CNN-based fast source device identification

Sara Mandelli, Davide Cozzolino, *Member, IEEE,* Paolo Bestagini, *Member, IEEE,*
Luisa Verdoliva, *Senior Member, IEEE,* and Stefano Tubaro, *Senior Member, IEEE*

***Abstract*—Source identification is an important topic in image forensics, since it allows to trace back the origin of an image. This represents a precious information to claim intellectual property but also to reveal the authors of illicit materials. In this paper we address the problem of device identification based on sensor noise and propose a fast and accurate solution using convolutional neural networks (CNNs). Specifically, we propose a 2-channel-based CNN that learns a way of comparing camera fingerprint and image noise at patch level. The proposed solution turns out to be much faster than the conventional approach and to ensure an increased accuracy. This makes the approach particularly suitable in scenarios where large databases of images are analyzed, like over social networks. In this vein, since images uploaded on social media usually undergo at least two compression stages, we include investigations on double JPEG compressed images, always reporting higher accuracy than standard approaches.**

I. Introduction

In this paper, we tackle the problem of image source device identification. State-of-the-art solutions rely on photo response non-uniformity (PRNU) [1], a characteristic noise trace left by the camera sensor on all acquired images. PRNU is caused by imperfections in the sensor manufacturing process and represents a unique pattern noise associated to a certain device. It can be used both to trace the exact origin of an image and to establish its integrity [2], [3].

The classic pipeline is based on the availability of a number of images coming from the same device in order to carry out a reliable PRNU estimation. Then, the image under test can be compared with the sensor fingerprint through denoising and peak to correlation energy (PCE) computation [4]. However, computing the PCE between a query image noise and a fingerprint might be computationally expensive, and may become a major bottleneck when source identification requires scanning huge databases of device fingerprints [5], [6], [7]. With the increasing resolution of digital sensors, the search time may become prohibitive even for a moderate fingerprint database.

S. Mandelli, P. Bestagini and S. Tubaro are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano 20133, Italy (e-mail: name.surname@polimi.it).

D. Cozzolino is with the Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione, University Federico II of Naples, 80125 Naples, Italy (e-mail: davide.cozzolino@unina.it).

L. Verdoliva is with the Dipartimento di Ingegneria Industriale, University Federico II of Naples, 80125 Naples, Italy (e-mail: verdoliv@unina.it).

This material is based on research sponsored by DARPA and AFRL under agreement number FA8750-16-2-0173. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA and AFRL or the U.S. Government. This work was supported by the PREMIER project, funded by the Italian Ministry of Education, University, and Research within the PRIN 2017 program. Hardware support was generously provided by the NVIDIA Corporation.

To manage large image databases some methods look for compressed or alternative representations of the sensor noise. One of the first methods introduces the concept of fingerprint digest: for each query fingerprint, only pixels with the largest magnitude are extracted [8]. Instead, in [9] the idea is to reduce the resolution of PRNU values by quantizing them into a single bit, while in [10] a composite fingerprint is introduced. In [11] compression is obtained using random projections followed by binary quantization of the projected values, and in [12] this technique is investigated for JPEG images. Alternatively, in [13] a tree-like search strategy is designed to split PRNU matching problem into a series of sub-problems.

We consider an alternative approach to compare PRNUs and test images by leveraging convolutional neural networks (CNNs) to achieve good performance on small image patches. Notice that relying on small patches makes PRNU-based forgery localization possible. Moreover, it enables analyses on social networks, where images are often available only in cropped formats. CNNs have been already successfully applied for camera model identification [14], [15], for tracing the origin of an image in a social network [16] or for PRNU anonymization [17]. However, less attention has been devoted to PRNU-based device identification. Only recently, in [18] sensor noise extraction has been improved by using a residual-based deep learning process that replaces the standard agnostic denoising procedure.

In this paper, we devise a data-driven approach for source identification based on PRNU and consider the scenario in which PRNU and images are geometrically synchronized. Differently from [18], our goal is to replace the PCE test in favour of an alternative method based on CNNs. Using deep learning to compare patches is a powerful approach [19]. It has been recently proposed for forensics applications in [20], [21], [22] by means of Siamese networks, in order to learn distinctive data representations that highlight camera-based artifacts. Unlike [19], our solution does not compare directly raw image pixels, but work on noise residuals. In our scenario, patches to be compared are extracted from the estimated PRNU and the noise residual of the test image, which do not possess any type of periodicity and are hard to compress. For this reason, we avoid embedded representations and rely on the whole original information, by proposing a network architecture that learns the best similarity function for source identification. Specifically, we design a 2-channel-based CNN, which, through proper training, is able to learn the best way to compare patches for device identification. We propose a shallow CNN architecture which turns out to be faster and more accurate than standard PCE approach. Furthermore, deeper CNNs are also investigated, enabling to achieve even better accuracy if processing time is not a constraint.

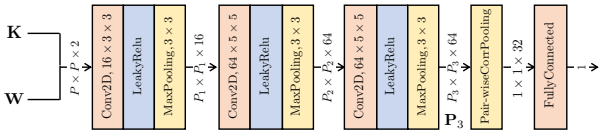

Fig. 1. Scheme of the proposed pair-wise correlation network (PCN).

II. Proposed method

Given a set of images coming from device d, it is possible to estimate the device PRNU $\mathbf{K}_d$ through denoising and maximum-likelihood estimation [1], [2]. Given a test image $\mathbf{I}$, the noise residual $\mathbf{W}$ can be extracted using a denoising-based procedure [2]. Standard PRNU-based device attribution methods compare $\mathbf{W}$ and $\mathbf{K}_d$ in order to detect a possible match [4]. In this work, we propose to perform this comparison through a CNN. For each pair of query image $\mathbf{I}$ and candidate device d, we feed $\mathbf{K}_d$ and $\mathbf{W}$ to the CNN, assuming they are geometrically synchronized. The network returns a CNN-based identification score C_s which is directly associated to the coherence between the image and the device. Analyzing C_s we can infer whether $\mathbf{I}$ belongs to d or not.

A. Network architectures

As the proposed 2-channel-based approach is independent from the used CNN, we investigate different architectures.

With the specific purpose of testing a shallow network enabling fast computations, the first proposed architecture is drawn using only 3 convolutional layers as depicted in Fig. 1. Then, a pair-wise correlation pooling layer and a fully-connected layer follow to obtain a single score C_s. The pair-wise correlation pooling layer is inspired to the conventional metrics used to identify the source device on images, i.e., normalized cross-correlation and PCE. Since these are based on cross-correlation between noise residual and PRNU, the pair-wise correlation pooling computes a correlation as well between pairs of input features. Being $\mathbf{P}_3$ the input of the pair-wise correlation pooling, the output of the layer is defined as:

$$[\mathbf{P}_4]_n = \frac{1}{P_3^2} \sum_{p_i=1}^{P_3} \sum_{p_j=1}^{P_3} [\mathbf{P}_3]_{p_i,p_j,2n-1} \cdot [\mathbf{P}_3]_{p_i,p_j,2n}, n \in [1, 32], \tag{1}$$

where n is the channel index. The pair-wise correlation pooling layer can be seen as a simplified version of bilinear pooling layer [23]. Actually, bilinear pooling computes the correlation between *all* the features maps, whereas pair-wise pooling evaluate this only for *adjacent* pairs of features. The main motivation behind pair-wise pooling is to provide fast computations. Indeed, evaluating the complete cross-correlation between all feature maps may be time-consuming.

The other networks we propose are known in literature as Inception-ResNet V2 [24] and EfficientNet models [25], which achieve very good results in computer vision tasks. Differently from the first solution which is trained from scratch, we initialize the network weights of EfficientNet and Inception models using the weights trained on Imagenet database [26].

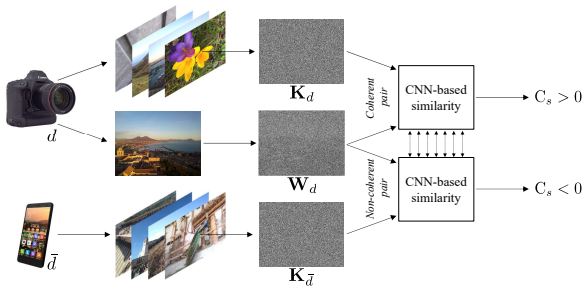

Fig. 2. Sketch of the proposed training strategy. For every device $d \in \mathcal{D}_\mathrm{T}$, the network is fed using noise residuals coming from device d, paired with PRNU $\mathbf{K}_d$ (coherent pair) and with PRNU $\mathbf{K}_{\bar{d}}$ coming from a different device (non-coherent pair). The CNN is able to learn a similarity measure C_s for the source identification task.

For brevity's sake, we hereafter investigate three EfficientNet models with increasing depth [25]: B0, B2 and B4. We denote the networks as PCN (Pair-wise Correlation Net), EFFB0, EFFB2, EFFB4 (EfficientNet) and INC (Inception).

B. Network training

To train the network, we assume that a set $\mathcal{D}_\mathrm{T}$ of N_d devices is available. For each device, we assume having a PRNU estimate $\mathbf{K}_d$ (obtained from a set of flat-field images) as well as a set of natural images. From each natural image belonging to device d, we extract the noise residue $\mathbf{W}_d$.

As shown in Fig. 2, the network is trained by concurrently feeding it with coherent pairs (i.e., PRNU and residue of the same device), and non-coherent pairs (i.e., PRNU and residue of diverse devices). In order to provide both coherent and non-coherent cases for each device inside one batch of data, we consider a batch-size of twice the cardinality of $\mathcal{D}_\mathrm{T}$. For every $d \in \mathcal{D}_\mathrm{T}$, the coherent pair $(\mathbf{K}_d, \mathbf{W}_d)$ is created by randomly picking $\mathbf{W}_d$ from the set of residues of device d, while the non-coherent pair $(\mathbf{K}_{\bar{d}}, \mathbf{W}_d)$ takes $\mathbf{K}_{\bar{d}}$ from another device $\bar{d}$ different from d, $\bar{d} \in \mathcal{D}_\mathrm{T}$.

As far as the loss function is concerned, we face the problem as a binary classification problem, adopting the standard sigmoid cross-entropy loss. In other words, label 1 is assigned to coherent pairs and label 0 otherwise. The network learns to return a score C_s, which is high for coherent pairs and low for non-coherent ones. We stop the training when the accuracy (i.e., the fraction of correct predictions) over the validation set is maximized. We use Adam optimization algorithm [27] with learning rate and patience initialized at 0.001 and 30, respectively, for a maximum number of 500 epochs.

C. Network deployment

Once the CNN is trained, we can use it in two scenarios:

- *closed-set* scenario: given an image, we aim at identifying the source among a finite pool of devices;
- *open-set* scenario: given an image and a candidate device, we aim at inferring if the device shot that image or not.

To solve these problems, we always feed the network with pairs of PRNU and image residue. Notice that, despite the CNN is trained over a specific set of devices $\mathcal{D}_\mathrm{T}$, it can be

used also to compare PRNUs and residues from devices never seen during training (i.e., $d \notin \mathcal{D}_{\mathrm{T}}$). As a matter of fact, the CNN learns how to compute a distance measure between a PRNU and a residue, thus it is not bounded to work on a closed-set of devices.

III. Experimental analysis

In this section, we first describe the dataset, the experimental setup and the evaluation metrics, then we report numerical results discussing the main achievements.

A. Dataset

The input to the network is always a pair of PRNU and image noise residue. These terms are assumed to be synchronized, i.e., no pixel misalignment is present. In our experiments, we always crop the central $P \times P$ pixel region from each image in order to limit network complexity. We let P range from $P = 80$ to $P = 720$ to study the relationship between image resolution and accuracy. As commonly done in CNN-based solutions, we normalize both PRNUs and residues by their standard deviation before feeding them to the network.

In order to test our method on a significant amount of devices, we consider both the Dresden image database [28] and the Vision dataset [29]. To avoid excessively old camera models, we only pick devices whose imagery have resolution greater than 720×720 pixels. For each device we exclusively investigate JPEG compressed images, as these represent the most frequent data for a forensics analyst to deal with.

Focusing on computing reliable PRNU fingerprints, we select devices if at least 25 JPEG images depicting scenes of flat surfaces are available. We end up with a total amount of 87 devices, 55 from Dresden and 32 from Vision. To estimate the PRNU we select only flat-field images, whereas, in order to fairly test our method, we make use of images showing natural scenes taken from indoor and outdoor scenarios. We work with more than 12000 images belonging to Dresden and almost 7000 images from Vision. On average, there are more than 200 natural images per device, randomly split in three disjoint datasets: training (50% of the images), validation (25%) and evaluation (remaining 25%).

The training dataset $\mathcal{D}_{\mathrm{T}}$ includes 20 devices from Vision and 16 devices from Dresden. It is worth mentioning that $\mathcal{D}_{\mathrm{T}}$ has been built using only devices of *different* models. In doing so, we avoid introducing an important constraint into the learning process, that is, training using cameras of the same model. Indeed, this may help enhancing the final performance, although requiring various instances of the same model at investigation side, which is actually unlikely to happen. Results are computed on images from the evaluation set of all 87 devices. Thereby, we are simulating a real situation in which the system is tested over *unknown* camera models as well.

B. Evaluation metrics

In the closed-set scenario, we test the residue of the query image against all 87 PRNUs. The camera returning the highest score C_s is associated to the image. To assess the attribution accuracy, we consider all query residues in the evaluation set for each device. The closed-set accuracy score A_{cs} is defined as the average fraction of correct predictions per device.

The open-set scenario is evaluated as a binary classification problem: distinguishing correlating pairs of PRNU-residue from non-correlating ones. Notice that we do not threshold the CNN score C_s associated to each test image, but we resort to receiver operating characteristic (ROC) curves. For each camera we consider all noise residues from that camera as positive samples, whereas the set of negatives includes all the residues not belonging to that camera. Each ROC curve draws the relationship between true positive rate (TPR) and false positive rate (FPR), computed over the set of available devices. As compact metric we use $\mathrm{AUC}_{\mathsf{os}}$, defined as the area under the curve (AUC) for the open-set problem. The goal is to achieve a high value of $\mathrm{AUC}_{\mathsf{os}}$, ideally 1.

C. Results

All tests have been run on a workstation equipped with one Intel® Core™ i9-9980XE (36 Cores @3.00 GHz), RAM 126 GB, one QUADRO P6000 (3840 CUDA Cores @1530 MHz), 24 GB, running Ubuntu 18.04.2. For the sake of clarity, we report at this link[1] the code used for the experiments, together with some additional materials to analyze results in details.

Closed-set scenario. Fig. 3 shows results for the closed-set scenario. A_{cs} is depicted as a function of the average computational time for testing one pair of PRNU-residue, and of the crop size P. For EFF and INC models we draw results only for $P \leq 320$, as larger image dimensions would require additional GPU memory. As state-of-the-art comparison, we perform the classic PCE test [4] between each noise residual in the evaluation dataset and all available PRNUs.

All the proposed CNNs show better accuracy than PCE, which achieves similar results only for the highest P values. EFF models always outperform results of INC, which actually requires more computation time as well. Whenever only very small patches are available (i.e., $P \leq 160$), EFFB4 represents a very good solution to achieve A_{cs} even larger than 80%. On the contrary, if time is a strong constraint, PCN is a good trade-off between accuracy and required computation time. For any patch-size, PCE and other CNNs require at least 4 times as much the computations of PCN.

For what concerns the required computational times, it is worth noting that the main advantage with respect to PCE is the possibility of feeding the CNNs with multiple pairs of PRNU-residue. These pairs can be processed in parallel by the GPU, at least until there is available memory for storing data. As a consequence, the larger the amount of candidate devices, the higher the temporal benefit compared to PCE. For instance, when testing a query image over 87 candidate cameras, we always take up less than 8 GB of GPU memory even using the INC configuration. Whether more devices were available, the average testing time would remain basically unchanged until reaching the maximum memory size.

[1] https://github.com/polimi-ispl/cnn-fast-sdi

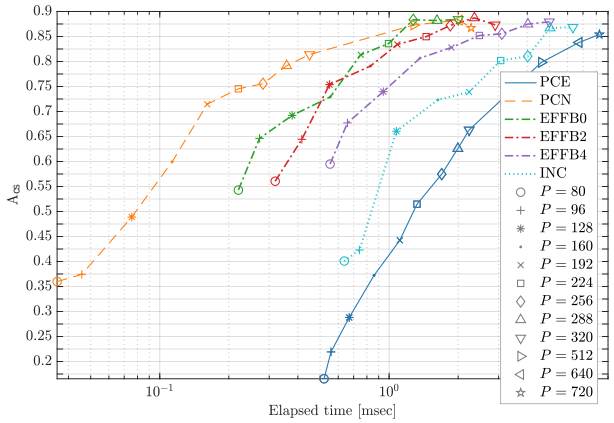

Fig. 3. Accuracy A_{cs} as a function of time [milliseconds] and crop size P.

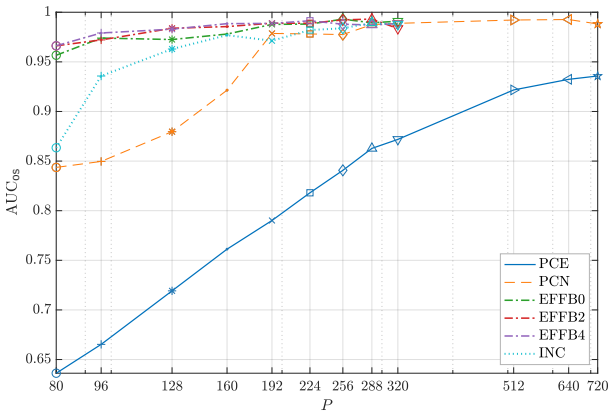

Fig. 4. Area under the ROC curve AUC_{os} as a function of crop size P.

Open-set scenario. Fig. 4 reports results for the open-set scenario. All proposed architectures outperform PCE for any P value, even though the highest AUC_{os} is obtained by EFF models. Notice that we are not analyzing AUC_{os} as a function of the required time, since the open-set scenario reduces investigations to only one pair of PRNU-residue. In this scenario, CNNs and PCE report comparable computational time, in the order of few milliseconds. However, as previously shown, whenever various images should be tested against the same PRNU, we could exploit the data parallelization property of GPU to test multiple images together, thus saving important computation time.

JPEG re-compression. In order to simulate scenarios where images underwent some post-processing operations, we test our method on double JPEG-compressed images. This step is purposely designed to simulate real case setups, in which only images from social networks (typically undergoing at least two compression steps) may be available. In this vein, we re-compress all the images using JPEG compression with quality factors 80 and 90. Then, we extract the noise residuals [2], defining them as $\mathbf{W}_{\text{JPEG}}$. As far as PRNUs are concerned, we can distinguish two scenarios: (i) whether flat-field single-compressed images are available, exploit the PRNUs $\mathbf{K}$ estimated from these; (ii) if only flat-field double-compressed images are available, use them to compute the PRNUs defined as $\mathbf{K}_{\text{JPEG}}$. As the analyst might have only double-compressed test data, it is important to understand whether she/he can exploit the network trained on single-compressed data, or some sort of domain adaptation must be applied. To this purpose, we evaluate different setups, considering training and testing on single and/or double-compressed data.

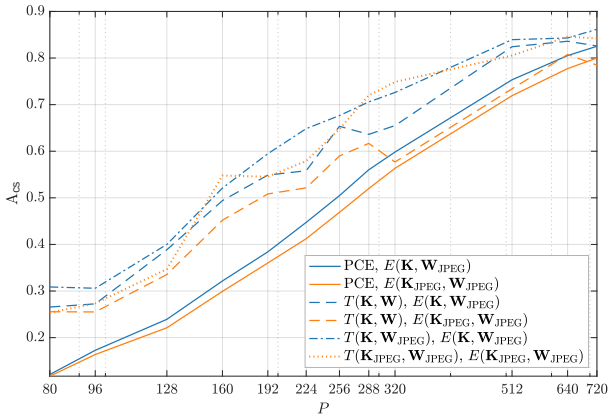

Fig. 5. Accuracy A_{cs} as a function of crop size P, considering double-JPEG compressed dataset with quality 90 and PCN network. $T(\cdot)$ and $E(\cdot)$ refer to training and evaluation datasets, respectively.

Experiments show that we can achieve also 20 percentage points of accuracy more than PCE for both JPEG qualities 80 and 90. This improvement is evident especially for the smallest patch-sizes (i.e., $P < 320$). For instance, Fig. 5 depicts the results achieved on the closed-set scenario using PCN configuration on double-compressed dataset with JPEG quality 90. As expected, it is worth noting that performing domain adaptation (i.e., training over the re-compressed dataset) always helps enhancing the performance with respect to training on the single-compressed dataset, even though every reported curve outperforms the corresponding PCE results.

IV. Conclusions

In this paper we propose a fast solution to the image device identification problem. In particular, we leverage PRNU and standard image noise residual extraction, but we substitute the correlation stage with a 2-channel-based CNN, able to learn the best similarity function for source device identification. Our method proves to be faster than PCE in case a large amount of potential provenance devices is investigated, requiring much less query image content to obtain enhanced attribution accuracy, given that images and PRNUs are assumed to be geometrically synchronized. Eventually, we evaluate the proposed methodology on images undergone to double-JPEG compression, in order to simulate more complex scenarios.

Given these promising results, our future work will focus on including the noise residual extraction step into the learning process as suggested by [18], as well as investigating how to deal with potential pixel misalignment between the image and the fingerprint. Moreover, video sequences will be the natural continuation of our research. Indeed, compression artifacts [30], [31] and stabilization issues [32], [33] make video source identification a complex task of undoubted interest.

REFERENCES

[1] J. Lukas, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 2, pp. 205–214, 2006.

[2] M. Chen, J. Fridrich, M. Goljan, and J. Lukás, "Determining image origin and integrity using sensor noise," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 1, pp. 74–90, 2008.

[3] G. Chierchia, G. Poggi, C. Sansone, and L. Verdoliva, "A Bayesian-MRF approach for PRNU-based image forgery detection," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 554–567, 2014.

[4] M. Chen, J. Fridrich, M. Goljan, and J. Lukáš, "Source digital camcorder identification using sensor photo response non-uniformity," in *Security, Steganography, and Watermarking of Multimedia Contents IX*, vol. 6505, 2007.

[5] X. Lin and C. Li, "Large-scale image clustering based on camera fingerprints," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 793–808, 2017.

[6] F. Marra, G. Poggi, C. Sansone, and L. Verdoliva, "Blind PRNU-based image clustering for source identification," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 9, pp. 2197–2211, 2017.

[7] Q. Phan, G. Boato, and F. De Natale, "Accurate and scalable image clustering based on sparse representation of camera fingerprint," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 7, pp. 1902–1916, 2018.

[8] M. Goljan and J. Fridrich, "Managing a large database of camera fingerprints," in *Media Forensics and Security II, SPIE Electronic Imaging Symposium*, 2010.

[9] S. Bayram, H. Sencar, and N. Memon, "Efficient Sensor Fingerprint Matching Through Fingerprint Binarization," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 4, pp. 1404–1413, 2012.

[10] ——, "Sensor fingerprint identification through composite fingerprints and group testing," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 597–612, 2015.

[11] D. Valsesia, G. Coluccia, T. Bianchi, and E. Magli, "Compressed fingerprint matching and camera identification via random projections," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 7, pp. 1472–1485, 2015.

[12] L. Bondi, P. Bestagini, F. Pérez-González, and S. Tubaro, "Improving PRNU compression through preprocessing, quantization, and coding," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 608–620, 2019.

[13] S. Taspinar, H. Sencar, S. Bayram, and N. Memon, "Fast camera fingerprint matching in very large databases," in *IEEE International Conference on Image Processing*, 2017.

[14] A. Tuama, F. Comby, and M. Chaumont, "Camera model identification with the use of deep convolutional neural networks," in *IEEE International Workshop on Information Forensics and Security*, 2016.

[15] L. Bondi, L. Baroffio, D. Guera, P. Bestagini, E. Delp, and S. Tubaro, "First steps toward camera model identification with convolutional neural networks," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 259–263, 2017.

[16] R. Caldelli, I. Amerini, and C. Li, "PRNU-based Image Classification of Origin Social Network with CNN," in *European Signal Processing Conference*, 2018, pp. 1357–1361.

[17] N. Bonettini, L. Bondi, D. Güera, S. Mandelli, P. Bestagini, S. Tubaro, and E. J. Delp, "Fooling PRNU-based detectors through convolutional neural networks," in *European Signal Processing Conference (EUSIPCO)*, 2018, pp. 957–961.

[18] M. Kirchner and C. Johnson, "SPN-CNN: Boosting Sensor-Based Source Camera Attribution with Deep Learning," in *IEEE International Workshop on Information Forensics and Security*, December 2019.

[19] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[20] M. Huh, A. Liu, A. Owens, and A. Efros, "Fighting fake news: Image splice detection via learned self-consistency," in *European Conference on Computer Vision*, 2018.

[21] O. Mayer and M. Stamm, "Forensic Similarity for Digital Images," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1331–1346, 2020.

[22] D. Cozzolino and L. Verdoliva, "Noiseprint: A cnn-based camera model fingerprint," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 144–159, 2020.

[23] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1449–1457.

[24] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[25] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[28] T. Gloe and R. Böhme, "The 'Dresden Image Database' for benchmarking digital image forensics," *Journal of Digital Forensic Practice*, vol. 3, pp. 150–159, 2010.

[29] D. Shullani, M. Fontani, M. Iuliani, O. Al Shaya, and A. Piva, "VISION: a video and image dataset for source identification," *EURASIP Journal on Information Security*, vol. 2017, no. 1, p. 15, 2017.

[30] W.-H. Chuang, H. Su, and M. Wu, "Exploring compression effects for improved source camera identification using strongly compressed video," in *IEEE International Conference on Image Processing*, 2011, pp. 1953–1956.

[31] E. Altinisik, K. Tasdemir, and H. Sencar, "Mitigation of H.264 and H.265 video compression for reliable PRNU estimation," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1557–1571, 2019.

[32] M. Iuliani, M. Fontani, D. Shullani, and A. Piva, "Hybrid reference-based video source identification," *Sensors*, vol. 19, no. 3, p. 649, 2019.

[33] S. Mandelli, P. Bestagini, L. Verdoliva, and S. Tubaro, "Facing device attribution problem for stabilized video sequences," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 14–27, 2020.