Learning Hierarchical Graph Representation for Image Manipulation Detection

Wenyan Pan1, Zhili Zhou1*, Miaogen Ling1,2*, Xin Geng3 and Q. M. Jonathan Wu4
1School of Computer and Software & Engineering Research Center of Digital Forensics Ministry of Education, Nanjing University of Information Science and Technology, Nanjing, China
2Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou, China
3School of Computer Science and Engineering, Southeast University, Nanjing, China
4Department of Electrical and Computer Engineering, University of Windsor, Windsor, Ontario, Canada
{Panwy, mgling}@nuist.edu.cn, zhou_zhili@163.com, xgeng@seu.edu.cn, jwu@uwindsor.ca

Abstract

The objective of image manipulation detection is to identify and locate the manipulated regions in the images. Recent approaches mostly adopt the sophisticated Convolutional Neural Networks (CNNs) to capture the tampering artifacts left in the images to locate the manipulated regions. However, these approaches ignore the feature correlations, i.e., feature inconsistencies, between manipulated regions and non-manipulated regions, leading to inferior detection performance. To address this issue, we propose a hierarchical Graph Convolutional Network (HGCN-Net), which consists of two parallel branches: the backbone network branch and the hierarchical graph representation learning (HGRL) branch for image manipulation detection. Specifically, the feature maps of a given image are extracted by the backbone network branch, and then the feature correlations within the feature maps are modeled as a set of fully-connected graphs for learning the hierarchical graph representation by the HGRL branch. The learned hierarchical graph representation can sufficiently capture the feature correlations across different scales, and thus it provides high discriminability for distinguishing manipulated and non-manipulated regions. Extensive experiments on four public datasets demonstrate that the proposed HGCN-Net not only provides promising detection accuracy, but also achieves strong robustness under a variety of common image attacks in the task of image manipulation detection, compared to the state-of-the-arts.

1 Introduction

Recently, determination of image authenticity has become a challenging problem. Due to the availability of powerful image-editing tools, the malicious users can manipulate digital images easily without leaving obvious manipulation traces. The existing image manipulation operations usually modify the image content by adopting several operations, i.e., copy-move, splicing, and removal. Thus, as an effective way to determine image authenticity, image manipulation detection has received more and more attention.

The basic idea of image manipulation detection is to capture the tampering artifacts to identify and localize the manipulated regions for image authenticity determination. The traditional manipulation detection approaches mainly extract handcrafted features to identify and localize manipulated regions, such as Color Filter Array (CFA) [Popescu and Farid, 2005], Discrete Cosine Transform (DCT) [Li et al., 2017], Scale-Invariant Feature Transform (SIFT) [Zhou et al., 2016]. Such approaches have achieved impressive performance. However, since the feature extraction algorithms are designed manually, these approaches tend to locate manipulated regions in only one or certain specific types of ma-
• The feature correlations within the selected feature maps are modeled as a set of fully-connected graphs. To the best of our knowledge, this is the first network that explores the graph structure for image manipulation detection.

• The HGCN-Net is proposed to learn the hierarchical graph representation from those fully-connected graphs. The learned representation can sufficiently capture the feature correlations across different scales, and thus it provides high discriminability for distinguishing manipulated and non-manipulated regions. In practice, the manipulated images transmitted on the Internet usually suffer from a variety of intentional or unintentional attacks such as JPEG compression and Gaussian noise addition. Since the feature correlations are relatively stable under those attacks, the learned representation also shows strong robustness and provides desirable results of manipulation detection under those attacks.

• Extensive experiments on four public manipulated image datasets demonstrate that the proposed HGCN-Net achieves promising detection performance and strong robustness under a variety of common image attacks, compared to the state-of-the-arts.

2 Related Works

2.1 Image Manipulation Detection

Handcrafted feature-based approaches. In early work, the image manipulation detection approaches usually extract handcrafted features, such as Discrete Cosine Transform (DCT) [Li et al., 2017], Scale-Invariant Feature Transform (SIFT) [Zhou et al., 2016], to capture the tampering traces left in the images for localizing the manipulated regions. However, since the handcrafted features can only capture the traces caused by one or several certain tampering operations, leading to limited generalization ability of these approaches.

CNN-based approaches. To improve the generalization ability of handcrafted feature-based approaches, some approaches [Salloum et al., 2018; Wu et al., 2018; Zhou et al., 2020] explore the sophisticated CNNs to automatically learn the tampering trace features for image manipulation detection. Some other approaches suppress the image content to facilitate the extraction of tampering trace features. [Zhou et al., 2018] proposed a two-stream Faster R-CNN, which pays more attention to the noise feature to locate the manipulated regions. [Rao and Ni, 2016] used the Steganalysis Rich Model (SRM) [Fridrich and Kodovsky, 2012] to initialize the kernel of the first convolution layer with steganalysis filter kernels, aiming to suppress the effects of image content for manipulation detection. [Bayar and Stamm, 2018] proposed the constrained convolutional layer to adaptively learn the tampering trace features. [Yang et al., 2020] proposed a two-stream network, which combines Convolutional Block Attention Module (CBAM) [Woo et al., 2018] and SRM to learn the tampering trace features for image manipulation detection. Although these approaches generally outperform the handcrafted feature-based approaches, they do not take into account the feature correlations between manipulated and non-manipulated regions. Thus, there is still a large room for performance improvement.

2.2 Graph Convolutional Network

The Graph Convolutional Networks (GCNs) [Kipf and Welling, 2016] and their variants have been applied in many computer vision tasks, such as image classification [Knyazev et al., 2019], object detection [Zhao et al., 2021], segmentation [Lu et al., 2020; Li and Gupta, 2018]. [Knyazev et al., 2019] built multi-graphs by segmenting each image as a set of superpixels and formulated image classification as a multigraph classification problem. [Hong et al., 2021] investigated the CNNs and miniGCN for hyperspectral image classification. In [Zhao et al., 2021], the graph feature pyramid...
network is proposed to capture the intrinsic image structures across different scales by mapping the superpixels of each image to the graph nodes. [Li and Gupta, 2018] formulated the semantic segmentation as the graph node classification task for semantic segmentation. Inspired by these GCN models, we attempt to learn the graph representation from images to describe the feature correlations between manipulated and non-manipulated regions for image manipulation detection.

3 The Proposed Approach

3.1 Pipeline of HGCN-Net

The pipeline of the proposed HGCN-Net is illustrated in Figure 2. The proposed HGCN-Net consists of two branches: the backbone network branch and the hierarchical graph representation learning (HGRL) branch. First, the backbone network branch extracts the feature maps for the graph construction in the HGRL branch; second, the HGRL branch is implemented by three following steps: constructing graphs from the feature maps, learning the hierarchical graph representation from the constructed graphs by hierarchical graph convolution, and transforming the learned representation to the original CNN space. Finally, the backbone network branch integrates the feature maps with the hierarchical graph representation learned by the HGRL branch to obtain the final detection results. The main contribution of HGCN-Net lies in the three steps of the HGRL branch. Therefore, in the following sections, we will elaborate each of the three steps, and introduce the loss function used in HGCN-Net.

3.2 Graph Construction

The graph construction is illustrated by the green box of Figure 3. After extracting the selected multi-scale feature maps from the different layers in the backbone network branch, we construct a set of fully-connected graphs to model the feature correlations by connecting all grids in each feature map. The multi-scale feature maps represent as \( \mathcal{P} = \{ P_1, P_2, P_3, P_4 \} \), where \( P_i \in \mathbb{R}^{B \times C_i \times W_i \times H_i}, i = 1, 2, 3, 4 \), \( B \), \( C \), \( W \), and \( H \) denotes the batch size, channel, width, and height of the feature map, respectively. Note that there is a large amount of grid in each \( P_i \), and thus the construction of those fully-connected graphs is time-consuming. To reduce time consumption, we first transform \( \mathcal{P} \) into a low-dimensional feature maps \( \mathcal{P}' = \{ P'_1, P'_2, P'_3, P'_4 \} \) by down-sampling process \( down_3 \). Formally, \( \mathcal{P}' \) is computed as:

\[
P' = down_3(\mathcal{P}), \quad P'_i \in \mathbb{R}^{B \times C_i \times W'_i \times H'_i}
\]

where \( W'_i = W_i / s_i, H'_i = H_i / s_i, s_i = \{ s_1, s_2, s_3, s_4 \} \) denotes the down-sampling factor. The down-sampling process is achieved by implementing a convolution operation with \( 3 \times 3 \) kernel and stride of 2, and batch normalization operation.

After the down-sampling process, the set of fully-connected graphs \( \mathcal{G} = \{ G_1, G_2, G_3, G_4 \} \) is constructed by connecting all grids in each \( P'_i \). The \( i \)-th fully-connected graph is represented as \( G_i = (V_i, E_i) \), where \( V_i \) denotes a set of \( N_i \) graph nodes, \( N_i = W'_i \times H'_i \), and \( E_i \) are set of edges represented as an adjacency matrix \( A_i \in \mathbb{R}^{N_i \times N_i} \), representing the spatial distances between every two graph nodes.

3.3 Hierarchical Graph Convolution

After the graph construction, the GCN is introduced to perform convolution operations on \( \mathcal{G} \) to learn the hierarchical graph representation. The red box of Figure 3 illustrates the hierarchical graph convolution. The convolution operation takes the \( \mathcal{G} \) and corresponding feature matrix \( P'_i \) as input. After implementing a 2-layer graph convolution operation on \( \mathcal{G} \) and \( P'_i \), we obtain the hierarchical graph representation \( \mathcal{F} = \{ F_1, F_2, F_3, F_4 \} \) by

\[
F_i = \sigma_s \left( \hat{A}_i \sigma_r \left( \hat{A}_i P'_i M^{(0)} M_i^{(1)} \right) \right)
\]

where \( F_i \in \mathbb{R}^{B \times C_i \times N_i^2} \), \( A_i \) denotes the adjacency matrix. \( M^{(0)} \) and \( M_i^{(1)} \) denote the weight matrix for GCN in the first and second layer. \( \sigma_s \) and \( \sigma_r \) denote the softmax and ReLU activation function, respectively. \( \hat{A}_i \) denotes the preprocessing step, represented by

\[
\hat{A}_i = D_i^{-1/2} A_i D_i^{-1/2}
\]

where \( \hat{A}_i = A_i + I \) denotes an adjacency matrix with self-loops, \( I \) denotes an identity matrix, \( D_i \) denotes the diagonal node degree matrix of \( A_i \). Then, the learned hierarchi-
In the training stage of the proposed HGCN-Net, we design the principal loss function $\mathcal{L}_S$ to supervise the detection results of the HGCN-Net. The designed loss function contains two parts. The loss $\mathcal{L}_C$ is utilized to supervise the detection results of the HGRL branch for optimizing the hierarchical graph representation for further fusion. While the loss $\mathcal{L}_B$ is utilized to supervise the detection results of the backbone network branch and the performance of the fusion operation. All the loss functions are Dice Loss. Furthermore, we use the parameter $\alpha$ to control the importance of the $\mathcal{L}_C$ and $\mathcal{L}_B$ during training. The principal loss function $\mathcal{L}_S$ is represented by

$$\mathcal{L}_S (X_m, g_t) = \alpha \mathcal{L}_C (R (X_m), g_t) + (1 - \alpha) \mathcal{L}_B (X_m, g_t) \quad (5)$$

where $X_m$ denote an input manipulated image, $R (X_m)$ denotes the detection results of the HGRL branch.

### 4 Experiments

#### 4.1 Datasets and Evaluation Metrics

To validate the effectiveness of the proposed HGCN-Net, we conduct extensive experiments on the four most widely used datasets: CASIA [Dong et al., 2013], COVER [Wen et al., 2016], NIST16 [Guan et al., 2019], and Columbia [Hsu and Chang, 2006]. The CASIA consists of two parts: CASIA 1.0 and CASIA 2.0, which contain 921 and 5124 manipulated images, respectively. We choose CASIA 2.0 for training and CASIA 1.0 for testing. The COVER dataset contains 100 manipulated images, in which we choose 75 images for training and 25 images for testing. The NIST16 contains 564 manipulated images, in which we choose 404 images for training and 160 for testing. The Columbia dataset, consisting of 180 manipulated images, is split in half to construct the train/test sets.

Similar to [Zhou et al., 2020; Rao and Ni, 2016], we use pixel-level $F_1$ score and Matthews Correlation Coefficient (MCC) for evaluating performance of the proposed HGCN-Net on the four benchmarks, we set the prediction threshold as 0.5 to get a binary prediction mask.

#### 4.2 Implementation Details

The HGCN-Net is implemented on the Pytorch with an NVIDIA RTX 3090 GPU. We apply the Adam optimizer with batch size 64 for CASIA and 8 for other datasets. The initial learning rate is 0.0001 with decaying 6% every two epochs. Training is terminated after 100 epochs. During the training, we use data augmentation techniques including flipping, rotation, padding, cropping, and adding Gaussian noise using open-source toolbox albumentation. We use the pretrained model on ImageNet to initialize the backbone network branch in HGCN-Net to improve the training stability.

During the training process, the input images are resized to $320 \times 320, 480 \times 480, 512 \times 512,$ and $512 \times 512$ for CAISA, COVER, NIST16, and Columbia, respectively. For the hierarchical graph projection, we set three different down-sampling factors $s$, e.g., $s_{11} = \{16, 8, 4, 2\}$, $s_{21} = \{8, 4, 2, 1\}$, and $s_{31} = \{4, 2, 1, 1\}$, note that $P_4$ does not need a down-sampling operation since the size of the feature map is small enough. For the loss function, For the loss function, we set the weight $\alpha$ as different values, i.e., $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ for comparison.
Table 1: Ablation study of HGCN-Net on CASIA with different settings.

| Approach                    | $F_1$ (%) | MCC (%) |
|-----------------------------|-----------|---------|
| R-101-FPN                  | 38.29     | 37.85   |
| R-101-FPN+GR(Z_1)-A        | 38.79     | 37.99   |
| R-101-FPN+GR(Z_2)-A        | 38.79     | 38.16   |
| R-101-FPN+GR(Z_3)-A        | 39.56     | 38.89   |
| R-101-FPN+GR(Z_4)-A        | 39.70     | 38.96   |
| R-101-FPN+HGR-A            | 39.71     | 39.30   |
| R-101-FPN+HGR-A(w/ $L_S$)  | 40.06     | 39.19   |
| R-101-FPN+HGR-C(w/ $L_S$)  | 40.81     | 39.39   |

Table 2: Ablation study of HGCN-Net on CASIA with different down-sampling factors.

| Approach                    | $F_1$ (%) | MCC (%) | Speed  |
|-----------------------------|-----------|---------|--------|
| HGCN-Net ($s_{11}$)         | 40.13     | 38.64   | 497.23 |
| HGCN-Net ($s_{21}$)         | 40.81     | 39.39   | 682.98 |
| HGCN-Net ($s_{31}$)         | 39.10     | 38.36   | 3613.52|

4.3 Experimental Results

Ablation Study of network architecture. To validate the impact of different network structures, we conduct ablation studies incrementally on CASIA. We test and compare the HGCN-Net with different settings: (1) R-101-FPN, in which ResNet101 network with the Feature Pyramid Network is used as the backbone network. (2) R-101-FPN+GR(Z_i)-A, in which R-101-FPN separately fuses with each graph representation $Z_1, Z_2, Z_3, \text{ and } Z_4$ by element-wise addition. (3) R-101-FPN+HGR-A(w/ $L_S$), in which R-101-FPN fuses with the hierarchical graph representation $Z$ by element-wise addition. (4) R-101-FPN+HGR-A(w/ $L_S$), in which R-101-FPN fuses with the hierarchical graph representation $Z$ by element-wise addition with the designed loss function $L_S$. (5) R-101-FPN+HGR-C(w/ $L_S$), in which R-101-FPN fuses with the hierarchical graph representation $Z$ by concatenation operation with the designed loss function $L_S$.

As shown in Table 1, R-101-FPN achieves 38.29% in $F_1$ score and 37.85% in MCC. R-101-FPN+GR(Z_i)-A generally outperforms R-101-FPN in $F_1$ score and MCC. Meanwhile, when we adopt R-101-FPN+HGR-A, the network improves the $F_1$ score and MCC performance over the R-101-FPN by 1.42% and 1.45%. Furthermore, when we adopt R-101-FPN+HGR-A(w/ $L_S$), we achieve higher $F_1$ score and MCC, i.e., 40.06% and 39.19%, when using the designed loss function $L_S$. Finally, it is clear that R-101-FPN+HGR-C(w/ $L_S$) outperforms the R-101-FPN by 2.52% in $F_1$ and 1.54% in MCC, and achieves highest accuracy, i.e., 40.81% in $F_1$ score and 39.39% in MCC, which indicate the concatenation operation further improve the performance. That might be because the concatenation fusion operation retains more graph representation, which is beneficial for detection accuracy. According to the above, we choose R-101-FPN+HGR-C(w/ $L_S$) as the default setting of HGCN-Net in the following experiments.

Ablation Study of different down-sampling factors. We also perform an ablation study of different down-sampling factors $s_i$. The different down-sampling factors $s_i$ lead to different sizes of the graphs. We also test the speed (inference time) of these approaches. Speed is measured by calculating the average inference time of the testing set on CASIA. From the results of Table 2, it can be clearly observed that $s_{21} = \{8, 4, 2, 1\}$ yields the best performance with acceptable speed. So, we choose the down-sampling factor $s_{21}$ as the default setting in all the following experiments.

Ablation Study of the weight of the loss function. In this part, we design an ablation study for $\alpha$, we test the different values of $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$. From the results of Table 3, it can be clearly observed that $\alpha = 0.5$ yields the best performance. Thus, in all following experiments, we choose the $\alpha = 0.5$ as the default setting.

Comparison with State-of-the-arts. We conduct experiments on the public datasets and compare the performance with prior approaches such as GSR-Net [Zhou et al., 2020], CR-CNN [Yang et al., 2020], EXIF-Consistency [Huh et al., 2018]. The pixel-level $F_1$ score and MCC are also used for performance evaluation. For the CR-CNN and GSR-Net, we use these networks to train on the training set and evaluate on test sets. The performance of these approaches is shown in Table 4. From Table 4, it is clear that the proposed HGCN-Net outperforms the CR-CNN and GSR-Net by a large margin on both CASIA and COVER. Moreover, our approach achieves the highest accuracy in Columbia, but slightly lower accuracy in the NIST16. Thus, the proposed HGCN-Net generally outperforms the other approaches. The result demonstrates that feature correlations is beneficial for improving the performance in image manipulation detection tasks.
Table 4: $F_1$ score and MCC comparison of different approaches on four public datasets. "−" denotes that the results are not available in the references.

| Approach       | CASIA | NIST16 | COVER | Columbia |
|----------------|-------|--------|-------|----------|
|                | $F_1$ (%) | MCC (%) | $F_1$ (%) | MCC (%) | $F_1$ (%) | MCC (%) | $F_1$ (%) | MCC (%) |
| ResNet101+FPN  | 38.29 | 37.85  | 66.26 | 65.86    | 36.85    | 30.65    | 91.80    | 89.41    |
| EXIF-Consistency | -    | -      | -    | -        | -        | -        | 88.00    | 80.00    |
| CR-CNN         | 24.68 | 26.12  | 89.87 | 89.59    | 26.64    | 19.90    | 91.18    | 88.07    |
| GSR-Net        | 35.80 | 35.97  | -    | -        | 25.27    | 23.34    | -        | -        |
| HGCN-Net       | 40.81 | 39.39  | 71.02 | 70.79    | 39.32    | 36.94    | 91.81    | 89.76    |

Robustness. In this section, we evaluate the robustness of the proposed HGCN-Net under a variety of common image attacks, including JPEG compression, resizing, Gaussian blur, blur, and rotation. We compare HGCN-Net with GSR-Net and CR-CNN using their publicly available codes. The results on the CASIA and COVER are shown in Figure 4. It can be clearly observed that the proposed HGCN-Net achieves high robustness with regard to all attacks on the CASIA and COVER datasets. This is because the feature correlations between manipulated and non-manipulated regions learned by HGCN-Net are relatively stable under those common attacks.

Qualitative Results. We first provide the qualitative result in Figure 5 to illustrate the performance of the proposed HGCN-Net with different settings listed in Table 1. The results show HGCN-Net with setting R-101-FPN+HGR-C(w/ $\mathcal{L}_S$) achieves the best performance. We also provide qualitative results in Figure 6 to visually compare HGCN-Net with CR-CNN and GSR-Net. As shown in Figure 6, the proposed HGCN-Net always produces smoother boundary and more accurate pixel-level localization results for detecting manipulated regions.

5 Conclusion

In this paper, we propose the HGCN-Net for image manipulation detection, which performs promising detection accuracy and strong robustness to a variety of common image attacks. This is achieved by sufficiently exploring the feature correlations within the manipulated images. Such feature correlations are captured by the proposed hierarchical graph representation, which is learned from the set of the fully-connected graphs constructed by connecting all grids in the feature maps of a given manipulated image. Experimental results demonstrate the effectiveness and strong robustness of our network on image manipulation detection tasks.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China under Grant 61972205, 62122032, 62032020, in part by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund, and in part by the Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CI-CAEET) fund, China.

References

[Bappy et al., 2017] Jawadul H Bappy, Amit K Roy-Chowdhury, Jason Bunk, Lakshmanan Nataraj, and...
[Fridrich and Kodovsky, 2012] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, 2012.

[Bayar and Stamm, 2018] Belhassen Bayar and Matthew C Stamm. Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection. *IEEE Transactions on Information Forensics and Security*, 13(11):2691–2706, 2018.

[Dong et al., 2013] Jing Dong, Wei Wang, and Tieniu Tan. ChinaSIP image tampering detection evaluation database. In *ChinaSIP*, pages 422–426. IEEE, 2013.

[Fridrich and Kodovsky, 2012] Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, 2012.

[Guan et al., 2019] Haiying Guan, Mark Kozak, Eric Robertson, Yooyoung Lee, Amy N Yates, Andrew Delgado, Daniel Zhou, Timothee Kheyrykhab, Jeff Smith, and Jonathan Fiscus. Mfc datasets: Large-scale benchmark datasets for media forensic challenge evaluation. In *WACV*, pages 63–72. IEEE, 2019.

[Hong et al., 2021] Danfeng Hong, Lianru Gao, Jing Yao, Bing Zhang, Antonio Plaza, and Jocelyn Chanussot. Graph convolutional networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 59(7):5966–5978, 2021.

[Hsu and Chang, 2006] J Hsu and SF Chang. Columbia uncompressed image splicing detection evaluation dataset. *Columbia DVMM Research Lab*, 2006.

[Huh et al., 2018] Minyoung Huh, Andrew Liu, Andrew Owens, and Alexei A Efros. Fighting fake news: Image splice detection via learned self-consistency. In *ECCV*, pages 101–117, 2018.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Knyazev et al., 2019] Boris Knyazev, Xiao Lin, Mohamed R Amer, and Graham W Taylor. Image classification with hierarchical multigraph networks. *arXiv preprint arXiv:1907.09000*, 2019.

[Li and Gupta, 2018] Yin Li and Abhinav Gupta. Beyond grids: Learning graph representations for visual recognition. *Advances in Neural Information Processing Systems*, 31:9225–9235, 2018.

[Li et al., 2017] Ce Li, Qiang Ma, Limei Xiao, Ming Li, and Aihua Zhang. Image splicing detection based on markov features in qdct domain. *Neurocomputing*, 228:29–36, 2017.

[Lu et al., 2020] Yi Lu, Yaran Chen, Dongbin Zhao, Bao Liu, Zhichao Lai, and Jianxin Chen. Cnn-g: convolutional neural network combined with graph for image segmentation with theoretical analysis. *IEEE Transactions on Cognitive and Developmental Systems*, 2020.

[Popescu and Farid, 2005] Alin C Popescu and Hany Farid. Exposing digital forgeries in color filter array interpolated images. *IEEE Transactions on Signal Processing*, 53(10):3948–3959, 2005.

[Rao and Ni, 2016] Yuan Rao and Jiangqun Ni. A deep learning approach to detection of splicing and copy-move forgeries in images. In *WIFS*, pages 1–6. IEEE, 2016.

[Salloum et al., 2018] Ronald Salloum, Yuzhuo Ren, and C-C J Y Kuo. Image splicing localization using a multi-task fully convolutional network (mfcn). *Journal of Visual Communication and Image Representation*, 51:201–209, 2018.

[Wen et al., 2016] Bihan Wen, Ye Zhu, Ramanathan Subramanian, Tian-Tsong Ng, Xuanjing Shen, and Stefan Winkler. Coverage—a novel database for copy-move forgery detection. In *ICIP*, pages 161–165. IEEE, 2016.

[Woo et al., 2018] Sanghyun Woo, Jongchan Park, Joonyoung Lee, and In So Kweon. Cbam: Convolutional block attention module. In *ECCV*, pages 3–19, 2018.

[Wu et al., 2018] Yue Wu, Wael Abd-Almageed, and Prem Natarajan. Busternet: Detecting copy-move image forgery with source/target localization. In *ECCV*, pages 168–184, 2018.

[Wu et al., 2019] Yue Wu, Wael Abd-Almageed, and Premkumar Natarajan. Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features. In *CVPR*, pages 9543–9552, 2019.

[Yang et al., 2020] Chao Yang, Huizhou Li, Fangting Lin, Bin Jiang, and Hao Zhao. Constrained r-cnn: A general image manipulation detection model. In *ICME*, pages 1–6. IEEE, 2020.

[Zhao et al., 2021] Gangming Zhao, Weifeng Ge, and Yizhou Yu. Graphfpn: Graph feature pyramid network for object detection. In *ICCV*, pages 2763–2772, 2021.

[Zhou et al., 2016] Zhili Zhou, Yunlong Wang, QM Jonathan Wu, Ching-Nung Yang, and Xingming Sun. Effective and efficient global context verification for image copy detection. *IEEE Transactions on Information Forensics and Security*, 12(1):48–63, 2016.

[Zhou et al., 2018] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Learning rich features for image manipulation detection. In *CVPR*, pages 1053–1061, 2018.

[Zhou et al., 2020] Peng Zhou, Bor-Chun Chen, Xintong Han, Mahyar Najibi, Abhinav Shrivastava, Ser-Nam Lim, and Larry Davis. Generate, segment, and refine: Towards generic manipulation segmentation. In *AAAI*, volume 34, pages 13058–13065, 2020.