PRNU-Net: a Deep Learning Approach for Source Camera Model Identification based on Videos Taken with Smartphone

Younes Akbari*, Noor Almaadeed*, Somaya Al-maadeed*,
Fouad Khelifi†, and Ahmed Bouridane‡,
*Department of Computer Science and Engineering , Qatar University, Doha, Qatar
†Department of Computer and Information Sciences, Northumbria University, UK,
‡Center for Data Analytics and Cybernetics, University of Sharjah, Sharjah, UAE.

Abstract—**Recent advances in digital imaging have meant that every smartphone has a video camera that can record high-quality video for free and without restrictions. In addition, rapidly developing Internet technology has contributed significantly to the widespread distribution of digital video via web-based multimedia systems and mobile applications such as YouTube, Facebook, Twitter, WhatsApp, etc. However, as the recording and distribution of digital video has become affordable nowadays, security issues have become threatening and have spread worldwide. One of the security issues is the identification of source cameras on videos. Generally, two common categories of methods are used in this area, namely Photo Response Non-Uniformity (PRNU) and Machine Learning approaches. To exploit the power of both approaches, this work adds a new PRNU-based layer to a convolutional neural network (CNN) called PRNU-Net. To explore the new layer, the main structure of the CNN is based on the MISLnet, which has been used in several studies to identify the source camera. The experimental results show that the PRNU-Net is more successful than the MISLnet and that the PRNU extracted by the layer from low features, namely edges or textures, is more useful than high and mid-level features, namely parts and objects, in classifying source camera models. On average, the network improves the results in a new database by about 4%.**

I. Introduction

In the last two decades, the cell phone technology has evolved significantly due to its economic advantages, functionality and accessibility [1]. The ability to create digital audiovisual content without constraints such as time, objects, or locations are clear advantages of the technology [2]. For forensic investigations and crime prosecutions, smartphone devices provide some important information in crucial ways [1], [3]. In areas such as medicine, law, and surveillance systems, where images and videos are examined for authenticity, these types of investigations have potential significance. Lossy video compression complicates the forensic analysis of videos much more than the analysis of images, since the current traces can be erased or significantly damaged by high compression rates, making it impossible or difficult to recover the entire processing data. While numerous forensic methods have been developed based on digital images [4], [5], [6], [7], [8], [9], the forensic analysis of videos has been less explored. It should be noted that methods based on images cannot also be applied directly to videos [10], [11], [12]. This is due to some challenges such as compression, stabilization, scaling, and cropping, as well as the differences between frame types that can occur when producing a video. By analyzing the video produced by digital cameras, video identification algorithms can identify and distinguish camera types. During the last few years, forensic specialists have been particularly interested in this topic. In general, images and videos can be identified in two ways: by extracting a unique fingerprint from the images or videos, or by examining the metadata associated with the images or videos (the DNA of the video). Lopez et al. [13] demonstrated that the internal elements and metadata of video can be used for source video identification. Since metadata can be removed from an image or video, identifying video or images based on fingerprint is a reliable method. Moreover, two concepts are considered for identifying camera: individual source camera identification (ISCI) and source camera model identification (SCMI). ISCI distinguishes cameras from both the same and different camera models, while SCMI is a subset of ISCI that distinguishes a particular camera model from others, but cannot distinguish a particular camera model from the same camera models. In this paper, we focus on the SCMI scenario.

Two common methods used in the field, namely Photo Response Non Uniformity (PRNU) [14], [15], [16], [17], [18] and Machine Learning approaches. PRNU, which is understood to be the unique fingerprint of the camera, is often referred to as residual noise or sensor pattern noise (SPN). PRNU is generated when the CCD (Charge Coupled Device) or CMOS (Complementary Metal Oxide Semiconductor) sensors process the input signal (light) and convert it into a digital signal. The output of the methods can be considered low-level features. In deep learning methods, which are a popular category of machine learning, this training step should be performed to extract the fingerprint of the video captured by the camera. The main challenges for these methods are the separation of content from noise. The challenge can be solved by introducing methods and algorithms to address the problem by, for example, adding new layers and loss functions. The architecture introduced by the Multimedia and Information Security Lab (MISL) [19] is one of the architectures. The MISLnet network is based on a so-called constrained convolutional layer. A Constrained Convolutional layer is added at the beginning of a CNN that is to perform forensic tasks as shown in Figure 1 (a). As a result of the layer, low-level features are extracted to suppress the image content. To design the layer, the convolutional layer filters are enforced by the following constraints:

$$\begin{cases} \omega_{k_j}^{(1)}(0,0) = -1 \\ \sum_{m,n\neq 0} \omega_{k_j}^{(1)}(m,n) = 1 \end{cases} \quad (1)$$

where $j = \{1,2,3\}$. Moreover, $\omega_{k_j}^{(1)}$ denotes the jth kernel of the kth filter in the first layer of the network. Despite promising results of the method [20], [21], because the degree of sensitivity in the field, an improvement in the field is always essential.

To add the benefits of PRNU approaches to CNNs, this paper presents a new PRNU-based layer that can improve the results of Deep Learning architectures in this application. The PRNU-based layer, which can be inserted into CNNs, adds an advantageous attribute to CNNs thus taking into account the fingerprint information extracted from frames in the network. The layer can pass the extracted fingerprint (low-level features) from each layer to the next layers. This means that the features can be extracted by layers with high, mid or low features. An overview of the structures is given in Figure 1 (b). In the structure, the new layer can be placed at any point in the network and retrieved several times like a convolutional layer. The goal of evaluating the new layer at different locations in the network is to make the effects of the PRNU extracted from the high, mid, and low-level features during learning clearer. Forward propagation and backpropagation are based on the PRNU method and the derivatives of the loss with respect to the input data of the layer, respectively. Two scenarios were performed in relation to the location and number of repetitions of the layer. To evaluate the approaches, the frames need be extracted. Generally, the frames consists of intra-coded picture (I-frame), predictive coded picture (P-frame), and bi-predictive coded picture (B-frames) showing promising results with I-frames [12], [22]. In our work, I-frames are extracted from Qatar University Forensic Video Database (QUFVD) which was created as part of this investigation. The database includes 6000 videos from 20 modern smartphone representing five brands, each brand has two models, and each model has two identical smartphone devices. The experiments show that the new layer can improve the results of the CNNs without it (MISL [19]). It is worth noting that, like all deep learning methods used to identify source cameras, this study deals with videos at the frame level instead of considering the video in a feature space representation.

The paper is organized as follows. Section II gives a review of available deep learning methods for source camera verification from videos is presented. Our new approach is then presented in Section III. Section IV describes our database used to identify source cameras from videos. Section V discusses the evaluation of the proposed while the last section concludes this work.

II. Literature Review

Source camera identification from videos can be classified into two categories: PRNU and Deep Learning methods. Since we focus on Deep Learning in this paper, we address these methods in this section.

In [23], a CNN based sensor pattern noise (SPN) method was presented, called SPN-CNN. The authors implemented the architecture based on the idea that CNN has the ability to extract signals characterised by noise from a set of images [24]. Therefore, the network was to obtain a noise pattern. The method was tested on the VISION database [25] and experimental results have shown that the results outperform those of the wavelet denoiser. Also, the authors showed that, when I-frames were considered to feed into CNN, the results were further improved.

References [20] and [21] proposed a deep learning method (MISLnet architecture) for source camera identification using video frames to train the network. They extended a version of a constrained convolutional layer introduced in [19] as mentioned in Section I. Moreover, a majority vote was considered to make the decision in video level using frames fed into the network. The constrained convolutional layer was added as the first layer that used three kernel with size 5. This layer is constructed in such a way that there are relationships between adjacent pixels that are independent of the content of the scene. The methods was tested on VISION database [25]. The experiments showed that the layer can improve results compared with deep learning architectures without the layer. The key difference between the two methods relates to the size of images and type of color modes. [20] and [21] used RGB and gray scale modes, respectively. Patches used in former is 480 while latter fed patched with 256.

Mayer et al. [26] used CNN proposed in [19] like the two previous studies to extract features and a similarity network to verify the source camera. The similarity network maps two input deep feature vectors to a 2D similarity vector. To achive this, the authors follow a design of the similarity network developed in [27]. To obtain a decision in video level, a fusion approach based on mean of the inactivated output layer from the similarity network was presented. This method was tested on SOCRatES dataset [28]. The experiments showed that the method improve traditional methods such as [29].

The structure of the CNN for the three studies is shown in Figure 1 (a). As shown in the figure, a constrained convolutional layer is added to a simple CNN.

III. PRNU-Net

Figure 1 (b) shows the structure of the network used in the study. As can be seen in the figure, only the constrained layer proposed by [19] (MISLnet architecture) is removed from our structure and the rest of our structure is the same as MISLnet. A new PRNU-based layer has been designed to replace the constrained convolutional layer and can be placed elsewhere in the network. The layer can extract PRNU from raw images (input layer) and feature maps of each convolutional layer.

To design our new layer, forward propagation and backward propagation are considered, which are explained in the following two subsections. Since PRNU is extracted from grayscale

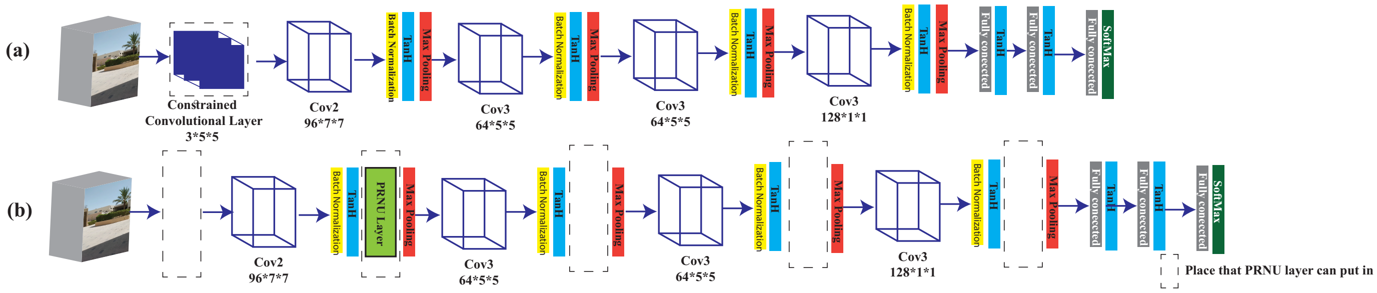


Fig. 1. Overview of (a) The CNN (MISLnet) presented in [19] with a constrained layer in the first layer of the network and (b) PRNU network structure with a layer based on PRNU (the dotted rectangle shows that different locations of the layer can be considered).

images, it should be noticed that the frames used as input are in grayscale mode.

A. Forward propagation

Let $B = \left\{X_{(l)}^{(j)}, Y^j\right\}_{j=1}^{N}$ be the training set with N samples. l shows the position of the layer as shown in Figure 1 (b). For each input of the layer, we consider $X_{(l)}^{(j)} = \{x_1, x_2, ..., x_d\}$, where d is the dimension of the input of the layer. For the network shown in Figure 1 (b), for $l = \{1, 2, 3, 4, 5\}$, d can be $d = \{1, 96, 64, 64, 128\}$, respectively. d guarantees that the PRNU can be extracted from raw images (input layer) and feature maps of the convolutional layers which are the input of the layer. For example, for a member of $X_{(2)}^{(j)}$, that is, x_1 with $d = 96$ denotes the first dimension of the input with 96 dimensions at the second position (after the first convolutional layer as shown in Figure 1 (b)) eligible for the layer. To obtain PRNU for the input, we have as [29]:

$$x_i = O + OK + \Theta \tag{2}$$

Where O refers to the original input multimedia file, K represents the PRNU factor and Θ is a random noise factor. To estimate K, noise residual W of the input should be obtained using denoising filter F:

$$W_i = x_i - F(x_i) \tag{3}$$

Estimation of K is obtained by the following maximum likelihood estimator:

$$\hat{K}_i = \frac{W_i x_i}{(x_i)^2} \tag{4}$$

Where $\hat{K}_i$ is output of the layer for input x_i.

The process of feature extraction by the PRNU layer is illustrated in Figure 2 at three different layer positions for one sample. The original image, the output of the first convolutional layer and the last convolutional layer are selected to obtain PRNU. A patch with a size of 350 × 350 is considered as input for the original image. To create feature maps in the convolutional layers, one of the convolutional kernels is used.

As mentioned above, a denoising filter F is used to extract the pattern noise. Wavelet-based filters can be considered to have better performance than approaches such as Wiener and median filters. Areas around the edges are usually misinterpreted by the latter two. For details on how this denoising filter works, see [4].

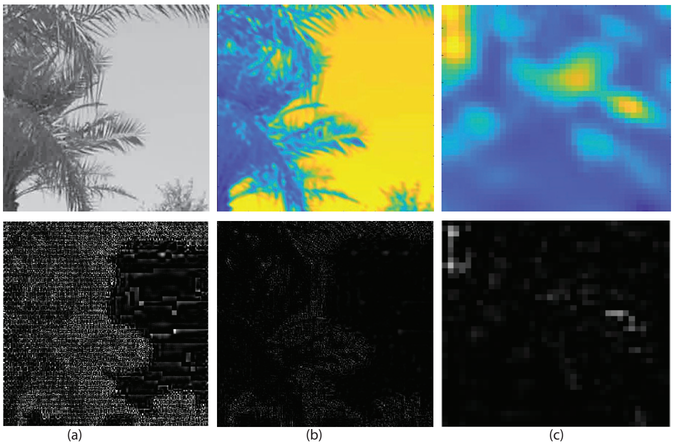


Fig. 2. Sample of PRNU layer results based on three positions. The first row shows (a) the original image, (b) the output of the first convolutional layer, and (c) the output of the last convolutional layer. The second row shows the PRNU extraction corresponding to each output (the displayed outputs of two convolutional layers are based on scaled colors and also the sizes have been adjusted to the original size).

B. BackPropagation

Backpropagation of the PRNU layer requires input and output of the forward function of the layer. To include a user-defined layer in a network, the forward function of the layer must accept the output of the previous layer and forward propagate arrays of the size expected by the next layer. Similarly, if the backward function is specified, it must accept inputs of the same size as the corresponding output of the forward function and backward propagate derivatives of the same size. The derivative of the loss with respect to the input data (X) is:

$$\frac{\partial L}{\partial X} = \frac{\partial L}{\partial f(X)} \frac{\partial f(X)}{\partial X} \tag{5}$$

where $\frac{\partial L}{\partial f(X)}$ is the gradient propagated from the next layer. Since in backpropagation scheme, we can use both input and output of forward propagation to derive the derivative of the activation. Let us consider $\hat{K}$ as:

$$\hat{K} = E \odot X \tag{6}$$

where E shows a matrix that perform the operation to obtain PRNU like (4), and $\odot$ is the element-wise product of the two matrix. Then, if we have $f(X) = \hat{K}$, the derivation is:

$$\frac{\partial f(X)}{\partial X} = E \tag{7}$$

Since we have both $\hat{K}$ and X from the backpropagation operation, we can easily obtain E. Algorithms 1 enumerates the learning schemes for forward propagation and backpropagation.

Algorithm 1 Training PRNU-Net with forward propagation and backpropagation

```
(Forward propagation)
Input: Training data $B = \left\{X_{(l)}^{(j)}, Y^j\right\}_{j=1}^{N}$
Output: PRNU estimated $\hat{K} = \left\{\hat{K}_i^{(j)}\right\}_{j=1}^{N}$
for j=1 to N
    for i=1 to d
        Compute Noise residual estimated $W_i^{(j)}$ using (3)
        Compute PRNU $\hat{K}_i^{(j)}$ using (4)
    end for
end for
Return: $\hat{K}$
(Backpropagation)
Input: $\hat{K}$, $X$, and $\frac{\partial L}{\partial f(X)}$
Output: $\frac{\partial L}{\partial X}$
Compute matrix of the PRNU operation: $E = \hat{K} \oslash X$
($\oslash$ is the element-wise division)
Compute $\frac{\partial f(X)}{\partial X}$ using (7)
Return: $\frac{\partial L}{\partial X} = \frac{\partial L}{\partial f(X)} E$
```

IV. Databases

Most of the databases used for source video identification provide recordings captured with video cameras, and only two databases offer recordings with smartphones, namely Daxing [1] and QUFVD [30]. Therefore, exploring the methods based on smartphone databases, which are developed with a rapid growth, can show whether the existing methods are useful on the new databases. Comparing the results of Daxing and QUFVD with older databases such as VISION will also show that the new smartphone-based databases are more challenging and thus improvement is inevitably needed. Although the Daxing database can be considered an important database in this field, as explored in [30], QUFVD is better suited to be used in Deep Learning methods. In fact, in Daxing database, source camera identification techniques based on machine learning may face a problem of unbalanced data since the number of training videos is small and differs across the devices. To make a fair comparison, the QUFVD database has been used. This includes five popular smartphone brands with two models per brand with two devices for each model, 6000 original videos, and 76,531 I-frames. Table I summarizes QUFVD with its features and Figure 3 shows samples of the database. The database is publicly available [1].

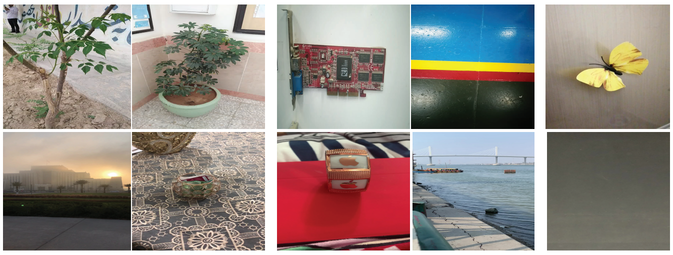


Fig. 3. Sample frames from captured videos of the database

V. Evaluation

PRNU-Net is evaluated using the SCMI scenario (a 10-class problem) with different settings. We divide the experiments into different scenarios showing the influence of some conditions on the results, which related to the position and repetition of the layer during training. Our proposed network is compared with MISLnet architecture [19] that has been used in several recent studies [20], [26], [21]. To show the impact of separating content from noise, MISLnet is considered with and without the constrained layer. Our implementation of the architecture is based more on [20], which was considered for identifying the source camera. The Stochastic gradient descent (SGD) is considered to train the model. The batch size is set to 100 and the parameters for momentum and decay of the stochastic gradient descent are set to 0.95 and 0.0005, respectively. The CNN is trained for 10 epochs in each experiment. The ratio of the database for the experiments uses 80% of these videos for training while the remaining 20% are considered as testing. Also, 20% of the training data is considered as validation data. As mentioned earlier, I-frames of the videos are extracted to evaluate the performances using the database. For each video and in each experimental setup, we selected all I-frames related to the videos in the training, testing, and validation tasks. A total of 76,531 I-frames were extracted. To identify a video based on its I-frames, all I-frames in the test set are considered. The scores obtained by the CNN based on the highest probability show which I-frames belong to which classes. At the video level, a majority vote then decides all the frames that belong to a video. It should be noted that all patches (about 90,000) for each class are used for training and the best results for both methods are based on patches with a size of 350×350. A 64-bit operating system (Ubuntu 18) with a CPU E5-2650 v4 @ 2.20 GHz, 128.0 GB RAM, and four NVIDIA GTX TITAN X. were used in order to run our experiments. Tables II lists the results of the frame and video levels in terms of accuracy (%) for the SCMI scenario for each smartphone model based on PRNU-Net and MISLnet with and without constrained layer. With

[1] https://www.dropbox.com/sh /nb543na9qq0wlaz/AAAc5N8ecjawk2KlVF8kfkrya?dl=0

TABLE I
THE DEVICES OF OUR DATABASE WITH THEIR CHARACTERISTICS.

Brand	Model	Resolution	Number of videos	Number of I-frames	Length in Secs	Operating system
Samsung	Galaxy A50 (device #1)	1080 × 1920	300	3654	11-15	Android 10.0
Samsung	Galaxy A50 (device #2)	1080 × 1920	300	3782	11-15	Android 10.0
Samsung	Note9 (device #1)	1080 × 1920	300	3956	11-15	Android 10.0
Samsung	Note9 (device #2)	1080 × 1920	300	3962	12-15	Android 10.0
Huawei	Y7 (device #1)	720 × 1280	300	3630	11-15	Android 9.0
Huawei	Y7 (device #2)	720 × 1280	300	3642	11-15	Android 9.0
Huawei	Y9 (device #1)	720 × 1280	300	4146	11-14	Android 9.0
Huawei	Y9 (device #2)	720 × 1280	300	4011	11-15	Android 9.0
iPhone	8 Plus (device #1)	1080 × 1920	300	3991	11-15	iOS 13
iPhone	8 Plus (device #2)	1080 × 1920	300	4080	11-14	iOS 13
iPhone	XS Max (device #1)	1080 × 1920	300	3893	11-15	iOS 13
iPhone	XS Max (device #2)	1080 × 1920	300	4074	11-15	iOS 13
Nokia	5.4 (device #1)	1080 × 1920	300	3350	11-13	Android 10.0
Nokia	5.4 (device #2)	1080 × 1920	300	3531	11-14	Android 10.0
Nokia	7.1 (device #1)	1080 × 1920	300	3904	11-13	Android 10.0
Nokia	7.1 (device #2)	1080 × 1920	300	3819	11-14	Android 10.0
Xiaomi	Redmi Note8 (device #1)	1080 × 1920	300	3776	11-14	Android 11.0
Xiaomi	Redmi Note8 (device #2)	1080 × 1920	300	3598	11	Android 11.0
Xiaomi	Redmi Note9 Pro (device #1)	1080 × 1920	300	3888	11-15	Android 11.0
Xiaomi	Redmi Note9 Pro (device #2)	1080 × 1920	300	3838	11-13	Android 11.0

TABLE II
THE RESULTS OF THE FRAME AND VIDEO LEVELS IN TERMS OF ACCURACY (%) FOR THE SCMI SCENARIO FOR EACH SMARTPHONE MODEL.

Model	I-frame			Video		
	[19] (without constrained layer)	[19] (with constrained layer)	Ours	[19] (without constrained layer)	[19] (with constrained layer)	Ours
Galaxy A50	69.1	72.8	75.0	71.0	73.3	77.2
Note9	74.2	78.7	78.8	88.4	95.8	95.8
Y7	65.7	68.0	71.5	80.0	84.2	86.0
Y9	70.8	76.9	77.3	82.4	86.7	91.6
8 Plus	65.5	67.8	73.9	83.4	84.2	85.5
XS Max	71.5	76.8	79.2	64.8	68.3	74.9
5.4	79.8	81.8	83.1	88.7	90.8	92.7
7.1	71.8	75.5	80.6	86.5	90.0	92.2
Redmi Note8	70.5	75.8	81.9	78.2	80.8	84.2
Redmi Note9 Pro	66.1	66.4	75.0	64.4	65.8	77.3
Overall accuracy	70.5	74.0	77.6	78.8	82.0	85.7

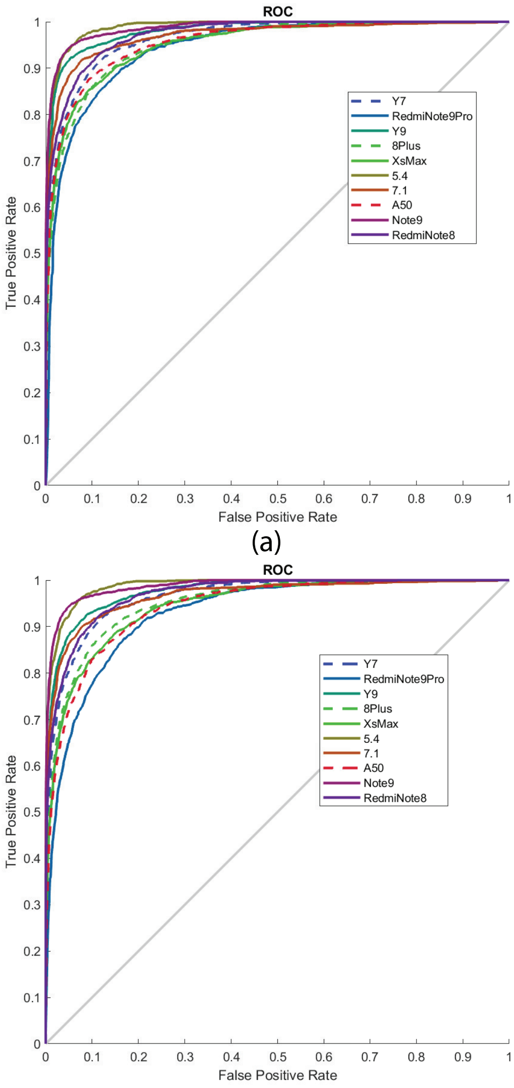
TABLE III
IMPACT OF PLACE AND REPETITION OF THE PRNU LAYER IN THE NETWORK (l)

Place	$l = 1$	$l = 2$	$l = 3$	$l = 4$	$l = 5$	$l = \{1, 2\}$	$l = \{1, 2, 3\}$	$l = \{1, 2, 3, 4\}$	$l = \{1, 2, 3, 4, 5\}$
	77.4	77.6	74.1	73.8	73.0	77.4	73.9	72.5	72.3

this premise, Figure 4 provides a more comprehensive picture of camera identification performance to check the quality of PRNU-Net compared to MISLnet by presenting the Receiver Operating Characteristic (ROC) curves for a selected group of ten classes (smartphone model) from our database. Two values are calculated for each threshold: True Positive Ratio (TPR) and False Positive Ratio (FPR). The TPR of a given class, e.g. Huawei Y7, is the number of outputs whose actual and predicted class is Huawei Y7 divided by the number of outputs whose predicted class is Huawei Y7. The FPR is calculated by dividing the number of outputs whose actual class is not Huawei Y7, but whose predicted class was Huawei Y7 by the number of outputs whose predicted class is not Huawei Y7. Impact of place and repetition of the layer in the network (l) is explored as shown in Table III. This shows that the position is more suitable for layers with high, mid, or low-level features.

A. Result discussion

Recently, Deep Learning methods have been introduced to solve source camera identification. The methods can help to improve the results obtained with traditional methods such as the PRNU methods. Overall, the results obtained at the frame and video levels suggest that PRNU-Net is more successful than MISLnet for the SCMI problem in all device models. For both methods, when the results are reported at the video level, improvement can be cleaely observed. In addition, the results of PRNU-Net and MISLnet with the constrained layer when compared against MISLnet without the constrained layer clearly show that the separation of content and noise is useful for source camera identification. The results are discussed in more detail below. As can be seen in Table II at the frame level, a few devices are hard to identify, such as the Y7, 8 Plus, and Redmi Note9 Pro, and this requires further analysis to find out the reason for the differences, which could be the resolution of the videos or the imaging technology used by the devices, etc. However, from other results, resolution does seem to be the reason, since Y7 and Y9 have the lowest resolution, but their identification results are not worse. The biggest improvement is for the Redmi Note9 Pro compared to MISLnet about 9%. At the video level, an overall improvement is achieved for all devices. The best result with 95% is also obtained for the Note 9 by both the methods. The best improvement on video level is also for Redmi Note9 pro compared to MISLnet about 12%. Figures 4 shows the TPR compared to the FPR for the two methods at different frame-level thresholds. As can be seen from the figure, all models achieve a larger Area Under Curve (AUC) value than MISLnet. A different analysis for the devices in terms of TPR and FPR as shown in the figure, shows that the best performance is obtained by Nokia 5.4 with Area Under Curve (AUC=0.991) for PRNU-Net (Figure 4 (a)) compared to AUC=0.989 for MISLnet (Figure 4 (b)). Table III shows the best result when the layer position is $l = 2$ and the layer repetition is 2, namely, $l = \{1, 2\}$ in frame level. When the repetition is set for all layers, a drop in performance is indicated showing that the fingerprint extracted can be effected by convolutional layers. This also shows that the layer gives better results when placed after layers with low-level features. This may be because PRNU extracts low level features and the features may be more accurate if the input is low level.



(a)

(b)

Fig. 4. True and false positive rates (ROC) obtained in SCMI scenario (a) 10 classes with PRNU-Net (b) 10 classes with MISLnet

VI. Conclusion

This paper has presented a new layer based PRNU extracted from videos taken with a smartphone to identify the camera source. In general, PRNU methods extract low-level features from frames, and we have studied the feature extraction by the methods using a deep-learning approach. For the new layer, forward propagation and backpropagation are defined based on the extracted PRNU and the derivative of the loss with respect to the input data, respectively. The method is evaluated with a database containing five popular smartphone brands with two models per brand and two devices for each model, 6000 original videos, and 76,531 I-frames. The results show that the approach achieves promising results compared to MISLnet, one of the most popular deep learning methods in the field. The best results are obtained when the layer located after low level inputs. However, it is obvious that it is essential to improve the results in future works.

To improve the results, especially when the layer is repeated, defining new learnable parameters can help to reduce the impact of the convolutional layers. The parallel use of other PRNU methods and filters can be considered as a bank of operators that can be used instead of convolutional layers. Also, it is possible to add the layer to other Deep Learning architectures. It would be a worthwhile endeavor for the future to change architecture so that videos are seen as a sequence of frames rather than focusing on single frames. Finally, the PRNU network should be tested using other scenarios such as ISCI to obtain a more accurate analysis.

Acknowledgment

This publication was made possible by NPRP grant # NPRP12S-0312-190332 from Qatar National Research Fund (a member of Qatar Foundation). The statement made herein are solely the responsibility of the authors.

References

[1] H. Tian, Y. Xiao, G. Cao, Y. Zhang, Z. Xu, and Y. Zhao, "Daxing smartphone identification dataset," *IEEE Access*, vol. 7, pp. 101 046–101 053, 2019.

[2] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, "An overview on video forensics," *APSIPA Transactions on Signal and Information Processing*, vol. 1, 2012.

[3] Y. Akbari, S. Al-maadeed, O. Elharrouss, F. Khelifi, A. Lawgaly, and A. Bouridane, "Digital forensic analysis for source video identification: A survey," *Forensic Science International: Digital Investigation*, vol. 41, p. 301390, 2022.

[4] 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.

[5] 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.

[6] A. Lawgaly and F. Khelifi, "Sensor pattern noise estimation based on improved locally adaptive dct filtering and weighted averaging for source camera identification and verification," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 392–404, 2016.

[7] A. Lawgaly, F. Khelifi, and A. Bouridane, "Weighted averaging-based sensor pattern noise estimation for source camera identification," in *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2014, pp. 5357–5361.

[8] X. Kang, Y. Li, Z. Qu, and J. Huang, "Enhancing source camera identification performance with a camera reference phase sensor pattern noise," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 393–402, 2011.

[9] F. Ahmed, F. Khelifi, A. Lawgaly, and A. Bouridane, "Comparative analysis of a deep convolutional neural network for source camera identification," in *2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3)*. IEEE, 2019, pp. 1–6.

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

[11] 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, 2019.

[12] E. Altinisik and H. T. Sencar, "Source camera verification for strongly stabilized videos," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 643–657, 2020.

[13] R. R. López, E. A. Luengo, A. L. S. Orozco, and L. J. G. Villalba, "Digital video source identification based on container's structure analysis," *IEEE Access*, vol. 8, pp. 36 363–36 375, 2020.

[14] S. McCloskey, "Confidence weighting for sensor fingerprinting," in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2008, pp. 1–6.

[15] W.-H. Chuang, H. Su, and M. Wu, "Exploring compression effects for improved source camera identification using strongly compressed video," in *2011 18th IEEE International Conference on Image Processing*. IEEE, 2011, pp. 1953–1956.

[16] M. Goljan, M. Chen, P. Comesaña, and J. Fridrich, "Effect of compression on sensor-fingerprint based camera identification," *Electronic Imaging*, vol. 2016, no. 8, pp. 1–10, 2016.

[17] A. Mahalanobis, B. V. Kumar, and D. Casasent, "Minimum average correlation energy filters," *Applied Optics*, vol. 26, no. 17, pp. 3633–3640, 1987.

[18] L. J. G. Villalba, A. L. S. Orozco, R. R. López, and J. H. Castro, "Identification of smartphone brand and model via forensic video analysis," *Expert Systems with Applications*, vol. 55, pp. 59–69, 2016.

[19] B. Bayar and M. C. Stamm, "Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2691–2706, 2018.

[20] D. Timmerman, S. Bennabhaktula, E. Alegre, and G. Azzopardi, "Video camera identification from sensor pattern noise with a constrained convnet," *arXiv preprint arXiv:2012.06277*, 2020.

[21] B. Hosler, O. Mayer, B. Bayar, X. Zhao, C. Chen, J. A. Shackleford, and M. C. Stamm, "A video camera model identification system using deep learning and fusion," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 8271–8275.

[22] B. C. Hosler, X. Zhao, O. Mayer, C. Chen, J. A. Shackleford, and M. C. Stamm, "The video authentication and camera identification database: A new database for video forensics," *IEEE Access*, vol. 7, pp. 76 937–76 948, 2019.

[23] M. Kirchner and C. Johnson, "Spn-cnn: Boosting sensor-based source camera attribution with deep learning," in *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2019, pp. 1–6.

[24] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[25] 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, pp. 1–16, 2017.

[26] O. Mayer, B. Hosler, and M. C. Stamm, "Open set video camera model verification," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 2962–2966.

[27] O. Mayer and M. C. Stamm, "Forensic similarity for digital images," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1331–1346, 2019.

[28] C. Galdi, F. Hartung, and J.-L. Dugelay, "Socrates: A database of realistic data for source camera recognition on smartphones." in *ICPRAM*, 2019, pp. 648–655.

[29] M. Goljan, J. Fridrich, and T. Filler, "Large scale test of sensor fingerprint camera identification," in *Media forensics and security*, vol. 7254. International Society for Optics and Photonics, 2009, p. 72540I.

[30] Y. Akbari, S. Al-Maadeed, N. Al-Maadeed, A. Al-Ali, F. Khelifi, A. Lawgaly *et al.*, "A new forensic video database for source smartphone identification: Description and analysis," *IEEE Access*, vol. 10, pp. 20 080–20 091, 2022.