	91.5%	90.2%
D19 - Apple iPhone 6 Plus	89.2%	86.2%
D20 - Apple iPad Mini	92.4%	91.2%
D21 - Wiko Ridge 4G	68.4%	60.0%
D22 - Samsung Galaxy Trend Plus	91.9%	90.1%
D23 - Asus Zenfone 2 Laser	92.5%	90.9%
D24 - Xiaomi Redmi Note 3	79.1%	74.2%
D25 - OnePlus A3000	85.3%	82.3%
D27 - Samsung Galaxy S5	84.6%	81.4%
D28 - Huawei P8	84.5%	81.5%
D29 - Apple iPhone 5	86.8%	83.7%
D30 - Huawei Honor 5c	91.0%	88.0%
D31 - Samsung Galaxy S4 Mini	83.5%	79.8%
D32 - OnePlus A3003	88.1%	85.1%
D33 - Huawei Ascend	81.2%	77.7%
D35 - Samsung Galaxy Tab A	93.1%	90.8%

Since VISION includes multiple devices of the same model, we tested how a net trained on one device would work on probes coming from another device *of the same model*. In particular, we evaluated the net trained on D01 on transformed probes coming from D26, another Samsung Galaxy S3 Mini. The percentage of correct detection was, in this case, respectively 38.7% with τ_{60}, and 33.1% with τ_{100}. Similarly, testing the D15 net on D06 probes (both Apple iPhone 6) yielded 18.1% with τ_{60}, and 14.2% with τ_{100}. These results suggest that the net is able to capture PRNU characteristics that are specific to a given individual device, and can therefore dis-

[1] Note that, in the VISION dataset, some of the acquired devices are of the same model (e.g. D02 and D10 are both Apple iPhone 4s): in this case, we selected for training and testing the device with the highest number of available probes (i.e., between D02 and D10, we used the former, since it has more natural images available).



(a) (b)

Fig. 5. Confusion matrices for multiple device attribution using τ_{60} (see text for details): (a) results for single patch sampling, and (b) results for triple patch sampling strategies. (Best viewed in color and zoomed in)

criminate well between two devices sharing the same model. Such net selectivity helps to keep low the number of false positives, as will be shown in the next Section, reporting experiments on multiple device attribution.

4.3.3. Multiple device attribution

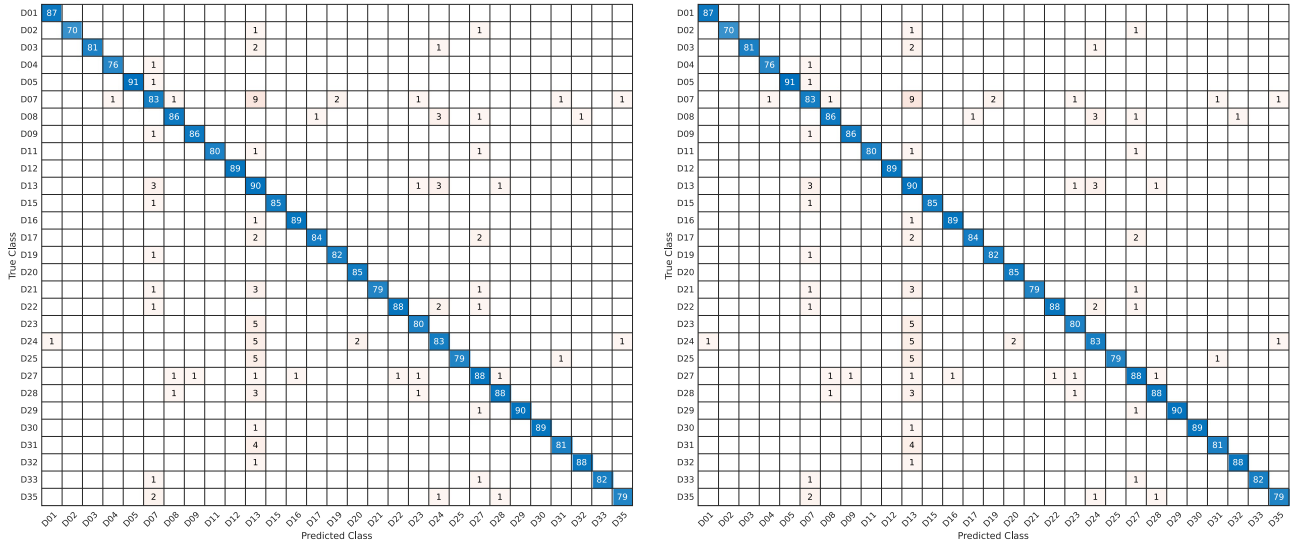
In this second test the performance of the trained networks in a multiple device attribution setup was assessed. Given a set of devices for which both the fingerprint and the trained net are available, this test evaluates the net's capability at selecting the correct source device for a scaled and rotated probe coming from any of the devices in the considered set. To accomplish such objective, the probe is compared against all the devices by firstly passing through the specific net to recover the transformation parameters, and then computing the PCE between the fingerprint and the registered residual noise. In this way, both True Positive (TP) and False Positive (FP) rates can be evaluated. For each considered device, 10 probes were selected and each probe was randomly transformed using 10 different scales and rotation angles, thereby obtaining 100 test cases for each device. This resulted in a total of 2900 test examples for the 29 devices from the VISION dataset. For each test example, 29 different PCE scores (one for each device in the dataset) were obtained.

Fig. 5a shows the confusion matrix obtained when *a single patch* from the probe image was randomly selected and the threshold was τ_{60}. The entry (i, j) of this matrix indicates how many times a probe from device i was attributed to device j (i.e., the jth PCE score was maximum among the 29 PCE scores computed). As a result, the TPs and FPs are respectively the diagonal $(i=j)$ and extra-diagonal $(i\neq j)$ entries of the matrix. If none of the 29 PCE scores was above the threshold, the probe was labelled as Not Assigned (NA) (this is why the matrix rows can add up to a number less than 100, which is the number of test cases for each device). Averaging the results over all devices, the performance using a single patch and τ_{60} was 72.0% TP, 5.3% FP, 22.7% NA. While the ratio between TP and FP is not at all bad, the single patch strategy for device attribution gives rise to a relatively large number of NA. Hence, in order to improve the performance, alternative device attribution strategies were devised and tested.

Triple patch strategy Instead than using a single patch, three patches for each probe are extracted and evaluated. All the three patches are then passed to the nets and three distinct PCE scores are computed. Finally, the highest PCE among the three is selected. As can be seen in Fig. 5b, using this approach gives better results than with a single patch. The average percentages over all devices are respectively 87.4% TP, 4.5% FP, 8.1% NA. It can be noticed that the number of NA probes was significantly reduced. However this strategy is three times slower than the single patch approach.

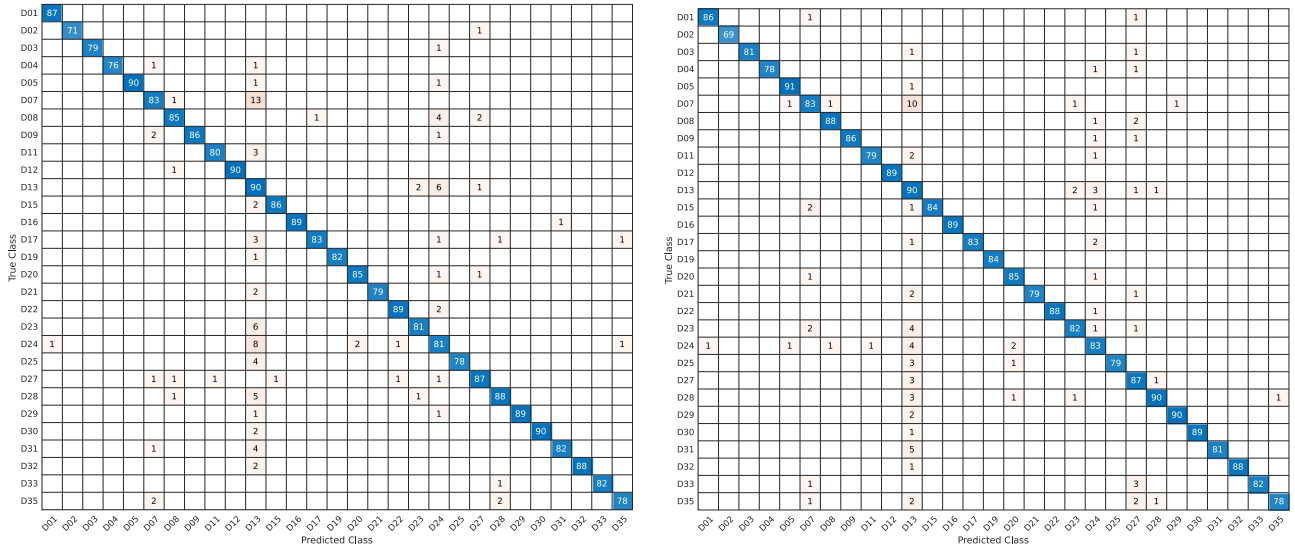
Multi-patch statistical analysis Since the forward pass of the net used to retrieve the transformation parameters is very fast compared to the time required to compute the PCE, alternative strategies can be designed based on a statistical analysis of the prediction results obtained with a large number N of input patches extracted from the probe at hand. Each patch is analyzed by the net and the scale and rotation parameters are recorded. This gives rise to two sets of N elements: S, with all the estimated scales and R, with the rotation angles. Then, robust statistical measures $\mathcal{M}$ are applied to S and R in order to extract a single pair of scale and angle estimates $(\hat{\sigma}, \hat{\theta})$ that summarize the entire set of N patches. In particular, two measures were tested: the *univariate median* $\mathcal{M}_u$, and the *geometric median* $\mathcal{M}_g$. Using the former, scale and rotation angles are used separately, i.e. $\hat{\sigma} = \mathcal{M}_u(\mathrm{S})$ and $\hat{\theta} = \mathcal{M}_u(\mathrm{R})$. Differently, when using the geometric median the two sets are considered jointly, i.e. $(\hat{\sigma}, \hat{\theta}) = \mathcal{M}_g(\mathrm{S}, \mathrm{R})$. Once obtained the $(\hat{\sigma}, \hat{\theta})$ parameters, a similarity transform is defined to register the probe, and a single PCE is evaluated, thus reducing the overall computational time w.r.t. the triple patch strategy. Results for the univariate median are reported in Fig. 6a for $N = 100$, and in Fig. 6b for $N = 1000$. Similarly, Fig. 7a and Fig. 7b reports, respectively, the results for the geometric median with $N = 100$ and $N = 1000$ patches. The two statistical measures yield similar results, with slightly lower FPs but also with lower TPs and higher NAs w.r.t. the triple patch strategy. Moreover, the univariate median obtains the same results using 100 or 1000 patches, while the geometric median slightly improves as the number of patches increases.

The results obtained with τ_{100} with all stretegies are reported in Table 2. The table shows only the diagonal entries of the confusion matrix, since with this higher threshold almost all false positives are correctly filtered out (the few residual FPs appear in brackets). However, as expected, by increasing the threshold, more NAs and less TPs are obtained. In general the triple patch strategy is the best performing, obtaining the highest TPs for 26 out of



(a) (b)

Fig. 6. Confusion matrices for multiple device attribution using τ_{60} and the univariate median strategy ($\mathcal{M}_u$) (see text for details). In (a) results with 100 patches, and in (b) for 1000 patches. (Best viewed in color and zoomed in).



(a) (b)

Fig. 7. Confusion matrices for multiple device attribution using τ_{60} and the geometric median strategy ($\mathcal{M}_g$) (see text for details). In (a) results with 100 patches, and in (b) for 1000 patches. (Best viewed in color and zoomed in).

Table 2
Correct detection rate for the multiple device attribution test, using τ_{100}. In bold the highest rate for each device. In brackets are reported the false positives. Note that, D07 is confused with D13 both for the Single patch sampling and the Geometric Median with 100 patches ($\mathcal{M}_g^{100}$).

	D01	D02	D03	D04	D05	D07	D08	D09	D11	D12	D13	D15	D16	D17	D19	D20	D21	D22	D23	D24	D25	D27	D28	D29	D30	D31	D32	D33	D35
Single	77	49	38	40	85	72 (1)	80	84	54	73	85	65	74	77	76	83	56	79	70	57	63	72	73	62	84	72	83	60	72
Triple	**90**	**67**	66	66	**92**	**85**	**90**	**91**	**78**	**93**	**91**	82	**89**	**88**	**92**	**91**	**80**	**89**	**84**	**81**	**84**	**89**	**92**	**92**	**90**	**88**	**92**	**86**	**82**
$\mathcal{M}_u^{100}$	86	62	78	72	90	81	84	86	**78**	89	90	84	**89**	82	80	84	78	87	80	79	76	85	88	89	88	80	87	81	77
$\mathcal{M}_u^{1000}$	86	62	78	72	90	81	84	86	**78**	89	90	84	**89**	82	80	84	78	87	80	79	76	85	88	89	88	80	87	81	77
$\mathcal{M}_g^{100}$	86	63	**79**	**73**	89	82 (1)	84	86	**78**	89	89	**85**	**89**	82	80	82	78	87	80	80	74	86	88	87	88	80	87	81	77
$\mathcal{M}_g^{1000}$	85	63	**79**	72	89	82	86	85	77	89	89	84	87	82	81	84	79	88	79	**81**	76	86	89	86	89	80	87	80	77

29 devices. However, for D03, D04, and (less significantly) for D15, multi-patch approaches achieved higher scores. Note that a similar behaviour for D03 and D04 is found also for $\tau = 60$. This is probably due to the peculiar characteristics of these devices, where the intensity of the PRNU is likely not uniform throughout the image. Hence, by sampling a larger number of patches, there is a higher probability to select a meaningful patch over which correct transformation parameters can be estimated. This hypothesis however would require a deeper analysis that is out of the scope of this paper and will be left for future investigations.

Table 3 summarizes the results obtained with the different device attribution strategies, using both the thresholds τ_{60} and τ_{100}.

Table 3
Resume of the multiple device attribution using different strategies, for τ_{60} and τ_{100}. Results are expressed as percentage; in brackets we reported the number of FP when 0.0% are achieved but some false positive still happen.

	τ_{60}			τ_{100}		
	TP	FP	NA	TP	FP	NA
Single	72.0%	5.3%	22.7%	69.5%	0.0% (1)	30.5%
Triple	87.4%	4.5%	8.1%	85.5%	0.0%	14.5%
$\mathcal{M}_u^{100}$	84.1%	3.5%	12.4%	82.4%	0.0%	17.6%
$\mathcal{M}_u^{1000}$	84.1%	3.5%	12.4%	82.4%	0.0%	17.6%
$\mathcal{M}_g^{100}$	83.9%	3.8%	12.2%	82.4%	0.0% (1)	17.6%
$\mathcal{M}_g^{1000}$	84.2%	3.4%	12.4%	82.4%	0.0%	17.6%

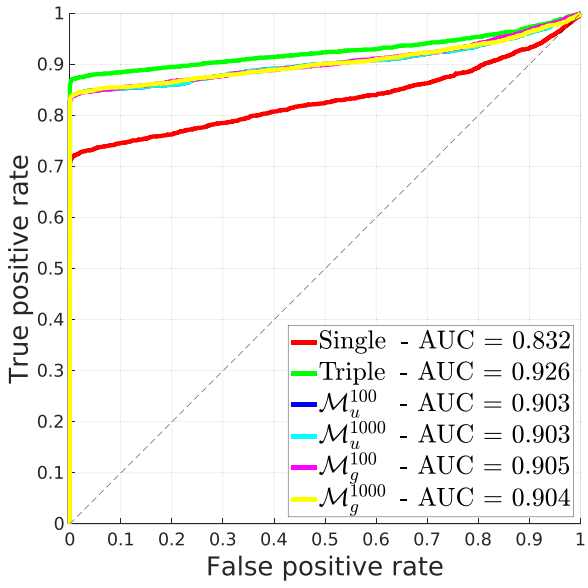


Fig. 8. ROC curves for multiple device attribution using our net and the six different strategies on 29 devices of the VISION dataset. (Best viewed in color and zoomed in).

Results are expressed in percentage w.r.t. overall number of test examples. TP columns were obtained by accumulating the values in the diagonal of the confusion matrices, FP columns those in the non-diagonal entries. NA columns take into account the probes that were neither TPs nor FPs. Whatever the thresholds, the worst performing strategy is the single patch strategy, which can be assumed as the baseline of the method. With τ_{60}, the best performing strategy in terms of both TP and NA percentage is the triple patch strategy (87.4% and 8.1%, respectively), followed by the geometric median using 1000 patches (84.2% and 12.4%). The latter strategy also has the best performance on FPs (3.4%). The second best on FPs is the univariate median, whose performance is the same with either 100 or 1000 patches. Concerning τ_{100}, all the strategies obtain negligible FPs, at the expense of lower TPs and higher NAs. The triple patch strategy is confirmed as the best of all, while the strategies based on robust statistics have almost identical, suboptimal performances.

In order to get an insight into the results for any possible value of the threshold, the Receiver Operating Characteristic (ROC) curve and the relative Area Under the Curve (AUC) obtained with the different approaches were also computed. For this experiment, the 2900 PCEs computed for all the test examples (and not only the maximum PCE for each test example, as done before with confusion matrices) were used, together with the true device attribution labels. As can be seen in Fig. 8, all the strategies show good performance with a high TP rate (greater than 0.7) even at extremely low FP rates. Among the proposed strategies, triple patch sampling is confirmed as the best performing one with an AUC of 0.926, followed by the robust statistic strategies that achieve similar results, with AUCs higher than 0.9. The baseline single patch sampling, with an AUC of 0.832, is the fastest but also the worst performing among the tested strategies.

4.4. Comparative evaluation

The proposed solution was compared against the brute-force search approach [18], the particle swarm method presented in [24], and the solution presented in [25] (that will be indicated as SCV4SSV).

For the brute-force case, given a probe, all the combinations of scales and rotation angles are evaluated by first registering the probe and then computing the PCE. The scale-angle pair obtaining the maximum PCE score is then selected. In order to be fair, we limited the search space to the values covered during the training of the net, i.e. [0.85,1.15] with steps of 0.01 for the scale, and $[-0.15, 0.15]$ degrees with steps of 0.01 for the rotation angle. Concerning the particle swarm, we adapted the source code released by the authors in order to work on single images instead than on video frames. Differently from the brute-force approach, this method performs a smarter search of the parameters by exploiting the particle swarm optimizer, that in each iteration focuses parameter search on the most promising particles. In our tests we used the default parameters selected by the authors, i.e., the maximum number of used particles $N_p = 50$ and the maximum number of iterations $\max_{it} = 50$ (see [24] for further details). As with the brute-force, also for the particle swarm we limited the search space to [0.85,1.15] for the scale, and to $[-0.15, 0.15]$ degrees for the rotation angle. SCV4SSV was also tested using an adaptation of the authors' source code to make it work on single images instead than on video frames. Once obtained the PRNU of the transformed probe, it is centrally cropped with a fixed window. Then, candidate transformations are searched for. To avoid extreme interpolation effects, the authors select a sub-set of transformation such that the shift of corner pixels of the cropped PRNU are limited in a given window. Moreover, to speed up the search, two solutions are provided: the *Three-Level Hierarchical Grid Search*, and its constrained version (which is the one implemented in the code). Finally, a validation procedure is carried out to exclude transformations not satisfying three user defined thresholds, namely PCE_{vld}, n_{sub}, and PCE_{sub} (see [25] for details). In order to select an appropriate value for last three thresholds, several trials were performed. Using the values provided in the code, i.e. $PCE_{vld} = 42$, $n_{sub} = 2$, and $PCE_{sub} = 2$, only 13% of probes were analysed. The best performances were achieved using $PCE_{vld} = 42$, $n_{sub} = 1$, and $PCE_{sub} = 2$, however 70% of probes were still discarded. In order to be fair with the other methods, that do not discard any probe, we decided to show results for the best performing trial where all the probes were analysed: This output was obtained by setting $PCE_{vld} = 21$, $n_{sub} = 0$, and $PCE_{sub} = 1$, with a minimal performance loss w.r.t. the best setup (the difference between their AUCs was 0.045 only).

Note that the high number of PCEs to be evaluated—by all the compared methods, in particular by the brute-force—is a strong limitation, since it requires huge computational times, so in this test we were forced to limit the analysis to only three devices. As done in the previous tests, 100 test cases for each device were considered. In Fig. 9 confusion matrices obtained with τ_{60} and τ_{100} are shown for the brute-force approach (Fig. 9a and 9e), the particle swarm method (Fig. 9b and 9f), SCV4SSV (Fig. 9c and 9g), and our single patch sampling, limited to the three devices considered (see Fig. 9d and 9h). As can be seen, our method, despite using the worst among the proposed strategies, obtains similar or better results in terms of TPs, FPs and NAs than its competitors.

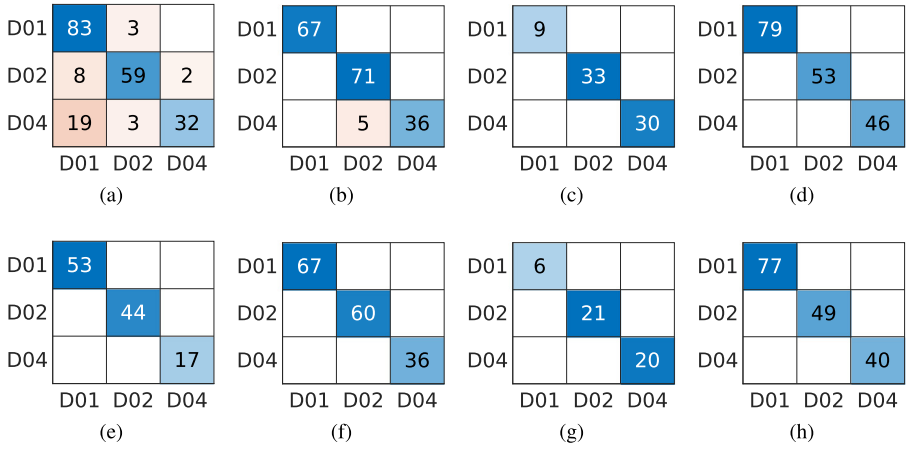


Fig. 9. Confusion matrices comparing the performance of the brute force approach with τ_{60} (a) and with τ_{100} (e). Similarly, in (b) and (f) are shown the results of the particle swarm method, and in (c) and (g) results for SCV4SSV, while in (d) and (h) results with our single patch strategy are reported. The *Predicted Class* run along the columns, while the *True Class* along the rows. (Best viewed in color and zoomed in).

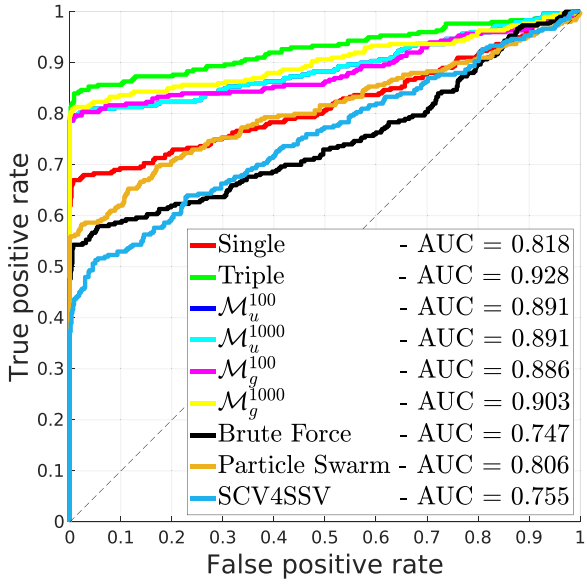


Fig. 10. ROC curves comparing our approaches w.r.t. brute force method, particle swarm solution, and SCV4SSV, using three devices (D01, D02, and D04). AUC values are reported in the plot legends. (Best viewed in color and zoomed in).

Indeed, using τ_{60}, our solution achieves 178(60%) TPs, 0 FPs and 122(40%) NAs, brute-force achieves 174(58%) TPs, 35(12%) FPs and 91(30%) NAs, particle swarm achieves 174(58%) TPs, 5(1.7%) FPs and 121(40.3%) NAs, and SCV4SSV obtains 72(24%) TPs, 0 FPs and 228(76%) NAs. Results with τ_{100} are even more favourable toward our approach, that obtains 166(55%) TPs, 0 FPs and 134(45%) NAs, against 114(38%) TPs, 0 FPs and 186(62%) NAs of brute-force search, 163(54.3%) TPs, 0 FPs and 137(45.7%) NAs of particle swarm optimization, and 47(15.7%) TPs, 0 FPs and 253(84.3%) NAs of SCV4SSV.

Fig. 10 shows the ROC curves and the related AUCs for all the proposed strategies, the brute-force method, the particle swarm optimizer, and the SCV4SSV approach (for the three devices considered). All the evaluated methods obtain better results than the brute-force approach: SCV4SSV achieves only a slightly improvement w.r.t. the brute-force, while particle swarm obtains a better AUC, only slightly worse than our single patch solution. AUCs for our solutions are always higher than those of the compared methods, with the triple patch strategy achieving the best results overall.

The bad performance of the brute-force approach is, in our opinion, due to the selected steps on the scales and angles to be tested. Probably, choosing a finer step would increase the performance, since there would be a higher probability to select the exact scale-angle pair used to warp the probe. However, reducing the step would also increase dramatically the number of trials to be carried out, and as a consequence the computational time, to a point that the problem would become intractable. Relating the AUC with the results shown in Fig. 9c and 9g, SCV4SSV seems to output low PCE scores, but it is able to provide relatively higher PCEs when comparing the residual noise of the correct device with its fingerprint, such to obtain a better AUC score w.r.t. the brute-force. On the other hand, the particle swarm approach is able to better identify the correct scale and angle parameters, since it focuses on the more promising parameters with an optimization strategy. Note that the particular transformation search strategies adopted by the particle swarm and SCV4SSV, not only provide better results, but also reduce the computational times considerably w.r.t. the brute-force. Still, our solutions achieve the best performance, since the trained nets are able to recover the transformation parameters both reliably and with extremely reduced computational times—as shown in the next Section.

4.4.1. Computational times

The computational times (expressed in seconds) for the proposed approaches, the brute-force method, the particle swarm solution, and SCV4SSV are shown in Table 4. All times were obtained on a PC equipped with an Intel Core i7-8700 CPU@3.20 GHz, with 32 GB of RAM. The neural net was implemented in Python using the Tensorflow library and can run either on an NVIDIA TITAN Xp GPU, with 12 GB of RAM, or on the CPU. A single forward pass on the trained net requires 0.03 s, if run on the CPU, 0.002 s if run on the GPU. Warping the probe takes around 0.02 s, while computing the PCE requires a time proportional to the probe resolution: In our experiments an average time of 2.8 s was required. The geometric median takes around 0.01 s on 100 samples, and 0.4 s on 1000 samples. The univariate median takes instead 0.0001 s, both for 100 and 1000 samples. The brute-force approach tested 31 values both for the scale and angles (using the limits and the steps defined above), for a total of $31 \times 31 = 961$ tries. The particle swarm and SCV4SSV also performed multiple tests, however since

Table 4
Estimates of computational times for a single probe evaluation (expressed in seconds). For each method we reported the number of required operations and the average time used by each operation; For the net forward pass (and as consequence for the total times) we reported the GPU time and, in brackets, the CPU time. Total times are obtained by multiplication and sum of the single operations. Note that, for the Particle Swarm and SCV4SSV we do not know how many warping and PCE computations have been used, so we reported the total average time obtained experimentally.

	Forward Pass Time	Forward Pass Num	Warp Time	Warp Num	PCE Time	PCE Num	Geom Median 100 Time	Geom Median 1000 Time	Univ Median Time	Total Time
Single	0.002 (0.03)	1	0.02	1	2.8	1	–	–	–	2.822 (2.850)
Triple	0.002 (0.03)	3	0.02	3	2.8	3	–	–	–	8.466 (8.550)
$\mathcal{M}_u^{100}$	0.002 (0.03)	100	0.02	1	2.8	1	–	–	0.0001	3.020 (5.820)
$\mathcal{M}_u^{1000}$	0.002 (0.03)	1000	0.02	1	2.8	1	–	–	0.0001	4.820 (32.820)
$\mathcal{M}_g^{100}$	0.002 (0.03)	100	0.02	1	2.8	1	0.01	–	–	3.030 (5.830)
$\mathcal{M}_g^{1000}$	0.002 (0.03)	1000	0.02	1	2.8	1	–	0.4	–	5.220 (33.220)
Brute Force	–	–	0.02	961	2.8	961	–	–	–	2710.020
Particle Swarm	–	–	0.02	n.a.	2.8	n.a.	–	–	–	1022.231
SCV4SSV	–	–	0.02	n.a.	2.8	n.a.	–	–	–	122.939

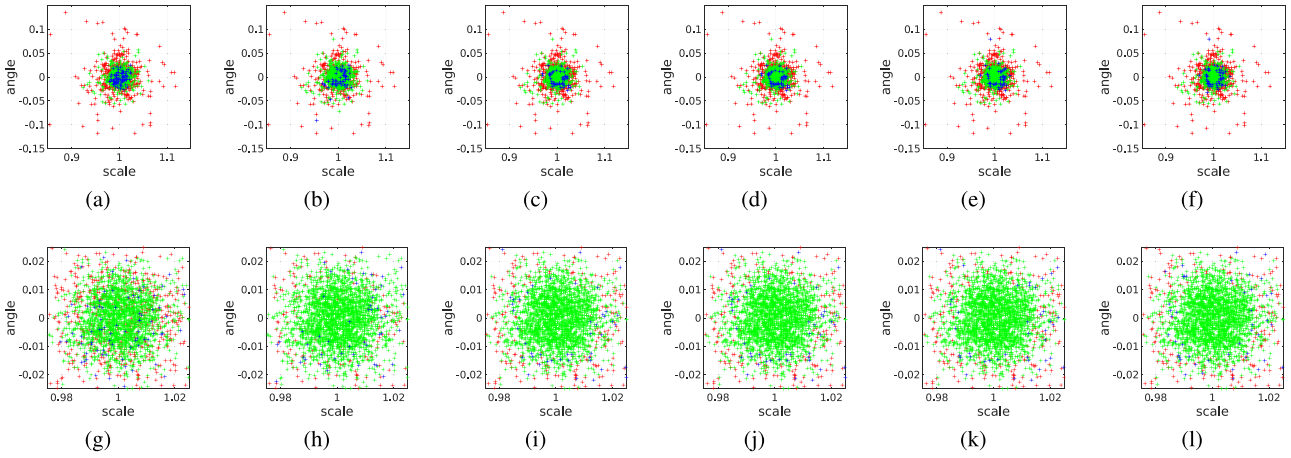


Fig. 11. Relation between the transformation parameters and the obtained PCE score. In all the plot, red crosses indicate PCE score inferior to 60, in blue if the PCE is between 60 and 100, in green PCE score higher than 100. Plots on the top row show the full parameter space, while on bottom a zoomed version (corresponding to the black bounding boxes) is presented. (a) and (g) are for the single patch case, while the triple patch case in shown in (b) and (h). Similarly, (c) and (i) are for $\mathcal{M}_u^{100}$; (d) and (j) are for $\mathcal{M}_u^{1000}$; (e) and (k) for $\mathcal{M}_g^{100}$, and finally (f) and (l) for $\mathcal{M}_g^{1000}$. (Best viewed in color and zoomed in).

their exact number is unknown, only the time required to evaluate a probe can be appreciated.

As can be seen in Table 4, the brute-force approach has extremely high computational times, due to the computation of 961 warps and PCEs. Particle swarm, using a smarter search of the registration parameters, is almost three times faster, and SCV4SSV, by carefully selecting which transformations to evaluate, works almost thirty time faster than brute-force. Our methods are way faster than their competitors, using only a few seconds per probe, especially if the GPU is available. Obviously, the three patch sampling is the most computationally demanding solution, since it requires the evaluation of three PCEs. However, if no GPU is available, the evaluation of 1000 patches (for both $\mathcal{M}_u^{1000}$ and $\mathcal{M}_g^{1000}$) is the most time consuming step.

4.4.2. Ablation study

Different net architectures were experimented in order to assess the best model to be used. In particular, an attempt was done to remove the max pooling operation after each convolutional block or replace it with the average pooling operation to assess if removing or changing the interpolation of the feature map could bring some positive effects. Deeper and wider net architectures were also tested by including additional convolutional blocks or by increasing to 128 the number of kernels in the first convolutional blocks. However, none of these changes improved significantly the results, so they were dropped, as they required higher training times, given the increased number of parameters to be tuned.

To gain a better understanding of the proposed approach, the relation between the transformation parameters (i.e., scale and angle) and the obtained PCE score after probe registration was also analyzed. To this aim, the results of Section 4.3.3, limited only to the cases where probe and reference come from the same device, were considered, and all the ground truth scales and angles and the respective PCE scores obtained with our six strategies (i.e. Single, Triple, $\mathcal{M}_u^{100}$, $\mathcal{M}_u^{1000}$, $\mathcal{M}_g^{100}$, and $\mathcal{M}_g^{1000}$) were retrieved. In Fig. 11 for each scale-angle pair the achieved results are shown, divided in three classes: (i) PCE $< \tau_{60}$ (red crosses), (ii) $\tau_{60} \leq$ PCE $< \tau_{100}$ (blue crosses), (iii) PCE $\geq \tau_{100}$ (green crosses). As can be noticed, stronger transformations on the marginal sides of the plots are those that achieve lower PCEs, hence are the most difficult cases. Notice also that, while in the single patch cases (Fig. 11a and 11g), some red crosses can be found also near the centre of the plots, with the other strategies most of the red crosses change to blue or green.

Similar conclusions can be drawn observing the 3D plots of Fig. 12. In this case the obtained PCE scores are used directly (without quantization) as real values, in order to obtain a

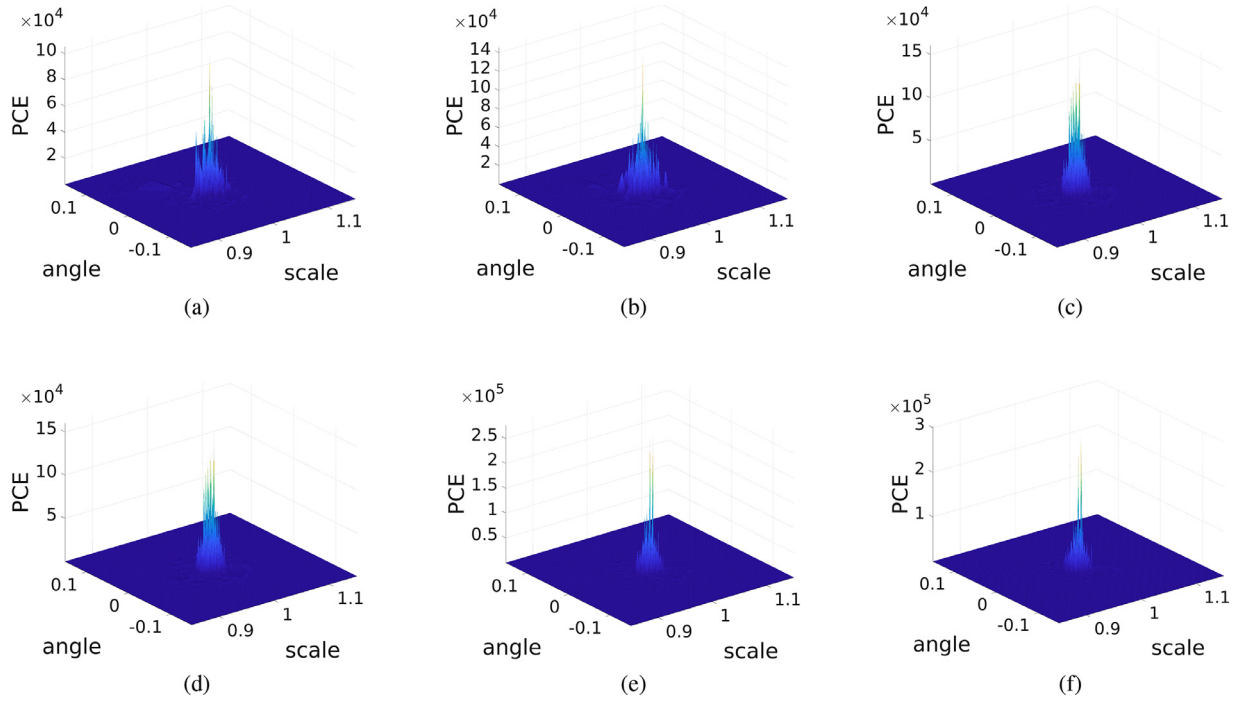


Fig. 12. Relation between the transformation parameters and the obtained PCE score presented as a 3D surface plot. (a) is for the single patch case, while the triple patch case in shown in (b). Similarly, (c) is for $\mathcal{M}_u^{100}$; (d) for $\mathcal{M}_u^{1000}$; (e) for $\mathcal{M}_g^{100}$, and finally (f) for $\mathcal{M}_g^{1000}$. (Best viewed in color and zoomed in).

3D surface plot. Higher PCEs are found near the center where the transform is close to the identity (i.e., unit scale and zero angle).

5. Conclusions

In this paper we presented a novel approach to solve the camera identification task when the probe image has undergone a scale and rotation transformation. Differently from the state of the art solutions, our methods do not test a high number of different registration possibilities. Using a trained convolutional network the proposed method is able to estimate sufficiently correct transformation values so as to let the correlation based PCE to work reliably. Several experiments carried out on a recent dataset of smartphone devices show the effectiveness of the proposed solution, both when focusing on a single device or working on multiple device attribution. In particular, the high true positive and the low false positive rates guarantee that our method can be used reliably in practical application scenarios. Moreover, at the cost of spending time to train the nets, the proposed solution, at test time, can output the registration parameters in just a few seconds, strongly reducing the time requirements w.r.t. the compared solutions, thus being particularly useful to analyze datasets with a huge number of probes. The main drawbacks of the proposed approach are two. Firstly, at this moment the approach was trained to reliably estimates similarity transformations only: An extension to deal with more general transformations (e.g., homographies) is planned for future development. Secondly, it requires to perform as many training sessions as the number of the distinct devices to test. However, this limitation is not so impacting on the method performance, since it has to be done only once, in an offline and automatic mode.

In future work, our single image approach will be extended to image sequences in order to work also with digitally stabilized videos. Also the introduction of correlation measures into the network loss function should be considered in order to better train the net for the specific problem of PRNU matching.

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.

Acknowledgment

The authors would gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] L. Verdoliva, Media forensics and deepfakes: an overview, IEEE J. Sel. Top. Signal Process. 14 (5) (2020) 910–932.
[2] P. Ferrara, T. Bianchi, A. De Rosa, A. Piva, Image forgery localization via fine–grained analysis of CFA artifacts, IEEE Trans. Inf. Forensics Secur. 7 (5) (2012) 1566–1577.
[3] M. Chen, J. Fridrich, M. Goljan, J. Lukas, Determining image origin and integrity using sensor noise, IEEE Trans. Inf. Forensics Secur. 3 (1) (2008) 74–90.
[4] V.I. Ivanov, J.S. Baras, Authentication of area fingerprint scanners, Pattern Recognit. 94 (2019) 230–249.
[5] D. Bhardwaj, V. Pankajakshan, A JPEG blocking artifact detector for image forensics, Signal Process. Image Commun. 68 (2018) 155–161.
[6] Q. Liu, An approach to detecting JPEG down-recompression and seam carving forgery under recompression anti-forensics, Pattern Recognit. 65 (2017) 35–46.
[7] E. Kee, J.F. O'Brien, H. Farid, Exposing photo manipulation with inconsistent shadows, ACM Trans. Graph. 32 (3) (2013).
[8] T. Carvalho, C. Riess, E. Angelopoulou, H. Pedrini, A. de Rezende Rocha, Exposing digital image forgeries by illumination color classification, IEEE Trans. Inf. Forensics Secur. (2013) 1182–1194.
[9] M. Fanfani, F. Bellavia, M. Iuliani, A. Piva, C. Colombo, FISH: face intensity-shape histogram representation for automatic face splicing detection, J. Vis. Commun. Image Represent. 63 (2019) 102586.
[10] B. Peng, W. Wang, J. Dong, T. Tan, Position determines perspective: investigating perspective distortion for image forensics of faces, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 1813–1821.
[11] B. Peng, W. Wang, J. Dong, T. Tan, Image forensics based on planar contact constraints of 3D objects, IEEE Trans. Inf. Forensics Secur. 13 (2) (2018) 377–392.
[12] M. Iuliani, M. Fanfani, C. Colombo, A. Piva, Reliability assessment of principal point estimates for forensic applications, J. Vis. Commun. Image Represent. 42 (2017) 65–77.

[13] M. Fanfani, M. Iuliani, F. Bellavia, C. Colombo, A. Piva, A vision-based fully automated approach to robust image cropping detection, Signal Process. Image Commun. 80 (2020) 115629.
[14] J. Lukas, J. Fridrich, M. Goljan, Digital camera identification from sensor pattern noise, IEEE Trans. Inf. Forensics Secur. 1 (2) (2006) 205–214.
[15] M. Kirchner, C. Johnson, Spn-cnn: boosting sensor-based source camera attribution with deep learning, in: Proceedings of the IEEE International Workshop on Information Forensics and Security (WIFS), IEEE, 2019, pp. 1–6.
[16] S. Mandelli, D. Cozzolino, P. Bestagini, L. Verdoliva, S. Tubaro, Cnn-based fast source device identification, IEEE Signal Process. Lett. 27 (2020) 1285–1289.
[17] Manisha, A. Karunakar, C.-T. Li, Identification of source social network of digital images using deep neural network, Pattern Recognit. Lett. 150 (2021) 17–25.
[18] M. Goljan, J.J. Fridrich, Camera identification from cropped and scaled images, in: Proceeding of the Security, Forensics, Steganography, and Watermarking of Multimedia Contents X, 2008.
[19] D. Shullani, M. Fontani, M. Iuliani, O.A. Shaya, A. Piva, VISION: a video and image dataset for source identification, EURASIP J. Inf. Secur. 2017 (1) (2017) 15.
[20] T. Höglund, P. Brolund, K. Norell, Identifying camcorders using noise patterns from video clips recorded with image stabilisation, in: Proceedings of the 7th International Symposium on Image and Signal Processing and Analysis (ISPA), 2011, pp. 668–671.
[21] S. Taspinar, M. Mohanty, N. Memon, Source camera attribution using stabilized video, in: Proceedings of the IEEE International Workshop on Information Forensics and Security (WIFS), 2016, pp. 1–6.
[22] M. Iuliani, M. Fontani, D. Shullani, A. Piva, Hybrid reference-based video source identification, Sensors 19 (3) (2019).
[23] S. Taspinar, M. Mohanty, N. Memon, Camera identification of multi-format devices, Pattern Recognit. Lett. 140 (2020) 288–294.
[24] S. Mandelli, P. Bestagini, L. Verdoliva, S. Tubaro, Facing device attribution problem for stabilized video sequences, IEEE Trans. Inf. Forensics Secur. 15 (2019) 14–27.
[25] E. Altinisik, H.T. Sencar, Source camera verification for strongly stabilized videos, IEEE Trans. Inf. Forensics Secur. 16 (2021) 643–657.
[26] F. Bellavia, M. Iuliani, M. Fanfani, C. Colombo, A. Piva, PRNU pattern alignment for images and videos based on scene content, in: Proceeding of the IEEE International Conference on Image Processing (ICIP), 2019.
[27] F. Bellavia, M. Fanfani, C. Colombo, A. Piva, Experiencing with electronic image stabilization and PRNU through scene content image registration, Pattern Recognit. Lett. (2021).
[28] M. Goljan, J. Fridrich, Sensor fingerprint digests for fast camera identification from geometrically distorted images, in: Media Watermarking, Security, and Forensics 2013, volume 8665, SPIE, 2013, pp. 85–94.
[29] R. Li, C.-T. Li, Y. Guan, Inference of a compact representation of sensor fingerprint for source camera identification, Pattern Recognit. 74 (2018) 556–567.
[30] C. Liu, M. Kirchner, Cnn-based rescaling factor estimation, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, 2019, pp. 119–124.
[31] B. Bayar, M.C. Stamm, A generic approach towards image manipulation parameter estimation using convolutional neural networks, in: Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, 2017, pp. 147–157.
[32] A. Maier, B. Lorch, C. Riess, Toward reliable models for authenticating multimedia content: detecting resampling artifacts with Bayesian neural networks, in: Proceedings of the IEEE International Conference on Image Processing (ICIP), 2020, pp. 1251–1255.
[33] D. DeTone, T. Malisiewicz, A. Rabinovich, Deep image homography estimation, arXiv eprint 1606.03798 (2016). 1606.03798.
[34] C. Harris, M. Stephens, A combined corner and edge detector, in: Proceedings of the Alvey Vision Conference, 1988, pp. 147–152.
[35] S. Baker, A. Datta, T. Kanade, Parameterizing Homographies, Technical Report, CMU-RI-TR-06-11, Carnegie Mellon University, Pittsburgh, PA, 2006.
[36] E. Goceri, Deep learning based classification of facial dermatological disorders, Comput. Biol. Med. 128 (2021) 104118.
[37] E. Goceri, Diagnosis of Alzheimer's disease with Sobolev gradient-based optimization and 3D convolutional neural network, Int. J. Numer. Methods Biomed. Eng. 35 (7) (2019a) e3225.
[38] E. Goceri, Analysis of deep networks with residual blocks and different activation functions: classification of skin diseases, in: Proceedings of the Ninth International Conference on Image Processing Theory, Tools and Applications (IPTA), 2019b, pp. 1–6.

Marco Fanfani received his Ph.D. degree in multimedia analysis from the University of Florence in 2015. From 2012 he has been working at the Computational Vision Group of the Department of Information Engineering of the University of Florence. His research activities focus on computer vision and media forensics.

Alessandro Piva is Associate Professor at the University of Florence. His research interests lie in the areas of Information Forensics and Security, and Image and Video Processing. In the above topics he has been co-author of more than 50 papers published in international journals. He is IEEE Fellow.

Carlo Colombo holds a PhD in Robotics (1996) from the Sant'Anna School of University Studies and Doctoral Research, Pisa, Italy. He is professor of Computational Vision at the Department of Information Engineering, University of Florence. He has authored more than 100 papers and has been general chair of ECCV 2012.