Patch Diffusion: A General Module for Face Manipulation Detection

Baogen Zhang[1], Sheng Li[1]*, Guorui Feng[2], Zhenxing Qian[1], Xinpeng Zhang[1]

[1]Fudan University
[2]Shanghai University
{bgzhang19, lisheng, zxqian, zhangxinpeng}@fudan.edu.cn, grfeng@shu.edu.cn

Abstract

Detection of manipulated face images has attracted a lot of interest recently. Various schemes have been proposed to tackle this challenging problem, where the patch-based approaches are shown to be promising. However, the existing patch-based approaches tend to treat different patches equally, which do not fully exploit the patch discrepancy for effective feature learning. In this paper, we propose a Patch Diffusion (PD) module which can be integrated into the existing face manipulation detection networks to boost the performance. The PD consists of Discrepancy Patch Feature Learning (DPFL) and Attention-Aware Message Passing (AMP). The DPFL effectively learns the patch features by a newly designed Pairwise Patch Loss (PPLoss), which takes both the patch importance and correlations into consideration. The AMP diffuses the patches through attention-aware message passing in a graph network, where the attentions are explicitly computed based on the patch features learnt in DPFL. We integrate our PD module into four recent face manipulation detection networks, and carry out the experiments on four popular datasets. The results demonstrate that our PD module is able to boost the performance of the existing networks for face manipulation detection.

Introduction

In recent years, with the tremendous progress made in computer vision and deep learning, it becomes easy to generate real-look alike fake face images. People can manipulate the face images by softwares such as ZAO, FaceApp, FakeApp, etc. Along with the fun offered by these softwares, malicious face manipulations would mislead the public, which may result in unforeseeable consequences, especially when relevant techniques are applied on celebrities. Therefore, we urgently need advanced and effective methods for manipulated face detection.

The schemes for face manipulation detection can be mainly divided into two categories, the video-based methods and the image-based methods. The video-based methods focus on the differences among different frames to find temporal manipulation traces for classification. Several interesting ideas such as eye blinking (Li, Chang, and Lyu 2018), physiological measurement (Fernandes et al. 2019; Ciftci, Demir, and Yin 2020) and emotions (Mittal et al. 2020) are proposed to achieve satisfactory performance. However, when the input is only a single image, the video-based methods may fail due to the loss of sequential information. This can be addressed by using the image-based method, whose main idea is to detect the manipulated traces inside a single image. Researchers have devoted efforts in developing effective image-based face manipulation techniques. Various manipulation traces have been investigated to facilitate the detection, including the anomaly in color (Li et al. 2020a), the blending boundary (Li et al. 2020b; Tarasiou and Zafeiriou 2020; Wang et al. 2020b), the anomaly in frequency (Qian et al. 2020), and etc. These schemes explore the inconsistency of the traces from the whole image for face manipulation detection, which may not work well when the manipulation traces are subtle.

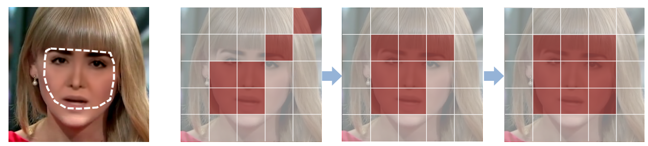

Figure 1: Our Patch Diffusion module diffuses the patches such that the manipulated region gradually merges to prompt the detection. From left to right, a manipulated face image with the boundary of the fake area marked in white, the intermediate prediction results during the patch diffusion. The masks shown in red refer to the patches predicted as fake. Better viewed in color.

Some studies (Chai et al. 2020; Mayer and Stamm 2019) have shown that, the subtle traces located inside small local patches such as manipulation contours or anomaly in texture (Zhang, Karaman, and Chang 2019), would be helpful for differentiating the genuine and fake area. Most recently, Chen *et al.* (Chen et al. 2021) and Zhao *et al.* (Zhao et al. 2021b) consider the similarity among different patches to boost the performance. Despite the advantage, these approaches treat the patches equally, which neglect the diverse impact of different patches for the task of face manipulation detection. More efforts are required to fully exploit the discrepancy among different patches in face manipulation detection.

In this paper, we propose a module named Patch Diffu-

*Sheng Li is the corresponding author: lisheng@fudan.edu.cn

sion (PD) for face manipulation detection. Unlike the existing patch-based schemes which either process the patches independently or simply compute the similarity among different patches, we propose to diffuse the patches using graph networks such that the manipulated region can be gradually merged to prompt the detection, as shown in Fig. 1. In particular, our PD module is composed of two parts: Discrepancy Patch Feature Learning (DPFL) and Attention-Aware Message Passing (AMP). The DPFL focuses on learning effective patch feature representations by taking both the patch importance and correlations into consideration, where a Pairwise Patch Loss (PPLoss) is newly designed for patch feature learning. The AMP constructs a graph from the patches and gradually diffuses the patches through message passing. We propose an attention-aware adjacent matrix for message passing in AMP, which is explicitly computed from the patch features learnt in DPFL.

To demonstrate the generalization ability of our PD module, we integrate it into four recent face manipulation detection networks including CNNDetect (Wang et al. 2020a), Xception (Rossler et al. 2019a), DSP-FWA (Li and Lyu 2019), and WM-Wsdan-Effb (Zhao, Cui, and Zhou 2020) to see how the performance could be boosted. We conduct experiments on four popular datasets. The results demonstrate that, by integrating our PD module into the existing networks, the performance of face manipulation detection can be further boosted in both the intra and inter-database scenarios. The main contributions of this paper are summarized as follows.

- We propose a patch diffusion module for the task of face manipulation detection, which can be integrated into the existing networks for performance boosting.
- We propose a novel patch feature learning scheme by incorporating the patch importance and correlation, which is achieved by a PPLoss newly designed.
- We propose an attention-aware adjacent matrix for patch diffusion, which is computed explicitly from the patch features.

Related Works

Face Manipulation Detection. Existing face manipulation detection schemes can be mainly categorized into the video-based methods and the image-based methods.

The video-based methods perform face manipulation detection based on the inconsistencies between frames. Works on heart rate estimation (Fernandes et al. 2019; Ciftci, Demir, and Yin 2020; Hernandez-Ortega et al. 2020; Qi et al. 2020) detect the manipulation through biological signals collected from videos. Li *et al.* (Li, Chang, and Lyu 2018) and Mittal *et al.*(Mittal et al. 2020) determine whether the frame has been altered based on the facial action and expression. Li *et al.* (Li et al. 2020c) introduces partial face attack in deepfake videos and propose a multiple instance learning framework for face manipulation detection. Several other works (Sabir et al. 2019; Masi et al. 2020; Chen et al. 2020) create new model structures based on the recurrent network.

The image-based methods detect inconsistencies between fake and genuine regions from a single image. Liu *et al.* (Liu, Qi, and Torr 2020) discover the texture differences between real and fake face images. They propose a Gram-Net which focuses on the extraction of global image textures. Works in (Li et al. 2020b; Wang et al. 2020b) focus on intrinsic image discrepancies across the blending boundaries caused by manipulation. The blending regions can be accurately predicted to distinguish the manipulated face images. Dang *et al.* (Dang et al. 2020) propose an attention mechanism to process the face feature maps, which highlights the informative regions for the improvement of detection ability. Similarly, Zhao *et al.* (Zhao et al. 2021a) formulate the face manipulation detection as a fine-grained classification problem, where a multi-attention network is proposed to explore discriminative regions in the face images. Wang *et al.* (Wang et al. 2020a) discuss the impact of data augmentation and indicate that using common image processing operations would be able to improve the generality. Identity-driven method (Dong et al. 2020) takes the idea of face recognition to determine whether the tampered identity is consistent with the original identity.

Recently, several works have been proposed by exploring the subtle manipulated traces located in the image patches, which are shown to be promising for face manipulation detection. Chai *et al.* (Chai et al. 2020) take advantage of a patch-based classifier with limited receptive fields in the image. This work demonstrates that the patch-based classifier is better than the classifiers trained from the whole face images for detecting the subtle manipulated traces. Chen *et al.* (Chen et al. 2021) and Zhao *et al.* (Zhao et al. 2021b) further consider the patch similarity in the spatial and frequency domain to improve the performance, where each patch is equally treated and processed during the patch feature learning.

Graph Network (GN). Using graphs to process data with structural information attracts a lot of research interest. With the development of neural networks, combining the graphs and networks together (Sperduti and Starita 1997; Gori, Monfardini, and Scarselli 2005; Scarselli et al. 2009) becomes an effective approach for various learning based applications. Several models such as message-passing neural network (Gilmer et al. 2017), non-local neural network (Wang et al. 2018), relation network (Santoro et al. 2017) and deep set (Zaheer et al. 2017) are proposed for message-passing in the graph neural networks.

In the field of computer vision , the GNs are increasingly used to tackle tasks such as image classification, object detection, semantic segmentation, etc. Chen *et al.* (Chen et al. 2019) propose a graph-based global reasoning network to globally aggregate the features located in different regions, which works well on image classification tasks. Zhang *et al.* (Zhang et al. 2020) propose a dynamic graph message passing network to adaptively sample nodes in the graph, which performs well in semantic segmentation tasks. Monti *et al.* (Monti et al. 2017) propose a spatial-domain model to generalize CNN architectures to graphs such that it can learn task-specific features. Qi *et al.* (Qi et al. 2018) propose a graph parsing neural network for inferring a parse graph in an end-to-end manner. Such a graph can be integrated into the existing object detection frameworks for better perfor-

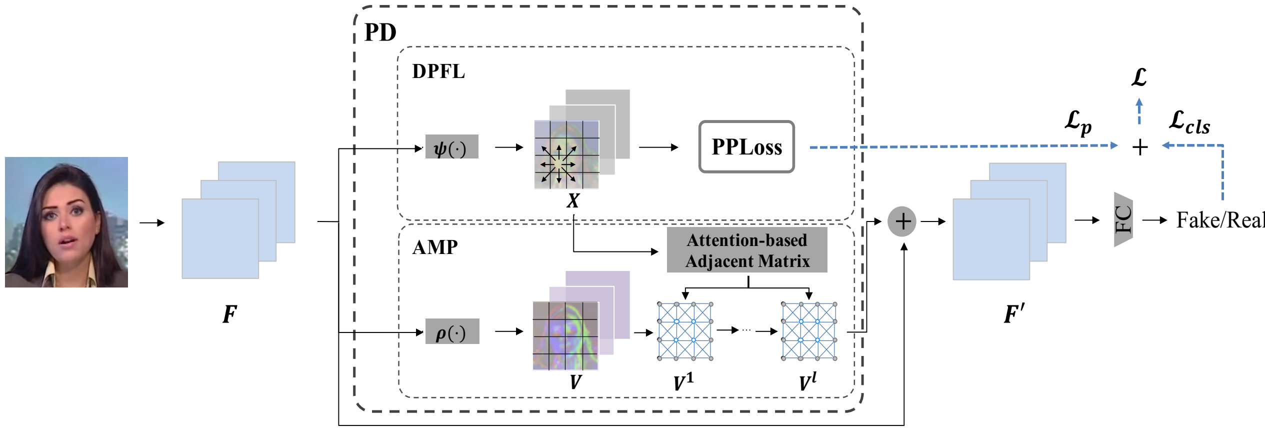

Figure 2: Patch diffusion for face manipulation detection.

mance.

The Proposed Method

The architecture of our proposed patch diffusion (PD) module is given in Fig. 2. The input of the PD module is the feature map of a convolutional layer of an existing network. Its output is a feature map with the same dimension as the input. Each spatial location in the feature map represents a local patch. Our PD model processes the input feature map by, 1) discrepancy patch feature learning (DPFL) and 2) attention-aware message passing (AMP). We propose a Pairwise Patch Loss (PPLoss) to learn effective patch feature representations in DPFL. These are used to compute an attention-aware adjacent matrix which explicitly serves as an attention mechanism for patch diffusion in AMP. In what follows, we explain the DPFL and AMP in detail, as well as the module integration and training.

Discrepancy Patch Feature Learning (DPFL)

Let's denote input feature map as $\mathbf{F} \in \mathbb{R}^{H \times W \times C}$, where H, W, C are the height, width and number of channels, respectively. Each spatial location in $\mathbf{F}$ corresponds to a local patch represented as a C dimensional vector, with a total of $N_p = H \times W$ patches.

Given a feature map $\mathbf{F}$, we first transform it into $\mathbf{X} = \psi(\mathbf{F}) \in \mathbb{R}^{H \times W \times C'}$, where $\psi(\cdot)$ is a 1×1 convolution layer with C' kernels. This is to make the feature dimension of each local patch invariant among different networks. For the ith patch, let's denote $\mathbf{x}_i$ as the corresponding feature vector in $\mathbf{X}$. Therefore, we have $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N_p}$ in terms of the N_p patches in the image.

We annotate the face image into three regions including the fake region, real region and the background region, as shown in Fig. 3. For the ith patch $\mathbf{x}_i$, let's denote $\mathcal{I}_i^{\mathrm{f}}$ and $\mathcal{I}_i^{\mathrm{r}}$ as the ratios of the fake and real region that occupy the patch, respectively. The ground truth of the ith patch is computed as

$$y_i = \begin{cases} 0, & \mathcal{I}_i^{\mathrm{f}} > \delta, \\ 1, & \mathcal{I}_i^{\mathrm{r}} > \delta \text{ and } \mathcal{I}_i^{\mathrm{f}} < \delta, \\ 2, & \text{otherwise} \end{cases}, \tag{1}$$

where δ is a threshold for the tolerance of annotation errors. $y_i = 0,\ 1$ or 2 refer to the case that the patch is annotated as fake, real or background, respectively. We set a relatively small δ with $\delta = 0.1$ for robust patch annotation.

As indicated in (Li et al. 2020b), the traces around the edges of the fake area are more distinct than that in the other areas, which plays an important role for manipulated face detection. This is to say, the importance of each patch should be different for such a detection task. More attentions should be paid on the patches which contain edges of different regions. We evaluate the importance of the ith patch by a score s_i which is formulated below.

$$s_i = \begin{cases} 2 - \mathcal{I}_i^{\mathrm{f}} & if\ y_i = 0 \\ 2 - \mathcal{I}_i^{\mathrm{r}} & if\ y_i = 1 \\ 2 - \mathcal{I}_i^{\mathrm{b}} & if\ y_i = 2 \end{cases}, \tag{2}$$

where $\mathcal{I}_i^{\mathrm{b}}$ is the ratio of the background region in the patch and $s_i \in [1, 2)$. The larger the s_i, the more important the patch will be.

With y_i and s_i available, we propose here a Pairwise Patch Loss (PPLoss) by considering the discrepancy between a pair of two patches (say $\mathbf{x}_i$ and $\mathbf{x}_j$). The discrepancy lies in the following two aspects, 1) the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$, which is computed by the L2-norm denoted as $d_{ij} = ||\mathbf{x}_i - \mathbf{x}_j||$, and 2) the difference between the patch ground truth (i.e., y_i and y_j), which is indicated by a hyper-parameter ϵ_{ij} given below.

$$\epsilon_{ij} = \begin{cases} 0 & if\ y_i = y_j \\ \alpha & if\ y_i \neq y_j \end{cases}, \tag{3}$$

where α is a hyperparameter depending on the ground truth of the two patches. The PPLoss is formulated by

$$\mathcal{L}_p = \frac{1}{N_e} \sum_{i=1}^{N_p} \sum_{j \in \mathcal{N}_i} \{f(s_i, s_j)|d_{i,j} - \epsilon_{ij}|\} \tag{4}$$

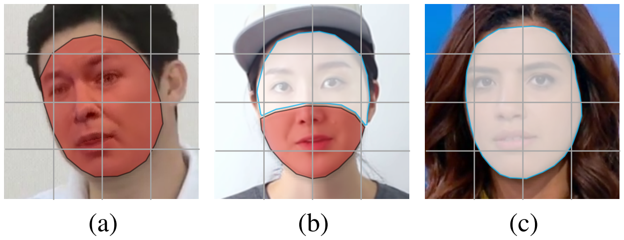

Figure 3: Illustrations of different regions in a face image, (a) a fake face image with a fake region and a background region, (b) a fake face image with a fake region, a real region and a background region, (c) a genuine face image with a real face region and a background region. The fake region and real region are shown in red masks and white masks, with the rest being the background region. Better viewed in color.

where

$$f(s_i, s_j) = \begin{cases} 1, & s_i = 1 \text{ and } s_j = 1 \\ s_i s_j \omega, & otherwise \end{cases} \tag{5}$$

is a function evaluating the importance of a pair of two patches with $\omega > 1$ as a hyperparameter, $|z|$ computes the absolute value of z, and $\mathcal{N}_i$ denotes the indexes of a local neighborhood (within two hops) of the ith patch, and N_e is the total patch pairs that are used to compute the loss.

Our PPLoss contains two parts: $|d_{i,j} - \epsilon_{ij}|$ and $f(s_i, s_j)$. Here, we refer them as D_{ij} and W_{ij} for short. The D_{ij} tries to maximize the inter-class distance and minimize the intra-class distance ($\epsilon_{ij} = 0$ when $i = j$). Since we have three classes of patches, it would be difficult to learn patch features that can maximize the inter-class distance for different pairs of classes. Instead, we regard such distance as hyper-parameters (i.e., α) whose optimal values can be determined in validation. Intuitively, α should be large enough for classes that are difficult to be differentiated (e.g., real and fake). This is in accordance with the optimal values of α found in validation (see Section for details). The W_{ij} additionally weights D_{ij} by taking the patch importance into consideration. It makes the patches with edges contribute more during the training, which effectively exploits the distinct traces left on the edges for learning representative patch features.

Next, we combine our PPLoss with the classification loss of the network to be integrated (say $\mathcal{L}_{cls}$) for feature learning, which is given as

$$\mathcal{L} = \lambda \mathcal{L}_p + \mathcal{L}_{cls}, \tag{6}$$

where λ is the weight balancing $\mathcal{L}_p$ and $\mathcal{L}_{cls}$. The PPLoss helps us to learn representative patch features, where patches belong to the same region (fake, real or background) tend to be similar. For the patches belong to different regions, their difference will be close to α. The PPLoss also makes sure that the patches containing edges of different regions contribute more during the learning, which effectively exploits the distinct traces left on the edges of the manipulation area.

Attention-aware Message Passing (AMP)

Initialization: First of all, we transform the feature map $\mathbf{F}$ by $\mathbf{V} = \rho(\mathbf{F}) \in \mathbb{R}^{H \times W \times C}$, where $\rho(\cdot)$ is a 1×1 convolution layer with C kernels. We denote each patch in $\mathbf{V}$ as a node. Thus, the feature map $\mathbf{V}$ contains $N_p = H \times W$ nodes with $\mathbf{V} = \{\mathbf{v}_i\}_{i=1}^{N_p}$. Then, we construct an undirected graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ with $\mathbf{V}$ as the nodes and $\mathbf{E}$ as the edges. Initially, each node is fully connected to all the other nodes in the graph to form the edges.

Message Passing: In order to perform the patch diffusion, we gradually updates $\mathbf{V}$ through message passing using a graph convolutional network (GCN) (Kipf and Welling 2017), which is formulated as

$$\mathbf{V}^{(l+1)} = \mathbf{A}\mathbf{V}^{(l)}\mathbf{W}^{(l)} \tag{7}$$

where $\mathbf{V}^{(l)}$ and $\mathbf{V}^{(l+1)}$ refer to the nodes at the lth and $(l+1)$th iteration with $\mathbf{V}^{(0)} = \mathbf{V}$, $\mathbf{W}^{(l)}$ is a layer-specific trainable weight matrix for intra-node feature fusion and $\mathbf{A}$ is the adjacent matrix for message passing with a dimension of $N_p \times N_p$.

The adjacent matrix defines how two nodes exchange information, which is important for message passing. Most of the existing works incorporate a binary adjacent matrix (Kipf and Welling 2017). In such a case, $\mathbf{A}(i, j) = 1$ means there is an edge between $\mathbf{v}_i$ and $\mathbf{v}_j$ (i.e., $\mathbf{v}_i$ and $\mathbf{v}_j$ belong to the same region), $\mathbf{A}(i, j) = 0$ means $\mathbf{v}_i$ and $\mathbf{v}_j$ are not connected (i.e., $\mathbf{v}_i$ and $\mathbf{v}_j$ belong to different regions). Some works (Chen et al. 2019; Te et al. 2020) implicitly define the adjacent matrix by a learnable 1×1 convolutional layer, which serves as an attention mechanism for message passing.

Instead of using a convolutional layer to learn the adjacent matrix, we propose here an explicit way to compute an attention-aware adjacent matrix. To be more specific, we explicitly compute a weight $\mathbf{A}(i, j)$ for the message passing between $\mathbf{v}_i$ and $\mathbf{v}_j$, where

$$\mathbf{A}(i, j) = \frac{\mathbf{x}_i \mathbf{x}_j^T}{||\mathbf{x}_i|| \cdot ||\mathbf{x}_j|| + \xi}, \tag{8}$$

where ξ is a scalar to prevent division by 0. As such, the correlations between any two nodes (i.e., patches) are considered for the message passing, which would be helpful to lead a smooth patch diffusion.

After l iterations, we can get a more representative feature map $\mathbf{V}^{(l)}$. The output feature map is computed as

$$\mathbf{F}' = \mathrm{BN}(\mathbf{V}^{(l)}) + \mathbf{F}, \tag{9}$$

where BN is the batch normalization.

Module Integration and Training

As a module, our PD can be integrated into the existing convolutional neural networks. In order to achieve satisfactory performance, the dimension of the input feature map for our PD should not be too small or too large. Because each spatial element in the feature map corresponds a local image patch. Feature maps with large dimensions refer to small image patches which do not contain sufficient information for patch diffusion. On the contrary, feature maps with small dimensions indicate that the image patches are large, which are not good for the detection of the subtle manipulation traces.

In the training phase, both the images and the corresponding patch ground truth will be fed into the integrated network, which will then be trained according to the loss given in Eq. (6). After the training, the integrated network can be used to predict whether an input face image is genuine or fake according to the output of the classification layer.

Experiments

Setup

Training dataset. We select FaceForensics++ (Rossler et al. 2019b) as the training dataset. It is a large-scale face manipulation dataset containing 1000 real videos, in which 720 videos are used for training, 140 videos are reserved for validation and 140 videos are for testing. Each real video has four manipulated versions which are generated by DeepFakes (DF) (Tora 2018), Face2Face (F2F) (Thies et al. 2016), FaceSwap (FS) (Kowalski 2018) and NeuralTextures (NT) (Thies, Zollhöfer, and Nießner 2019). In addition, each video has three compression levels: RAW, High Quality (HQ) and Low Quality (LQ), respectively. We evenly select 16 frames from each video. For each frame, we extract 81 facial landmarks [1] to crop the facial area.

For the fake face images generated by DF and F2F, we obtain the fake and background regions according to the facial mask given in the database, while the real regions are automatically annotated according to the facial landmarks. For those generated by NT and FS, we also annotate different regions according to the facial landmarks. With the regions annotated, the ground truth of the patches are calculated based on Eq. (1). Therefore, there is no human labor involved for the annotations in our experiment.

Test datasets. To have a thorough demonstration for the generality of our propose scheme, we conduct the testing on the following databases: (1) FaceForensics++ (Rossler et al. 2019b): all the videos in the test set are used for testing. (2) DeepfakeDetection [2] (DFD): DFD is a large dataset for synthetic video detection, we use all the RAW videos for testing. (3) Celeb-DF (Li et al. 2020d): Celeb-DF is a large-scale challenging DeepFake video dataset with two versions: Celeb-DFv1 (CD1) and Celeb-DFv2 (CD2), and we test on both of them. (4) DeeperForensics (DFR) (Jiang et al. 2020): DFR is a recently released face swap video dataset with 60,000 videos, it is a representative dataset for face manipulation detection, and we test on all the videos.

Implementation details. We choose four recent face manipulation detection networks, including CNNDetect (Wang et al. 2020a), Xception (Rossler et al. 2019b), DSP-FWA (Li and Lyu 2019) and WM-Wsdan-Effb (Zhao, Cui, and Zhou 2020) for the integration of our PD module. The CNNDetect (Wang et al. 2020a) is shown to have good performance for deepfake detection. The Xception (Rossler et al. 2019b) is a baseline model in FaceForensics++, which is a popular method in the area of face manipulation detection. The DSP-FWA (Li and Lyu 2019) achieves satisfactory performance on Celeb-DF. The WM-Wsdan-Effb (Zhao, Cui, and Zhou 2020) is the runner up in the facebook Deepfake Detection Challenge (Dolhansky et al. 2019). Note that the winner of this challenge [3] drops out part of the image for data augmentation, which is not appropriate for our module.

For each image, we adopt the method proposed in (Rossler et al. 2019b) to automatically and conservatively crop the facial area into a square, which is then resized to 224×224 for training and testing. Our module is integrated after a convolutional block with a feature map of $14 \times 14 \times C$ as the output. In other words, we set $H = W = 14$ and $N_p = 196$ fixed during the experiment, while C various among different models and we set $C' = C/2$. We set $\alpha = 2, 15, 17$ for a pair of patches whose ground truth are real and background, fake and background, real and fake, respectively. Other hyperparameters are set by $\lambda = 1, \omega = 3, l = 1, \xi = 10^{-8}$. The code is available at https://github.com/starxchina/Patch-Diffusion

Intra-database Evaluation for the PD Module

In this section, we evaluate the performance of the four existing networks on FaceForensics++ before and after the integration of our PD module. Three indicators are used to evaluate the network performance including: ACC (correctly classified images/total images), AUC (area under the Receiver Operating Characteristic curve) and EER (Equal Error Rate). In the following discussions, all the three indicators are reported in terms of percentages.

We use all the subsets of the FaceForensics++ (DF, F2F, FS, NT, Real) to train the models. Each training image has three different compression levels, and we conduct the training and testing on the same compression level. Table 1 gives the performance on FaceForensics++ for different compression levels, where RAW means no compression, HQ and LQ indicate high and low compression quality, respectively. It can be seen that our PD module is able to boost the performance for majority of the cases regardless the networks. The only two cases that we are not able to achieve better results (by using the PD module) is the ACC and EER on RAW for WM-Wsdan-Effb. This is mainly because the performance of WM-Wsdan-Effb is already very high on RAW. On the other two compression levels (i.e., HQ and LQ), our PD module is able to improve the performance consistently on all the four existing networks. For most of the existing networks, more performance gain can be achieved using our PD module on LQ, where the performance of the four existing networks are not satisfactory. After integrating the PD module for the detection of LQ, the improvement of the ACC is 4.73%, 1.96%, 0.46% and 0.88% for the CNNDetect, Xception, DSP-FWA, and WM-Wsdan-Effb, respectively. Such improvements are reduced to 2.6%, 1.1% and 0.54% for the CNNDetect, Xception and WM-Wsdan-Effb on HQ. For the DSP-FWA, however, we are able to gain more improvement of the ACC (i.e., 5%) on the HQ than on the LQ.

[1] https://github.com/codeniko/shape_predictor_81_face_landmarks

[2] https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html

[3] https://github.com/selimsef/dfdc_deepfake_challenge

Method	RAW			HQ			LQ		
	ACC	AUC	EER	ACC	AUC	EER	ACC	AUC	EER
CNNDetect (Wang et al. 2020a)	96.97	99.46	3.61	91.56	96.21	10.70	79.04	82.30	26.91
Xception (Rossler et al. 2019a)	98.60	99.68	1.61	94.33	97.92	7.31	84.41	84.15	24.25
DSP-FWA (Li and Lyu 2019)	98.44	99.57	1.76	86.70	90.26	17.89	81.05	74.45	32.80
WM-Wsdan-Effb (Zhao, Cui, and Zhou 2020)	**99.50**	**99.87**	**0.52**	94.62	98.02	6.41	85.23	87.50	21.03
CNNDetect+PD	**98.70**	**99.78**	**1.28**	**94.16**	**98.06**	**6.84**	**83.77**	**84.86**	**23.81**
Xception+PD	**99.30**	**99.75**	**0.71**	**95.43**	**98.71**	**5.55**	**86.37**	**89.99**	**18.88**
DSP-FWA+PD	**99.25**	**99.66**	**0.81**	**91.70**	**96.25**	**10.63**	**81.51**	**80.37**	**27.81**
WM-Wsdan-Effb+PD	99.43	**99.87**	0.57	**95.16**	**98.51**	**5.46**	**86.11**	**89.29**	**18.79**

Table 1: The performance on FaceForensics++ on different quality levels before and after the integration of our PD module.

Method	Training Set	Test Set (AUC (%))			
		DF	F2F	FS	NT
Xception	DF	**100.00**	66.92	36.78	84.98
	F2F	96.75	99.50	62.29	76.51
	FS	67.85	75.81	99.96	59.46
	NT	98.70	82.82	49.74	99.24
Xception+PD	DF	**100.00**	**82.32**	**38.78**	**91.50**
	F2F	**99.58**	**99.68**	**77.78**	**91.98**
	FS	**83.49**	**93.27**	**100.00**	**82.12**
	NT	**99.87**	**93.13**	**61.53**	**99.52**

Table 2: Performance of cross-database evaluation on different manipulation types in FaceForensics++.

Method	Test Set (AUC (%))			
	DFD	CD1	CD2	DFR
CNNDetect	89.70	67.03	75.99	22.11
Xception	93.45	65.90	73.27	60.18
DSP-FWA	90.14	59.93	68.06	58.15
WM-Wsdan-Effb	96.46	75.15	80.80	68.23
CNNDetect+PD	**93.99**	**70.80**	**77.03**	**70.32**
Xception+PD	**95.74**	**68.23**	**74.22**	**65.57**
DSP-FWA+PD	**93.07**	**69.45**	**76.01**	**67.99**
WM-Wsdan-Effb+PD	**97.38**	**75.39**	**82.25**	**68.41**

Table 3: Performance of the cross-database evaluation among different databases.

Cross-database Evaluation for the PD Module

On different manipulation types. By following the same protocol as the work in (Chai et al. 2020; Dang et al. 2020), we conduct cross-database evaluation between different subsets in the FaceForensics++, where the AUC is served as the indicator. In particular, we select one subset for training, and then test the performance on the other three subsets. Both the training and testing are carried out on the RAW level, and we choose Xception as a representative network. The results are given in Table 2. It can be seen that integrating our PD module significantly improves the performance. For example, when trained on FS training and test on the rest three subsets, the improvements of the AUC are 15.64%, 17.46% and 22.66% for DF, F2F and NT, respectively. This indicates that our PD works well for the detection of unseen manipulation types.

On different databases. Next, we conduct the cross-database evaluation on different databases. In particular, we carry out the training on the FaceForensic++ and the testing on DFD, Celeb-df (CD1), Celeb-df (CD2) and DFR, the results of which are shown in Table 3. We find that our PD module can consistently improve the AUC for different networks. But the improvement here is less when compared with the cross-database evaluation for different manipulation types.

DPFL	AMP	ACC	AUC	EER
-	-	84.41	84.15	24.25
✓	-	85.10	89.11	19.46
-	✓	83.42	82.46	25.53
✓	✓	**86.37**	**89.99**	**18.88**

Table 4: Ablation studies on DPFL and AMP.

	ACC	AUC	EER
Xception+DPFL	85.10	89.11	19.46
Xception+PD′	85.57	88.94	19.46
Xception+PD	**86.37**	**89.99**	**18.88**

Table 5: Ablation studies on the adjacent matrix.

Ablation Studies

In this section, we conduct ablation studies on different components of our PD module. All the ablation studies are carried out on the LQ of FaceForensics++ with Xception as the basic network.

DPFL and AMP. We gradually switch off the DPFL and AMP in our PD module to see how the performance changes. As shown in Table 4, we can see that using both of them achieves the best performance, with 1.96% of higher ACC than the original Xception, 1.27% higher ACC than using DPFL only, and 2.95% higher ACC than using the AMP only. We observe that using the AMP along will degrade the performance of the Xception, which indicates the effective-

Patch importance	ACC	AUC	EER
same score	85.53	88.32	20.36
proposed score	**86.37**	**89.99**	**18.88**

Table 6: Ablation studies on patch importance.

Method	ACC RAW	AUC RAW	ACC HQ	AUC HQ
Face X-ray	-	98.80	-	87.40
LRL	**99.87**	99.92	97.59	99.46
PCL	-	99.79	-	-
WM-Wsdan-Effb	99.62	**100.00**	96.59	99.49
WM-Wsdan-Effb+PD	99.62	**100.00**	**98.48**	**99.92**

Table 7: Comparisons of intra-database evaluation with the state-of-the-art.

ness of our DPFL for representative patch feature learning.

Attention-based adjacent matrix. To demonstrate the effectiveness of our attention-based adjacent matrix for message passing. We replace it with an existing attention-based adjacent matrix (PD′) which are learnt by a network (Chen et al. 2019; Te et al. 2020). Table 5 gives the performance of the Xception+PD by using different adjacent matrixes. It can be seen that both are able to further improve the performance compared with using only the DPFL (i.e., Xception+DPFL). But using our explicitly computed adjacent matrix works better than using the one learnt from a network.

Patch importance. When learning the patch features in DPFL, we propose and incorporate a score to measure the importance of each patch in the PPLoss. In order to see the impact of the patch score, we conduct experiment by setting the score of all the patches as 1, the results of which is shown in Table 6. It can be seen that incorporating the proposed patch scores is indeed helpful. So more emphasis should be put on the patches containing edges of different regions.

Comparisons with the State-of-the-art

In this section, we compare the networks integrated with the PD module against the state-of-the-art work including the Face X-ray (Li et al. 2020b), LRL (Chen et al. 2021), PCL (Zhao et al. 2021b) and Patch Forensics (Chai et al. 2020). All these works as well as our PD module require to annotate the patches for training, and the work in (Li et al. 2020b) and (Zhao et al. 2021b) generate additional data from the training set for augmentation. For fair comparison, we follow the same protocols as suggested in these works to e-valuate our method, where the results of the existing works are duplicated from the literature.

First of all, we conduct the intra-database comparison on FaceForensics++ by following the protocol given in (Li et al. 2020b; Chen et al. 2021; Zhao et al. 2021b). In particular, we use the training sets of RAW and HQ in FaceForensics++ for training, and the corresponding test set for testing. Table 7 gives the comparisons among different schemes. It can be seen that, by integrating PD with WM-Wsdan-Effb, we can achieve better performance than the existing scheme in most of the cases.

Method	Test Set (AUC (%))	
	DFD	CD2
Face X-ray	95.40	-
LRL	89.24	78.26
PCL	-	81.80
WM-Wsdan-Effb	96.46	80.80
WM-Wsdan-Effb+PD	**97.38**	**82.25**

Table 8: Comparisons of cross-database evaluation with the state-of-the-art.

Method	Training Set	Test Set (AP (%))			
		DF	NT	F2F	FS
Patch-Forensics	DF	99.41	74.99	71.74	**58.74**
	F2F	84.39	80.88	97.66	63.21
	FS	61.77	53.44	62.00	97.13
	NT	70.32	92.23	65.04	52.79
Xception+PD	DF	**100.00**	**89.57**	**83.18**	42.63
	F2F	**99.62**	**91.40**	**99.58**	**79.04**
	FS	**85.09**	**82.65**	**94.48**	**100.00**
	NT	**99.87**	**99.14**	**92.79**	**61.60**

Table 9: Comparisons of cross-database evaluation on different manipulation types in FaceForensics++.

We then conduct the cross-database comparison on DFD and CD2 as suggested in (Li et al. 2020b; Chen et al. 2021; Zhao et al. 2021b), where all the models are trained in FaceForensics++. As shown in Table 8, our WM-Wsdan-Effb+PD performs consistently better than the existing schemes. Our scheme performs significantly better than the LRL here, with over 8% higher AUC on DFD and around 4% higher AUC on CD2. When compared with (Li et al. 2020b) and (Zhao et al. 2021b), the improvement is not significant because both of these schemes use additional data for data augmentation. It might be a good choice to combine such strategy with our proposed PD module.

Next, we compare our method with the Patch Forensics (Chai et al. 2020) in Table 9, where the average precision (AP) of Patch-Forensics are the best results on different X-ception blocks reported in (Chai et al. 2020). It can be seen that our method performs better than the Patch Forensics in majority of the cases with over 10% higher AP.

Conclusion

In this paper, we propose a face manipulation detection module by taking advantage of the graph network. This module is termed as the Patch Diffusion (PD) module, which consists of Discrepancy Patch Feature Learning (DPFL) and Attention-Aware Message Passing (AMP). The DPFL learns the patch features according to a newly designed Pairwise Patch Loss. While the AMP performs the message passing to diffuse the patches according to an adjacent matrix explicitly computed from the patch features. Various experiments demonstrate the effectiveness of our PD module in the existing networks for face manipulation detection.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62072114, U20A20178, 62072295, U20B2051 and U1936214, in part by Alibaba Group through Alibaba Innovative Research (AIR) Program, in part by the Shanghai Science and Technology Committee under Grant 21010500200.

References

Chai, L.; Bau, D.; Lim, S.-N.; and Isola, P. 2020. What makes fake images detectable? Understanding properties that generalize. In *European Conference on Computer Vision*, 103–120. Springer.

Chen, P.; Liu, J.; Liang, T.; Zhou, G.; Gao, H.; Dai, J.; and Han, J. 2020. FSSPOTTER: Spotting Face-Swapped Video by Spatial and Temporal Clues. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6. IEEE.

Chen, S.; Yao, T.; Chen, Y.; Ding, S.; Li, J.; and Ji, R. 2021. Local Relation Learning for Face Forgery Detection. In *AAAI*, 1081–1088.

Chen, Y.; Rohrbach, M.; Yan, Z.; Shuicheng, Y.; Feng, J.; and Kalantidis, Y. 2019. Graph-based global reasoning networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 433–442.

Ciftci, U. A.; Demir, I.; and Yin, L. 2020. Fakecatcher: Detection of synthetic portrait videos using biological signals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Dang, H.; Liu, F.; Stehouwer, J.; Liu, X.; and Jain, A. K. 2020. On the detection of digital face manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5781–5790.

Dolhansky, B.; Howes, R.; Pflaum, B.; Baram, N.; and Canton-Ferrer, C. 2019. The Deepfake Detection Challenge (DFDC) Preview Dataset. *ArXiv*, abs/1910.08854.

Dong, X.; Bao, J.; Chen, D.; Zhang, W.; Yu, N.; Chen, D.; Wen, F.; and Guo, B. 2020. Identity-Driven DeepFake Detection. *arXiv preprint arXiv:2012.03930*.

Fernandes, S.; Raj, S.; Ortiz, E.; Vintila, I.; Salter, M.; Urosevic, G.; and Jha, S. 2019. Predicting heart rate variations of deepfake videos using neural ode. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 0–0.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, 1263–1272. PMLR.

Gori, M.; Monfardini, G.; and Scarselli, F. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, 729–734 vol. 2.

Hernandez-Ortega, J.; Tolosana, R.; Fierrez, J.; and Morales, A. 2020. DeepFakesON-Phys: DeepFakes Detection based on Heart Rate Estimation. *arXiv preprint arXiv:2010.00400*.

Jiang, L.; Li, R.; Wu, W.; Qian, C.; and Loy, C. C. 2020. DeeperForensics-1.0: A Large-Scale Dataset for Real-World Face Forgery Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2889–2898.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

Kowalski, M. 2018. FaceSwap. https://github.com/marekkowalski/faceswap. Accessed: 2020-08-01.

Li, H.; Li, B.; Tan, S.; and Huang, J. 2020a. Identification of deep network generated images using disparities in color components. *Signal Process.*, 174: 107616.

Li, L.; Bao, J.; Zhang, T.; Yang, H.; Chen, D.; Wen, F.; and Guo, B. 2020b. Face x-ray for more general face forgery detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5001–5010.

Li, X.; Lang, Y.; Chen, Y.; Mao, X.; He, Y.; Wang, S.; Xue, H.; and Lu, Q. 2020c. Sharp Multiple Instance Learning for DeepFake Video Detection. In *Proceedings of the 28th ACM International Conference on Multimedia*, 1864–1872.

Li, Y.; Chang, M.-C.; and Lyu, S. 2018. In ictu oculi: Exposing ai created fake videos by detecting eye blinking. In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 1–7. IEEE.

Li, Y.; and Lyu, S. 2019. Exposing DeepFake Videos By Detecting Face Warping Artifacts. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Li, Y.; Yang, X.; Sun, P.; Qi, H.; and Lyu, S. 2020d. Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3207–3216.

Liu, Z.; Qi, X.; and Torr, P. H. 2020. Global texture enhancement for fake face detection in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8060–8069.

Masi, I.; Killekar, A.; Mascarenhas, R. M.; Gurudatt, S. P.; and AbdAlmageed, W. 2020. Two-branch recurrent network for isolating deepfakes in videos. In *European Conference on Computer Vision*, 667–684. Springer.

Mayer, O.; and Stamm, M. C. 2019. Forensic similarity for digital images. *IEEE Transactions on Information Forensics and Security*, 15: 1331–1346.

Mittal, T.; Bhattacharya, U.; Chandra, R.; Bera, A.; and Manocha, D. 2020. Emotions Don't Lie: An Audio-Visual Deepfake Detection Method using Affective Cues. In *Proceedings of the 28th ACM International Conference on Multimedia*, 2823–2832.

Monti, F.; Boscaini, D.; Masci, J.; Rodola, E.; Svoboda, J.; and Bronstein, M. M. 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5115–5124.

Qi, H.; Guo, Q.; Juefei-Xu, F.; Xie, X.; Ma, L.; Feng, W.; Liu, Y.; and Zhao, J. 2020. DeepRhythm: exposing deepfakes with attentional visual heartbeat rhythms. In *Proceedings of the 28th ACM International Conference on Multimedia*, 4318–4327.

Qi, S.; Wang, W.; Jia, B.; Shen, J.; and Zhu, S.-C. 2018. Learning human-object interactions by graph parsing neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 401–417.

Qian, Y.; Yin, G.; Sheng, L.; Chen, Z.; and Shao, J. 2020. Thinking in frequency: Face forgery detection by mining frequency-aware clues. In *European Conference on Computer Vision*, 86–103. Springer.

Rossler, A.; Cozzolino, D.; Verdoliva, L.; Riess, C.; Thies, J.; and Niessner, M. 2019a. FaceForensics++: Learning to Detect Manipulated Facial Images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Rossler, A.; Cozzolino, D.; Verdoliva, L.; Riess, C.; Thies, J.; and Nießner, M. 2019b. Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE International Conference on Computer Vision*, 1–11.

Sabir, E.; Cheng, J.; Jaiswal, A.; AbdAlmageed, W.; Masi, I.; and Natarajan, P. 2019. Recurrent Convolutional Strategies for Face Manipulation Detection in Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Santoro, A.; Raposo, D.; Barrett, D. G. T.; Malinowski, M.; Pascanu, R.; Battaglia, P. W.; and Lillicrap, T. 2017. A simple neural network module for relational reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, 4967–4976.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2009. The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, 20(1): 61–80.

Sperduti, A.; and Starita, A. 1997. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*, 8(3): 714–735.

Tarasiou, M.; and Zafeiriou, S. 2020. Extracting Deep Local Features to Detect Manipulated Images of Human Faces. In *IEEE International Conference on Image Processing, ICIP*, 1821–1825. IEEE.

Te, G.; Liu, Y.; Hu, W.; Shi, H.; and Mei, T. 2020. Edge-aware Graph Representation Learning and Reasoning for Face Parsing. In *European Conference on Computer Vision*, 258–274. Springer.

Thies, J.; Zollhöfer, M.; and Nießner, M. 2019. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4): 1–12.

Thies, J.; Zollhofer, M.; Stamminger, M.; Theobalt, C.; and Nießner, M. 2016. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2387–2395.

Tora, M. 2018. Deepfakes. https://github.com/deepfakes/faceswap/tree/v2.0.0. Accessed 2020-08-01.

Wang, S.-Y.; Wang, O.; Zhang, R.; Owens, A.; and Efros, A. A. 2020a. CNN-Generated Images Are Surprisingly Easy to Spot... for Now. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7794–7803.

Wang, X.; Yao, T.; Ding, S.; and Ma, L. 2020b. Face Manipulation Detection via Auxiliary Supervision. In *International Conference on Neural Information Processing*, 313–324. Springer.

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Póczos, B.; Salakhutdinov, R.; and Smola, A. J. 2017. Deep Sets. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, 3391–3401.

Zhang, L.; Xu, D.; Arnab, A.; and Torr, P. H. 2020. Dynamic graph message passing networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3726–3735.

Zhang, X.; Karaman, S.; and Chang, S.-F. 2019. Detecting and simulating artifacts in gan fake images. In *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, 1–6. IEEE.

Zhao, H.; Cui, H.; and Zhou, W. 2020. Kaggle DFDC. https://github.com/cuihaoleo/kaggle-dfdc.

Zhao, H.; Zhou, W.; Chen, D.; Wei, T.; Zhang, W.; and Yu, N. 2021a. Multi-attentional Deepfake Detection. *arXiv preprint arXiv:2103.02406*.

Zhao, T.; Xu, X.; Xu, M.; Ding, H.; Xiong, Y.; and Xia, W. 2021b. Learning Self-Consistency for Deepfake Detection. *To appear in ICCV 2021*.