On-device Filtering of Social Media Images for Efficient Storage

Dhruval Jain
On-device AI
Samsung R&D Institute
Bengaluru, India
dhruval.jain@samsung.com

Debi Prasanna Mohanty
On-device AI
Samsung R&D Institute
Bengaluru, India
debi.m@samsung.com

Sanjeev Roy
On-device AI
Samsung R&D Institute
Bengaluru, India
sanjeev.roy@samsung.com

Naresh Purre
On-device AI
Samsung R&D Institute
Bengaluru, India
naresh.purre@samsung.com

Sukumar Moharana
On-device AI
Samsung R&D Institute
Bengaluru, India
msukumar@samsung.com

***Abstract*—Artificially crafted images such as memes, seasonal greetings, etc are flooding the social media platforms today. These eventually start occupying a lot of internal memory of smartphones and it gets cumbersome for the user to go through hundreds of images and delete these synthetic images. To address this, we propose a novel method based on Convolutional Neural Networks (CNNs) for the on-device filtering of social media images by classifying these synthetic images and allowing the user to delete them in one go. The custom model uses depthwise separable convolution layers to achieve low inference time on smartphones. We have done an extensive evaluation of our model on various camera image datasets to cover most aspects of images captured by a camera. Various sorts of synthetic social media images have also been tested. The proposed solution achieves an accuracy of 98.25% on the Places-365 dataset and 95.81% on the Synthetic image dataset that we have prepared containing 30K instances.**

***Index Terms*—CNNs, Depthwise separable convolutions**

I. Introduction

In recent years, there has been a huge flux of data on the internet. Social media platforms have contributed significantly in increasing the volume of images circulated. Images are being used to communicate opinions on trending news through memes. According to a survey, 3.2 billion images are shared each day over the internet. A large proportion of these images are wholly synthetic and become irrelevant to the user in a short period. Images captured from the camera have very few sharp edge transitions and are characterized by sensor pattern noise. Jan Lukas et al. [1] proposed the identification of digital cameras based on the sensors pattern noise by using a correlation detector to investigate the presence of the reference noise pattern in the given image. Corripio et al. [2] computed wavelet features for smartphone camera identification. Artificially crafted memes or seasonal greetings have sharp edge transitions. If the image is entirely artificially generated, it doesn't have noise in its raw form. Some noise is added by the lossy compression techniques used by social media platforms. These synthetic images are generally created by adding artificial text on the camera image or stacking multiple camera images together. In this process, certain regions of the image get sharper edge transitions and uniform pixel intensities.

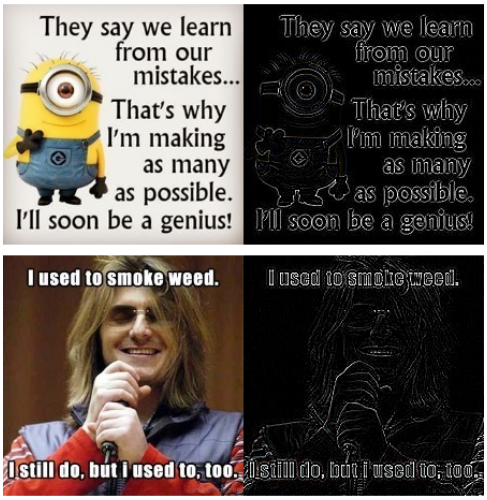
Convolutional Neural networks can learn complex latent features based on image content by which they can distinguish between images portraying different objects or scenes. This has led to their widespread success in the object recognition task. We show that CNNs can perform equally well in distinguishing between two images that portray the same object or scene but one is either synthetically generated or has added artificial characteristics in the form of text, clip-art or image croppings. Depthwise separable convolutions significantly reduce the number of parameters to build light weight deep neural network architectures. MobileNet [3] uses 3×3 depthwise separable convolutions which need 8 to 9 times lesser computations than standard convolutions. MobileNet-224 [3] used 4.2 million parameters and suffered a drop in accuracy of only 0.9% on the Imagenet classification task compared to VGG16 [4] which used 138 million parameters.

This paper is divided into six sections. We have built our custom deep learning architecture whose details are discussed in section III. Section IV provides an extensive evaluation of the proposed model along with visual representations illustrating the learning of the model.

II. Related Work

Previous works like [5] and [6] computed hand-crafted features based on color and space correlation of pixels for classification. It assumed synthetic images to have more colors than natural images as they tend to have large uniform regions of the same color. These assumptions do not hold good with artificially manipulated camera images like memes that are very popular on social media today. Due to the noise added by the lossy JPEG compression, uniform color regions may have

978-1-7281-6926-2/20/$31.00 ©2020 IEEE



(a) Synthetic Images

(b) Camera Images

Fig. 1: Edge Residuals

varying pixel intensities. In [5], eight features were computed including saturation average, SGLD histogram [7], etc and various classifiers like AdaBoost, SVMs and neural networks were tested. These feature values are averaged globally across the image, thereby failing to describe the image locally. Some camera images may have very sharp edges distributed locally in certain regions and the globally averaged feature values may fall near to those of synthetic images. [5] and [6] also highlighted that camera images have faded edges, unlike the synthetic ones. But with advanced camera sensors that we have today, a handful of these features do not suffice and may lead to misleading results.

Classification of images based on image content has been an area of active research with the advent of deep learning. But the classification of images based on statistical properties has not received much attention. V Andrearczyk et al. [8] introduced Texture CNN. In their work, they proposed that features extracted by the fully connected layers of CNN architectures like AlexNet [9], based on shape information , are of very little importance in texture understanding. Wavelet CNNs proposed by S Fujieda et al. [10] incorporate spectral information to enhance texture classification.

Bayar et al. [11] proposed a CNN architecture for detecting image manipulation that suppresses the image content to learn features based on tampering. The scope of manipulation was only restricted to gaussian blurring, resampling using bilinear interpolation, median filtering, and additive white gaussian noise. They did not consider adding artificial text or image-croppings.

Other works in this domain include [12], [13] and [14] which focus on distinguishing photorealistic computer graphics from camera images. The scope of our problem is entirely different from them as we focus on artificially crafted social media images that become junk to the user in a short period of time. To best of our knowledge, we are the first to propose a lightweight CNN based architecture to perform this task on smartphones with a low inference time.

III. Methodology

We begin by examining the edge residuals of the camera and synthetic images. High pass filters designed by Fridrich et al. [15] suppress the low-frequency components that represent the image content. These filters, when applied to synthetic images, are expected to produce strong edge residuals. Camera images which have very few sharp edge transitions compared to synthetic images tend to have weaker edges residuals. Fig1 (a) shows how artificial text is demarcated from the background, whereas in Fig1 (b), the edge residuals of camera images are weak even in case of the scenic text. These edge residuals must be analyzed locally, and therefore we use CNNs to learn latent features to enhance classification. The depthwise separable convolution is composed of two layers, a depthwise convolution layer followed by a pointwise convolution layer. In depthwise convolutions, unlike the standard convolutions, a single filter is applied per input channel. In pointwise convolutions, 1×1 kernels are used to compute the linear combination of the output of the depthwise convolution layer. The m^{th} channel of output feature map $\hat{\mathbf{G}}$ is given by

$$\hat{\mathbf{G}}_{k,\ l,\ m} = \sum_{i,j} \hat{\mathbf{K}}_{i,j,m} \cdot \mathbf{F}_{k+i-1,\ l+j-1,\ m} \qquad (1)$$

where $\hat{\mathbf{K}}$ is the depthwise convolution kernel which is applied to the m^{th} channel of the input $\mathbf{F}$.

Images are downsampled to 224×224 before they are fed into the model to achieve faster inference on-device. Even though this results in a significant loss of information, our model is able to approximate the target function well and produce good results on the standard datasets as shown in Table III. No other preprocessing is required. We have come up with three different architectures to test our hypothesis. They are described as follows.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

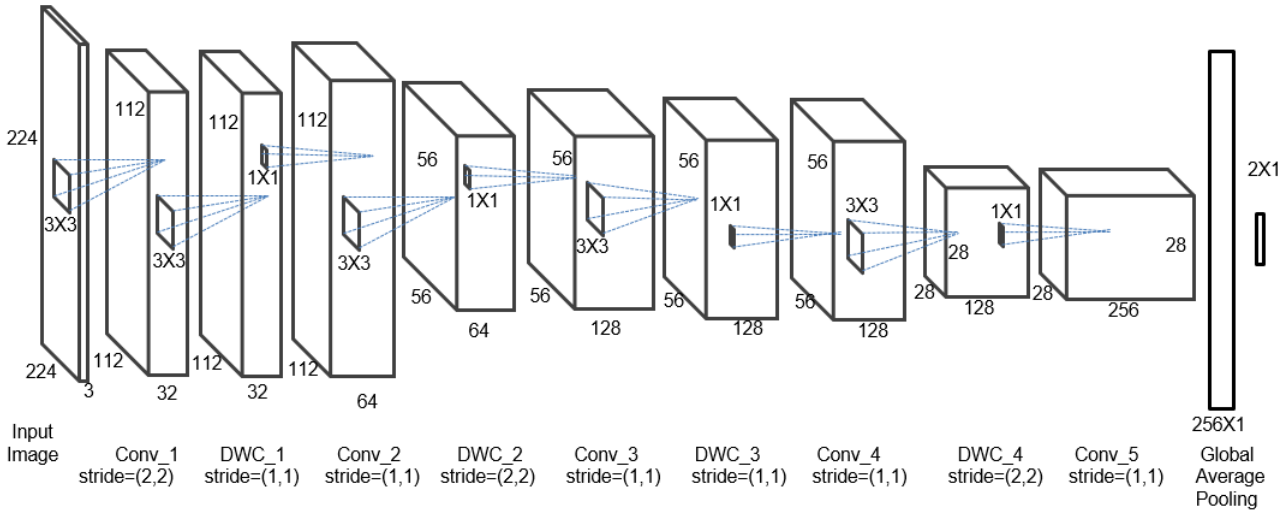


Fig. 2: Model Architecture for DWS_1

- Fig2. shows our model architecture which we call **DWS_1**. Input image has dimensions $224 \times 224 \times 3$. Each block shown in the figure is the output of the previous layer. Conv_i layers, where *i = 1, 2 . . 5* are standard convolution layers and DWC_i, where *i = 1, 2, 3, 4* are depthwise convolution layers. Conv_1 layer has 32 filters and performs 3×3 convolutions to produce an output of shape $112 \times 112 \times 32$. The rest of the convolution layers, Conv_i, where *i = 2, 3 . . 5* perform 1×1 convolutions to compute a linear combination of the previous output. A 1×1 convolution layer produces the desired number of output channels equal to the number of filters used. Batch normalization [16] applied after each layer produced better results whilst also accelerating training. We used ReLU nonlinearities as activations for all layers.
- **DWS_1** is modified such that Conv_5 layer performs 3×3 convolutions and that is why we call it **DWS_3**. Increasing the kernel size to 3×3 in the Conv_5 layer helps in incorporating the neighborhood information in order to learn more complex features. However, it also increases the inference time on-device.
- We replaced all the four depthwise separable convolution layers in **DWS_1** with standard convolution layers keeping the number of filters the same. The kernel size is kept 3×3 in all the layers. We call this model **FCONV_3**. Changing the kernel size to 5×5, we end up with **FCONV_5**.

IV. Experiments

A. Dataset

All the images used in training and testing are in the JPEG format. We prepared our dataset for synthetic images, which we call **SocialMedia** dataset. We scraped seasonal greeting images and memes from various web sources. SocialMedia consists of 40K images out of which 10K images were used for testing. Our training set for synthetic images consists of around 34,994 images, including 4,994 images from Reddit Memes Dataset [1]. For further evaluation for our solution, we used TextRecognitionDataGenerator [2] to add artificial text on camera images to make another test set for synthetic images having 30K instances. It is referenced as TRDG in Table1.

For natural image dataset, we picked random 50 images (if found) from each of the 602 classes from the Open Images Dataset V5 [3], 15,620 images from Indoor Scene Recoginition Database proposed by A. Quattoni et al. [17] which has 67 indoor categories. We also included 1,000 Motion and out of focus camera images from Blur Detection dataset [18]. To incorporate camera images with scenic text, we included 1,555 images from Total-Text [19] dataset proposed by CK Chng et al. The total size of our training set for camera images is around 43K. A natural scene may have different lighting conditions, various objects with contrast backgrounds, etc. To cover most of these aspects, we tested on various camera image datasets. These include test set from MIT Places365-Standard [20] comprising of 384K images, 41K images from MS-COCO [21] 2014 validation set and 41K images from MS-COCO 2017 test set. Photographic Image dataset proposed by Tokuda et al. [22] having 4,850 images was also evaluated.

B. Experimental Setup

- We implemented the proposed architectures in the Tensorflow framework and all of the experiments were conducted on a GeForce GTX 1080ti GPU.

[1]https://www.kaggle.com/sayangoswami/reddit-memes-dataset/metadata
[2]https://github.com/Belval/TextRecognitionDataGenerator
[3]https://storage.googleapis.com/openimages/web/download.html

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

- The loss function used is categorical cross-entropy. We have used Adam [23] optimizer with the decay of 1e-6. The initial learning rate was set to 0.001.
- We have used 5 fold cross-validation, where we randomly select one-fifth portion of the dataset for validation and train the model on the remaining. Batch size was kept 128 and we ran 200 epochs for training.

C. Evaluation

We have listed the performance of our architectures on various datasets discussed above in Table II. All images have quality factor of 0.95. We see that DWS_1 outperforms the other proposed models. The first convolution layer with kernel dimensions $3 \times 3 \times 32$ is common for all the models. It learns simple features based on edges like orientation and sharpness. Convolution layers in FCONV3 and FCONV5 models learn more complex features based on image content compared to the depthwise separable convolution layers in DWS_1 and DWS_3. These features become vital in tasks like objection recognition but may lead to misclassification in our scenario. This is evident in the results that we have obtained.

For testing the practical utility of the model, we evaluated DWS_1 with different JPEG quality factors as listed in Table III. The extent of noise introduced increases with decreasing quality factors. Edges become less sharp and images tend to lose finer details. Hence, we can expect growth in classification accuracy in case of camera images. And, this shall adversely affect accuracy in case of synthetic images. This underlying hypothesis is well supported by the results obtained in Table III. The low running time of the model is important for a real-time experience. Fig.3 shows the plot for Inference time versus the Resolution factors of the input image. Resolution factor of r has image size of $224r \times 224r$.

We have also compared our approach to traditional methods [5], [6], where classification is based on hand-crafted features. For this purpose, we prepared a dataset containing 10K images for train and 2K images for test equally distributed between both the classes. Following [5], we computed ten features per image for classification. Based on colour, different colour ratio and saturation average are used. To incorporate edge information, space correlation amongst pixels was measured in terms of features (average) like spatial gray level dependence [7], color correlogram [24] (for each color channel), gray histogram [5] and farthest neighbor [5]. Table I shows the performance of various classifiers on these hand-crafted features.

TABLE I

Classifiers	Test Accuracy
SVM	61.21%
Neural Network	61.79%
Random Forest	**65.23**%

From the above table, we see that the traditional approach does not perform well on typical social media images shared over the internet today. These failure cases are shown in Fig. 4. The presence of small artificial text on the camera image background does not add much to the value of features based on edges sharpness. Therefore, it is prone to misclassification. Moreover, with the advanced camera sensors, camera images may have high average gradient magnitudes that can result in classifiers detecting them as synthetic.

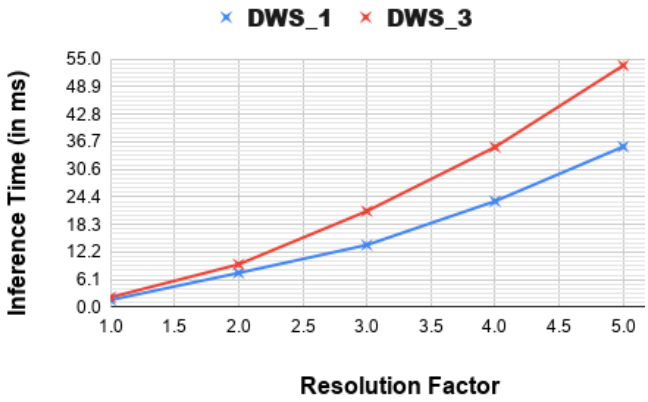


Fig. 3: Plot for Inference Time versus Input size

(a) Synthetic Images

(b) Camera Images

Fig. 4: Failure Cases of Traditional Approaches

D. Results

Fig.5 and Fig.6 display test images along with their heat maps. These heat maps are generated from the output of Conv_5 layer of DWS_1 model to visualize its learning. The intensity of added artificial characteristics ranges from dark blue (lowest) to red (highest). Heat maps of synthetic images in Fig.3 have connected patches of very high intensity collectively representing the region of artificial nature whereas the heat maps of camera images shown in Fig.5 have sparsely distributed blocks (mostly single blocks) of high intensities. We have presented four different types of social media images.

- A thin white strip with small text is added on top of the camera image as shown in Fig.5 (a). This region

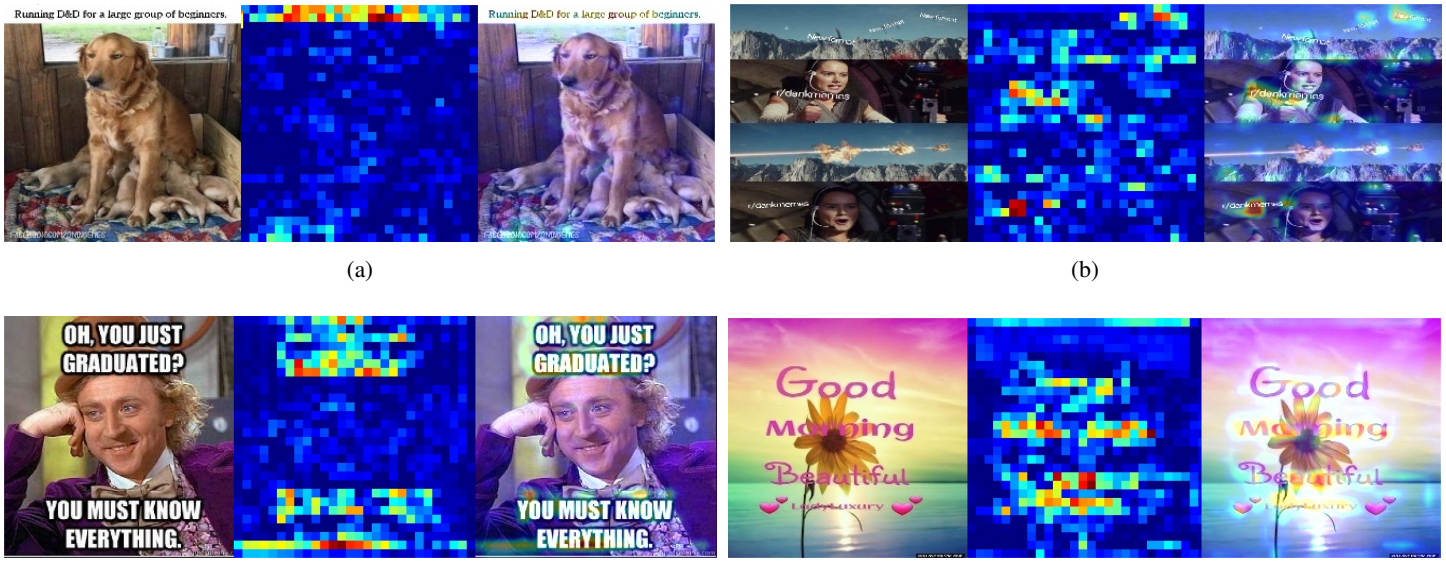
978-1-7281-6926-2/20/$31.00 ©2020 IEEE

TABLE II: Accuracy obtained on various public datasets

Models	No. of parameters	Datasets					
		Places-365	Tokuda et al.	COCO-val14	COCO-test17	SocialMedia	TRDG
DWS_1	67K	98.25%	**97.59%**	**96.39%**	**96.52%**	**96.23%**	**95.81%**
DWS_3	328K	**98.73%**	96.72%	95.14%	95.02%	95.53%	95.49%
FCONV3	537K	93.47%	95.61%	93.53%	94.47%	93.27%	93.18%
FCONV5	1,488K	97.52%	94.24%	95.12%	95.04%	91.35%	93.89%

TABLE III: Performance of DWS_1 on different Quality factors

Quality factors	Datasets					
	Places-365	Tokuda et al.	COCO-val14	COCO-test17	SocialMedia	TRDG
0.45	98.30%	**98.39%**	**98.22%**	**98.31%**	94.45%	93.52%
0.55	97.81%	98.27%	98.09%	98.29%	94.49%	93.54%
0.65	98.43%	98.28%	98.17%	98.17%	94.67%	93.82%
0.75	**98.63%**	97.71%	98.18%	98.11%	95.09%	94.38%
0.85	98.28%	97.28%	98.22%	98.27%	**95.37%**	**94.79%**



(a) (b)

(c) (d)

Fig. 5: Synthetic images with their heat maps

is completely identified and marked with high density pixels.

- Four different camera images having very small text being stacked together to form a meme as shown in Fig.5 (b). The horizontal sharp edge so formed after stacking is identified by the horizontal line of higher intensity pixels.
- Text added on the camera image being well demarcated from the background in the heat map as shown in Fig.5 (c).
- Fig.5 (d) shows a complete synthetic image. We can see the regions of connected high density pixels.

Various scenes are presented in Fig.6. In (d), the scenic text is shown in a contrasting white background. There are only small structures of higher intensity blocks seen over the text area and the model correctly predicts the image with confidence of 95.86%. The solution was applied to a real scenario in a smartphone environment. We invited volunteers to test the solution. The solution was tested on around 51 users belonging to various demographics. A smart assistant interface was used to notify the users about the need for reviewing the stored or received images of various resolutions from social media. Our solution rans on the Samsung Galaxy S10 device with an average inference time of 24.3 milliseconds per image. In a real-world setting, it has been observed that the proposed approach has enriched users' efficacy in managing the device storage. We noted an average of 18% extra available storage space (profiled weekly) for each user.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

(a) (b)

(c) (d)

Fig. 6: Camera images with their heat maps

V. Conclusion

In this paper, we demonstrate the use of convolutional neural networks for solving the highly relevant problem of filtering social media messages. This work is the first of its kind in this problem domain and we achieved an average accuracy of 97.18% on camera images and 96.02% on synthetic images. We show that depthwise separable convolutions perform well in learning features to distinguish artificial features added in an image. The model shows promising results on various standard camera image datasets. Additionally, we have tested the relevance of the approach in practical scenarios. The high accuracy of the model ensures that we do not recommend a user to delete a photo that should be archived. We would like to extend this work for distinguishing photo-realistic computer graphics where edge transitions are smoother than other synthetic images on social media.

VI. 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] J. R. Corripio, D. A. González, A. S. Orozco, L. G. Villalba, J. Hernandez-Castro, and S. J. Gibson, "Source smartphone identification using sensor pattern noise and wavelet transform," 2013.

[3] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[5] M. Garbarino, *Automatic classification of natural and synthetic images*. Skolan för datavetenskap och kommunikation, Kungliga Tekniska högskolan, 2008.

[6] R. W. Lienhart and A. Hartmann, "Classifying images on the web automatically," *Journal of electronic imaging*, vol. 11, no. 4, pp. 445–455, 2002.

[7] S. Prabhakar, H. Cheng, Z. Fan, J. C. Handley, and Y.-w. Lin, "Picture/graphics classification system and method," Jan. 3 2006, uS Patent 6,983,068.

[8] V. Andrearczyk and P. F. Whelan, "Using filter banks in convolutional neural networks for texture classification," *Pattern Recognition Letters*, vol. 84, pp. 63–69, 2016.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[10] S. Fujieda, K. Takayama, and T. Hachisuka, "Wavelet convolutional neural networks for texture classification," *arXiv preprint arXiv:1707.07394*, 2017.

[11] B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, 2016, pp. 5–10.

[12] Y. Yao, W. Hu, W. Zhang, T. Wu, and Y.-Q. Shi, "Distinguishing computer-generated graphics from natural images based on sensor pattern noise and deep learning," *Sensors*, vol. 18, no. 4, p. 1296, 2018.

[13] D. Bhalang Tarianga, P. Senguptab, A. Roy, R. Subhra Chakraborty, and R. Naskar, "Classification of computer generated and natural images based on efficient deep convolutional recurrent attention model," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 146–152.

[14] N. Rahmouni, V. Nozick, J. Yamagishi, and I. Echizen, "Distinguishing computer graphics from natural images using convolution neural networks," in *2017 IEEE Workshop on Information Forensics and Security (WIFS)*. IEEE, 2017, pp. 1–6.

[15] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 868–882, 2012.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[17] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 413–420.

[18] J. Shi, L. Xu, and J. Jia, "Discriminative blur detection features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2965–2972.

[19] C. K. Ch'ng and C. S. Chan, "Total-text: A comprehensive dataset for scene text detection and recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 935–942.

[20] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.
[22] E. Tokuda, H. Pedrini, and A. Rocha, "Computer generated images vs. digital photographs: A synergetic feature and classifier combination approach," *Journal of Visual Communication and Image Representation*, vol. 24, no. 8, pp. 1276–1292, 2013.
[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[24] J. Huang, S. R. Kumar, M. Mitra, and W.-J. Zhu, "Image indexing using color correlograms," Jun. 12 2001, uS Patent 6,246,790.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE