Toward Effective Image Manipulation Detection With Proposal Contrastive Learning

Yuyuan Zeng, Bowen Zhao, Shanzhao Qiu, Tao Dai, and Shu-Tao Xia

Abstract—Deep models have been widely and successfully used in image manipulation detection, which aims to classify tampered images and localize tampered regions. Most existing methods mainly focus on extracting global features from tampered images, while neglecting the relationships of local features between tampered and authentic regions within a single tampered image. To exploit such spatial relationships, we propose Proposal Contrastive Learning (PCL) for effective image manipulation detection. Our PCL consists of a two-stream architecture by extracting two types of global features from RGB and noise views respectively. To further improve the discriminative power, we exploit the relationships of local features through a proxy proposal contrastive learning task by attracting/repelling proposal-based positive/negative sample pairs. Moreover, we show that our PCL can be easily adapted to unlabeled data in practice, which can reduce manual labeling costs and promote more generalizable features. Extensive experiments among several standard datasets demonstrate that our PCL can be a general module to obtain consistent improvement. The code is available at https://github.com/Sandy-Zeng/PCL.

Index Terms—Image manipulation detection, contrastive learning, proposal-based, local features.

I. INTRODUCTION

DIGITAL images can be easily manipulated owing to easy-to-use image processing software. Malicious image manipulation, such as in fake news, insurance fraud and malicious rumors can have a great negative impact on society. In reality, splicing (paste objects from one image to another), copy-move (copy and move some elements from one region to another within one image) and removal (erase some regions from an image followed by inpainting) are the most commonly used image manipulation techniques. Some examples of these image manipulations are given in Fig. 1. It is difficult to identify the tampered regions directly due to the imperceptible tampering artifacts. Faced with a considerable number of tampered images and exquisite image manipulation techniques, it is crucial to develop reliable image manipulation detection techniques.

Early methods mainly focus on using handcrafted or predetermined features to distinguish authentic regions and tampered regions, such as doubly compressed JPEG [1], color filter array [2], local noise features [3], [4] and etc. However, the handcrafted features cannot generalize well on unseen manipulation types. In recent years, deep learning (DL)-based methods have achieved impressive performance on image manipulation detection based on various network architectures, including convolutional neural network [5], autoencoder [6], long short-term memory [7], Faster R-CNN [8] and etc. However, most existing DL-based methods mainly focus on designing sophisticated networks to extract global features from tampered images, while they neglect the intrinsic relationships of local features between tampered and authentic images.

| Authentic Images | Tampered Images | Our Results |
|------------------|----------------|-------------|
| Splicing         |                |             |
| Copy-Move        |                |             |
| Removal          |                |             |

![Fig. 1. Tampered images with different manipulation techniques. Red boxes indicate the ground-truth bounding boxes and the blue ones are our detection results. Best viewed in color.](https://example.com/image.png)

Fig. 1. Tampered images with different manipulation techniques. Red boxes indicate the ground-truth bounding boxes and the blue ones are our detection results. Best viewed in color.
regions within a tampered image, thus hindering the discriminative ability of these networks. For example, we commonly figure out the tampered regions by comparing the differences between normal (authentic) and abnormal (tampered) areas within an image. We believe that such spatial correlation is helpful to improve the discriminative ability of the feature representations for manipulation detection.

To exploit such rich spatial relationships of local features, in this paper, we propose an effective Proposal Contrastive Learning (PCL) for image manipulation detection, inspired by the success of contrastive learning in self-supervised representation learning [9], [10]. As shown in Fig. 2, our PCL module is built on a two-stream architecture to classify and locate tampered regions. As shown by previous methods [8], [11], the noise view is beneficial to reveal the semantic-agnostic features, we adopt the two-stream architectures and construct the noise views either by SRM filters [8] or constrained convolutional layers [11]. Different from the conventional contrastive learning in image classification, we explore contrastive learning at proposal-level within the object detection framework. Moreover, the two-stream features for image manipulation detection can naturally provide the positive and negative data pairs instead of constructing them with extra data augmentation as in self-supervised image classification [9], [10]. The relationships between tampered and authentic regions are exploited by attracting the features of positive pairs and repelling the features of negative pairs. Moreover, our PCL can be easily applied to unlabeled data to further improve the performance, which facilitates the potential applications in practice, as numerous available tampered images without annotations are disseminated in the wild, especially on various social media platforms. The experimental results demonstrate that PCL is both effective on unlabeled data and labeled data. Extensive experiments among several standard datasets demonstrate that PCL can achieve consistent improvements and lead to more discriminative feature representations, showing great potential.

The main contributions of this work are summarized as follows:

- We propose a simple yet effective proposal contrastive learning module to leverage the relationships of local features between tampered and authentic regions for discriminative representation learning.
- Our method can be easily applied to unlabeled data to further improve manipulation detection performance. To the best of our knowledge, no efforts have been devoted to incorporating the massive unlabeled data for image manipulation detection.
- Extensive experiments on several standard datasets demonstrate the effectiveness of our proposed method on both labeled and unlabeled datasets.

II. RELATED WORK

A. Image Manipulation Detection

Traditional methods for image manipulation detection mainly rely on handcrafted or predetermined features. Early work [1] employed the statistics of DCT coefficients of doubly compressed JPEG images to distinguish the authentic and tampered regions. Color filter array was also utilized to detect the inconsistency of the tampered regions in an image [2]. Moreover, the local noise features introduced by the sensors and post-processing [3], [4] and the inconsistencies of the illuminant color or lighting [12], [13] are also clues for image splicing detection. Usually, these handcrafted features are defined for a specific image manipulation type, therefore it is difficult for them to achieve good performance in practice.

Recently, extensive deep learning-based architectures are adopted to extract the tampered characteristics adaptively, leading to better generalization ability in face of different types of manipulation. Instead of relying on manual feature extraction, early proposed methods utilized convolutional neural network to determine whether an image is tampered. Besides, long short-term memory (LSTM) based network was also employed to classify and locate the tampered regions [7]. The two-stream network utilized an RGB stream and a noise stream to learn rich features for image manipulation detection [8], which relies on the modern object detection models. In other work, deep semantic segmentation models, like fully convolutional networks (FCN) [14], hybrid models [15], [16] were also employed for image manipulation detection. In CR-CNN [11], they proposed a coarse-to-fine architecture based on Mask R-CNN, which also incorporates the constrained convolutional layer and attention mechanism. SNIS [17] propose a signal noise separation module to separate tampered region for post-processing forgery detection. Instead of designing more sophisticated network structures, in this work, we focus on (i) how to obtain more discriminative feature representations, (ii) how to make full use of unlabeled data to alleviate the data-scarce problem.

Moreover, there are other studies utilizing the patch consistency within the images to identify the tampered regions, such as [18], [19], and [20]. They rely on different camera attributes to identify tampered regions, such as EXIF [18], Noiseprint [19] and other adaptive features learned from data directly [20]. Thus these methods require data with meta-information about camera, which is often not available. While we use the manipulated images with only pixel information. Previous work explicitly learns the patch consistency by dividing images into multiple overlapped patches and performing patch pairings to identify differences. While our method implicitly models patch consistency through contrastive learning at proposal level. Therefore, our method is different from previous work in terms of the data condition and the utilization of patch consistency.

B. Contrastive Learning

Recently, contrastive learning has made important progress in self-supervised learning [9], [10], [21], which aims to learn effective representations without human supervision, i.e., from unlabeled data. In addition, the contrastive representation has been shown to play a beneficial role in other tasks, like semi-supervised learning [22], supervised image recognition [23] and unsupervised video representation learning [24]. Instead of defining loss functions to measure the difference between predictions and fixed targets, contrastive losses measure
the similarities of data pairs in representation space. The contrastive losses push apart dissimilar sample pairs and pull together similar sample pairs. How choose positive and negative data pairs is extremely important in contrastive learning. In self-supervised image recognition, the most common practice is to treat differrent data augmentations of the same sample as positive data pairs, and the augmentations from other images are regarded as negative data pairs. Others like different modalities [25], [26], different feature representations [21] can also provide positive data pairs.

In this work, we explore contrastive learning at proposal-level for image manipulation detection. Rellying on two-stream inputs (RGB and noise), the positive and negative sample pairs are constructed. Our proposal contrastive learning is employed to enhance the feature representations under both the supervised paradigm and the semi-supervised paradigm. The concurrent work [27], [28] also utilizes the tool “contrastive learning” to enhance image manipulation detection. However, there are many differences between our work with them: (i) The concurrent work is based on the semantic segmentation framework, which must require per-pixel annotations. While our model is based on the object detection framework and can be trained with the bounding-box annotations, which are much easier and cheaper than the per-pixel annotations. (ii) Also for the above reason, our work mainly outputs the predicted bounding boxes and has marginal advantages in terms of pixel-level evaluation metrics. However, in many real applications, there is actually no need to discover every tampered pixel. Instead, the complete and accurate bounding boxes are enough. Thus, the detection-framework-based methods are also valuable, which can meet the need and do not require costly annotations. (iii) Our contrastive learning module allows the unlabeled data to participate in training and brings additional gains, which is important and urgently required in the real world. However, the concurrent work does not offer this capability (their contrastive learning mechanism must rely on labeled information).

III. METHODOLOGY

A. Basic Detector

We first introduce the widely used basic detector [8] for manipulation detection.

1) Overall Framework: As shown in Fig. 2(a), the basic model consists of two streams: an RGB stream and a noise stream. The RGB stream operates on the original RGB image, and the noise stream takes the noise feature map processed by SRM filters [30] or constrained convolutional layers [11] as input. As indicated by [8], the noise stream can pay more attention to the tampered characteristics rather than the semantic image content. Faster R-CNN is adopted to the design of the two-stream network architecture. The RGB stream and noise stream have independent backbone networks (e.g., ResNet-101). These two streams share one Region Proposal Network (RPN), which is elaborated below. Then, the proposal features extracted by the two streams are fed into the Region-based CNN (RCNN), which contains a classification head and a localization head, see details below.

2) Region Proposal Network (RPN): The RPN only relies on the RGB stream to learn and generate region proposals. The noise stream directly borrows the RPN module from the RGB stream. Formally, the training loss for the RPN consists of a classification loss $L_{cls}^{rp nm}$ (e.g., the cross-entropy loss) and a localization loss $L_{loc}^{rp m}$ (e.g., the smooth-L1 loss) as follows,

$$L_{rpn} = L_{cls}^{rp nm}(\mathbf{p}_k, [p_k^r, p_k^n]) + \lambda_1 L_{loc}^{rp m}(\mathbb{I}_k, \mathbb{I}_k^r),$$

where $\mathbf{p}_k$ and $\mathbf{p}_k^r$ denote the probability of anchor $k$ being a potential tampered region, the corresponding true label. $\mathbb{I}_k$ and $\mathbb{I}_k^r$ are the predicted 4-dimensional bounding box coordinates for anchor $k$ and the corresponding ground-truth coordinates respectively. $\lambda_1$ is the hyper-parameter to balance the two losses.

3) Region-based CNN (RCNN): After RPN, features are cropped from the RGB and noise feature maps for each proposal. Next, the features are resized to a fixed size through an RoI Pooling layer and fed into RCNN, which is responsible for giving the class probabilities for each proposal and refining the corresponding boxes. The RCNN is composed of a classification head and a localization head. In the classification head, the RoI features $\mathbf{h}'$ and $\mathbf{h}''$ generated from RGB stream and noise stream are combined via bilinear pooling ($\text{Bi Pool}$), which can preserve the spatial information and fuse the features from two streams naturally. The fused features for classification are given by

$$\mathbf{h}' = \text{Bi Pool} (\mathbf{h}', \mathbf{h}'').$$

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Considering the RGB information is more effective for localization, only the features of RGB stream are provided to the localization head, as shown in Fig. 2(a). Overall, the training loss for RCNN is formulated as

\[ L_{rcnn} = L_{rcnn}^{cls}(\{p_i\}, \{p_i^*\}) + \lambda_2 L_{rcnn}^{loc}(l_i, \{l_i^*\}), \]  

where \( L_{rcnn}^{cls} \) and \( L_{rcnn}^{loc} \) are the classification and localization loss, \( p_i \) denotes the probability of proposal \( i \) being a potential tampered region and \( p_i^* \) is the ground truth label. \( l_i \) and \( l_i^* \) are the predicted 4-dimensional bounding box coordinates for proposal \( i \) and the corresponding ground-truth coordinates respectively. \( \lambda_2 \) is the hyper-parameter to balance the two losses. \( L_{rcnn}^{cls} \) is related to the fused features \( h^f \), and \( L_{rcnn}^{loc} \) is associated to the RGB features \( h^r \) alone.

\[ L_{rcnn}^{cls}(\{p_i\}, \{p_i^*\}) = \sum_{i} \log \frac{1}{\sum_j \exp(s(p_i^*, p_j^*))}, \]

\[ L_{rcnn}^{loc}(l_i, \{l_i^*\}) = \sum_{i} \sum_{j} \log \frac{\exp(s(l_i^*, l_j^*), \tau))}{\sum_j \exp(s(l_i^*, l_j^*), \tau))}, \]

where \( s(\cdot, \cdot, \tau) \) stands for the similarity between two inputs \( \cdot \) and \( \cdot \), \( \tau \) is the hyper-parameter of temperature, \( N \) is the number of positive pairs. Here, the similarity of features is measured by dot product. The PCL loss maximizes the agreement between the RGB and noise features of tampered regions, which helps reveal the view-invariant factors here; on the other hand, it can repel the features of authentic regions.

Equ. (6) is composed of two terms. The first term (denoted as \( L_{pcl-rgb} \)) treats the RGB features from tampered regions \( \{t_i^f\} \) and the noise features from authentic regions \( \{a_i^f\} \) as negative pairs, dubbed as PCL-RGB. The second term (denoted as \( L_{pcl-noise} \)) regards the noise features from tampered regions \( \{t_i^f\} \) and the RGB features from authentic regions \( \{a_i^f\} \) as negative pairs oppositely, dubbed as PCL-Noise. In Sec. IV-B, we conduct an ablation study on different loss functions and empirically demonstrate that we can obtain better performance when using the PCL loss. Overall, the total loss function for image manipulation detection with labeled data is given by

\[ L^l = L_{rpm} + L_{rcnn} + \beta L_{pcl}, \]

where \( \beta \) balances the proxy contrastive learning task and the original tasks.

\[ L_{rpm} = \sum_{i} \sum_{j} \log \frac{\exp(s(t_i^f, t_j^f), \tau))}{\sum_j \exp(s(t_i^f, a_j^f), \tau))}, \]

\[ L_{pcl-rgb} = \sum_{i} \sum_{j} \log \frac{\exp(s(t_i^f, a_j^f), \tau))}{\sum_j \exp(s(t_i^f, t_j^f), \tau))}, \]

\[ L_{pcl-noise} = \sum_{i} \sum_{j} \log \frac{\exp(s(t_i^f, a_j^f), \tau))}{\sum_j \exp(s(t_i^f, t_j^f), \tau))}. \]
which can incorporate a lot of “free” data into training, leading to enriched feature representations. For unlabeled data, the ground-truth boxes are not available to identify the positive and negative pairs like for labeled data, thus we turn to rely on the prediction results to construct sample pairs. There are two prediction scores related to a proposal: the score in RPN and that in RCNN. Here, we examine two different sampling strategies: (i) one depends on the predicted classification scores of RPN; (ii) the other depends on the predicted classification scores of RCNN. We hope that the prediction scores can reflect whether the proposals are related to tampered regions, which can help to select more suitable sample pairs. In Fig. 4, we draw the relationship of IoU (between the proposals and ground-truth boxes) and the prediction scores in RPN (Fig. 4(a)) and RCNN (Fig. 4(b)) respectively. It can be observed in Fig. 4(a) that the proposals distribute uniformly with RPN scores due to the RPN usually guarantees recall rather than precision. While RCNN scores are more correlated with IoU as depicted in Fig. 4(b). We also calculate the Pearson Correlation Coefficient (PCC) [33] to indicate the linear correlation of IoU and prediction scores. The PCC of IoU and RPN scores is 0.57, while for IoU and RCNN scores is 0.83, which indicates that RCNN scores are more reliable. In Sec. IV-B, we also conduct an ablation study to show the detection performance with RPN and RCNN scores as guidance for unlabeled data.

Based on the above observations, we apply the RCNN scores as guidance to select positive and negative pairs. The proposal \( i \) is considered as “tampered” when the RCNN score is larger than the threshold \( \delta \), otherwise labeled as “authentic”. Now, like in Sec. III-C, we can divide the projected feature representations \( \{ z'_i \}, \{ z''_i \} \) of unlabeled data into \( \{ t'_i \}, \{ t''_i \} \) and \( \{ a'_i \}, \{ a''_i \} \). Then, the same strategy can be applied to construct positive and negative pairs. Finally, we can compute the contrastive loss \( L_{pcl}^u \) for unlabeled data by Eq. (6).

Finally, the labeled data and unlabeled data are utilized together and contributed to different loss functions, i.e., \( L^l \) and \( L_{pcl}^u \) respectively. Overall, the total loss function is given by

\[
L^u = L^l + \omega(m) \cdot L_{pcl}^u,
\]

where \( L^l \) is computed as Eq. (7) on labeled data, \( L_{pcl}^u \) is computed on unlabeled data weighted by a weight function \( \omega(m) \) which increases along as the training schedule. Following [34], the weight function is defined as

\[
\omega(m) = \exp\left(\frac{-m}{\tau} \cdot \left[ 1 - \left( \frac{m}{M} \right)^2 \right] \right),
\]

where \( m \) is the number of current iteration, and \( M \) is the number of warm up iterations. Since the prediction results of RCNN are not reliable at the beginning of network training, a smaller loss weight is adopted for unlabeled data at first.

IV. EXPERIMENTS

We conduct experiments on several standard image manipulation datasets and compare our method with other counterparts to demonstrate the effectiveness of proposal contrastive learning. Moreover, as our method can be applied to unlabeled data, we further incorporate unlabeled data into training to improve the performance.

A. Experimental Settings

1) Implementation Details: The proposed model is trained in an end-to-end manner. During training, image flipping and Gaussian noise are used for data augmentation. The input images are resized such that their shorter side equals 600 pixels. For anchors, we use four scales with box areas of \( 8^2, 16^2, 32^2 \) and 64\(^2 \) pixels, and three aspect ratios of 1:1, 1:2, and 2:1. The anchor scales are smaller than that in original Faster R-CNN [35] because we focus on regions that are likely to be manipulated instead of meaningful objects in the original Faster R-CNN. The batch size of RPN proposals is set to 32 for training and set to 100 for testing. Following the default and common setting [35], we set IoU threshold to 0.7 for potential positive regions and 0.3 for negative regions for anchors in RPN network. We use the minimum boxes containing ground-truth masks as ground-truth bounding boxes for our model training. The hyper-parameter \( \lambda_1 \) in Eq. (1) is set to 10, and \( \lambda_2 \) in Eq. (3) is set to 1. We set IoU threshold \( \epsilon \) for labeled data and predicted scores threshold \( \delta \) for unlabeled data to 0.5 to distinguish tampered regions from authentic regions. Temperature \( \tau \) and \( \beta \) in Eq. (6) are set to 0.1 and 0.05 in default setting. We pre-train our model with COCO synthetic dataset [8] for 110K steps. The warm up iterations \( M \) for unlabeled data is set to 60K steps. The learning rate is initially set to \( 1e^{-3} \) and reduced to \( 1e^{-4} \) after 40K steps. All the experiments are conducted in a single GTX 1080Ti.
2) Evaluation Metrics: We use AP$_{50}$ and AP$_{50-95}$, the common metrics [36], [37], to compare our PCL with other detection-based methods. AP$_{50}$ corresponds to the average precision at IoU threshold of 0.5, and AP$_{50-95}$ means the average AP at IoU thresholds from 0.5 to 0.95 with a step size of 0.05. To evaluate the image-level manipulation detection (recognition) performance, we use image-level Area Under Curve (AUC) score and F1 score. For the pixel-level localization performance, we use pixel-level F1 score and AUC for comparison with those segmentation-framework-based methods following previous work [14].

3) Evaluation Datasets: We use the same COCO synthetic dataset in [8] to pre-train and evaluate our models. The proposed methods are compared with others on four public image manipulation benchmarks: NIST16 [38], CASIA [39], [40], COVERAGE [41], Columbia [42] and IMD20 [43]. The training and testing split protocol of each benchmark follows [8]. The details of the evaluation datasets are presented as below:

- COCO synthetic dataset is created from COCO [36] by randomly selecting objects from one image with the segmentation mask and splicing to other images which is used for model pre-training. The dataset contains 11K manipulated images with the ground-truth tampered masks.
- NIST16 is a challenging dataset of Nimble Challenge Evaluation 2016. It contains three tampering types. Masks are provided to indicate where manipulation takes place. The dataset is split into two groups: 404 images for fine-tuning and 160 images for evaluation.
- CASIA contains splice and copy-move tampered images. By carefully selecting the tampered areas and using post-processing methods like blurring, the tampered images are as much as realistic to human eyes. We obtain the ground-truth masks by subtracting tampered images from their original images. We use CASIA 2.0 with 5,123 images for fine-tuning and CASIA 1.0 with 921 images for evaluation.
- COVERAGE is a large copy-move dataset. A tampered image is created by copying a small object from a duplicated region and pasting it onto the same image. Ground-truth masks are provided. We use 75 images for fine-tuning and 25 images for evaluation.
- Columbia is a small dataset containing 180 splicing images. Ground-truth masks are provided. We directly use the pre-trained model to evaluate Columbia without fine-tuning as the same in [8].
- IMD20 consists of 2,010 real-life manipulated images collected from Internet, it contains all three types of manipulation.

B. Ablation Study

In this section, we analyze different components of our proposed method including loss functions, different noise views, effects of hyper-parameter $\beta$ and proposal sampling strategies for unlabeled data. Experiments in this section are conducted on COCO synthetic dataset unless otherwise specified.

### Table I

| Loss Function | AP$_{50}$ | AP$_{50-95}$ |
|---------------|----------|-------------|
| Baseline      | 74.1     | 50.3        |
| PCL-RGB       | 76.2     | 51.5        |
| PCL-Noise     | 71.6     | 48.9        |
| PCL-FCL       | 76.3     | 52.6        |
| PCL(Ours)     | 78.0     | 53.2        |

1) Different Loss Functions: Firstly, we test different loss functions e.g. PCL-RGB, PCL-Noise, PCL-FCL and PCL, which reveals different strategies for constructing positive and negative pairs as discussed in Sec. III-C. PCL-RGB and PCL-Noise indicate that we only apply the $\mathcal{L}_{pcl-rgb}$ or $\mathcal{L}_{pcl-noise}$ in Eqn. (6) respectively. PCL-FCL means the utilization of vanilla fully contrastive strategy to construct the positive and negative pairs as illustrated in Fig. 3. The baseline method is RGB-N [8] which utilizes SRM filters to construct the noise view. From Table I, we can observe that model trained with PCL can improve AP$_{50}$ of baseline from 74.1% to 78.0% and enhance AP$_{50-95}$ from 50.3% to 53.2%, which indicates that contrastive learning leads to better feature representations for image manipulation detection. The comparison of PCL-RGB, PCL-Noise and PCL indicates that bilateral contrastive learning is necessary for better performance. In the case of PCL-FCL, AP$_{50}$ is 76.3%, outperforming the baseline by 2.1%. However, it is still inferior to PCL (76.3% v.s. 78.0%), which demonstrates that selecting suitable positive and negative sample pairs plays an important role in contrastive learning.

2) Different Noise Views: Most image manipulation detection methods aim to exploit the different noise distribution between authentic and tampered parts to detect tampered images [8], [11]. The noise views of input images are generated by the pre-defined filters such as SRM filters in RGB-N [8] or the trainable counterparts such as the constrained convolutional layers denoted as Constrained Conv in CR-CNN [11]. In our method, we adopt the two-stream architecture, for the noise stream, we evaluate the two choices of noise views in this section. The experimental results are summarized in Table II. We adopt RGB-N to indicate using SRM filters for noise view construction, and RGB-C to represent the one with Constrained Conv as the input layer. The experimental results indicate that the Constrained Conv is better than SRM filters for noise feature extraction. For both two choices of noise views, when combined with PCL, the performance can be improved by 1%-3%, which shows that PCL can adaptively extract the valuable features from the two different views. Moreover, we compare RGB-C (a two-stream architecture) with CR-CNN (a single-stream architecture with single noise feature extraction). The CR-CNN results are reproduced with their open-source code. It can be observed that RGB-C outperforms CR-CNN in AP$_{50-95}$ on COCO synthetic dataset (53.4% v.s. 52.9%). We can continuously
TABLE II

EXPERIMENTS ON DIFFERENT NOISE VIEWS WITH PCL ON COCO SYNTHETIC DATASET. AP50 (%) AND AP50−95(%) ARE REPORTED

| Methods        | Noise View   | AP50 | AP50−95 |
|----------------|--------------|------|---------|
| RGB-N [8]      | SRM filter   | 74.1 | 50.3    |
| +PCL (Ours)    |              | 78.0 | 53.2    |
| CR-CNN [11]    | Constrained Conv | 78.3 | 52.9    |
| RGB-C          | Constrained Conv | 78.1 | 53.4    |
| +PCL (Ours)    |              | 79.4 | 55.1    |

Fig. 5. Performance on COCO by varying values of $\beta$. It can be observed that large PCL weight would hurt the detection performance.

improve the performance by combining RGB-C with PCL, e.g. AP50 is 79.4% and AP50−95 is 55.1%, which shows that two-stream architectures can provide more diverse feature representations than the single-stream ones.

3) Effect of $\beta$: In Equ. (7), we introduce a hyper-parameter $\beta$ to balance the PCL loss with the original RPN and RCNN loss. Here, we analyze the influence of the value of $\beta$. We vary the value of $\beta$ to [0.001, 0.005, 0.01, 0.05, 0.1], and the results in term of AP50−95 are reported in Fig. 5. It can be observed that the performance begins to drop when the weight of PCL loss is larger than 0.01 for RGB-N and 0.05 for RGB-C, which is mainly due to that too large weight will cause the model to be dominated by the auxiliary task and deviate from the main detection task. Thus, in default setting, the value of $\beta$ is set to 0.01 for RGB-N and 0.05 for RGB-C.

4) Effectiveness of PCL for Unlabeled Data: Here, we further conduct experiments to evaluate the effectiveness of PCL on unlabeled data. We randomly split the training data in COCO synthetic dataset to labeled data $D_l$ (50%) and unlabeled data $D_u$ (50%). In Table III, Labeled PCL indicates applying PCL on the labeled data, Unlabeled PCL with RPN Score and Unlabeled PCL with RCNN Score mean applying PCL on unlabeled data with two different proposal sampling strategies as discussed in Sec. III-D. The baseline here is RGB-N [8] trained with labeled data $D_l$ which achieves 70.0% of AP50 and 44.6% of AP50−95. When PCL is applied on labeled data, we can obtain 71.0% of AP50 and 45.4% of AP50−95. When incorporating with unlabeled data $D_u$, we observe that using RCNN scores as guidance to construct sample pairs achieves better performance, which further enhances AP50 to 73.0% and AP50−95 to 46.2%. The overall improvement from unlabeled data is 3% on AP50 here.

In Sec. IV-C.2, we further investigate the effectiveness of PCL on unlabeled data for downstream standard datasets such as NIST16 [38] and CASIA [39], [40], where we can obtain over 13% improvement on AP50.

C. Experiments on Standard Datasets

In this section, we firstly evaluate our method on four standard image manipulation datasets NIST16 [38], CASIA [39], [40], COVERAGE [41], Columbia [42] and one real-world dataset IMD20 [43] on labeled data in supervised learning setting. Then we evaluate our methods on NIST16 [38] and CASIA [39], [40] datasets on unlabeled data in semi-supervised learning setting. We fine-tune the COCO pre-trained models on standard datasets except for the Columbia dataset which is directly tested on the COCO pretrained models following previous work [8].

1) PCL on Labeled Data: We first conduct experiments in supervised learning regime with RGB-N and RGB-C as basic detectors respectively. We report AP50 and AP50−95 in Table IV and Table V. For both RGB-N and RGB-C, we find that PCL can consistently improve the AP among most datasets. The most significant improvement of AP50 is on COVER for RGB-N, increased by 5.8%, and on NIST16 for RGB-C, increased by 4%. On Columbia, we observe our method achieves improvement in terms of AP50−95, while shows degradation of AP50. This means that the model misses more objects, while for the detected objects, the results tend to be more accurate. We speculate that this phenomenon possibly because the model is not fine-tuned on this dataset and the domain gap between datasets are the main reasons for the low recall. The consistent improvement shows that PCL can exploit the spatial relationship in tampered images and help to obtain more discriminative feature representations on labeled data.

TABLE III

EXPERIMENTS OF PCL ON UNLABELED DATA WITH DIFFERENT PROPOSAL SAMPLING STRATEGIES, RGB-N IS ADOPTED AS THE BASIC DETECTOR. $D_l$ INDICATES LABELED DATA (50% COCO TRAINING DATA), $D_u$ INDICATES UNLABELED DATA (ANOTHER 50% COCO TRAINING DATA). MODELS ARE VALIDATED WITH THREE RANDOM LABELED-UNLABELED SPLITS, THE MEAN AP50 (%)/AP50−95(%) AND STANDARD VARIANCE ARE REPORTED

| Training Data | Labeled PCL | RPN Score | RCNN Score | AP50 | AP50−95 |
|---------------|-------------|-----------|------------|------|---------|
| $D_l$          | ×           | ×         | ×          | 70.0±0.2 | 46.4±0.8 |
| $D_l$          | ✓           | ×         | ×          | 71.0±1.8 | 45.4±1.8 |
| $D_l + D_u$    | ✓           | ✓         | ×          | 72.5±1.4 | 45.5±1.1 |
| $D_l + D_u$    | ✓           | ✓         | ✓          | 73.0±0.9 | 46.2±1.0 |

TABLE IV

AP50(%) COMPARISON ON STANDARD DATASETS

| Methods        | NIST16 | Columbia | COVER | CASIA | IMD20 |
|----------------|--------|----------|-------|-------|-------|
| RGB-N          | 84.6   | 78.8     | 55.1  | 33.1  | 27.9  |
| +PCL (Ours)    | 87.4   | 78.5     | 60.9  | 35.9  | 29.4  |
| RGB-C          | 82.2   | 81.3     | 62.0  | 39.3  | 31.4  |
| +PCL (Ours)    | 86.2   | 79.8     | 62.7  | 39.3  | 32.5  |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Furthermore, we evaluate PCL with RGB-N and RGB-C as backbone on different manipulation types based on NIST2016 dataset. The results (in terms of AP50−95) are listed in Table VI. It is shown that PCL can obtain improvement on almost all the manipulation types. Most of the improvement stems from the success in removal. On RGB-N, we achieve 5.3% AP50−95 improvement on removal tampered images, while on RGB-C, we get 15.8% improvement on removal tampered images.

2) PCL on Unlabeled Data: To demonstrate that PCL can be effectively used on real datasets with unlabeled data, we examine two semi-supervised learning settings. First, we randomly sample 10%, 30%, 50%, 70% and 90% training data in NIST16, CASIA and IMD20 as the labeled set Dl and regard the rest of training data as unlabeled set Du. Models are trained by the loss function Equ. (8) with unlabeled data incorporated. Second, we use the entire training data as the labeled set and resort to the development dataset of NC2017 (the licensed dataset of OpenMFC 2017 [38], [44]) as the unlabeled set. The unlabeled set here is not necessarily in-domain. NC2017 has 2,512 images with different types of manipulation. Specifically, we combine the unlabeled images from NC2017 with COCO synthetic dataset for model pre-training. Then, we fine-tune the pre-trained models on CASIA, NIST16 and IMD20 datasets. As demonstrated in previous sections, RGB-C obtains better performance than RGB-N, we conduct experiments with RGB-C as the basic detector in the following sections.

The AP50 comparison is shown in Table VII, and AP50−95 comparison is shown Table VIII. Since semi-supervised learning has not been widely studied for image manipulation detection task yet, we mainly compare PCL in semi-supervised regime with the vanilla supervised learning regime which only uses labeled data. On the three datasets NIST16 CASIA and IMD20, our method can consistently enhance the performance under different ratios of labeled data. We find PCL to be particularly effective under the setting of small ratios of labeled data, e.g., we get the improvement of 36.7% AP50 for NIST16 with 10% labeled data (40 labeled images) and 6.7% improvement for CASIA with 30% labeled data (1533 labeled images). Interestingly, PCL is proven to be more data-efficient in semi-supervised settings than supervised learning for image manipulation detection task (65.8% for PCL semi-supervised learning with 10% labeled data v.s. 64.2% for supervised learning with 50% labeled data on NIST16). Furthermore, during training, we find that the model’s performance is beneficial from longer training when more unlabeled data are incorporated which indicates that semi-supervised learning can reduce over-fitting in the low-label regime. From these results, we prove that PCL is effective for unlabeled data too. It is expected that we can obtain more generalizable feature representations with more data incorporated.

We summarize the experimental results in Table IX, when the model is trained with 100% labeled data and extra unlabeled data. It is shown that AP50 on NIST16 can be improved by 7.1%, AP50 on CASIA is increased by 2.5% and AP50 on IMD20 is enhanced by 1.4%. In this setting, we achieve state-of-the-art detection results on NIST16 (AP50 of 89.26%) and CASIA (AP50 of 41.82%). Here, we demonstrate that incorporating more extra realistic data in pre-training can benefit the downstream image manipulation detection.

a) Ablation study of PCL in semi-supervised learning: We further examine the effectiveness of PCL on labeled data and unlabeled data in semi-supervised learning. CASIA and NIST16 are split to 50% labeled data Dl and 50% unlabeled data Du. We evaluate the methods over three different data folds and report results in Table X. It can be observed that when applying PCL on labeled data, we can improve AP50 from 64.2% to 67.3% on NIST16, while on CASIA AP50 is enhanced from 33.2% to 33.5%. When we further incorporate unlabeled data into training, more significant improvement is obtained (10.5% on NIST16 and 3.7% on CASIA), which shows that PCL is effective for both unlabeled data and labeled data.

b) Different choices of threshold of RCNN score δ in semi-supervised learning: Different choices of δ indicate the correctness of the pseudo-label of proposals in unlabeled data. The larger values of δ, the higher the correctness usually, however less number of proposals can be applied to PCL loss. We experiment with different values of δ on NIST16 dataset with 50% labeled data and 50% unlabeled data, the experimental results are shown in Table XI. It can be observed that with different values of δ, we can obtain better performance than only learning with 50% labeled data (64.2% AP50 / 46.6% AP50−95), i.e., the method performs well under a wide range of hyper-parameters.

3) Evaluation on Image-Level Manipulation Detection (Recognition): In this section, we demonstrate the effectiveness of PCL on image-level manipulation detection (recognition) task. Image-level manipulation detection (recognition) aims to distinguish manipulated images from pristine ones via image-level binary classification, the evaluation also involves pristine images and reflects the false alarms of the models. Since CASIA are the only test dataset that owing the corresponding pristine images. We construct the test set with CASIA consisting of 50% manipulated and 50% pristine
TABLE VII

Experiments on NIST16, CASIA and IMD20 Under Different Ratios of Labeled Data in Semi-Supervised Learning. AP50 (%) Is Reported

| Datasets | Methods | 10% | 30% | 50% | 70% | 90% |
|----------|---------|-----|-----|-----|-----|-----|
| NIST16   | RGB-C   | 29.1| 45.5| 64.2| 68.0| 76.7|
|          | +PCL(Ours) | 65.8(+36.7) | 67.0(+21.5) | 77.8(+13.6) | 80.4(+12.4) | 86.4(+9.7) |
| CASIA    | RGB-C   | 28.4| 30.0| 33.2| 35.5| 38.1|
|          | +PCL(Ours) | 34.3(+5.9) | 36.7(+6.7) | 37.2(+4.0) | 39.1(+3.6) | 39.3(+1.2) |
| IMD20    | RGB-C   | 15.5| 21.2| 23.9| 29.8| 30.6|
|          | +PCL(Ours) | 26.7(+11.2) | 27.9(+6.7) | 28.8(+4.9) | 31.7(+1.9) | 32.3(+1.7) |

TABLE VIII

Experiments on NIST16, CASIA and IMD20 Under Different Ratios of Labeled Data in Semi-Supervised Learning. AP50−95 (%) Is Reported

| Datasets | Methods | 10% | 30% | 50% | 70% | 90% |
|----------|---------|-----|-----|-----|-----|-----|
| NIST16   | RGB-C   | 15.4| 27.9| 46.6| 50.2| 58.8|
|          | +PCL(Ours) | 31.2(+15.8) | 38.9(+11.0) | 51.0(+4.4) | 56.4(+6.2) | 63.3(+4.5) |
| CASIA    | RGB-C   | 16.9| 17.8| 22.3| 23.7| 25.9|
|          | +PCL(Ours) | 19.3(+2.4) | 23.0(+5.2) | 23.9(+1.6) | 25.7(+2.0) | 26.0(+0.1) |
| IMD20    | RGB-C   | 7.2 | 11.1| 13.0| 16.4| 16.2|
|          | +PCL(Ours) | 12.3(+5.1) | 14.1(+3.0) | 15.7(+2.7) | 17.3(+0.9) | 17.9(+1.7) |

TABLE IX

Experiments on NIST16, CASIA and IMD20 in Semi-Supervised Learning With 100% Labeled Data and Extra Out-of-Domain Unlabeled Data. AP50 (%) and AP50−95 (%) Are Reported

| Methods | Training Data | NIST16 | CASIA | IMD20 |
|---------|---------------|--------|--------|-------|
| RGB-C   | $D_L$         | 82.2/61.1 | 39.3/25.8 | 31.4/17.5 |
| +PCL(Ours) | $D_L + D_u$ | 89.3/68.1 | 41.8/27.2 | 32.8/17.5 |

TABLE X

Ablation Study of PCL in Semi-Supervised Learning. $D_L$ Indicates Labeled Data (50% Training Data), $D_u$ Indicates Unlabeled Data (Another 50% Training Data). Models Are Validated With Three Random Labeled-Unlabeled Splits, the Mean AP50 (%) and Standard Variance Are Reported

| Methods | Training Data | NIST16 | CASIA |
|---------|---------------|--------|--------|
| RGB-C   | $D_L$         | 64.2±2.1 | 33.2±0.8 |
| +PCL(Ours) | $D_L$ | 67.3±1.3 | 33.5±0.1 |
| +PCL(Ours) | $D_L + D_u$ | 77.8±0.9 | 37.2±0.5 |

TABLE XI

Different Choices of the Threshold of RCNN Scores $\delta$ in Semi-Supervised Learning on NIST16

| $\delta$ | AP50 (%) | AP50−95 (%) |
|----------|----------|-------------|
| 0.4      | 75.2     | 51.0        |
| 0.5      | 77.8     | 51.0        |
| 0.6      | 79.6     | 53.5        |
| 0.7      | 75.6     | 48.8        |

we regard the max score among the detected boxes as the image-level binary classification score. For the compared methods, ManTraNet [16], SPAN [46], and PSCCNet [45] are segmentation-framework-based methods, the averages of the predicted maps are regarded as their scores.

As shown in Table XII, when RGB-C pre-trained on coco synthetic datasets with 11K data, we obtained the AUC of 72.04%, when combined with PCL, the AUC can be further improved to 73.96% which demonstrates that the proposed module PCL helps yield more discriminative features. As the compared counterparts trained their models with a large amount of data, we further pre-trained our models on a larger synthetic datasets released by PSCCNet [45] with 100K data. It is shown that RGB-C trained with 100K data achieved the AUC of 75.47% and RGB-C+PCL outperforms with AUC of 76.38% and F1 of 72.85%.

4) Comparison With Semantic-Segmentation-Based Methods: In the literature on image manipulation detection, another group is segmentation-based methods, such as classic unsupervised methods: ELA [47], NOI1 [48], and CFA1 [2]...
Fig. 6. Qualitative comparison between RGB-N and RGB-N+PCL. Mega columns from left to right are results from COVER, CASIA V1 and NIST16. Red boxes are the ground-truth bboxes, blue ones are the predicted bboxes with highest scores.

| Methods           | NIST16 | Columbia | COVER | CASIA  | IMD20 |
|-------------------|--------|----------|-------|--------|-------|
| classic unsupervised methods |        |          |       |        |       |
| ELA [47]          | 23.6/42.9 | 47.0/58.1 | 22.2/58.3 | 21.4/61.3 | -/   |
| NOI [48]          | 28.5/58.7 | 57.4/54.6 | 26.9/58.7 | 26.3/61.2 | -/   |
| CFAI [2]          | 17.4/30.1 | 46/772.0 | 19.0/48.5 | 20.7/52.2 | -/   |
| semantic segmentation-based methods |        |          |       |        |       |
| MFCN [14]         | 57.1/-   | 61.2/-   | -/-   | 54.1/- | -/-   |
| SPAN [46]         | 58.2/96.1 | 81.5/93.6 | 55.8/93.7 | 38.2/83.8 | -75.0 |
| GSR-Net [49]      | -/-      | -/-      | 48.9/- | 57.4/- | -/-   |
| MVSS-Net [50]     | 81.4/84.2 | -/-      | 50.4/84.9 | 52.2/87.7 | -/-   |
| MantraNet [16]    | -79.5/-  | -82.4/-  | -81.9/- | -81.7/- | -74.8 |
| PSCC-Net [45]     | 74.2/99.1 | -/-      | 72.3/94.1 | 55.4/87.5 | -80.6 |
| object detection-based methods |        |          |       |        |       |
| RGB-N [8]         | 72.2/93.7 | 69/75.8 | 43.7/81.7 | 40.8/79.5 | -/-   |
| RGB-C(Ours)       | 81.6/95.6 | 67/74.4 | 61.3/88.3 | 46.3/74.9 | 49.9/82.4 |
| +PCL(Ours)        | 78.0/94.6 | 69/76.1 | 62.0/91.7 | 46/77.5 | 50.3/82.3 |

Table XIII

Pixel-Level F1 and AUC Score (%) Comparison with Other Counterparts on Five Standard Datasets

and deep learning-based methods: MFCN [14], SPAN [46], GSR-Net [49], MVSS-Net [50], Mantra-Net [16] and PSCC-Net [45]. In this section, we compare our method with these models in terms of pixel-level metrics (F1 score and AUC score). To achieve this, we extend our model with a segmentation module as done in CR-CNN [11] and fine-tune the model end-to-end on downstream datasets. The results are reported in Table XIII (the results of the compared counterparts are extracted from the corresponding papers).

It can be observed that our methods can get better performance compared to classic unsupervised methods such as ELA [47], NOI [48], CFAI [2] and object detection-based methods such as RGB-N [8]. While compared to semantic segmentation-based methods, our methods can outperform MFCN [14], SPAN [46], GSR-Net [49], MVSS-Net [50] and MantraNet [16] on NIST16, COVER and IMD20 datasets under the evaluation of F1, however, slightly lag behind SPAN [46] under the evaluation of AUC, which indicates that we can get higher precision, however the false positive rate is also higher (false positive predictions increase, meanwhile, true positive predictions grow more). We think that the reason for this phenomenon is that our approach is based on the detection framework, which makes more background areas in the bounding boxes have high responses. Most of semantic segmentation-based methods conduct pixle-wise prediction and design fine-gained segmentation modules, for example, both MFCN [14] and GSR-Net [49] holds an edge prediction module which is especially beneficial for CASIA dataset (which contains lots of fine-grained manipulation), our approach lags behind them on CASIA. As SPAN adopts the pre-trained model released by [16], which was pre-trained with millions of unreleased manipulation images, it outperforms others on Columbia. When compared with PSCC-Net [45], we outperform on NIST16 (under the evaluation of F1) and IMD20 dataset. We lag behind on COVER and CASIA for the reason that our models are pretrained on an 11K COCO synthetic dataset following previous work [8], while PSCC-Net is pretrained on a 100K synthetic manipulation dataset which contains various types of manipulation such as splice, copy-move and removal, thus PSCC-Net has a better generalization ability.

It is worth noting that our work is based on the object detection framework and is primarily designed to output the predicted bboxes, it has marginal advantages in terms of pixel-level evaluation metrics compared to the segmentation-based methods. However, in many real applications, there is actually no need to discover every tampered pixel. Instead, the complete and accurate bounding boxes are enough. More importantly, the semantic-segmentation-based methods must require per-pixel annotations for training, while the object-detection-based methods can be trained with the bounding-box annotations, which are much easier and cheaper than the per-pixel annotations (35× – 85× improvement in annotation speed [29]). Therefore, the object-detection-based methods are also valuable, which can meet the need and do not require costly annotations, though they have a natural disadvantage in terms of pixel-level evaluation metrics.

D. Qualitative Results

We show some qualitative results from COVER, CASIA V1 and NIST16 in Fig. 6. We compare the detection results of RGB-N and RGB-N+PCL. The images displayed...
show that the proposal contrastive learning can achieve better performance in both manipulation classification and localization.

**Error Analysis.** We also provide some error analysis of our method in Fig. 7. We find that most of the failure cases of our method are indeed hard examples, especially the images with manipulated regions that are very similar to the background. The failure cases show that although PCL has improved the performance of manipulation detection, it still cannot identify some cunning tampering well, which may require more specific methods.

**V. Limitation and Future Work**

The proposed Proposal Contrastive Learning (PCL) method facilitates the feature representation in manipulation detection models and helps to make full use of extra unlabeled data which we believe can offer insights for practical applications. Though we demonstrate the advantage of our method with the consistent improvement in AP in both supervised learning and semi-supervised learning settings, our method has marginal advantages in terms of pixel-level evaluation metrics compared to the segmentation-based methods.

In future work, (i) we plan to extend the local contrastive learning idea to the segmentation-based methods to enhance the pixel-level localization performance; (ii) further explore semi-supervised learning in the area of image manipulation localization; (iii) moreover, large-scale pre-training data can be utilized to improve the ability of image manipulation models in dealing with the more challenging real-life manipulated images.

**VI. Conclusion**

In this paper, we propose a Proposal Contrastive Learning (PCL) module for image manipulation detection. Our method is deployed on a two-stream architecture by contrasting the features from different views (e.g., RGB and noise features here). PCL can capture the view-invariant factors between different views and thus enhance the feature representations. Moreover, our method can be adapted to unlabeled data to alleviate the problem of insufficient training samples. Extensive experiments among several standard datasets show the effectiveness of PCL for image manipulation detection.
[25] S.-W. Chung, J. S. Chung, and H.-G. Kang, “Perfect match: Improved cross-modal embeddings for audio-visual synchronisation,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), May 2019, pp. 3965–3969.

[26] P. Morgado, N. Vasconcelos, and I. Misra, “Audio-visual instance discrimination with cross-modal agreement,” 2020, arXiv:2004.12943.

[27] Q. Yin, J. Wang, W. Lu, and X. Luo, “Contrastive learning based multi-task network for image manipulation detection,” Signal Process., vol. 201, Dec. 2022, Art. no. 108709.

[28] W. Pan, Z. Zhou, G. Liu, T. Huang, H. Yan, and Q. M. J. Wu, “Auto-focus contrastive learning for image manipulation detection,” 2022, arXiv:2211.10922.

[29] V. Kulkarni, S. Chandra, A. Agrawal, P. Torr, and A. Tyagi, “Box2Seg: Attention weighted loss and discriminative feature learning for weakly supervised segmentation,” in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2020, pp. 290–308.

[30] J. Fridrich and J. Kodovsky, “Rich models for steganalysis of digital images,” IEEE Trans. Inf. Forensics Security, vol. 7, no. 3, pp. 868–882, Jun. 2012.

[31] T. Huynh, S. Kornblith, M. R. Walter, M. Maire, and M. Khademi, “Boosting contrastive self-supervised learning with false negative cancellation,” 2020, arXiv:2011.11765.

[32] A. Van Den Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” 2018, arXiv:1807.03748.

[33] K. Goh, M. E. Cusick, D. Valle, B. Childs, M. Vidal, and A. L. Barabási, “The human disease network,” Proc. Nat. Acad. Sci. USA, vol. 104, no. 21, pp. 8685–8690, 2007.

[34] S. Laine and T. Aila, “Temporal ensembling for semi-supervised learning,” 2016, arXiv:1610.02242.

[35] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” 2015, arXiv:1506.01497.

[36] T.-Y. Lin et al., “Microsoft COCO: Common objects in context,” in Proc. Eur. Conf. Comput. Vis. (ECCV) Cham, Switzerland: Springer, 2014, pp. 740–755.

[37] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal visual object classes (VOC) challenge,” Int. J. Comput. Vis., vol. 88, no. 2, pp. 303–338, 2009.

[38] H. Guan et al., “MFC datasets: Large-scale benchmark datasets for media forensic challenge evaluation,” in Proc. IEEE Winter Appl. Comput. Vis. Workshops (WACVW), Jan. 2019, pp. 63–72.

[39] J. Dong, W. Wang, and T. Tan. (2010). Casia Image Tampering Detection Evaluation Database. [Online]. Available: http://forensics.idealtest.org

[40] J. Dong, W. Wang, and T. Tan, “CASIA image tampering detection evaluation database,” in Proc. IEEE China Summit Int. Conf. Signal Inf. Process., Jul. 2013, pp. 422–426.

[41] B. Wen, Y. Zhu, R. Subramanian, T.-T. Ng, X. Shen, and S. Winkler, “COVERAGE—A novel database for copy-move forgery detection,” in Proc. IEEE Int. Conf. Image Process. (ICIP), Sep. 2016, pp. 161–165.

[42] T.-T. Ng, J. Hsu, and S.-F. Chang, “Columbia image splicing detection evaluation dataset,” DVM Lab. Columbia Univ., CalPhotos Digit. Libr., New York, NY, USA, Tech. Rep., 2009. [Online]. Available: https://www.ee.columbia.edu/ln/dvm/downloads/AuthSplicedDataSet/AuthSplicedDataSet.html

[43] A. Novozamsky, B. Mahdian, and S. Saie, “IMD2020: A large-scale annotated dataset tailored for detecting manipulated images,” in Proc. IEEE Winter Appl. Comput. Vis. Workshops (WACVW), Mar. 2020, pp. 71–80.

[44] H. Guan et al., “User guide for NIST media forensic challenge (MFC) datasets,” Nat. Inst. Standards Technol., USA, Tech. Rep., 2021. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=931782

[45] A. Fujita, Y. Liu, J. Chen, and X. Liu, “PSSC-Net: Progressive spatial-channel correlation network for image manipulation detection and localization,” IEEE Trans. Circuits Syst. Video Technol., vol. 32, no. 11, pp. 7505–7517, Nov. 2022.

[46] X. Hu, Z. Zhang, Z. Jiang, S. Chaudhuri, Z. Yang, and R. Nevatia, “Span: Spatial pyramid attention network for image manipulation localization,” in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2020, pp. 312–328.

[47] N. Krawetz. (2007). A Pictures Worth Digital Image Analysis and Forensics. [Online]. Available: https://www.hackerfactor.com/papers/bh-usa-07-krawetz-wp.pdf

[48] B. Mahdian and S. Saie, “Using noise inconsistencies for blind image forensics,” Image Vis. Comput., vol. 27, no. 10, pp. 1497–1503, Sep. 2009.

[49] P. Zhou et al., “Generate, segment, and refine: Towards generic manipulation segmentation,” in Proc. AAAI Conf. Artif. Intell., 2020, vol. 34, no. 7, pp. 13058–13065.

[50] X. Chen, C. Dong, J. Ji, J. Cao, and X. Li, “Image manipulation detection by multi-view multi-scale supervision,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 14185–14195.