Forensic Symmetry for DeepFakes

Gen Li, Xianfeng Zhao, *Senior Member, IEEE*, and Yun Cao, *Member, IEEE*

***Abstract*— In this paper, we propose a new DeepFakes forensics approach called forensic symmetry, which determines whether two symmetrical face patches contain the same or different natural features. To do this, we propose a multi-stream learning structure composed of two feature extractors. The first feature extractor obtains symmetry feature from the front face images. The second feature extractor obtains similarity feature from the side face images. Symmetry feature and similarity feature are collectively called natural feature. Forensic symmetry system maps the pair of symmetrical face patches into the angular hyperspace to quantify the difference of their natural features. The greater the difference of natural features, the higher the tamper probability of face images. The heuristic prediction algorithm is designed to compute the tamper probability of DeepFakes at video level. A series of experiments are carried out to evaluate the effectiveness of our proposed forensic symmetry system. Experimental results show that our approach is effective for DeepFakes detection under the scenarios of homologous detection, heterogeneous detection, and re-compression detection.**

***Index Terms*— DeepFakes forensics, face symmetry detection, multi-stream learning, multi-margin loss, heuristic prediction.**

I. Introduction

DEEPFAKES are synthetic multimedia in which the identity or expression of a target subject is replaced by that of another source subject. DeepFake technologies bring potential threats and concerns to everyone. Illegal information such as fake news and manipulated pornographic videos may cause a profound distrust in society, threaten national and political security, and violate individual rights and interests [1]. Recently, the DeepFakes generation applications such as *ZAO*, *FakeApp*, and *DeepFaceLab* have been released opening the door to anyone to create fake face videos [2], without too much professional experience in the field needed. Several independent implementations have been used for creating DeepFake videos in scale, such as *faceswap* and *faceswap-GAN* [3], [4]. The manipulation process is automatic and runs with little manual intervention. Some high quality DeepFake videos generated through those methods are hardly to distinguish by the eyes. Therefore, it is of great importance to develop effective methods for DeepFakes forensics.

In response to increasingly sophisticated and realistic manipulated face images and videos, there have been immense research efforts in developing effective DeepFakes detectors. As the release of large-scale DeepFake datasets, it enables the training of Deep Neural Networks (DNN) to identify DeepFakes with high accuracy. For example, exploiting visual artifacts, head poses inconsistencies, eye blinking patterns, optical flow motion, reconstruction distortion, and multi-domain features to expose DeepFakes [5], [6], [7], [8], [9], [10], [11], [12], [13], [14].

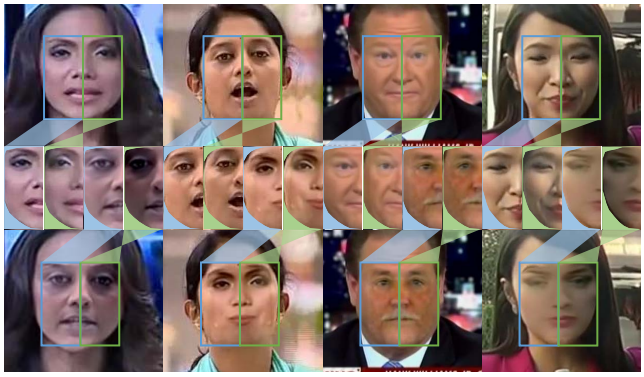

Fig. 1. Examples of real face images (top row) and DeepFake images (bottom row). As shown in the pictures, there are unnatural traces and visual artifacts in symmetrical face areas or face boundaries of DeepFake images.

However, there are three main drawbacks to many of the existing DeepFakes forensics methods. First is that many detection models were trained with the whole face images in undifferentiated ways. While, the facial inner attributes such as face pose considered in our method, were ignored during learning progress. The second drawback is that many investigators assess the performance of their methods at image level purely. As a matter of fact, the DeepFake videos became more widespread on social networks than DeepFake images. Thus, it is necessary to introduce a new computing strategy to improve the performance of the detection models at video level. The third drawback is that most forensics methods use traditional classification loss function to train classifier. However, classification loss cannot well vectorize the difference between inter-class samples.

To solve the above drawbacks, in this paper, we propose a novel DeepFakes forensics approach called *forensic symmetry*, which determines whether two symmetrical face patches contain the same or different natural features. The motivation of our approach is that there are unnatural traces and visual artifacts such as color, texture, and illumination in symmetrical face areas or face boundaries of DeepFake images if we look and compare closely. Examples of real face images and DeepFake images are shown in Fig. 1. To magnify those inconsistencies, we propose a multi-stream learning structure composed of two feature extractors. First, we detect face area and extract facial landmarks. Then, we compute the transform

Manuscript received 14 June 2022; revised 26 October 2022 and 30 December 2022; accepted 31 December 2022. Date of publication 9 January 2023; date of current version 16 January 2023. This work was supported by the National Key Technology Research and Development Program under Grant 2020AAA0140000. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Husrev Taha Sencar. *(Corresponding author: Xianfeng Zhao.)*

The authors are with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100195, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100195, China (e-mail: ligen1@iie.ac.cn; zhaoxianfeng@iie.ac.cn; caoyun@iie.ac.cn).

Digital Object Identifier 10.1109/TIFS.2023.3235579

matrices to align the face to a standard configuration. Based on the face pose estimation method, we divide the face images into *front face dataset* and *side face dataset*. In the first feature extractor, we extract *symmetry feature* from two symmetrical face patches in the front face dataset. In the second feature extractor, we extract *similarity feature* from a side face patch and an incomplete face patch in the side face dataset. Symmetry feature and similarity feature are collectively called *natural feature* in this paper. During the natural feature extraction process, we adopt a novel multi-margin angular loss function to optimize feature extractor. After that, we map the natural features into angular hyperspace to quantify the difference of natural features of symmetrical face patches. The greater the difference of natural features, the higher the tamper probability of face images. Lastly, we adopt a specially designed heuristic prediction algorithm to compute the tamper probability of DeepFakes at video level. Compared with our previous work [15], in this paper, we innovated the learning structure of natural features and the prediction algorithm of face video tamper probability. We added various ablation studies regarding network branch, loss function, and prediction algorithm. And, we presented abundant visualization analysis.

We experimentally evaluate the effectiveness of our proposed forensic symmetry system on five DeepFake datasets. The experimental results show that our approach has very good performance on DeepFakes forensics under multiple scenarios, including homologous detection, heterogeneous detection, re-compression detection, occlusion detection, and multi-face detection. The main contributions of our work are summarized as follows:

- **Face symmetry detection**. We exploited the difference between symmetrical face areas to expose DeepFake images and videos.
- **Multi-stream learning structure**. We proposed a multi-stream learning structure to extract natural features from symmetrical face patches.
- **Multi-margin angular loss function**. We designed a multi-margin angular loss function to enhance the discrimination of detection model.
- **Heuristic prediction algorithm**. We designed a heuristic prediction algorithm to compute the tamper probability of DeepFakes at video level.

The remainder of this paper is organized as follows. In Section II, we present an overview about DeepFakes forensics methods. In Section III, we describe the overall pipeline of our proposed forensic symmetry system in detail. In Section IV, we assess our approach through a series of experiments. Lastly, Section V concludes our work.

II. Related Work

In this section, we briefly review the DeepFakes forensics methods. The DeepFakes forensics methods are divided into single-stream based methods and multi-stream based methods.

A. Single-Stream Based Methods

Single-stream based methods mostly exploit various types of DNN to expose manipulation traces and inconsistencies exhibited in the spatial domain and time domain of DeepFakes.

In the aspect of spatial domain, Matern et al. [5] exploited visual artifacts such as eye color, missing reflections, and missing details in the eyes and teeth areas to expose DeepFakes. Li et al. [6] observed that some DeepFake algorithms can only generate limited resolution face images. Thus, the authors proposed a method based on Convolutional Neural Networks (CNN) to detect such peculiar traces. Yang et al. [7] performed a study based on the difference between head poses estimated using a full set of facial landmarks and a part of facial landmarks to differentiate DeepFake videos from real videos. Li et al. [16] focused on face boundaries detection. Guarnera et al. [11] taken the convolutional trace left in DeepFakes generation process as detection clue. Wang et al. [17] taken layer-wise neuron behavior as distinguishable features, as opposed to final-layer neuron output compared to the other learning based method. Dang et al. [18] adopted CNN and attention mechanisms to generate the feature maps of the classifier model. To overcome new DeepFake data scarcity limitation, Khalid et al. [19] taken the Variational AutoEncoder (VAE) as backbone network to train one-class detection model. Besides, CapsuleNet, MesoNet, MesoInceptionNet, XceptionNet, and EfficientNet were also introduced to detect DeepFakes [12], [20], [21], [22], [23].

In the aspect of time domain, Korshunov et al. [24] adopted Recurrent Neural Networks (RNN) based on Long Short Term Memory (LSTM) to detect the inconsistencies between lip movements and audio speech. Delp et al. [25] and Sabir et al. [26] proposed two similar detection systems which learns and infers in an end-to-end manner and outputs the probability of the video being a DeepFake one. Montserrat et al. [27] described a method to automatically weight different face regions by the Automatic Face Weighting (AFW) layer and the Gated Recurrent Unit (GRU). Amerini et al. [28] exploited optical flow field dissimilarities to discriminate original and DeepFake videos. Li et al. [8] proposed an algorithm to analyze changes in the eye blinking patterns, which has a specific frequency and duration in humans and not replicated in early DeepFake videos. Agarwal et al. [9] proposed a method based on both facial expressions and head movements across frames. Fernandes et al. [10] trained the Neural Ordinary Differential Equations (Neural-ODE) model to extract and detect the heart rate of fake videos. Besides, the motion information of face were also introduced to detect DeepFake videos [29], [30].

B. Multi-Stream Based Methods

Multi-stream based methods mostly exploit the fusion of multi-level or multi-domain features to expose DeepFakes. Nguyen et al. [31] designed an encoder and a Y-shaped decoder to simultaneously detect manipulated images and locate the manipulated regions. Tolosana et al. [32] provided an evaluation framework consists of entire face model, eyes model, nose model, mouth model, and rest model. Similarity, Kumar et al. [33] proposed a multi-stream network which divides face images into four local face regions through equal grids and feds them into Deep Residual Networks

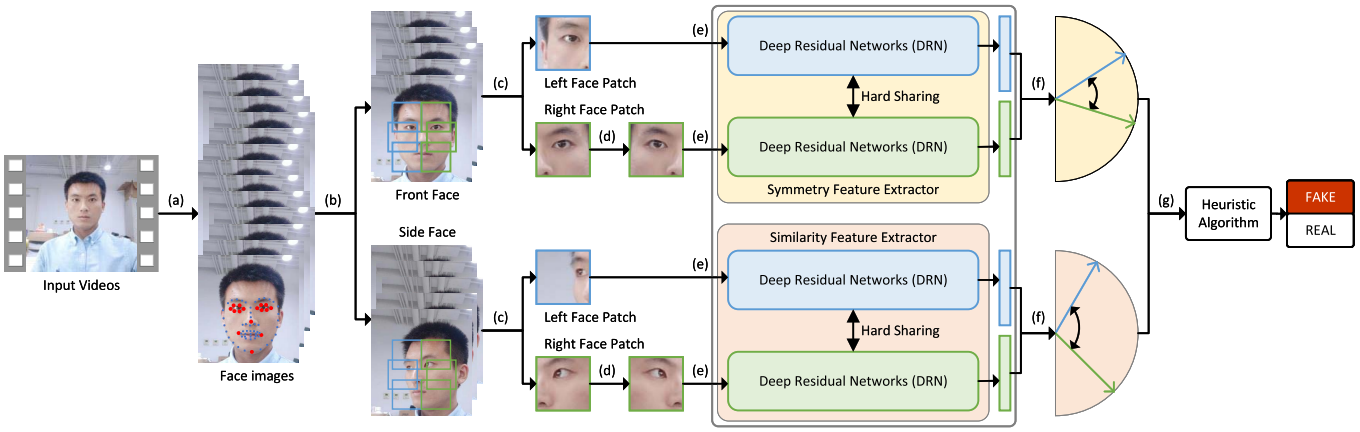

Fig. 2. Overall pipeline of our proposed forensic symmetry system. (a) Face detection and facial landmarks localization, face pose estimation, face alignment and face resize. (b) Data partition. (c) Patches cropping. (d) Right face patch mirroring. (e) Natural feature extraction. (f) Feature mapping. (g) Tamper probability computing.

(DRN) [34] respectively to extract multi-level features. Zhou et al. [35] trained a face classification model to expose tampering artifacts in the first stream and trained a triplet network to capture local noise residuals in the second stream. Besides, using multi-domain fusion features such as spatial domain feature, frequency domain feature, and temporal domain feature based on multi-stream network to expose DeepFakes has been proved effective [14], [36], [37], [38], [39], [40], [41], [42].

III. Proposed Approach

In this section, we describe our proposed forensic symmetry system from five main parts. They are data pre-processing, neural network architecture, angular loss function, training methodology, and heuristic prediction algorithm. The overall pipeline of our approach is shown in Fig. 2.

A. Data Pre-Processing

We will introduce our data pre-processing method from the aspects of face detection and facial landmarks localization, face pose estimation, face alignment and face resize, data partition and patches cropping.

1) Face Detection and Landmarks Localization: The first step of our data pre-processing method is detecting face area and locating facial landmarks through the open library *dlib* [43]. This library is a collection of miscellaneous algorithms in machine learning, computer vision, image processing, and linear algebra. The *dlib* face detector can produce both face bounding boxes and facial landmarks simultaneously. The facial landmarks are locations on the face carrying important structural information including the tip of cheek, eyebrows, nose, eyes, and mouth. Fig. 3 (a) represents an example of 68 facial landmarks over a sample face image, where

- *Cheek*: landmarks from number 0 to 16,
- *Left eyebrow*: landmarks from number 17 to 21,
- *Right eyebrow*: landmarks from number 22 to 26,
- *Nose*: landmarks from number 27 to 35,
- *Left eye*: landmarks from number 36 to 41,
- *Right eye*: landmarks from number 42 to 47,
- *Mouth*: landmarks from number 48 to 67.

We choose this face detector to detect face area and to locate facial landmarks for the reason that it provides good detection performance and reliable landmarks even for variational face pose, which plays an important part in the following data processing steps.

2) Face Pose Estimation: In the field of computer vision, the face pose refers to its relative orientation and position with respect to a camera. A three-dimensional (3D) face has only two kinds of motions with respect to a camera when we regard the 3D face as a rigid object roughly. The two kinds of motions are translation and rotation. Moving the face from current 3D location to a new 3D location in a direction is called translation. Rotating the face about an axis is called Rotation. The 3D facial landmarks in camera coordinates can be projected onto the image coordinates using the intrinsic parameters of the camera such as focal length and optical center. In this paper, we use three coordinates, world coordinates, image coordinates, and camera coordinates, to estimate 3D face pose based on the plane face image.

In order to visually understand the coordinate transformation of facial landmarks among the above three coordinates, we draw a perspective as shown in Fig. 4. Taking one of the facial landmarks as an example. The world coordinate P^w is $[U, V, W]^T$. The image coordinate P^i is $(x, y)^T$. The camera coordinate P^c is $[X, Y, Z]^T$. The coordinate transformation among P^w, P^i, and P^c can be formulated as

$$\begin{cases} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \vec{r} \begin{bmatrix} U \\ V \\ W \end{bmatrix} + \vec{t} \\ s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = F \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \end{cases}, \tag{1}$$

where $\vec{r}$ is the rotation vector whose dimension is 1×3. $\vec{t}$ is the translation vector whose dimension is 3×1. s is an unknown scaling factor for the reason that we do not know the depth

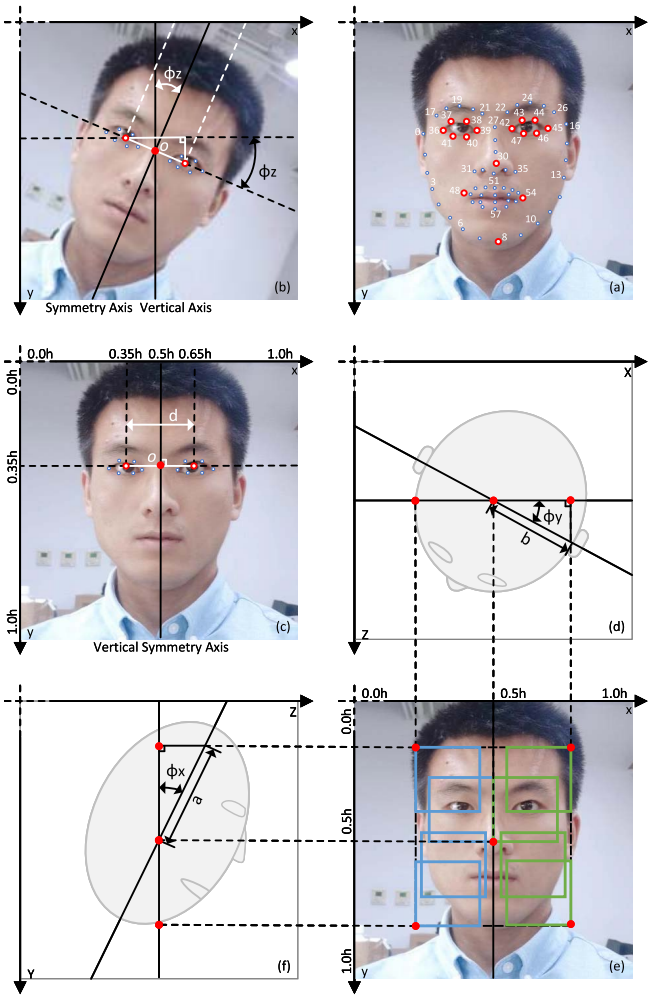

Fig. 3. Illustration of our data pre-processing method. (a) Example of 68 facial landmarks. (b) Example face image before the treatment of face alignment and face resize, for showing the roll angle ϕ_z. (c) Example face image after the treatment of face alignment and face resize. (d) Top view of 3D face image, for showing the yaw angle ϕ_y. (e) Front view of 3D face image, for showing the process of face patches cropping. (f) Left view of 3D face image, for showing the pitch angle ϕ_x.

of face image. F is the coefficient matrix whose dimension is 3×3, which is defined as

$$F = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2}$$

where (c_x, c_y) is the optical center. f_x is the focal length in x direction. f_y is the focal length in y directions.

Next, s, $\vec{r}$, and $\vec{t}$ can be estimated by multiple pairs of coordinates (P^i, P^w) when we know the coefficient matrix F. In this paper, the optical center (c_x, c_y) is approximated by the center of the video frame. The focal lengths f_x and f_y are approximated by the width of the video frame. The image coordinate P^i can be extracted through the face detector. The world coordinate P^w can be approximated by a standard 3D average face model. To improve computational efficiency, we use six facial landmarks to estimate 3D face pose. The locations and world coordinates of six facial landmarks in the 3D average face model are

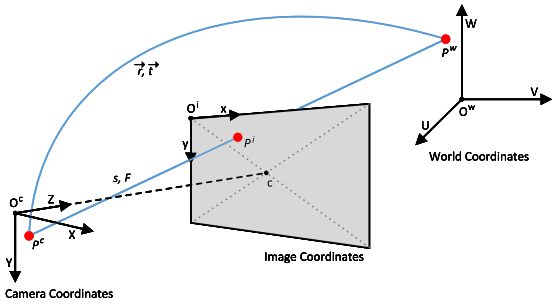

Fig. 4. Perspective of coordinate transformation among world coordinates, image coordinates, and camera coordinates. P^w is world coordinate. P^i is image coordinate. P^c is camera coordinate.

- *Left corner of left eye*: (-225.0, 170.0, -135.0),
- *Right corner of right eye*: (225.0, 170.0, -135.0),
- *Tip of nose*: (0.0, 0.0, 0.0),
- *Left corner of mouth*: (-150.0, -150.0, -125.0),
- *Right corner of mouth*: (150.0, -150.0, -125.0),
- *Tip of chin*: (0.0, -330.0, -65.0).

Then, the objective function of 3D face pose estimation can be formulated as

$$f\left(\vec{r}, \vec{t}, s\right) = \sum_{j \in J} \left\| s \begin{bmatrix} x_j \\ y_j \\ 1 \end{bmatrix} - F\left(\vec{r} \begin{bmatrix} U_j \\ V_j \\ W_j \end{bmatrix} + \vec{t} \right) \right\|^2, \tag{3}$$

where the set $J = \{36, 45, 30, 48, 54, 8\}$ stands for the indexes of facial landmarks. After the iterative optimization process of Eq. 3, we can obtain the optimal value of rotation vector $\vec{r}$. And, the array of Euler angles Φ of 3D face pose can be figured out by the formulas as

$$\begin{cases} \vec{r} = \theta \vec{e} \\ = \|\vec{r}\|^2 \left[e_x, e_y, e_z \right] \\ Q = \left[q_k, q_x, q_y, q_z \right] \\ = \left[\cos\frac{\theta}{2}, e_x \sin\frac{\theta}{2}, e_y \sin\frac{\theta}{2}, e_z \sin\frac{\theta}{2} \right] \\ \Phi = \begin{bmatrix} \phi_x \\ \phi_y \\ \phi_z \end{bmatrix} \\ = \begin{bmatrix} \arctan \frac{2(q_k q_x + q_y q_z)}{1 - 2({q_x}^2 + {q_y}^2)} \\ \arcsin\left(2\left(q_k q_y - q_z q_x\right)\right) \\ \arctan \frac{2(q_k q_z + q_x q_y)}{1 - 2({q_y}^2 + {q_z}^2)} \end{bmatrix} \end{cases}, \tag{4}$$

where $\vec{e}$ is the unit vector of rotation axis. θ is the rotation angle. e_x, e_y, and e_z are the components of $\vec{e}$ in the directions of X-axis, Y-axis, and Z-axis. Q represents an axis of rotation $\vec{e}$ and a rotation angle θ around that axis. Q is a quaternion. ϕ_x, ϕ_y, and ϕ_z are the angle components of Φ in the directions of X-axis, Y-axis, and Z-axis.

Up to now, the estimates of Euler angles ϕ_x, ϕ_y, and ϕ_z of 3D face pose are figured out indeed. They are the rotation of the world coordinates regarding the camera coordinates. The result of this step is crucial to the following steps of face alignment and face resize, and patches cropping.

3) Face Alignment and Face Resize: The purpose of face alignment is keeping the symmetry axis of face overlaps with

the vertical axis of the image. Face alignment can be achieved by rotating face image in a certain angle about a fixed rotation center. The purpose of face resize is correcting the face area to a unified size. Face resize can be achieved by scaling face image in a certain scaling factor after face alignment.

Fig. 3 (b) and (c) show an example face image before and after the processes of face alignment and face resize. In our method, the rotation angle is the roll angle ϕ_z which has been figured out in the step of face pose estimation. The rotation center is the center of two eyes. The scaling factor is the ratio of expected value to actual value of the distance of the center of two eyes. Based on the image coordinates of facial landmarks P^i, the rotation center (o_x, o_y) and scaling factor m can be calculated by the formulas as

$$\begin{cases} (l_x, l_y) = \left(\frac{1}{6}\sum_{j=36}^{41} x_j, \frac{1}{6}\sum_{j=36}^{41} y_j\right) \\ (r_x, r_y) = \left(\frac{1}{6}\sum_{j=42}^{47} x_j, \frac{1}{6}\sum_{j=42}^{47} y_j\right) \\ (o_x, o_y) = \left(\frac{l_x+r_x}{2}, \frac{l_y+r_y}{2}\right) \\ m = \frac{d}{\sqrt{(r_x-l_x)^2+(r_y-l_y)^2}} \end{cases}, \tag{5}$$

where (l_x, l_y) is the center of left eye. (r_x, r_y) is the center of right eye. d is the expected value of the distance of the center of two eyes. In our method, the dimension of unified square face image is $h \times h \times 3$ and h is 256 in pixels. The horizontal distance from the center of left eye to the center of right eye is $0.3h$, which is equal to d. The horizontal distance from the left border of face image to the center of left eye is $0.35h$. The horizontal distance from the right border of face image to the center of right eye is $0.35h$. The vertical distance from the top border of face image to the center of two eyes is $0.35h$. The vertical distance from the bottom border of face image to the center of two eyes is $0.65h$. The unified square face images will be used in the next step.

4) Data Partition and Patches Cropping: After the step of face pose estimation, we can figure out the pitch angle ϕ_x about X-axis, the yaw angle ϕ_y about Y-axis, and the roll angle ϕ_z about Z-axis of 3D face pose in the camera coordinates. The roll angle ϕ_z goes to zero after the process of face alignment. In order to crop the symmetrical face patches, the pitch angle ϕ_x and the yaw angle ϕ_y need to be considered. The pitch angle ϕ_x determines the locations of the top border and the bottom border of face area. The larger the absolute value of ϕ_x, the smaller the face area is. In the same way, the yaw angle ϕ_y determines the locations of the left border and the right border of face area. The larger the absolute value of ϕ_y, the smaller the face area is.

Fig. 3 (d), (e), and (f) show the way of symmetrical face patches cropping. Taking the center of face image as the center of face area. Taking the top left corner of face image as coordinate origin. The location of the face area is

$$\begin{cases} \left(l_x^t, l_y^t\right) = \left(\frac{1}{2}h - b\cos\phi_y, \frac{1}{2}h - a\cos\phi_x\right) \\ \left(r_x^t, r_y^t\right) = \left(\frac{1}{2}h + b\cos\phi_y, \frac{1}{2}h - a\cos\phi_x\right) \\ \left(l_x^b, l_y^b\right) = \left(\frac{1}{2}h - b\cos\phi_y, \frac{1}{2}h + a\cos\phi_x\right) \\ \left(r_x^b, r_y^b\right) = \left(\frac{1}{2}h + b\cos\phi_y, \frac{1}{2}h + a\cos\phi_x\right) \end{cases}, \tag{6}$$

where a is the half-height of face. b is the half-width of face. In our method, a is 97 and b is 83 in pixels. $\left(l_x^t, l_y^t\right)$, $\left(r_x^t, r_y^t\right)$, $\left(l_x^b, l_y^b\right)$, and $\left(r_x^b, r_y^b\right)$ are the locations of top left corner, top right corner, bottom left corner, and bottom right corner of face area. The size of face area is different in different face poses. In particular, the size of face area is zero when the pitch angle ϕ_x is 0.5π in clockwise direction or -0.5π in anticlockwise direction. In this case, no face could be detected by face detector. The size of left face area is zero when the yaw angle ϕ_y is 0.5π in clockwise direction. The size of right face area is zero when the yaw angle ϕ_y is -0.5π in anticlockwise direction.

The length of the side of face area should be considered when we crop the symmetrical face patches. We can't obtain the complete symmetrical face patches when the half-width or half-height of face area smaller than the length of the side of face patch. In our method, the face images are divided into the front face image or the side face image according to the pitch angle ϕ_x and the yaw angle ϕ_y. The data partition method is

$$\begin{cases} D_f, & \left(|\phi_x| \le \arccos\frac{p}{a}\right) \,\&\, \left(|\phi_y| \le \arccos\frac{p}{b}\right) \\ D_s, & \left(|\phi_x| > \arccos\frac{p}{a}\right) \,\|\, \left(|\phi_y| > \arccos\frac{p}{b}\right) \end{cases}, \tag{7}$$

where D_f stands for the front face dataset. D_s stands for the side face dataset. p is the length of the side of face patch. In our method, p is 64 in pixels. The dimension of face patch is $p \times p \times 3$. The thresholds of ϕ_x and ϕ_y are

$$\begin{cases} \phi_x^t = \pm\arccos\frac{64}{97} \approx \pm\frac{5}{18}\pi \\ \phi_y^t = \pm\arccos\frac{64}{83} \approx \pm\frac{4}{18}\pi \end{cases}, \tag{8}$$

where ϕ_x^t and ϕ_y^t are the thresholds of ϕ_x and ϕ_y, which decide the fact that the face images belong to the front face dataset or the side face dataset. To be specific, the face image is part of the front face dataset when the absolute value of pitch angle ϕ_x no greater than $|\phi_x^t|$ and the absolute value of yaw angle ϕ_y no greater than $|\phi_y^t|$. Conversely, the face image is part of the side face dataset when the absolute value of pitch angle ϕ_x greater than $|\phi_x^t|$ or the absolute value of yaw angle ϕ_y greater than $|\phi_y^t|$.

To crop the symmetrical face patches from the front face dataset D_f, we randomly crop face area through symmetrical sliding windows about the symmetry axis of face image. To crop a side face patch and an incomplete face patch from the side face dataset D_s, we randomly crop the maximum face area through symmetrical sliding windows about the symmetry axis of face image. In this case, the height and the width of the maximum face area are $2a$ and $2b$. In order to retain the same structural information of the left face path and the right face patch, we mirror the right face patch.

Finally, we obtain two sub face datasets of front face dataset and side face dataset. In our method, the front face dataset is used to train the symmetry feature extractor. The side face dataset is used to train the similarity feature extractor. Compared with the other data pre-processing methods that they only focus on the whole face area or a part of face area, our data pre-processing method focuses on the symmetrical face area, which is a novelty of our method.

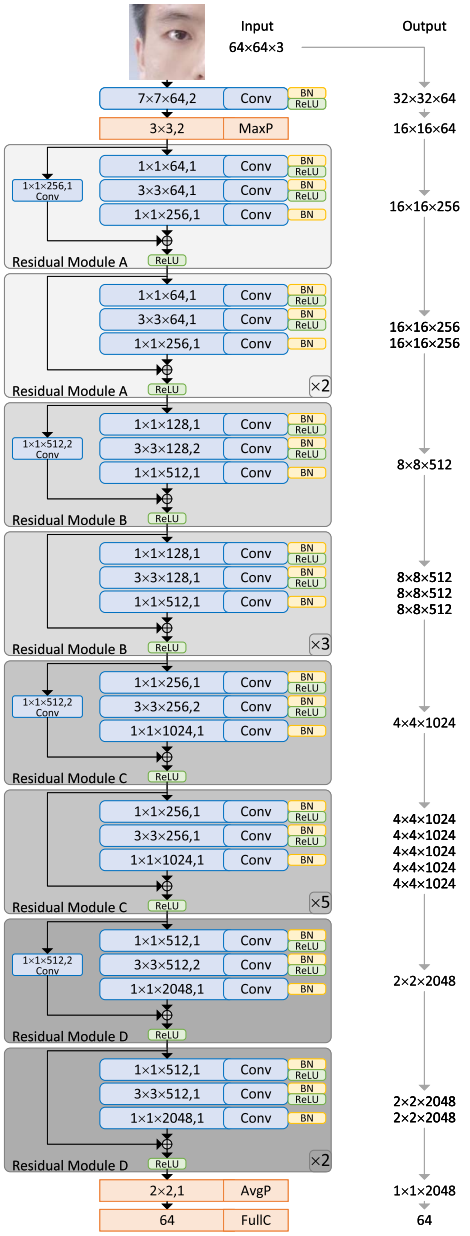

Fig. 5. The backbone network used in our proposed forensics symmetry system. Conv, MaxP, AvgP, FullC, BN, and ReLU stand for Convolutional layer, Max Pooling layer, Average Pooling layer, Fully Connected layer, Batch Normalization layer, and Rectified Linear Unit.

B. Neural Network Architecture

To build the natural feature extractors, we take DRN as backbone network. DRN has been proposed to deepen the network and solve the problems of gradient disappearance and gradient explosion in the training process of deep CNN model. Fig. 5 shows the architecture of DRN used in our method. The input of backbone network is the face patch. The output of backbone network is the feature vector, which stands for the natural feature of the input face patch.

In our method, we use four identical DRN to extract the natural features. Two of them are used to extract symmetry features from two input symmetrical face patches cropped from the front face image, which is called symmetry feature extractor. The other two DRN are used to extract similarity features from a side face patch and an incomplete face patch cropped from the side face image, which is called similarity feature extractor. The symmetry feature extractor and the similarity feature extractor are collectively called natural feature extractor. Hard sharing is used in the training process of natural feature extractors for sharing parameters.

Next, we map the natural features into angular hyperspace to quantify their difference. The details of our designed multi-margin angular loss function will be described in the next part.

C. Angular Loss Function

An appropriate loss function can enhance the discrimination of detection model. In order to quantify the difference of natural features in angular hyperspace, the loss function we need is metric loss function instead of classification loss function. Accordingly, we designed a multi-margin angular loss function, which is presented as

$$L = \frac{1}{n}\sum_{i=1}^{n} \max\left(\left(1 - y_i\right)\left(\beta_i - \alpha_0\right) + y_i\left(\alpha_1 - \beta_i\right), 0\right), \quad (9)$$

where n is the batch size. y_i is 0 if the i-th sample is cropped from the real sample. y_i is 1 if the i-th sample is cropped from the fake sample. α_0 and α_1 are two additive angular margin penalties. β_i is the angular distance of natural features extracted from symmetrical face patches of i-th sample, which is formulated as

$$\beta_i = \arccos \frac{\vec{f}_i^{\,l} \cdot \vec{f}_i^{\,r}}{\left\|\vec{f}_i^{\,l}\right\|^2 \left\|\vec{f}_i^{\,r}\right\|^2}, \quad (10)$$

where $\vec{f}_i^{\,l}$ is the natural feature vector of the left face patch of i-th sample. $\vec{f}_i^{\,r}$ is the natural feature vector of the right face patch of i-th sample.

From our designed multi-margin angular loss function, we can see that given an input real sample, a part of angular distance loss $(\beta_i - \alpha_0)$ is activated, the other part of angular distance loss $(\alpha_1 - \beta_i)$ remains silent mode. The angular distance loss will be zeroed when β_i is smaller than additive angular margin penalty α_0. Similarly, when the input is fake sample, a part of angular distance loss $(\alpha_1 - \beta_i)$ is activated, the other part of angular distance loss $(\beta_i - \alpha_0)$ remains silent mode. The angular distance loss will be zeroed when β_i is greater than additive angular margin penalty α_1.

There are two important hyper-parameters α_0 and α_1 mentioned times in this part. They decide the borders of angular distance between real sample and fake sample. In the section of Experimental Evaluation, we experimentally compared the performance of our approach under the different settings of hyper-parameters α_0 and α_1. The experimental processes and results will be described in more details in the following.

D. Training Methodology

In this section, we describe the training methodology of natural feature extraction models of our proposed forensic

symmetry system. To be sure, the natural feature extraction models consist of symmetry feature extraction model and similarity feature extraction model. The symmetry feature extraction model is trained with symmetry feature extractor on the front face dataset. The similarity feature extraction model is trained with similarity feature extractor on the side face dataset.

At the beginning of training, the parameters of backbone network are initialized with the ImageNet pre-trained weights, except the last fully connected layer. We iteratively train natural feature extraction models through Adaptive moment (Adam) optimizer. To balance the scale of training data and the speed of training process, the same number of real samples and fake samples are randomly selected to train natural feature extraction models during different epochs. The scale of training dataset is detailed in the section of Datasets. The initial learning rate is 10^{-4}. The learning rate is one-tenth of the original when the angular loss no longer fall during three consecutive epochs. The minimum of learning rate is 10^{-8}. All natural feature extraction models are trained through the deep learning framework of Pytorch and single graphics card of NVDIA Tesla-V100. After the training process, we can obtain the natural feature extraction model.

E. Heuristic Prediction Algorithm

In this section, we describe the computing method of tamper probability of DeepFakes at image level and video level.

At image level, the pitch angle and the yaw angle of face can be computed through Eq. 4. According to Eq. 7 and Eq. 8, we choose one of the feature extraction models to extract natural feature from face image. Specifically, when the values of the pitch angle and the yaw angle stay within the range of the front face dataset, we extract natural feature by symmetry feature extraction model. Otherwise, when the values of the pitch angle and the yaw angle stay within the range of the side face dataset, we extract natural feature by similarity feature extraction model. In order to decrease the disturbance bring by image local noise, face image tamper probability is the mean tamper probability of n pairs of symmetrical face patches. Face image tamper probability p^{img} is formulated as

$$p^{img} = \frac{1}{n}\sum_{j=1}^{n}\frac{\min\left(\max\left(\beta_j, \alpha_0\right), \alpha_1\right) - \alpha_0}{\alpha_1 - \alpha_0}, \tag{11}$$

where β_j is the angular distance of natural features extracted from j-th pair of symmetrical face patches. α_0 and α_1 are two angular margin penalties which have been mentioned in Eq. 9. Besides, the number of the pair of symmetrical face patches n is 10 in our method.

At video level, we designed a heuristic prediction algorithm to compute face video tamper probability based on the computing method of face image tamper probability. First, we extract face frames from the face video and compute the list of face image tamper probability P through Eq. 11. Then, we introduce two threshold parameters p^t and r^t where $p^t \in [0.5, 1.0]$ stands for a constant of probability and $r^t \in [0.5, 1.0]$ stands for a constant of proportion. When the proportion of face images whose tamper probability no smaller than p^t and greater than r^t, the face video tamper probability is the mean tamper probability of all fake face images. Or else, when the proportion of face images whose tamper probability no greater than $\left(1 - p^t\right)$ and greater than r^t, the face video tamper probability is the mean tamper probability of all real face images. If above two conditions are not met, the face video tamper probability is the mean tamper probability of all face images. Pseudo code outlining the computing process of face video tamper probability is given in Algorithm 1.

Algorithm 1 Heuristic Prediction Algorithm for Computing Face Video Tamper Probability.

Input: List of face image tamper probability P, probability threshold p^t, proportion threshold r^t.
Output: Face video tamper probability p^{vie}.

initial $n_0 = 0$, $n_1 = 0$, $n_2 = 0$
initial $s_0 = 0$, $s_1 = 0$, $s_2 = 0$
for $p \in P$ **do**
  if $p \geq p^t$ **then**
    $n_0 = n_0 + 1$
    $s_0 = s_0 + p$
  else if $p \leq \left(1 - p^t\right)$ **then**
    $n_1 = n_1 + 1$
    $s_1 = s_1 + p$
  else
    $n_2 = n_2 + 1$
    $s_2 = s_2 + p$
  end if
end for
$r_0 = \frac{n_0}{n_0+n_1+n_2}$
$r_1 = \frac{n_1}{n_0+n_1+n_2}$
if $r_0 \geq r^t$ **then**
  $p^{vie} = \frac{s_0}{n_0}$
else if $r_1 \geq r^t$ **then**
  $p^{vie} = \frac{s_1}{n_1}$
else
  $p^{vie} = \frac{s_0+s_1+s_2}{n_0+n_1+n_2}$
end if
return p^{vie}

To obtain the optimal probability threshold p^t and proportion threshold r^t, we experimentally compared the performance of our proposed forensic symmetry system under different threshold combinations. The experimental results will be described in the section of Experimental Evaluation.

IV. Experimental Evaluation

In this section, we conduct a series of experiments to evaluate the effectiveness of our proposed forensic symmetry system under multiple scenarios. First, we describe the information of five DeepFake datasets used in our experiments. Then, we evaluate the performance of our approach under the scenarios of homologous detection and heterogeneous detection respectively. Additionally, we compare the impact of different loss functions and prediction algorithms. In order to extend our work to the real case of online environment, we also

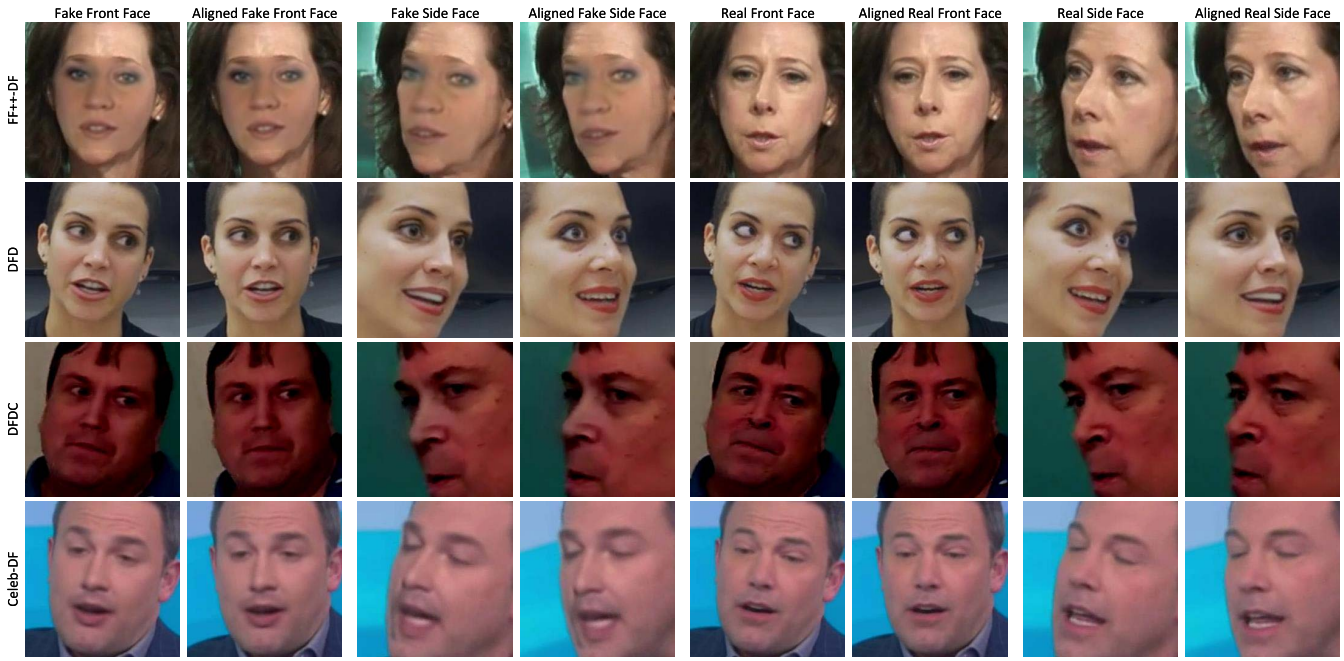

Fig. 6. Examples of fake front face images, fake side face images, real front face images, real side face images, and their aligned face images of FF++-DF, DFD, DFDC, and Celeb-DF datasets.

TABLE I
THE SCALE OF DEEPFAKE DATASETS USED IN THE TRAINING AND TESTING PROCESSES OF OUR EXPERIMENTS

Dataset	Real					Fake				
	Video	Frame	Front Face	Side Face	All Face	Video	Frame	Front Face	Side Face	All Face
DF-TIMIT	320	103.5k	81.5k	8.7k	90.2k	320	34.0k	34.0k	0.0k	34.0k
FF++-DF	1000	509.9k	451.7k	41.8k	493.5k	1000	509.9k	466.2k	25.3k	491.5k
DFD	363	315.4k	212.6k	51.1k	263.7k	363	265.4k	192.9k	43.1k	236.0k
DFDC	1000	300.0k	225.6k	32.5k	258.1k	1000	300.0k	226.4k	27.1k	253.5k
Celeb-DF	588	237.0k	160.1k	62.2k	222.3k	588	220.7k	153.9k	64.9k	218.8k

discuss the robustness of our approach under re-compression detection scenario. Finally, we give visualization analysis.

A. Datasets

DeepFakes forensics methods based on deep learning require large-scale training data. Currently, some DeepFake datasets have been constructed and opened by the investigators. Five frequently used DeepFake datasets are *DeepFake-TIMIT* (DF-TIMIT) dataset [24], *FaceForensics++-Deepfakes* (FF++-DF) dataset [44], *DeepFake Detection* (DFD) dataset [45], *DeepFake Detection Challenge* (DFDC) dataset [46] and *Celeb-DF* (Celeb-DF) dataset [47]. The scale of DeepFake datasets used in the training process and testing process of our experiments is shown in Table I.

Based on the extracted face images of each dataset, we divide them into the front face datasets and the side face datasets according to the face pose. The examples of fake front face images, fake side face images, real front face images, real side face images, and their aligned face images of DeepFake datasets are shown in Fig. 6. The detailed analysis of face pose of each dataset is illustrated in Fig. 7. As can be seen, there are more complex face pose in the yaw angle than the pitch angle in all DeepFake datasets. The ratio of front face images highlighted in blue line to side face images highlighted in yellow line is different with different DeepFake datasets. In particular, there are almost no side face images in DF-TIMIT dataset. The face pose of FF++-DF dataset is more compact than DFD, DFDC, and Celeb-DF datasets. In our experiments, training data consists of five in six face images of each dataset. Testing data consists of the other one-sixth face images of each dataset. The identities of training data and testing data are not overlap each other, which is important to perform a fair evaluation and to predict the generalization capacity of our approach against unseen identities.

B. Data Source Comparison

In this section, we evaluate the performance of our approach under homologous detection scenario and heterogeneous detection scenario. And, we compare our approach with the existing DeepFake detection methods. The five natural feature extraction models are trained on five DeepFake datasets respectively. It should be noted that we train only the symmetry feature extraction model using DF-TIMIT front face dataset for the reason that there are no fake samples in

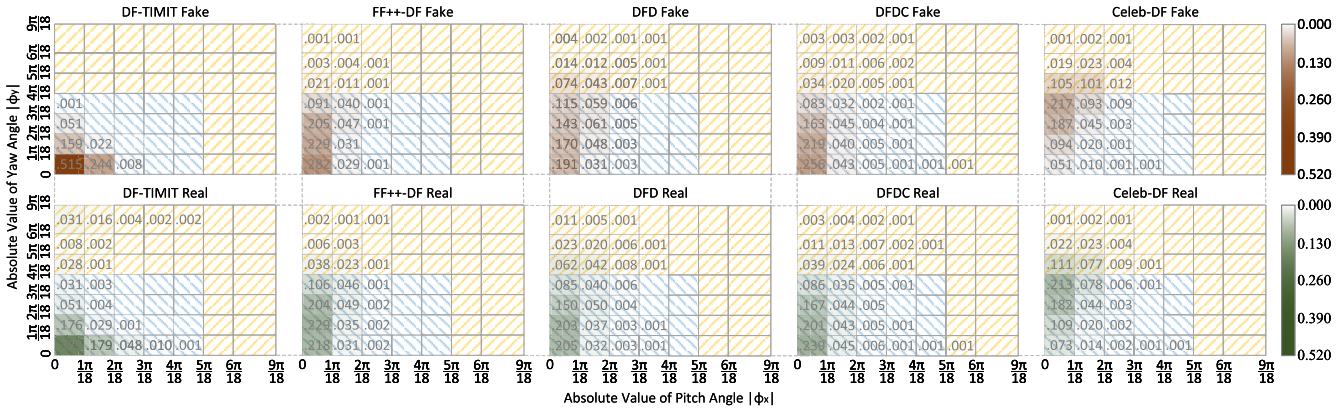

Fig. 7. The details of face pose of DF-TIMIT, FF++-DF, DFD, DFDC, and Celeb-DF datasets. Pictures on the top row stand for fake face images. Pictures on the bottom row stand for real face images. The front face images are highlighted in blue stripe. The side face images are highlighted in yellow stripe.

TABLE II
PERFORMANCE OF FORENSIC SYMMETRY SYSTEM UNDER HOMOLOGOUS DETECTION SCENARIO

Dataset	AUC	F1	PRE	REC	ACC
Image-level					
DF-TIMIT	0.9936	0.9325	0.9416	0.9235	0.9599
FF++-DF	0.9944	0.9585	0.9535	0.9634	0.9584
DFD	0.8622	0.7992	0.7424	0.8655	0.7903
DFDC	0.7835	0.7104	0.6672	0.7597	0.6887
Celeb-DF	0.7659	0.7846	0.7417	0.8328	0.7169
Video-level					
DF-TIMIT	1.0000	0.9920	1.0000	0.9841	0.9943
FF++-DF	0.9988	0.9821	0.9897	0.9746	0.9822
DFD	0.9185	0.8531	0.9104	0.8026	0.8581
DFDC	0.8131	0.7678	0.7168	0.8265	0.7494
Celeb-DF	0.8164	0.8397	0.8010	0.8824	0.7928

TABLE III
IMAGE-LEVEL AND VIDEO-LEVEL AUC OF FORENSIC SYMMETRY SYSTEM UNDER HETEROGENEOUS DETECTION SCENARIO

Model	.TIMIT	FF++.	DFD	DFDC	Celeb.
Image-level					
DF-TIMIT	0.9936	0.7338	0.7077	0.5105	0.6086
FF++-DF	0.8150	0.9944	0.7873	0.5462	0.5810
DFD	0.7717	0.7272	0.8622	0.5568	0.4674
DFDC	0.3878	0.6027	0.6214	0.7835	0.6396
Celeb-DF	0.6461	0.7316	0.6801	0.6076	0.7659
Video-level					
DF-TIMIT	1.0000	0.7533	0.7182	0.5004	0.6351
FF++-DF	0.9615	0.9988	0.8388	0.5497	0.6197
DFD	0.8737	0.7551	0.9185	0.5791	0.4717
DFDC	0.3867	0.6613	0.6354	0.7931	0.7077
Celeb-DF	0.7046	0.7888	0.6902	0.6374	0.8164

DF-TIMIT side face dataset. In this case, the similarity feature extraction model stays in inactivation state.

1) Homologous Detection: The testing datasets of homologous detection scenario and the training datasets of evaluation models belong to the same DeepFake datasets. We present the homologous detection performance of our approach in Table II. The metrics are area under the curve (AUC), balanced F score (F1), precision (PRE), recall (REC), and accuracy (ACC) at image level and video level. Taking DF-TIMIT dataset as an example, our approach correctly identifies their ground truth 95.99% at image level and 99.43% at video level. Compared with the testing results at image level, the values of AUC, F1, PRE, REC, and ACC rose by 0.83%, 5.95%, 5.84%, 6.06%, and 3.44% at video level respectively. Based on Celeb-DF dataset, our approach correctly identifies their ground truth 71.69% at image level and 79.28% at video level. Compared with the testing results at image level, the values of the five metrics rose by 5.05%, 5.51%, 5.93%, 4.96%, and 7.59% at video level respectively. Besides, our approach also works very well on the other DeepFake datasets.

Fig. 8 shows the histograms of tamper probability of five DeepFake datasets at image level and video level. Taking FF++-DF dataset as an example, the proportion of real samples whose tamper probability smaller than 0.1 is 67.35% and the proportion of fake samples whose tamper probability greater than 0.9 is 54.79% at image level. At video level, the two metrics are 67.01% and 62.45%. Based on DFD dataset, the proportion of real samples whose tamper probability smaller than 0.1 is 30.67% and the proportion of fake samples whose tamper probability greater than 0.9 is 31.56% at image level. At video level, the proportion of real samples whose tamper probability lie between 0.3 and 0.4 is 25.00% and the proportion of fake samples whose tamper probability lie between 0.6 and 0.7 is 22.37%. The experimental result shows that our approach can differentiate the real samples and fake samples effectively.

2) Heterogeneous Detection: The testing datasets of heterogeneous detection scenario and the training datasets of evaluation models belong to the different DeepFake datasets. We present the heterogeneous detection performance of our approach in Table III. The metrics are image-level AUC and video-level AUC. Taking Celeb-DF model as an example, the values of AUC on FF++-DF dataset reach 73.16% at image level and 78.88% at video level, which only fall by 3.43% and 2.76% respectively compared with the homologous detection scenario. There are some detection scenarios that our approach

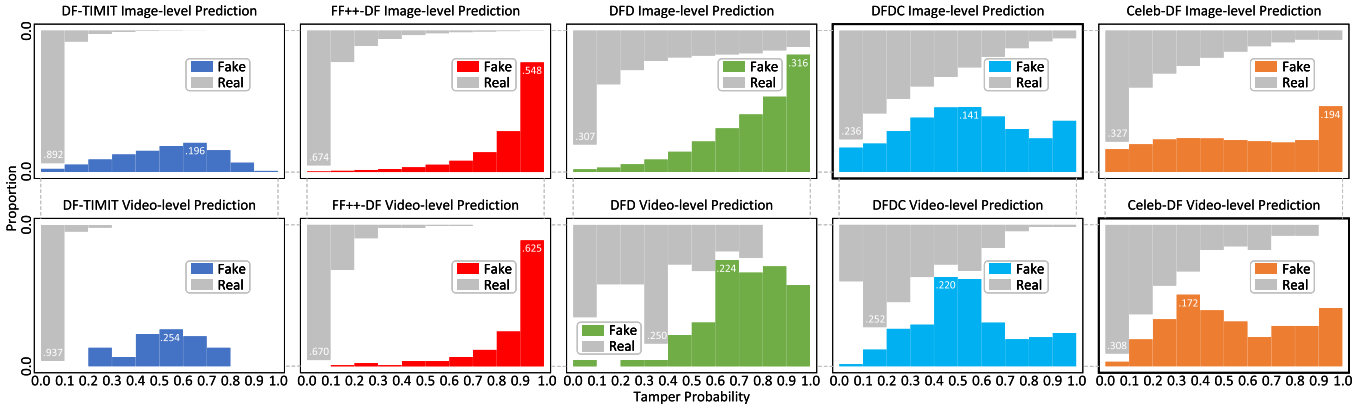

Fig. 8. Histograms of tamper probability of DF-TIMIT, FF++-DF, DFD, DFDC, and Celeb-DF datasets at image level and video level. Pictures on the top row are image-level prediction histograms. Pictures on the bottom row are video-level prediction histograms. The gray parts stand for real images and videos. The color parts stand for fake images and videos.

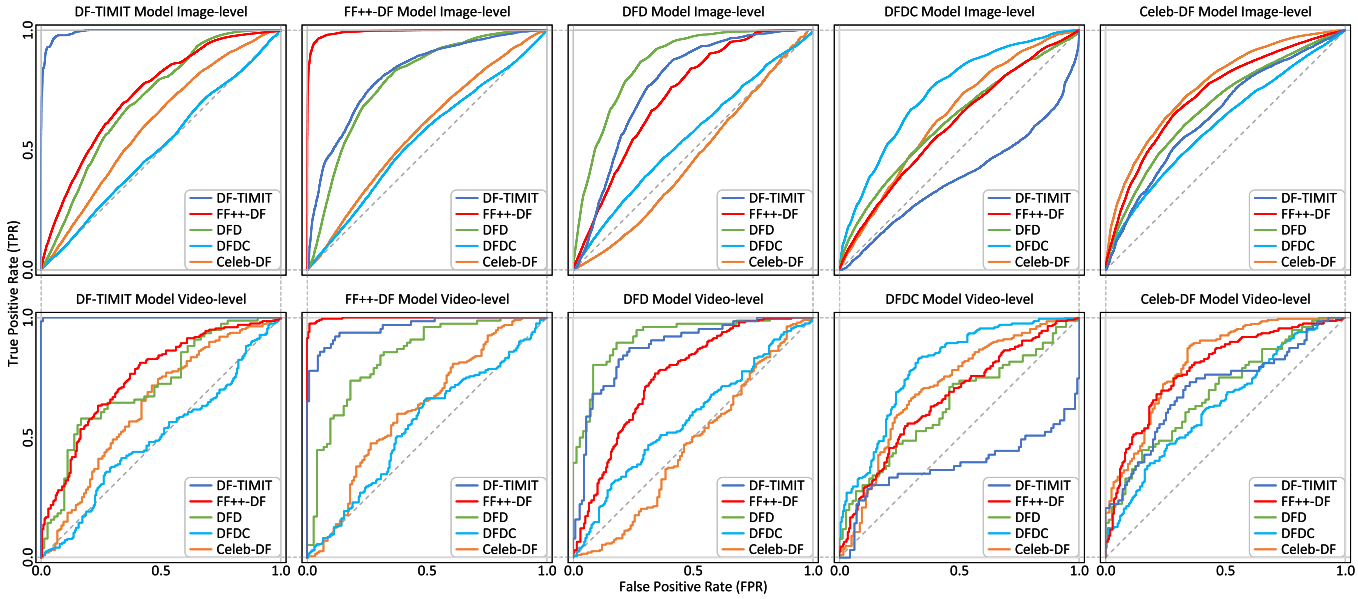

Fig. 9. ROC curves of DF-TIMIT, FF++-DF, DFD, DFDC, and Celeb-DF models on different DeepFake datasets at image level and video level. Pictures on the top row are image-level ROC curves. Pictures on the bottom row are video-level ROC curves.

doesn't achieve good performance. For example, the values of AUC of DFDC model on DF-TIMIT dataset only reach 38.78% at image level and 38.67% at video level, which fall by 39.57% and 40.64% respectively compared with the homologous detection scenario. This is likely due to the large difference of generation method and visual quality between DF-TIMIT dataset and DFDC dataset.

Fig. 9 shows the ROC curves of five natural feature extraction models on five DeepFake datasets at image level and video level. We can see that the performance of our approach on heterogeneous data decreases in a certain extent, which is consistent with the differences of DeepFake videos in visual quality. The higher the visual quality of DeepFake videos, the harder the difficulty of discrimination. In general, the experimental result shows that our approach can effectively extract the difference of natural features of symmetrical face areas even on unknown data sources.

3) Comparison With Existing Methods: In order to compare the performance of our approach with the existing methods, we present the image-level AUC of multiple DeepFakes detection methods under homologous detection scenario and heterogeneous detection scenario in Table IV. Besides, we also evaluate the individual performance of the symmetry feature extractor branch and similarity feature extractor branch of our approach. For the sake of fairness, all evaluation models are trained on FF++-DF dataset.

Experimental result shows that, compared with the existing methods listed, our approach achieves the state-of-the-art performance on FF++-DF dataset under homologous detection scenario. There are some existing methods achieve better performance under heterogeneous detection scenario, however, the values of AUC on FF++-DF dataset under homologous detection scenario are behind our approach. From the comparison, we can see that improving the generalization

TABLE IV
IMAGE-LEVEL AUC OF FORENSICS SYMMETRY SYSTEM AND THE EXISTING DEEPFAKES DETECTION METHODS

Method	FF++-DF	Celeb-DF
Two-stream [35]	0.7010	0.5380
Meso4-Net [12]	0.8470	0.5480
MesoInception4-Net [12]	0.8300	0.5360
FWA [6]	0.8010	0.5690
VA-MLP [5]	0.6640	0.5500
XceptionNet [44]	0.9908	0.3619
Multi-task [31]	0.7630	0.5430
Face X-ray [16]	0.9917	0.7476
CapsuleNet [20]	0.9660	0.5750
DSP-FWA [6]	0.9300	0.6460
Two-branch [40]	0.9320	0.7340
SMIL [41]	0.9680	0.5630
F3-Net [13]	0.9797	0.6517
SPSL [14]	0.9691	0.7688
FInfer [42]	0.9567	0.7060
Ours (Symmetry-branch)	0.9875	0.5561
Ours (Similarity-branch)	0.9530	0.5410
Ours (Multi-stream)	0.9944	0.5810

of heterogeneous detection has long been a major problem in the field of DeepFake forensics.

To confirm the effectiveness of using multi-stream learning structure, we evaluate the performance of the symmetry-branch and similarity-branch of our approach individually. From the experimental result, we can see that it is effective to expose DeepFake images using either symmetry feature extractor or similarity feature extractor. Compared with multi-stream learning structure, the values of AUC of symmetry-branch and similarity-branch on FF++-DF dataset fall by 0.69% and 4.14% respectively. It is easy to understand, symmetry-branch cannot handle side face images very well for the reason that symmetry feature extractor is trained with only front face images. And, similarity-branch cannot handle front face images very well for the reason that similarity feature extractor is trained with only side face images. Our multi-stream learning structure combines the advantages of the symmetry-branch and similarity-branch.

C. Impact of Loss Function

In this section, we evaluate the performance of our approach with different hyper-parameter combinations in our designed multi-margin angular loss function. And, we compare the performance of the multi-margin angular loss function with other metric loss functions. The basic dataset used in this part of the experiments is FF++-DF dataset.

1) Determination of Hyper-Parameter Combination: In order to get the optimal hyper-parameter combination of α_0 and α_1 in the multi-margin angular loss function, we investigate the impact of different hyper-parameter combinations on the performance of forensic symmetry system. To do this, we form 10 hyper-parameter combinations to perform comparative experiments according to the range of $\alpha_0 \in [0, \pi]$ and $\alpha_1 \in [0, \pi]$. We set α_0 and α_1 are from 0.00π to 1.00π every 0.25π. According to the principle of multi-margin angular loss function, the precondition of the setting of hyper-parameter combination is that α_0 should be smaller than α_1. Therefore, it is not workable when α_0 is 1.00π, α_1 is 0.00π, and α_0 greater than α_1.

TABLE V
IMAGE-LEVEL AND VIDEO-LEVEL AUC OF FORENSIC SYMMETRY SYSTEM UNDER DIFFERENT HYPER-PARAMETER COMBINATIONS

$\alpha_0 \downarrow \alpha_1 \rightarrow$	0.00π	0.25π	0.50π	0.75π	1.00π
Image-level					
0.00π	-	0.9899	0.9944	0.9845	0.9327
0.25π	-	-	0.9910	0.9754	0.9527
0.50π	-	-	-	0.9208	0.9268
0.75π	-	-	-	-	0.9065
1.00π	-	-	-	-	-
Video-level					
0.00π	-	0.9956	0.9988	0.9948	0.9706
0.25π	-	-	0.9975	0.9943	0.9844
0.50π	-	-	-	0.9835	0.9740
0.75π	-	-	-	-	0.9664
1.00π	-	-	-	-	-

TABLE VI
IMAGE-LEVEL AND VIDEO-LEVEL AUC OF FORENSIC SYMMETRY SYSTEM UNDER DIFFERENT METRIC LOSS FUNCTIONS

Loss Function	Image-level	Video-level
L1-norm Loss	0.9800	0.9856
L2-norm Loss	0.9832	0.9891
Contrastive Loss	0.9848	0.9920
Cosine Loss	0.9756	0.9809
Ours (Multi-margin Angular Loss)	0.9944	0.9988

Table V shows the values of image-level AUC and video-level AUC under different hyper-parameter combinations on FF++-DF dataset. From the experimental result, we can see that the values of AUC reach maximum of 99.44% at image level and 99.88% at video level when the hyper-parameter α_0 is 0.00π and α_1 is 0.50π. The values of AUC reach minimum of 90.65% at image level and 96.64% at video level when the hyper-parameter α_0 is 0.75π and α_1 is 1.00π. Based on the experimental result, the optimal hyper-parameter combination is that α_0 is 0.00π and α_1 is 0.50π. In this paper, we carry out all experiments using the optimal hyper-parameter combination.

2) Comparison With Other Loss Functions: In order to prove the superiority of our designed multi-margin angular loss function, we investigate the impact of different loss functions on the performance of forensic symmetry system. To do this, we set up a series of experiments in which we use different metric loss functions to optimize natural feature extraction models. The loss functions considered in our experiments are L1-norm loss function, L2-norm loss function, contrastive loss function, and cosine loss function.

Table VI shows the values of image-level AUC and video-level AUC under different loss functions on FF++-DF dataset. From the experimental result, we can see that the values of AUC reach maximum of 99.44% at image level and 99.88% at video level when adopting the multi-margin angular loss function. Compared with the best performance of the other metric

TABLE VII

VIDEO-LEVEL AUC DIFFERENCE OF FORENSIC SYMMETRY SYSTEM UNDER DIFFERENT THRESHOLD COMBINATIONS

$r^t \downarrow p^t \rightarrow$	0.5	0.6	0.7	0.8	0.9
0.5	-0.0318	-0.0276	-0.0121	-0.0059	-0.0048
0.6	-0.0267	-0.0115	-0.0094	-0.0023	+0.0008
0.7	-0.0092	-0.0086	-0.0025	-0.0011	+0.0004
0.8	-0.0052	-0.0038	-0.0029	-0.0022	+0.0000
0.9	+0.0002	+0.0015	+0.0001	+0.0000	+0.0000

TABLE VIII

IMAGE-LEVEL AUC OF FORENSIC SYMMETRY SYSTEM UNDER DIFFERENT MANIPULATION METHODS

Method	DF	F2F	FS	NT
XceptionNet [44]	0.9908	0.9377	0.9742	0.8423
Face X-ray [16]	0.9917	0.9906	0.9920	0.9893
SMIL [41]	0.9680	0.9143	0.9464	0.8857
SPSL [14]	0.9691	0.9462	0.9810	0.8049
Ours (Multi-stream)	0.9944	0.9913	0.9925	0.9736

TABLE IX

IMAGE-LEVEL AND VIDEO-LEVEL AUC OF FORENSIC SYMMETRY SYSTEM UNDER RE-COMPRESSION DETECTION SCENARIO

Model	HQ-data	MQ-data	LQ-data
Image-level			
HQ-model	0.9944	0.8742	0.7521
MQ-model	0.9798	0.9685	0.8838
LQ-model	0.9391	0.9325	0.9249
Video-level			
HQ-model	0.9988	0.9351	0.8150
MQ-model	0.9875	0.9820	0.9389
LQ-model	0.9693	0.9677	0.9601

loss functions listed, the values of AUC of the multi-margin angular loss function rose by 0.96% at image level and 0.68% at video level. Thus, our designed multi-margin angular loss function improves upon the performance of forensic symmetry system over the other metric loss functions.

D. Impact of Prediction Alogrithm

In this section, we evaluate the performance of our approach with different threshold combinations in our designed heuristic prediction algorithm. And, we compare the performance of the heuristic prediction algorithm with the time series neural network.

1) Determination of Threshold Combination: In order to get the optimal threshold combination of p^t and r^t in the heuristic prediction algorithm, we investigate the impact of different threshold combinations on the performance of forensic symmetry system. To do this, we form 25 threshold combinations to perform comparative experiments according to the range of $p^t \in [0.5, 1.0]$ and $r^t \in [0.5, 1.0]$. We set r^t and p^t are from 0.5 to 0.9 every 0.1. According to the principle of the heuristic prediction algorithm, the result of heuristic prediction algorithm is same as the result of averaging method when r^t is 1.0 or p^t is 1.0. The averaging method is taking the mean value of face image tamper probability as face video tamper probability.

Table VII shows the values of video-level AUC difference under different threshold combinations compared with the averaging method on all testing datasets. From the experimental result, we can see that the value of video-level AUC difference reach maximum of +0.15% when the threshold r^t is 0.9 and p^t is 0.6. On the contrary, the value of video-level AUC difference reach minimum of -3.18% when the threshold r^t and p^t are all 0.5. Based on the experimental result, the optimal threshold combination is that r^t is 0.9 and p^t is 0.6. In this paper, we carry out all experiments using the optimal threshold combination.

Besides, we also investigate the optimal threshold combinations on different DeepFake datasets. Experimental result shows that the values of video-level AUC difference reach maximum of 0.09%, 0.14%, 0.18%, 0.15%, and 0.14% on DF-TIMIT, FF++-DF, DFD, DFDC, and Celeb-DF datasets respectively. In this case, the optimal threshold combination of each DeepFake dataset is that r^t is 0.9 and p^t is 0.6. Thus, the selection of threshold combination in the heuristic prediction algorithm is robust to the characteristics of the testing datasets.

2) Comparison With Time Series Neural Network: In order to prove the superiority of our designed heuristic prediction algorithm, we compare the prediction performance of the heuristic prediction algorithm with the time series neural network. To do this, we construct a time series neural network composed of a sequential LSTM unit and a softmax layer. The input of sequential LSTM unit is the natural feature of face frame sequence. The output of softmax layer is face video tamper probability. The length of face frame sequence is 50, which is equal to the number of face frames used in the heuristic prediction algorithm. The face frame sequence is extracted sequentially from each video. The basic dataset used in this part of the experiments is FF++-DF dataset.

After the training process of the time series neural network, we obtained a time series prediction model. It can be used to compute the face video tamper probability. Then, we tested the time series prediction model on FF++-DF dataset. Experimental result shows that the value of video-level AUC reach 98.20%, which fall by 1.68% compared with the performance of the heuristic prediction algorithm. Thus, our designed heuristic prediction algorithm is better than the time series neural network.

E. Effect of Manipulation method

In this section, we investigate the effect of manipulation method on the performance of forensic symmetry system. To do this, we train four natural feature extraction models using four kinds of face datasets which are generated by four typical manipulation methods. They are DeepFakes subset (DF), Face2Face subset (F2F), FaceSwap subset (FS), and NeuralTextures subset (NT) of FF++ dataset. DeepFakes [3] and NeuralTextures [48] are deep learning-based manipulation methods. The former is facial identity replacement method. The latter is facial expression reenactment

method. Face2Face [49] and FaceSwap [50] are computer graphics-based manipulation methods. The former is facial expression reenactment method. The latter is facial identity replacement method.

Table VIII shows the values of image-level AUC under different manipulation methods on FF++ dataset. From the experimental result, we can see that, compared with the best performance of the other methods listed, the values of image-level AUC of our approach rose by 0.27%, 0.07%, and 0.05% on DF, F2F, and FS datasets respectively. And, our approach also works very well on NT dataset. Therefore, our approach is effective to extract asymmetrical natural feature from the face generated by different manipulation methods.

F. Effect of Re-Compression Operation

In the real scenario, the fake content producers may release fake images and videos that have undergone additional compression operation. In this section, we investigate the effect of re-compression operation on the performance of forensic symmetry system. To do this, we train three natural feature extraction models using three kinds of re-compression datasets. They are high quality subset (HQ-data), middle quality subset (MQ-data), and low quality subset (LQ-data) of FF++-DF dataset. The high quality videos with Constant Rate Factor (CRF) parameter equal to 0. The middle quality videos with CRF parameter equal to 23. The low quality videos with CRF parameter equal to 40. After the training process, we obtained high quality model (HQ-model), middle quality model (MQ-model), and low quality model (LQ-model). Then, we evaluated the performance of different natural feature extraction models on different re-compression datasets respectively.

Table IX shows the values of image-level AUC and video-level AUC under different re-compression operations on FF++ dataset. Taking low quality model as an example, the value of AUC on high quality data reach 96.93% at video level, which only fall by 0.16% and 0.92% on middle quality data and low quality data respectively. In general, the performance of our approach on high quality data is better than the performance on low quality data. The experimental result shows that our approach is robust to re-compression operation.

G. Visualization Analysis

In this section, we provide some visual explanations regarding the metrics of our approach. First, we employ Gradient-weighted Class Activation Mapping (Grad-CAM) [51] to visualize the natural features between front face images and side face images. Second, we present detection results of the face images that contain occlusions. Third, we illustrate the process of multi-face video detection. Fourth, we show some failed examples of false negative cases and false positive cases.

1) Feature Visualization: In order to understand what clues the forensic symmetry system rely on to detect DeepFakes, we visualize the natural feature maps using Grad-CAM. To do this, we feed front face images into the symmetry feature extractor to generate symmetry feature maps. We feed side face images into the similarity feature extractor to generate

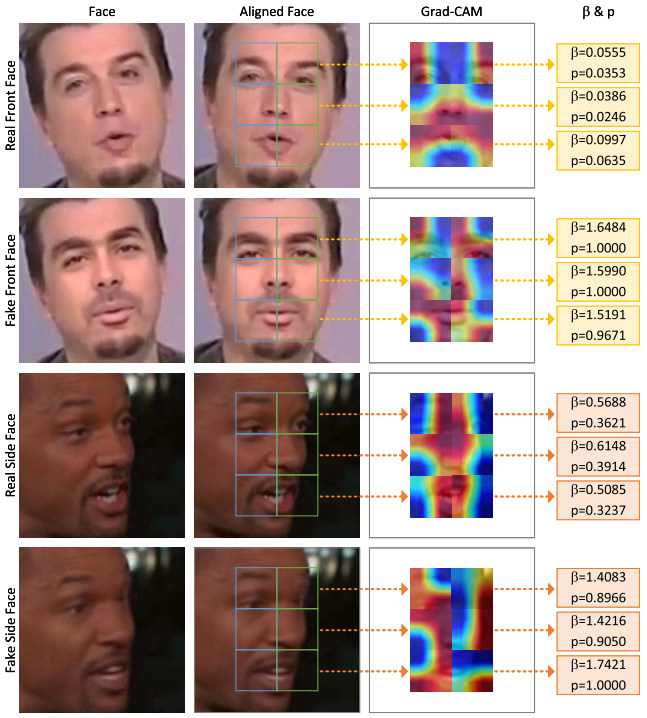

Fig. 10. Visualization of natural feature maps. Pictures on the first two rows are symmetry feature maps. Pictures on the last two rows are similarity feature maps. β stands for the angular distance of symmetrical face patches. p stands for the tamper probability of symmetrical face patches.

similarity feature maps. The symmetry feature maps are shown in the first two rows of Fig. 10. We can clearly observe that our approach produces intense activation around the symmetrical face areas of real front face image. However, the fake front face image present rather disordered and dispersed activation patterns of essential face areas compared to that in the real front face images. The similarity feature maps are shown in the last two rows of Fig. 10. We can clearly observe that the real side face images present rather dense activation patterns of face areas compared to that in the fake side face images. The similarity feature extractor puts more focus on background areas of the incomplete face patches. That is understandable that fake face images generated through DeepFake technique tend to contain asymmetrical artifacts and lack natural details compared to the real face images. The visualization of natural features indicates that our approach can learn reasonable features. Therefore, our approach is useful for DeepFake detection.

2) Occluded Face Detection: In order to evaluate the effect of occlusions on the performance of our approach, we present detection results of the face images that contain occlusions. As shown in Fig. 11, we give results on four pairs of occluded face images. Each pair of face images consists of a real face image and a corresponding fake face image. The occlusions include speckle occlusion, hat occlusion, glasses occlusion, and microphone occlusion. We can see that the face detector used in our approach can extract facial landmarks precisely from occluded face images. Our approach can correctly identify the ground truth of occluded face images. Therefore, our approach is robust to solve occluded face detection problem.

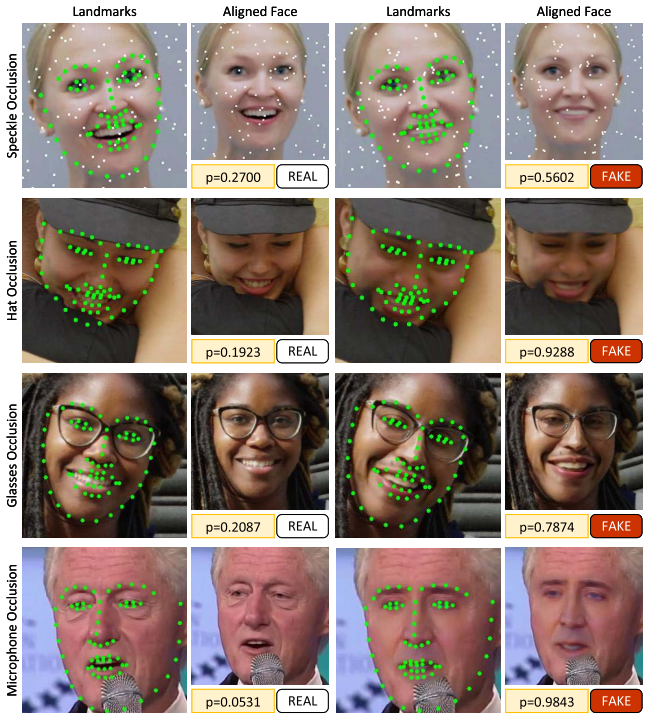

Fig. 11. Examples of occluded face detection. Pictures on the first row are speckle occlusion. Pictures on the second row are hat occlusion. Pictures on the third row are glasses occlusion. Pictures on the fourth row are microphone occlusion. p stands for face image tamper probability.

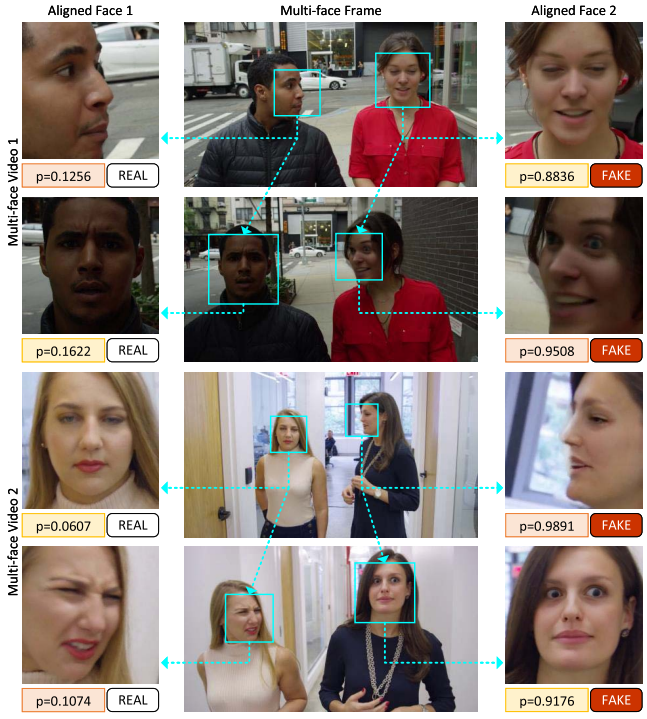

Fig. 12. Examples of multi-face video detection. p stands for face image tamper probability. Compared with single-face video detection, face tracking technique is performed on every frame of the multi-face video. According to the identity information, everyone is treated independently.

3) Multi-Face Video Detection: In order to handle multi-face videos, we illustrate the process of multi-face video detection in Fig. 12. Each frame of multi-face videos has two faces. First, we extract all faces from the first frame through face detector. Then, we perform the face tracking technique on the following frames of the video. According to the identity information, we treat everyone independently. Lastly, we feed all face images into the forensic symmetry system to compute face image tamper probability and face video tamper probability. From the experimental results, we can see that our approach correctly identifies the ground truth of face images in the multi-face videos. Therefore, our approach can handle the problem of multi-face detection properly.

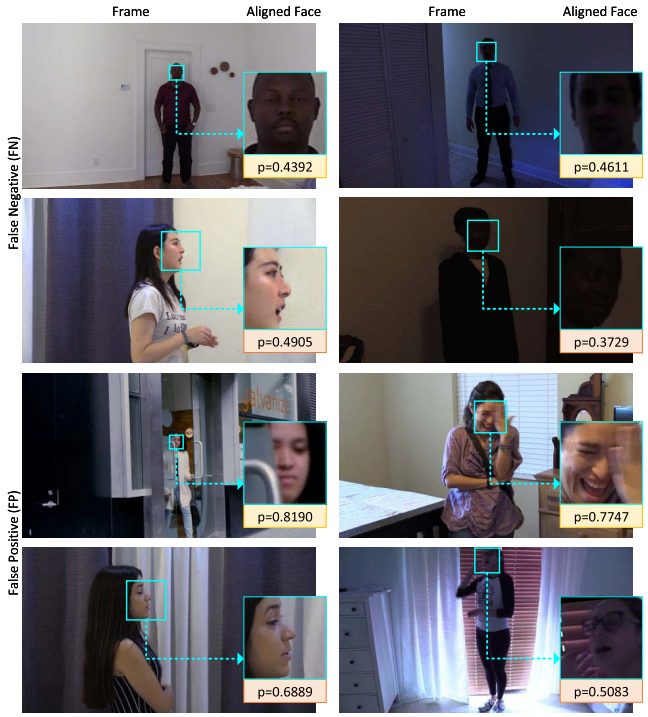

Fig. 13. Failed examples of false negative cases and false positive cases. Pictures on the first two rows are false negative face images. Pictures on the last two rows are false positive face images. p stands for face image tamper probability.

4) Failed Examples: Despite the good performance of our approach on various detection scenarios, we acknowledge that the forensic symmetry system is not without limitations. Fig. 13 shows some failed examples of false negative cases and false positive cases. Our approach would probably not detect the face images and videos in which half or more of face is occluded or could not be seen. In this case, the forensic symmetry system cannot extract natural features from the symmetrical face patches.

V. Conclusion

In this paper, we proposed a new DeepFakes forensics approach called forensic symmetry, which determines whether two symmetrical face patches contain the same or different natural features. Forensic symmetry system is implemented through multi-stream learning structure composed of a symmetry feature extractor and a similarity feature extractor. The symmetry feature extractor obtains symmetry feature from the front face images. The similarity feature extractor

obtains similarity feature from the side face images. Symmetry feature and similarity feature are collectively called natural feature. The difference of natural features is quantified using their angular distance in the angular hyperspace. Besides, we designed a heuristic prediction algorithm to compute the face video tamper probability. We evaluated the performance of our approach under homologous detection scenario, heterogeneous detection scenario and re-compression detection scenario. The experimental results show that our proposed forensic symmetry system is effective in a variety of detection scenarios.

REFERENCES

[1] L. Verdoliva, "Media forensics and DeepFakes: An overview," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 5, pp. 910–932, Aug. 2020.

[2] R. Tolosana, R. Vera-Rodríguez, J. Fiérrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Inf. Fusion*, vol. 64, pp. 131–148, Dec. 2020.

[3] Deepfakes. (2019). *Faceswap*. [Online]. Available: https://github.com/deepfakes/faceswap

[4] shaoanlu. (2019). *Faceswap-GAN*. [Online]. Available: https://github.com/shaoanlu/faceswap-GAN

[5] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *Proc. IEEE Winter Appl. Comput. Vis. Workshops (WACVW)*, Jan. 2019, pp. 83–92.

[6] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPR)*, Nov. 2019, pp. 46–52.

[7] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 8261–8265.

[8] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing AI created fake videos by detecting eye blinking," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2018, pp. 1–7.

[9] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting world leaders against deep fakes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPR)*, Jun. 2019, pp. 38–45.

[10] S. Fernandes et al., "Predicting heart rate variations of deepfake videos using neural ODE," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 1721–1729.

[11] L. Guarnera, O. Giudice, and S. Battiato, "DeepFake detection by analyzing convolutional traces," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 2841–2850.

[12] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A compact facial video forgery detection network," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2018, pp. 1–7.

[13] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao, "Thinking in frequency: Face forgery detection by mining frequency-aware clues," 2020, *arXiv:2007.09355*.

[14] H. Liu et al., "Spatial-phase shallow learning: Rethinking face forgery detection in frequency domain," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 772–781.

[15] G. Li, Y. Cao, and X. Zhao, "Exploiting facial symmetry to expose deepfakes," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2021, pp. 3587–3591.

[16] L. Li et al., "Face X-ray for more general face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5000–5009.

[17] R. Wang et al., "FakeSpotter: A simple yet robust baseline for spotting AI-synthesized fake faces," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 3444–3451.

[18] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, "On the detection of digital face manipulation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5780–5789.

[19] H. Khalid and S. S. Woo, "OC-FakeDect: Classifying deepfakes using one-class variational autoencoder," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 2794–2803.

[20] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-forensics: Using capsule networks to detect forged images and videos," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2307–2311.

[21] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1800–1807.

[22] H. Zhao, T. Wei, W. Zhou, W. Zhang, D. Chen, and N. Yu, "Multi-attentional deepfake detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2185–2194.

[23] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Mach. Learn. Res.*, vol. 97, Jun. 2019, pp. 6105–6114.

[24] P. Korshunov and S. Marcel, "DeepFakes: A new threat to face recognition? Assessment and detection," 2018, *arXiv:1812.08685*.

[25] D. Guera and E. J. Delp, "Deepfake video detection using recurrent neural networks," in *Proc. 15th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Nov. 2018, pp. 1–6.

[26] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, "Recurrent convolutional strategies for face manipulation detection in videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPR)*, Jan. 2019, pp. 80–87.

[27] D. M. Montserrat et al., "Deepfakes detection with automatic face weighting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 2851–2859.

[28] I. Amerini, L. Galteri, R. Caldelli, and A. Del Bimbo, "Deepfake video detection through optical flow based CNN," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 1205–1207.

[29] A. Haliassos, K. Vougioukas, S. Petridis, and M. Pantic, "Lips don't lie: A generalisable and robust approach to face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 5039–5049.

[30] Z. Sun, Y. Han, Z. Hua, N. Ruan, and W. Jia, "Improving the efficiency and robustness of deepfakes detection through precise geometric features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 3609–3618.

[31] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, "Multi-task learning for detecting and segmenting manipulated facial images and videos," in *Proc. IEEE 10th Int. Conf. Biometrics Theory, Appl. Syst. (BTAS)*, Sep. 2019, pp. 1–8.

[32] R. Tolosana, S. Romero-Tapiador, J. Fiérrez, and R. Vera-Rodríguez, "Deepfakes evolution: Analysis of facial regions and fake detection performance," in *Proc. Pattern Recognit. ICPR Int. Workshops Challenges*, vol. 12665, 2020, pp. 442–456.

[33] P. Kumar, M. Vatsa, and R. Singh, "Detecting Face2Face facial reenactment in videos," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 2578–2586.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[35] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Two-stream neural networks for tampered face detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1831–1839.

[36] Y. Hu, H. Zhao, Z. Yu, B. Liu, and X. Yu, "Exposing deepfake videos with spatial, frequency and multi-scale temporal artifacts," in *Proc. Digit. Forensics Watermarking 20th Int. Workshop (IWDW)*, vol. 13180, 2021, pp. 47–57.

[37] R. Durall, M. Keuper, and J. Keuper, "Watch your up-convolution: CNN based generative deep neural networks are failing to reproduce spectral distributions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7887–7896.

[38] M. Li, B. Liu, Y. Hu, L. Zhang, and S. Wang, "Deepfake detection using robust spatial and temporal features from facial landmarks," in *Proc. IEEE Int. Workshop Biometrics Forensics (IWBF)*, May 2021, pp. 1–6.

[39] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz, "Leveraging frequency analysis for deep fake image recognition," in *Proc. 37th Int. Conf. Mach. Learn., (ICML)*, vol. 119, 2020, pp. 3247–3258.

[40] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed, "Two-branch recurrent network for isolating deepfakes in videos," 2020, *arXiv:2008.03412*.

[41] X. Li et al., "Sharp multiple instance learning for deepfake video detection," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 1864–1872.

[42] J. Hu, X. Liao, J. Liang, W. Zhou, and Z. Qin, "FInfer: Frame inference-based deepfake detection for high-visual-quality videos," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 1, 2022, pp. 951–959.

[43] D. E. King, "Dlib-ml: A machine learning toolkit," *J. Mach. Learn. Res.*, vol. 10, pp. 1755–1758, Dec. 2009.

[44] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to detect manipulated facial images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1–11.
[45] Google AI. (2019). *Contributing Data to Deepfake Detection Research*. [Online]. Available: https://ai.googleblog.com/2019/09/contributing-data-to-deepfakedetection.html
[46] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. Canton Ferrer, "The deepfake detection challenge (DFDC) preview dataset," 2019, *arXiv:1910.08854*.
[47] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for deepfake forensics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 3204–3213.
[48] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–12, Aug. 2019.
[49] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Niessner, "Face2Face: Real-time face capture and reenactment of RGB videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2387–2395.
[50] FaceSwap. (2016). *Faceswap*. [Online]. Available: https://github.com/MarekKowalski/FaceSwap
[51] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.

Gen Li received the B.S. degree from the University of Science and Technology at Beijing, Beijing, China, in 2016, and the M.S. degree from the Second Institute of China Aerospace Science and Industry, Beijing, in 2019. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing. His research interests include multimedia security and digital forensics.

Xianfeng Zhao (Senior Member, IEEE) received the Ph.D. degree in computer science from Shanghai Jiao Tong University, China, in 2003. Since 2012, he has been a Professor with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences (CAS), Beijing, where he is the Leader of the Multimedia Security Group. Since 2015, he has also been a Professor with the University of CAS. His research interests are the fields in multimedia security including information hiding, multimedia forensics, and the related AI technologies. In these fields, he has published more than 100 papers. He organized International Workshop on Digital-Forensics and Watermarking (IWDW) five times as co-chairs. He is an Editorial Board Member of the journals on forensics, including *International Journal of Digital Crime and Forensics* (IJDCF) and *Forensic Science International: Reports* (FSI-R).

Yun Cao (Member, IEEE) received the B.S. degree from Tsinghua University, Beijing, China, in 2006, and the Ph.D. degree from the University of Chinese Academy of Sciences, Beijing, in 2012. He is currently an Associate Professor with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, and the School of Cyber Security, University of Chinese Academy of Sciences. His research interests include information hiding, digital forensics, and video processing.