LETTER

Efficient Multi-Scale Feature Fusion for Image Manipulation Detection*

Yuxue ZHANG[†], ***Nonmember and*** **Guorui FENG**[†a)], ***Member***

SUMMARY Convolutional Neural Network (CNN) has made extraordinary progress in image classification tasks. However, it is less effective to use CNN directly to detect image manipulation. To address this problem, we propose an image filtering layer and a multi-scale feature fusion module which can guide the model more accurately and effectively to perform image manipulation detection. Through a series of experiments, it is shown that our model achieves improvements on image manipulation detection compared with the previous researches.

key words: *image manipulation detection, convolutional neural network, high-pass filter, multi-scale feature fusion*

1. Introduction

The combination of computer technology and Internet makes it easy for everyone with various intentions to modify digital images by using advanced software tools. Image manipulation detection focuses on detecting manipulation traces and belongs to the field of image forensics. Generally speaking, image manipulation includes noise removal/addition, contrast enhancement, filtering enhancement, re-sampling, de-blurring, JPEG compression, and edge enhancement etc. [1], [2]. In particular cases, it is vital to find out this manipulation which an image has undergone.

In the past few years, researchers had developed various forensics technologies to confirm the authenticity of digital images or detect their processing history [3], [4]. According to these works, it was critical to find an effective feature representation that was helpful to distinguish between manipulated images and origin ones. For example, various methods to detect median filtering were also proposed in [5]–[7], where features were extracted from the first order difference map [5], either the order or the quantity of the gray levels [6], and the histogram bins of the first-order difference image [7] were studied and evaluated to be the detectors of median filtering. Whereas, it was a hard work to assess the true effectiveness of these feature representations, especially under the circumstances of distinguishing among a variety of filtering enhancement methods and taking the JPEG compression into account.

Manuscript received October 27, 2021.
Manuscript revised December 21, 2021.
Manuscript publicized February 3, 2022.

[†]The authors are with School of Communication and Information Engineering, Shanghai University, China.

[*]This work is supported by the National Science Foundation of China under Grant 62072295 and supported in part by Natural Science Foundation of Shanghai under Grant 19ZR1419000.

a) E-mail: fgr2082@aliyun.com

DOI: 10.1587/transinf.2021EDL8099

Nevertheless, the forensic algorithms described above were intended to detect a specific type of manipulation, and required strong theoretical knowledge and extensive experience. When distinguishing multiple image manipulations, these forensic algorithms were difficult to extract effective features.

With the rapid development of CNN, researchers have started to apply them to the field of image forensics [8]. At present, the application of deep learning methods in image forensics can be divided into three categories. The first category is the simple migration of those networks frequently used in Computer Vision (CV) tasks to image forensics tasks [9]. The second category is the tentative changes imposed on the input of the network [10]. Authors in [10] added an MFR layer before the conventional CNN which could suppress the interference caused by image edges and textures so that they could expose the trace left by median filtering. The third category is the modification of the network architectures [11]–[14]. In [12], authors proposed a new training algorithm specially applied to the first layer to learn prediction error filters. Structures or thoughts of principles based on GoogLeNet [15] and ResNet [13] were applied in these researches. Recently, Tan et al. proposed Efficient-Nets [16] which achieved the better performance in ImageNet classification.

In this paper, we propose a new approach to perform image manipulation detection which is able to distinguish traces of different image editing operations. The major contributions of our work are summarized as follows. 1) We add an image filtering layer before EfficientNet-B0. In this layer, we use eight high-pass filters of different directions, scales and sizes, which can capture weak and various manipulated traces. 2) We design a multi-scale feature fusion module to improve detection performance. In order to reduce the loss of manipulation trace information, we extract features of different scales from EfficientNet-B0, and then fuse these features in order to improve detection performance.

2. Proposed Model for Image Manipulations Detection

2.1 Comprehensive Framework

The proposed model is based on the state-of-the-art Efficient-Nets and uses the B0 architecture pre-trained on ImageNet for the trade-off between performance and complexity. The upper part of Fig. 1 shows the overall framework which is mainly composed of image filtering layer,

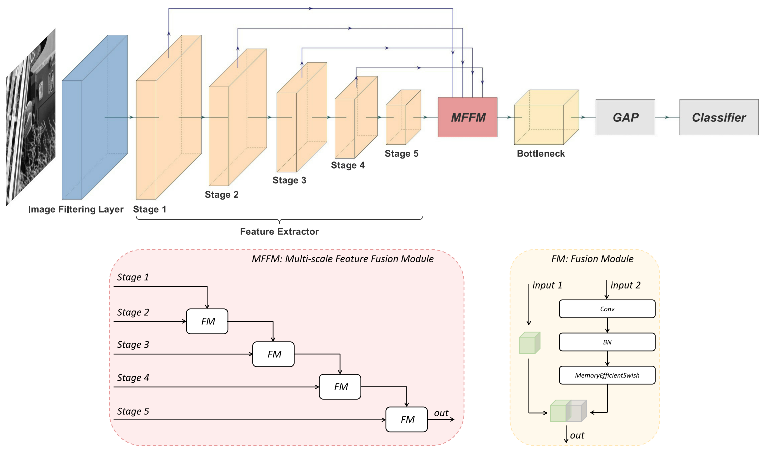

Fig. 1 The upper part is the framework of the proposed model architecture. According to the resolution of the feature map, the feature extractor of the EfficientNet-B0 architecture can be divided into five stages. *MFFM* represents the Multi-scale Feature Fusion Module which fuses the features of five stages to extract abundant features. In order to extract cross-channel features, we use 1 × 1 convolutional layer after *MFFM*. *GAP* represents Global Average Pooling.

feature extractor, multi-scale feature fusion module, and bottleneck layer. The size of the input grayscale images is 32×32.

2.2 Image Filtering Layer

We propose an image filtering layer to pre-process the input images prior to the subsequent processing. In [17], authors proposed two filter bases of different directions, scales and sizes which could be convolved with different base filters or the same base filter for three times to obtain a total of 328 filters. The residual of the filtered image and the original image is expressed as follows:

$$C^n_{i,k} = F_n(x_{i,k}) - x_{i,k} \tag{1}$$

where i denotes the image category and $0 \le i \le 5$, k denotes the k-th image in i-th image set, n denotes the n-th filter and $0 \le n < 328$, $F_n(\cdot)$ represents the convolutional operation of the n-th filter on the input image, and $C^n_{i,k}$ is the residual between the filtered image and the image itself. In order to reduce the impact of the specific content of the images and focus on the manipulated traces, we use the residual value to calculate. We perform the calculation of Eq. (1) on all the images in the i-th category, and the sums of them denote the result of these images processed by filter n. Here is the expression:

$$C^n_i = \sum_{k=1}^{s} C^n_{i,k} \tag{2}$$

The larger difference between different types of images generated by the same filter is, the stronger the ability of the filter to capture different manipulated traces is. Therefore, we subtract the sum of the residuals of different types of images in pairs, and take the minimum value to express the performance of the filter, denoted as S_n. The expression is as follows:

$$S_n = min||C^n_i - C^n_j||, \quad 0 \le i < j \le 5 \tag{3}$$

The bigger value of S_n, the better performance of filter to extract multiple manipulation traces. Finally, sort S_n and select the eight filters with the largest S_n as the convolutional kernel of the image filtering layer.

Table 1 Structure of the feature extractor. W/H/C: weight/height/channel.

Stage	Operator	Output Shape $(W \times H \times C)$
1	Conv3×3	32×32×32
	MBConv1, k3×3	32×32×16
2	[MBConv6, k3×3]×2	16×16×24
3	[MBConv6, k5×5]×2	8×8×40
4	[MBConv6, k3×3] ×3	4×4×80
	[MBConv6, k5×5] ×3	4×4×112
5	[MBConv6, k3×3] ×3	2×2×192
	[MBConv6, k5×5] ×3	2×2×320

2.3 Feature Extractor

According to the resolution of the feature map, the feature extractor of the EfficientNet-B0 architecture can be divided into five stages, and Table 1 lists the details of the architecture. The header *Stage* represents the current stage, *Operator* means the specific operation and *Output Shape* means the size of output feature map.

2.4 Multi-Scale Feature Fusion Module

Feature fusion must change along with the iteration of network, so we should start from the shallowest stage, and then merge the deeper stages iteratively in order that we can make full use of shallow information from different fusion stage.

For example, given f_1, f_2 and f_3, we first fuse f_1 with f_2 to generate f_1', and then fuse f_1' with f_3. Therefore, we propose a kind of connections shown in Fig. 1 between each stage. The fusion function J is as follows:

$$J(f_1, \ldots, f_n) = \begin{cases} f_1 & n = 1 \\ J(L(f_1, f_2), \ldots, fn) & n > 1 \end{cases} \tag{4}$$

where L is the Fusion Module (FM). Although any kind of network structure can be used as feature extraction module, we choose to use the structure of convolution-BN-activation function for the sake of reducing the amount of calculation. The convolutional kernel size is 2×2 and stride is 2. The expression is as follows:

$$L(f_1, f_2) = Concatenation(\sigma(BN(Wf_1 + b)), f_2) \tag{5}$$

where *Concatenation* represents the feature map fusion operation, σ represents the non-linear activation, *BN* represents batch normalization and W and b represents weight and bias respectively.

3. Experimental Results

3.1 Experimental Setups

We test on the UCID [18] database containing 1338 gray-scale images of size 512×384. In order to have sufficient data, we crop the UCID database. After cropping each image, we obtain 192 blocks of 32×32 sized from each of the 1338 images and thus 256,896 images are made. Five editing techniques in Table 2 are applied to these original images. In our experiments, 50% of images are randomly selected as the training set, 25% as the validation set, and the remaining 25% as the testing set.

In every experiment, the training process uses the Adam optimization method. The learning rate is defined as 10^{-3}. The loss function uses the cross-entropy loss function. We train the model for 20 epochs in each experiment. The input for each training iteration is a mini-batch of 128 images. Our experiments are all run on an NVIDIA TITAN Xp GPU. The datasets used in this work are all converted to the numpy format.

3.2 Image Manipulation Detection

In this part, we evaluate the performance of the proposed model in performing multiple image manipulation detection. The average accuracy of the proposed model in identifying multiple image manipulation is 97.41%. Table 3 shows the confusion matrix of the classification results of the model performing multiple image manipulation detection. In this table, *OR* means the original image, *JPEG* means JPEG compression, *RS* means Re-sampling, *AWGB* means Additive White Gaussian Noise, *MF* means Median Filtering, and *GF* means Gaussian Filtering. From the table, we can see that the accuracy of each image editing technique detected exceeds 94%, which is a satisfactory result. Moreover, except for the original and additive white Gaussian noise images that are detected with accuracy of 94.5% and 95.46%, each of the other manipulation is identified with an accuracy of greater than 97%. These results show that the proposed model can both accurately detect manipulated images and identify the type of image manipulation.

3.3 Comparison with Three CNNs

Using the same training, validation, and testing set described in Sect. 3.1, we compare the performance of our trained model for multiple image manipulation detection with MISLNet [12], Densely Connected CNN [19] and ModuleQNN-A [20]. In order to make a fair comparison, we implement all models with the Pytorch 1.1 deep learning framework. Besides, we only select the best CNN (ModuleQNN-A) automatically searched out in the multi-purpose forensics in [20] for re-implementation. Table 4 shows the accuracy of four CNNs for multiple image manipulation detection on the testing set. For each image manipulation, the best detection results are marked in bold. In contrast, our model can achieve 97.41% accuracy. From the table, it can be notice that our model achieves a better recognition rate in the original image, re-sampling, additive white Gaussian noise, median filtering and Gaussian filtering. These results show that the performance of our model is more effective.

Table 2 Five different editing techniques with parameters used to create experimental database for our experiments.

editing technique	Parameter
JPEG compression	Quality factor = 70
Resampling	Scaling factor = 1.5
Additive White Gaussian Noise	$\sigma = 2$
Median Filtering	$K_{size} = 5$
Gaussian Filtering	$K_{size} = 5, \sigma = 1.1$

Table 3 Confusion matrix (%) for identifying the multiple image manipulation Listed in Table 2.

Actual / Predicted	Original	JPEG	RS	AWGB	MF	GF
Original	94.50	0.55	0.11	4.68	0.15	0.01
JPEG	0.22	99.70	*	0.08	*	*
RS	0.14	0.18	99.55	*	0.07	0.06
AWGB	4.47	0.07	*	95.46	*	*
MF	0.85	0.46	0.16	*	97.36	0.33
GF	0.04	0.44	0.16	*	1.45	97.91
						Average: 97.41

* represents a prediction accuracy below 0.01%

Table 4 Image manipulation detection accuracy (%) using the proposed model compared with three CNNs.

CNN / Techniques	Original	JPEG	RS	AWGB	MF	GF	Average
MISLnet [12]	85.99	98.90	97.82	91.06	93.35	95.24	93.63
Densely Connected CNN [19]	94.30	98.95	99.30	91.74	93.66	96.49	95.74
ModuleQNN-A [20]	92.71	**99.81**	99.36	95.13	95.53	95.69	96.37
The Proposed Model	**94.50**	99.70	**99.55**	**95.46**	**97.36**	**97.91**	**97.41**

Table 5 Accuracy (%) of image manipulation detection with different selections of image filtering layer.

Image filtering layer	Accuracy
SRM filters [21]	96.37
Init high-pass filters [22]	97.14
Base filters [17]	97.21
Our method	**97.41**

Table 6 Accuracy (%) of image manipulation detection with different multiscale feature fusion strategies.

Number	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Accuracy
1	√	√	√	√	√	**97.41**
	√		√	√	√	97.36
2	√	√		√	√	97.37
	√	√	√		√	97.32
	√	√			√	97.20
3	√		√		√	97.19
	√			√	√	97.20

3.4 Ablation Experiment

3.4.1 Selection of Image Filtering Layer

We will evaluate the impact of different filtering layer on the performance of image manipulation detection through four sets of experiments: (1) using three SRM filters in [21]; (2) initializing eight 3×3 filters using the method proposed in [22], denoted as init high-pass filters; (3) using eight base filters in [17]; and (4) the eight filters selected in this paper. Only the image filtering layer is changed in the experiment, keeping the rest of the network architecture described in Sect. 2 unchanged. The detection accuracies based on different image filtering layers are shown in Table 5. As can be seen from the table, the accuracy of our method improves by 0.2% compared with the method using eight base filters, demonstrating the effectiveness of filter selection strategy in Sect. 2.2. Compared with the init high-pass filters, the accuracy of our method is improved by 0.27%. This is because we use high-pass filters of different scales, directions, and sizes. Compared with fixed-size init high-pass filters, our method increases the diversity of filters, which in consequence can capture more different manipulation traces. Compared with SRM filters, the accuracy improvement of our method is more significant, with an increase of 1.04%. This is because SRM filters can not capture all the manipulated traces, which affects the performance of image manipulation detection. These experimental results demonstrate the advantages of our proposed eight filters in extracting manipulated features.

3.4.2 Multi-Scale Feature Fusion Strategy

We select a multi-scale feature fusion strategy by designing three sets of experiments: (1) all five stages are fused, (2) four stages are fused, and (3) three stages are fused, where each set of experiments selects the features of the fifth stage. The accuracy of image manipulation detection based on different fusion strategies is shown in Table 6. Comparing the accuracy of the three sets of experiments, it can be found that the accuracy of classification decreases as the number of fusion stages decreases, indicating that using only some of the stages to construct multi-scale feature fusion is not conducive to extracting generalized features that are friendly to image manipulation detection, thus reducing the detection performance of the model.

4. Conclusion

In this paper, we propose an approach using an improved deep learning model for image manipulation detection. Eight filter kernels of different directions, scales and sizes are selected and then used as convolutional kernels in the first layer of the proposed model to help it to generate better features for classification. The features of different levels are fused with the features extracted by EfficientNet-B0, so that the model can extract abundant features and effectively improve network performance. Through a series of experiments and comparisons, we have demonstrated that the proposed model is able to detect multiple image manipulation with higher accuracy compared with the previous work.

References

[1] A.C. Popescu and H. Farid, “Exposing digital forgeries by detecting traces of resampling,” IEEE Transactions on signal processing, vol.53, no.2, pp.758–767, 2005.

[2] W. Luo, J. Huang, and G. Qiu, “Jpeg error analysis and its applications to digital image forensics,” IEEE Transactions on Information Forensics and Security, vol.5, no.3, pp.480–491, 2010.

[3] C. Tang, C. Hou, Y. Hou, P. Wang, and W. Li, “An effective edge-preserving smoothing method for image manipulation,” Digital Signal Processing, vol.63, pp.10–24, 2017.

[4] M.C. Stamm, M. Wu, and K.R. Liu, “Information forensics: An overview of the first decade,” IEEE access, vol.1, pp.167–200, 2013.

[5] G. Cao, Y. Zhao, R. Ni, L. Yu, and H. Tian, “Forensic detection of median filtering in digital images,” 2010 IEEE International Conference on Multimedia and Expo, pp.89–94, IEEE, 2010.

[6] H.-D. Yuan, “Blind forensics of median filtering in digital images,” IEEE Transactions on Information Forensics and Security, vol.6, no.4, pp.1335–1345, 2011.

[7] M. Kirchner and J. Fridrich, “On detection of median filtering in digital images,” Media forensics and security II, p.754110, International Society for Optics and Photonics, 2010.

[8] J. Yang, Z. Liang, Y. Gan, and J. Zhong, “A novel copy-move forgery detection algorithm via two-stage filtering,” Digital Signal Processing, vol.113, p.103032, 2021.

[9] L. Baroffio, L. Bondi, P. Bestagini, and S. Tubaro, “Camera identification with deep convolutional networks,” arXiv preprint arXiv: 1603.01068, 2016.

[10] J. Chen, X. Kang, Y. Liu, and Z.J. Wang, “Median filtering forensics based on convolutional neural networks,” IEEE Signal Processing Letters, vol.22, no.11, pp.1849–1853, 2015.

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

[12] B. Bayar and M.C. Stamm, “Constrained convolutional neural net-

works: A new approach towards general purpose image manipulation detection," IEEE Transactions on Information Forensics and Security, vol.13, no.11, pp.2691–2706, 2018.

[13] H. Tang, R. Ni, Y. Zhao, and X. Li, "Detection of various image operations based on cnn," 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, pp.1479–1485, IEEE, 2017.

[14] W. Quan, K. Wang, D.-M. Yan, and X. Zhang, "Distinguishing between natural and computer-generated images using convolutional neural networks," IEEE Transactions on Information Forensics and Security, vol.13, no.11, pp.2772–2787, 2018.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," Proceedings of the IEEE conference on computer vision and pattern recognition, pp.1–9, 2015.

[16] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," International Conference on Machine Learning, pp.6105–6114, PMLR, 2019.

[17] G. Feng, X. Zhang, Y. Ren, Z. Qian, and S. Li, "Diversity-based cascade filters for jpeg steganalysis," IEEE Transactions on Circuits and Systems for Video Technology, vol.30, no.2, pp.376–386, 2019.

[18] G. Schaefer and M. Stich, "Ucid: An uncompressed color image database," Storage and Retrieval Methods and Applications for Multimedia 2004, pp.472–480, International Society for Optics and Photonics, 2003.

[19] Y. Chen, X. Kang, Y.Q. Shi, and Z.J. Wang, "A multi-purpose image forensic method using densely connected convolutional neural networks," Journal of Real-Time Image Processing, vol.16, no.3, pp.725–740, 2019.

[20] Y. Chen, Z. Wang, Z.J. Wang, and X. Kang, "Automated design of neural network architectures with reinforcement learning for detection of global manipulations," IEEE Journal of Selected Topics in Signal Processing, vol.14, no.5, pp.997–1011, 2020.

[21] P. Zhou, X. Han, V.I. Morariu, and L.S. Davis, "Learning rich features for image manipulation detection," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.1053–1061, 2018.

[22] I.C. Camacho and K. Wang, "A simple and effective initialization of cnn for forensics of image processing operations," Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, pp.107–112, 2019.