An Image Splicing and Copy-Move Detection Method Based on Convolutional Neural Networks with Global Average Pooling

Qian Zhang[1,2], Jun Sang[1,2(✉)], Weiqun Wu[1,2], Bin Cai[1,2], Zhongyuan Wu[1,2], and Haibo Hu[1,2]

[1] Key Laboratory of Dependable Service Computing in Cyber Physical Society of Ministry of Education, Chongqing University, Chongqing 40004, China
jsang@cqu.edu.cn
[2] School of Big Data and Software Engineering, Chongqing University, Chongqing 401331, China

Abstract. Splicing and copy-move are two well-known methods of image tampering, while detection of image splicing and copy-move forgery is an important research topic in image forensics. In this paper, a method based on convolutional neural network with global average pooling was proposed for splicing and copy-move tampering detection. To detect image tampering, the inconsistency between the authentic images and the tampered images should be captured regardless of the image contents. So, the existing strategy using high-pass filter in SRM as initialization of the first layer was improved to reduce the influence of image content and make the features more diverse on each channel at the same time. In order to reduce the number of parameters in the fully connected layers and avoid overfitting, global average pooling was utilized before fully connected layers in the proposed model. Experiments on three public image tampering datasets demonstrated that the proposed method outperformed some state-of-the-art methods.

Keywords: Image tampering detection · Image splicing · Copy-move · Convolutional neural networks (CNNs) · SRM (Spatial Rich Model) · Global average pooling

1 Introduction

With the development of digitization and informatization, digital image has become an important information carrier. At the same time, the emergence of more and more powerful and sophisticated image processing software makes it easy to modify and manipulate images without leaving any traceable proof. However, illegal use of the forged images results in negative impact on human cognition and judgment of the objective world. Therefore, image forensics is an urgent problem to be solved, and a research hotspot in the field of information security.

Among many image tampering methods, splicing and copy-move are the most commonly used. In image splicing forgery, one region is cut and pasted to another image; while in copy-move forgery, one region is copied and pasted over other region

Y. Zhao et al. (Eds.): ICIG 2019, LNCS 11903, pp. 255–265, 2019.
https://doi.org/10.1007/978-3-030-34113-8_22

in the same image [1]. Many splicing detection methods [2–5] and copy-move detection approaches [1, 6–8] based on handcrafted features have been proposed to detect these two kinds of image tampering, respectively.

In recent years, some researchers have applied deep learning to image tampering detection. Rao et al. [9] proposed a convolutional neural network (CNN) to automatically learn feature from images, and used the high-pass filters in Spatial Rich Model (SRM) [10] as the initialization of the first layer to suppress the image content. In [11], a Stacked Autoencoder model was used to learn the complex feature to detect tampered images with different image formats. Chen et al. [12] proposed a new technique based on the analysis of the camera response functions combining with CNN for splicing and copy-move detection and localization. In [13], resampling features and deep learning were used to detect and localize image manipulations. Pomari et al. [14] combined the high representation power of Illuminant Maps with CNN to eliminated the laborious feature engineering process and locate forgery region. In [15], the noise features were extracted from a SRM filter layer to discover the noise inconsistency for tampered regions detection.

In this paper, a CNN model referencing to VGGNet [16] for image splicing and copy-move detection was proposed by analyzing the structure in [9]. Inspired by [9, 15], our initialization strategy of the first layer using 30 high-pass filters in SRM ensured not only suppression for image content, but also more diverse features on each channel. In addition, as there were too many parameters in the 5×5 convolutional layer in [9], two 3×3 convolutional layers were stacked to replace the 5×5 convolutional layer. In order to further reduce overfitting and improve network performance, global averaging pooling was applied before the fully connected layer of the network. Experimental results showed that the proposed method outperformed some existing methods on 3 public datasets.

The rest of the paper is organized as follows. In Sect. 2, the proposed method is described in detail. Section 3 presents and discusses experimental results and analysis. Finally, Sect. 4 contains the conclusion and future work.

2 The Proposed Method

2.1 Global Average Pooling

Convolutional neural networks (CNNs) usually use the fully connected layer in the last several layers to weight the features of the convolutional layer. Each neuron in the fully connected layer is connected to all of the neurons in the previous layer, so the fully connected layers usually have many parameters. Also, the fully connected layers tend to overfitting because of too many parameters, thus hampering the generalization ability of the overall network.

In [17], global average pooling (GAP) was proposed to replace fully connected layers, which required the last convolutional layer to output the same number of feature maps as the number of categories. The global average pooling calculates the average value for each feature map as the output, and there is no parameter to optimize. However, the number of the output feature maps of the last convolutional layer was

inconsistent with the number of categories in our proposed method. So, the global average pooling could not directly replace the fully connected layer and was added before the fully connected layer to decrease the dimension of the feature and reduce the parameters in fully connected layers. Figure 1 shows the difference between the fully connected layer and GAP.

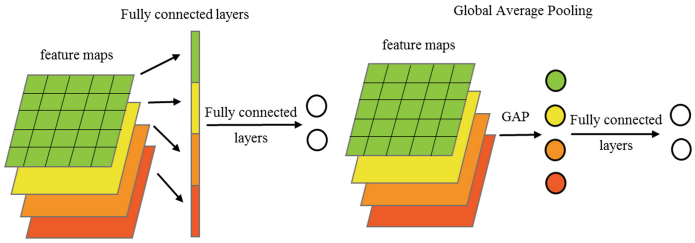

Fig. 1. The difference between the fully connected layer and GAP.

In Fig. 1, suppose there are four feature maps with size of 5 × 5, and the final output category is 2. Without applying GAP, the feature maps are constructed into a 100-dimensional vector, and then the vector is fully connected with the output. In this way, there are 200 parameters. By applying GAP, the global average value of each feature map is calculated to get a 4-dimensional vector, and then the vector is fully connected with the output. So, only 8 parameters are needed. By using GAP, the number of parameters for fully connected layer is greatly reduced.

Considering the finiteness of training data and the subtlety of tampering features, global averaging pooling was applied in our proposed network to avoid overfitting and improve performance.

2.2 High Pass Filtering

In 2012, Fridrich et al. [10] proposed a steganalysis method based on SRM, which constructed a set of high-pass filters of linear and nonlinear to obtain a variety of residual images. The image steganalysis methods based on SRM model the residual image rather than directly model the image itself, aiming at suppressing the influence of image content on steganalysis and strengthen the steganographic signal. There were 30 basic filters in a set of high-pass filters and they were divided into six categories, including "1st", "2nd", "3rd", "SQUARE", "EDGE3 × 3", and "EDGE5 × 5".

Image forgery detection, like image steganalysis, does not focus on the content of image, but only on the subtle artifacts (noise) introduced by the tampering operations. So the high pass filters in SRM have been used in image forgery detection to suppress the image content. Cozzolino et al. [18] and Verdoliva et al. [19] computed the residuals with a third-order ("3rd") linear high-pass filter to ensure a good performance for forgery detection. Based on the high-pass filter residuals, tampering detection and localization were carried out through a series of operations such as quantization and computation of co-occurrences histogram.

High-pass filters in SRM were not only used in image forgery detection based on traditional methods, but also in the deep learning based methods. Rao et al. [9] proposed a 10-layer CNN using the 30 basic high-pass filters in SRM for initialization of the first layer to detect splicing and copy-move images. The initialization served as a regularizer to suppress the image content and capture the subtle artifacts introduced by the tampering operations. Zhou et al. [15] proposed a two-stream Faster R-CNN for image tampering detection and localization, in which three high-pass filters in SRM were also used to extract noise features.

In our proposed method, the first convolutional layer was initialized with 30 high-pass filters in SRM to focus on the subtle tampering artifacts referencing to [9]. But compared with the initialization strategy in [9], our strategy was simpler and made the features diverse on each channel for color image. For grayscale images, the input image had only one channel, and the first layer was initialized directly with 30 high-pass filters. For color images, the input includes 3 channels and the weight of each channel was initialized with 30 high-pass filters. The initialization of each channel weight is defined as follows:

$$W_{CNN}(x, y) = W_{SRM}(x, y) \tag{1}$$

Where W_{CNN} and W_{SRM} are the weight matrix of size 5 × 5 used on every channel and the filter kernels of size 5 × 5 in SRM, respectively, and x, y = 1, 2, 3, 4, 5.

2.3 Network Architecture

In [9], a 10 layers CNN (denote as RaoNet) was designed for image splicing and copy-move detection, which can automatically learn feature representations from the input images. The 10-layer CNN structure was shown in Fig. 2. It consisted of 8 convolutional layers, 2 pooling layers and a fully connected layer with a softmax classifier. The first and second convolutional layers had 30 filters of size 5 × 5, while other convolutional layers all had 16 filters of size 3 × 3. The first convolutional layer was initialized with 30 basic high-pass filters in SRM, while other layers were initialized with "xavier". The stride of the second convolutional layer was set to 2, and the stride of other convolutional layers was set to 1. Max pooling with filter of size 2 × 2 was used

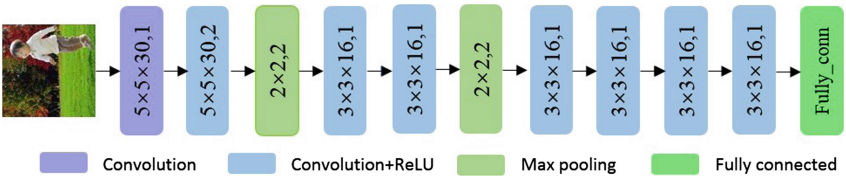

Fig. 2. The architecture of RaoNet [9]. {5 × 5 × 30, 1} denotes 30 convolution kernels of 5 × 5, and the stride is 1. {2 × 2, 2} denotes pooling of 2 × 2, and the stride is 2.

after the second and fourth convolutional layers. Activation function was Rectified Linear Units (ReLU), and fully connected layer was followed by dropout technique.

Three observations can be drawn from Fig. 2. Firstly, there were two convolutional layers with 5 × 5 filter in the RaoNet. It was verified in VGGNet that the stack of two 3 × 3 convolutional layers had an equivalent receptive field of 5 × 5, and the number of parameters was also decreased. Therefore, we replaced the second 5 × 5 convolutional layer in RaoNet with two 3 × 3 convolutional layers. In order to further reduce the number of parameters, the number of convolutional kernels was also modified to 16. Secondly, it was noted that the convolutional layer with a stride 2 was followed by the pooling layer with a stride 2 in RaoNet, which made the size of the feature map rapidly reduced and the feature information seriously lost. To solve this problem, the structure of RaoNet was improved by adjusting the location of pooling layer and increasing the number of network layers. Lastly, only one necessary fully connected layer in RaoNet was used at the end of the network aiming at reducing parameters. To further reduce the number of parameters and avoid overfitting, we used global average

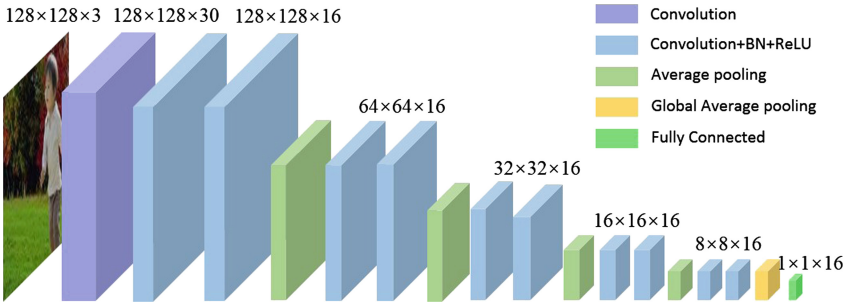

Fig. 3. The proposed CNN architecture.

pooling before the fully connection layer in the proposed model. Based on the above analysis, we proposed a CNN model as shown in Fig. 3.

The proposed network model contained 11 convolutional layers, 4 average pooling, 1 global average pooling, and 1 fully connected layer with 2-way softmax classifier. The first convolutional layer had 30 filters of size 5 × 5 and was initialized with 30 basic high-pass filters in SRM. Other convolutional layers had 16 filters of size 3 × 3 and were initialized with "xavier". Except for the first layer, BN and ReLU were used after other convolutional layers. Average pooling was used in the proposed model, and global average pooling was applied before the fully connected layers. RaoNet had 41,390 parameters to be trained, while our proposed model had 27,338 parameters. Compared with RaoNet, the layer number of the proposed network model was increased, but the number of parameters was only 66.03% of RaoNet. The number of parameters in the fully connected layer were also reduced from 800 to 32. The detailed network structure of the proposed model was shown in Table 1.

Table 1. The detailed structure of the proposed CNN.

Layer	Input size	Process	Kernels size	Output size
1	128 × 128 × 3	High pass filtering	30 × 5 × 5 (stride 1)	128 × 128 × 30
2	128 × 128 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	128 × 128 × 16
3	128 × 128 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	128 × 128 × 16
4	128 × 128 × 16	Average pooling	2 × 2 (stride 2)	64 × 64 × 16
5	64 × 64 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	64 × 64 × 16
6	64 × 64 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	64 × 64 × 16
7	64 × 64 × 16	Average pooling	2 × 2 (stride 2)	32 × 32 × 16
8	32 × 32 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	32 × 32 × 16
9	32 × 32 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	32 × 32 × 16
10	32 × 32 × 16	Average pooling	2 × 2 (stride 2)	16 × 16 × 16
11	16 × 16 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	16 × 16 × 16
12	16 × 16 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	16 × 16 × 16
13	16 × 16 × 16	Average pooling	2 × 2 (stride 2)	8 × 8 × 16
14	8 × 8 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	8 × 8 × 16
15	8 × 8 × 16	Conv-BN-ReLU	16 × 3 × 3 (stride 1)	8 × 8 × 16
16	8 × 8 × 16	GAP	16 × 8 × 8	1 × 1 × 16
17	1 × 1 × 16	Fully connected	–	1 × 1

3 Experiments

The CPU and GPU of the experimental computer were Intel (R) Xeon CPU e5-2683 v3@2.00 GHz and NVIDIA TESLA K80, respectively. The experimental platform was equipped with 64 bit ubuntu14.04, Anaconda3.4, CUDA Toolkit9.0, Opencv3.4. The proposed model was implemented using TensorFlow framework.

3.1 Dataset for Experiments

The datasets used in our experiment to analyze the accuracy of proposed method were the CASIA v1.0 [20], CASIA v2.0 [20] and Columbia gray datasets [21]. The CASIA v1.0 dataset consists of 800 authentic and 921 forged color images with size of 384 × 256. All of the images are in JPEG format without applying any post processing. The CASIA v2.0 dataset contains 7,491 authentic and 5,123 tampered color images with size ranging from 240 × 160 to 900 × 600 pixels in JPEG, BMP and TIFF formats. The geometric transforms, such as scaling and rotation, are applied for cropped image regions for the CASIA v1.0 and CASIA v2.0 datasets. In addition, post-processing is also adopted to forged image for the CASIA v2.0 datasets. The Columbia gray dataset contains 933 authentic and 912 spliced gray images with size of 128 × 128 in BMP format, and no post-processing is applied to the splicing image.

In the experiment, images in the CASIA v1.0 and CASIA v2.0 datasets were cropped to 128 × 128, which is helpful for CNN to learn more representative samples. For authentic image, a patch of 128 × 128 was cropped randomly from the image. For the tampered image, a patch of 128 × 128 was also cropped, while the ratio between the tampered region and the original region was kept as 1:1 as possible.

3.2 Experimental Settings

Due to the large difference between the number of the authentic images and that of the tampered images in the CASIA v2.0 dataset, 5123 authentic image were randomly selected from 7413 authentic images to keep balance between authentic and forged images. To evaluate the performance of the proposed method, 5/6 of the images were selected for training and 1/6 of the remaining images were used for testing. The training data were augmented by flipping horizontally and vertically for Columbia gray dataset. A six fold cross-validation evaluation protocol was adopted to validate the performance of proposed method. Adam was used to optimize the network. Initial learning rate, batch size and maximum epoch were set to 0.001, 32, and 500, respectively.

3.3 Results and Analysis

Effectiveness of Initialization with SRM. The first layer of the network was initialized with the 30 basic high-pass filters in SRM, which was helpful for the network to learn tampering features. Table 2 showed the comparison results between using SRM as the initialization of the first convolutional layer (denoted as CNN-SRM) and using "xavier" as the initialization of the first layer (denoted as CNN-xavier) for proposed model on Columbia gray, CASIA v1.0, and CASIA v2.0 datasets.

Table 2. Detection accuracy (%) with and without SRM initialization.

Method	Columbia gray	CASIA v1.0	CASIA v2.0
CNN-xavier	94.90	93.93	99.27
CNN-SRM	97.82	99.30	99.70

As can be seen from Table 2, when the high-pass filters were used to initialize the first convolutional layer, the detection accuracy was improved by 2.92%, 5.37% and 0.43% on Columbia gray, CASIA v1.0 and CASIA v2.0 datasets, respectively. Compared with the Columbia gray and CASIA v1.0 datasets, CASIA v2.0 dataset has more samples in it, and the improvement was not obvious for the CASIA v2.0 dataset.

We further compared the convergence of proposed model using SRM and not using SRM. Figure 4 showed the evolutions of training loss versus number of epochs on 3 datasets. It was observed that the CNN-SRM converged much faster and had smaller fluctuations than CNN-xavier in training loss.

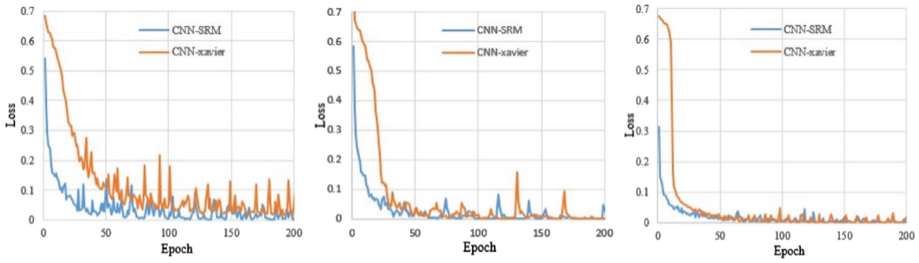

Fig. 4. The convergence comparison between CNN-SRM and CNN-xavier on Columbia gray, CASIA v1.0 and CASIA v2.0 datasets, respectively.

Performance Comparisons with Different Pooling Methods. Image tampering detection does not focus on the content of images, but on the noise features introduced by tampering. In order to explore the suitable pooling for image tampering detection, comparative experiments were conducted with average pooling and max pooling on Columbia gray, CASIA v1.0 and CASIA v2.0 datasets. Table 3 presented the detection accuracy of the comparison.

Table 3. Detection performance (%) of the proposed method with different pooling.

Method	Columbia gray		CASIA v1.0		CASIA v2.0	
	Average pooling	Max pooling	Average pooling	Max pooling	Average pooling	Max pooling
RaoNet	–	96.38	97.71	97.39	97.48	97.83
Ours	97.82	97.57	99.30	99.06	99.70	99.79

For Columbia gray dataset, the detection accuracy was 96.36% for RaoNet using max pooling, while it cannot obtain a decent result using average pooling. And the accuracy of the proposed model with average pooling was better than with max pooling on Columbia gray dataset. On CASIA v1.0 dataset, RaoNet and the proposed model achieved better performance using average pooling. While both RaoNet and the proposed model obtained higher accuracy using max pooling on CASIA v2.0 dataset. Overall, the difference of detection accuracy with average pooling or max pooling was not very obvious on the three datasets.

Effectiveness with GAP. In order to verify the effectiveness with GAP, the comparison results with and without GAP were shown in Table 4. The detection accuracy with GAP was higher than that of without GAP on Columbia gray, CASIA v1.0 and CASIA v2.0 datasets, regardless of whether the max pooling or average pooling were used. And it can be observed that the performance improvement with GAP was more obvious on the CASIA v1.0 dataset. It may be that there was less data in CASIA v1.0 dataset and the parameters in fully connected layer were not adequately trained, resulting in

overfitting. Therefore, GAP can improve network performance by reducing the number of parameters.

Table 4. Detection accuracy (%) with and without GAP.

Method	Columbia gray		CASIA v1.0		CASIA v2.0	
	Average pooling	Max pooling	Average pooling	Max pooling	Average pooling	Max pooling
Ours (without-GAP)	97.07	97.45	96.11	96.10	99.33	99.45
Ours (with-GAP)	97.82	97.57	99.30	99.06	99.70	99.79

Performance Comparisons with Other Methods. Finally, to evaluate the performance of the proposed method, the proposed method was compared with other existing methods and the experimental results were shown in Table 5.

Table 5. Detection accuracy (%) compared with other methods on three datasets.

Methods	Columbia gray	CASIA v1.0	CASIA v2.0
Muhammad [7]	–	94.89	97.33
Goh [8]	–	90.18	96.21
Rao [9]	96.38	98.04	97.83
Prakash [1]	–	99.44	98.89
Han [5]	91.28	98.95	97.28
Ours	97.82	99.30	99.70

As shown in Table 5, the proposed method outperformed some existing methods with an accuracy of 97.82%, 99.30% and 99.70% on Columbia gray, CASIA v1.0 and CASIA v2.0 datasets, respectively. Since the Columbia gray dataset consists of grayscale images, the methods [1, 7, 8] based on *YCbCr* channels were not applicable. While the proposed method was applicable not only to color images, but also to grayscale images. The accuracy of our method was lower than that of the method in [1] by 0.14% on CASIA v1.0 dataset, but our proposed method outperforms the method in [1] by 0.81% on the more challenging CASIA v2.0 dataset. Compared with RaoNet in [9], the accuracy of the proposed method was higher by 1.44%, 1.26% and 1.87% on Columbia gray, CASIA v1.0 and CASIA v2.0 datasets, respectively.

4 Conclusions

In this paper, we presented a method based on CNN with GAP to detect image tampering, which could be applied to the splicing and copy-move tampering at the same time. The proposed CNN model included 11 convolutional layers, 4 average pooling, 1 global average pooling, and 1 fully connected layer. Every two 3×3 convolutional layers was followed by a pooling layer. In order to reduce the influence of image content on tampering features and made the features diverse on each channel, a simple strategy was adopted to initialize the first convolutional layer of the proposed CNN with 30 high-pass filters in SRM. Global average pooling was adopted before fully connected layers to reduce the parameters and avoid the network overfitting. The experiments demonstrated that the proposed method outperformed some existing methods. The detection accuracies were 97.82%, 99.30%, and 99.70% on Columbia gray, CASIA v1.0, and CASIA v2.0 datasets, respectively. In future work, the effectiveness of the proposed method will be tested on more complex dataset, and the proposed method will be introduced into other image tampering detection.

References

1. Prakash, C.S., Kumar, A., Maheshkar, S., Maheshkar, V.: An integrated method of copy-move and splicing for image forgery detection. Multimedia Tools Appl. **77**, 26939–26963 (2018)
2. Ng, T.T., Chang, S.F., Sun, Q.: Blind detection of photomontage using higher order statistics. In: IEEE International Symposium on Circuits and Systems 2004, vol. 5, pp. V–V. IEEE Computer Society, Washington (2004)
3. Fu, D., Shi, Y.Q., Su, W.: Detection of image splicing based on Hilbert-Huang transform and moments of characteristic functions with wavelet decomposition. In: Shi, Y.Q., Jeon, B. (eds.) IWDW 2006. LNCS, vol. 4283, pp. 177–187. Springer, Heidelberg (2006). https://doi.org/10.1007/11922841_15
4. Pun, C.M., Liu, B., Yuan, X.C.: Multi-scale noise estimation for image splicing forgery detection. J. Vis. Commun. Image Represent. **38**, 195–206 (2016)
5. Han, J.G., Park, T.H., Moon, Y.H., Eom, I.K.: Quantization-based Markov feature extraction method for image splicing detection. Mach. Vis. Appl. **29**(3), 543–552 (2018)
6. Huang, H., Guo, W., Zhang, Y.: Detection of copy-move forgery in digital images using SIFT algorithm. In: 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, vol. 2, pp. 272–276. IEEE Computer Society, Washington (2008)
7. Muhammad, G., Al-Hammadi, M.H., Hussain, M., Bebis, G.: Image forgery detection using steerable pyramid transform and local binary pattern. Mach. Vis. Appl. **25**(4), 985–995 (2014)
8. Goh, J., Thing, V.L.: A hybrid evolutionary algorithm for feature and ensemble selection in image tampering detection. Electron. Secur. Digit. Forensics **7**(1), 76–104 (2015)
9. Rao, Y., Ni, J.: A deep learning approach to detection of splicing and copy-move forgeries in images. In: IEEE International Workshop on Information Forensics and Security 2016, pp. 1–6. IEEE Computer Society, Washington (2016)
10. Fridrich, J., Kodovsky, J.: Rich models for steganalysis of digital images. IEEE Trans. Inf. Forensics Secur. **7**(3), 868–882 (2012)

11. Zhang, Y., Goh, J., Win, L.L., Thing, V.L.: Image region forgery detection: a deep learning approach. In: Mathur, A., Roychoudhury, A. (eds.) Singapore Cyber Security R&D Conference (SG-CRC) 2016, pp. 1–11. IOS, Berlin (2016)
12. Chen, C., McCloskey, S., Yu, J.: Image splicing detection via camera response function analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017, pp. 5087–5096. IEEE Computer Society, Washington (2017)
13. Bunk, J., et al.: Detection and localization of image forgeries using resampling features and deep learning. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) 2017, pp. 1881–1889. IEEE Computer Society, Washington (2017)
14. Pomari, T., Ruppert, G., Rezende, E., Rocha, A., Carvalho, T.: Image splicing detection through illumination inconsistencies and deep learning. In: 25th IEEE International Conference on Image Processing (ICIP) 2018, pp. 3788–3792. IEEE Computer Society, Washington (2018)
15. Zhou, P., Han, X., Morariu, V.I., Davis, L.S.: Learning rich features for image manipulation detection. In: IEEE Conference on Computer Vision and Pattern Recognition 2018, pp. 1053–1061. IEEE Computer Society, Washington (2018)
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
17. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint arXiv:1312.4400 (2013)
18. Cozzolino, D., Poggi, G., Verdoliva, L.: Splicebuster: A new blind image splicing detector. In: IEEE International Workshop on Information Forensics and Security (WIFS) 2015, pp. 1–6. IEEE Computer Society, Washington (2015)
19. Verdoliva, L., Cozzolino, D., Poggi, G.: A feature-based approach for image tampering detection and localization. In: IEEE International Workshop on Information Forensics and Security (WIFS) 2014, pp. 149–154. IEEE Computer Society, Washington (2014)
20. Dong, J., Wang, W.: CASIA tampered image detection evaluation (TIDE) database, v1.0 and v2.0 (2011). http://forensics.idealtest.org/
21. Ng, T.T., Hsu, J., Chang, S.F.: Columbia image splicing detection evaluation dataset (2009). http://www.ee.columbia.edu/ln/dvmm/downloads/AuthSplicedDataSet/AuthSplicedDataSet.htm