Generalized Deepfake Detection Algorithm based on Inconsistency between Inner and Outer Faces

Jie Gao[1,2*], Sara Concas[2], Giulia Orrù[2], Xiaoyi Feng[1], Gian Luca Marcialis[2], and Fabio Roli[3]

[1] Northwestern Polytechnical University, Xi'an, China
[2] University of Cagliari, Italy
[3] University of Genova, Italy
{sara.concas90c,giulia.orru,marcialis}@unica.it
jie_gao@mail.nwpu.edu.cn,fengxiao@nwpu.edu.cn,fabio.roli@unige.it

Abstract. Deepfake refers to using artificial intelligence (AI) and machine learning techniques to create compelling and realistic media content, such as videos, images, or recordings, that appear real but are fake. The most common form of deepfake involves using deep neural networks to replace or superimpose faces in existing videos or images on top of other people's faces. While this technology can be used for various benign purposes, such as filmmaking or online education, it can also be used maliciously to spread misinformation by creating fake videos or images. Based on the classic deepfake generation process, this paper explores the Inconsistency between inner and outer faces in fake content to find synthetic defects and proposes a general deepfake detection algorithm. Experimental results show that our proposed method has certain advantages, especially regarding cross-method detection performance.

Keywords: deepfake detection · generalization · manipulations

1 Introduction

Deepfake refers to manipulating facial identities through automated algorithms to forge videos/images. The earliest deepfake was [10] created in 2017 by a Reddit user named "deepfakes". Since then, this technology has become more advanced and accessible, increasing the creation and dissemination of deepfakes while raising concerns about their potential harm to individuals and society.

Although there are already multiple deepfake detection methods [16], the generalization problem is still open and crucial. To the best of our knowledge, state-of-the-art (SOTA) methods have achieved high levels of in-domain performance. However, these methods are helpless in the face of cross-domain, which refers to detecting unknown manipulations, different from those present in the training data [7,11]. Given the shortcomings of existing methods, this work aims to study the inconsistency between the foreground and the background of faces

* Corresponding author

in images to detect deepfakes since, in manipulated images, the information of the faces of a source and a target subject coexist. Regardless of the manipulation technique used, the coexistence of this dual information exists in a deepfake. For this reason, designing a system capable of detecting it corresponds to designing a detector capable of generalizing even on unknown manipulations.

The contributions of this work are as follows:

(1) Considering that the forged mask is difficult to obtain in the deepfake content, we directly use facial landmarks to extract the mask and segment the inner face and outer face during the feature extraction process, thereby reducing the dependence on prior knowledge.
(2) We design two different encoders to extract features of inner and outer faces separately and build a consistency loss to distinguish real from fake. Furthermore, we build a parameter-sharing decoder to force it to recover images from low-rank feature vectors.
(3) We further constrain the encoder to extract rich features with reconstruction loss, resulting in improved generalization. The experimental results reveal that our proposed technique is competitive in cross-method experiments.

The remaining chapters of the paper are arranged as follows. Section 2 introduces the existing deepfake detection methods. Section 3 explains the architecture and details of our proposed method. In Section 4, we show the experimental setup and results. Finally, Section 5 discusses the proposed approach and draws some conclusions.

2 Related works on Deepfake Detection

As one of the significant threats to face recognition technology, deepfake technology has been widely spread in social media. Recently, many works [15] have been dedicated to resisting the harm brought by deepfake technology spread [12].

Although detection technologies are increasingly precise and various techniques are exploited [4–6], many methods are trained for specific deepfake manipulations, resulting in a substantially decreased cross-method detection performance.

To solve this problem, some deepfake detection methods focusing on improving cross-method generalization [1] have been proposed. For example, Face X-ray [8] proposed a new way to judge whether a face image has been replaced. They designed a detector based on the observation that there must be a blending boundary in the forgery step to distinguish authenticity by judging whether a face has a blending border. Kaede *et al.* [14] present novel synthetic training data called self-blended images (SBIs) to detect deepfakes. Kim *et al.* [7] proposed a new domain adaptation framework. Specifically, they constructed a student and teacher models based on knowledge distillation and representation learning. The student model can quickly adapt to novel deepfakes by extracting knowledge from the pre-trained teacher model and applying transfer learning without using source domain data during domain adaptation.

Although good generalization performance was reported in the above papers, they redesigned fake synthetic data or complex models for training, increasing the computational complexity and difficulty of training. In order to further solve the above challenges, this paper proposes a simple deepfake detection method, which can reduce the dependence on the training set while improving the generalization ability.

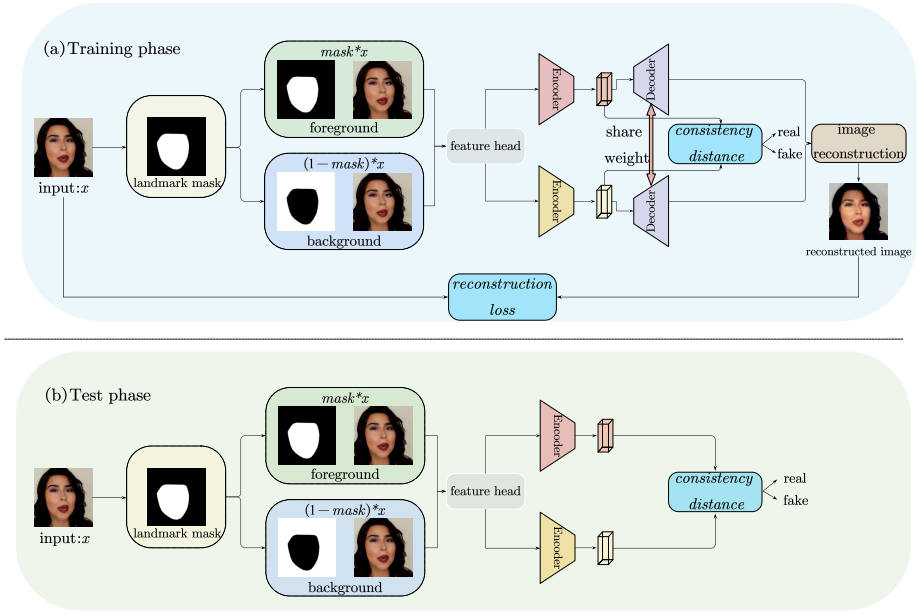

Fig. 1: Proposed architecture. The training step is depicted in (a): the input image is divided into two parts based on face landmarks: foreground and background. The foreground (inner face) is derived from the target image, while the background (outer face) is derived from the source image. The image reconstruction loss function is used to force different encoders to extract detailed information. The consistency loss function is used to differentiate between the consistency of the inner and outer faces. The test phase pipeline is depicted in (b). In the test stage, we only use the feature vector distance obtained by the two encoders for inference.

3 The Proposed Method

To address the generalization problem, we investigate the nature of the face as a biometric trait. In fact, source and target faces differ from individual identities'

perspectives in manipulated content. Based on this phenomenon, this work exploits inner and outer faces inconsistencies in forged faces to detect manipulated content.

Considering the feasibility of the above analysis, we need to assume that any input image consists of two parts (inner face and outer face). Specifically, we consider an input image composed of two components, one related to the source image and the other to the target image. If the image is real, the two components are consistent because the image is associated with a single individual and has not been manipulated. In deepfakes, on the other hand, these two components are very different because they belong to two different identities.

In order to describe an input face image more concretely and intuitively, we assume that the face image is x, and the corresponding label is y (Equation (1)).

$$y = \begin{cases} 0, x \in real \\ 1, x \in fake \end{cases} \tag{1}$$

First, we express the input image x in a general expression as shown in Equation (2), where the two contributions of the target x_{target} and source x_{source} are divided by the segmentation of the face from the background. This binary segmentation allows us to obtain a mask containing the foreground (inner face) and its inverse containing the background (outer face). In particular, the inner face refers to the inner contour, including eyes, eyebrows, and nose, relative to the target image contribution; the outer face refers to the rest of the area and is relative to the source image contribution. Examples of segmentation masks are shown in Fig 2.

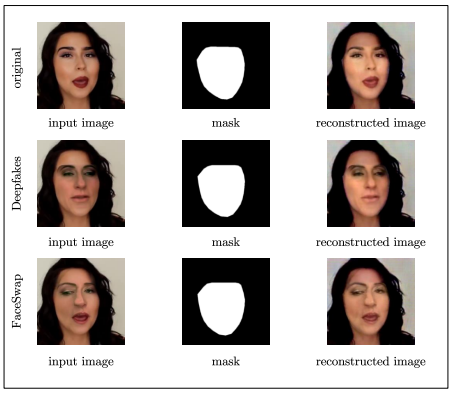

Fig. 2: Examples of the input image processing of the proposed architecture: the input image is segmented and reconstructed by the autoencoder. The reconstructed image is then compared with the original.

$$x = mask \odot x_{target} + (1 - mask) \odot x_{source} \tag{2}$$

According to Equation (2), in order to represent all input images as the relationship between x_{source} and x_{target}, we consider that a real input image satisfies: $x_{source} =$

x_{target}, and a deepfake image satisfies: $x_{source} \neq x_{target}$. Considering that x_{source} and x_{target} are difficult to describe intuitively, we try to map them to distance space to explore their similarity. Therefore, the intuitive idea is distinguishing real images from deepfakes based on consistency loss.

Fig. 1 shows the schematic diagram of our proposed deepfake detection method. As shown in Fig. 1, we can assume that the input image consists of a source image x_{source} and a target image x_{target}. First, we use a facial landmark detection algorithm to obtain the foreground mask, then perform inner and outer face segmentation based on it. Second, we construct a feature head to initialize the masked inner and outer face regions and further construct two different encoders to extract the features of the inner and outer faces, respectively. Our aim is to map these features into the distance space and construct an inconsistency loss function to perform real-fake discrimination. Finally, inspired by the deepfake synthesis process (different encoders, same decoder) and in order to reduce computational burden and enable faster reconstruction of learned distinct facial features for both inner and outer faces, we designed a parameter-shared decoder. We use a direct adding strategy to reconstruct the image since we obtain corresponding inner and outer face features based on landmark mask segmentation. Moreover, we further set the reconstruction loss function as an additional constraint to enable the encoder to extract detailed information. The encoder and decoder structure of the proposed method is shown in Table 1 and Table 2.

Table 1: The encoder of the proposed method consists of four convolutional and pooling layers, followed by a fully connected (FC) layer and a batch normalization (BN) step.

Layer Name	Kernel	Input	Output
Input	-	224x224x3	224x224x3
Feature Head	3x3, 2 filters	224x224x3	224x224x64
Conv1	3x3, 2 filters	224x224x64	224x224x64
MaxPool1	2x2, stride 2	224x224x64	112x112x64
Conv2	3x3, 2 filters	112x112x64	112x112x128
MaxPool2	2x2, stride 2	112x112x128	56x56x128
Conv3	3x3, 2 filters	56x56x128	56x56x256
MaxPool3	2x2, stride 2	56x56x256	28x28x256
Conv4	3x3, 2 filters	28x28x256	28x28x512
MaxPool4	2x2, stride 2	28x28x512	14x14x512
FC	-	14x14x512	512
BN	-	512	512

For the distance between x_{source} and x_{target}, we define it as $D(x_{target}, x_{source})$, and set the threshold τ as the basis for the binary classifications, which can be described as Equation (3).

$$\begin{cases} D\left(x_{target}, x_{source}\right) < \tau, y = 0, x \in real \\ D\left(x_{target}, x_{source}\right) > \tau, y = 1, x \in fake \end{cases} \tag{3}$$

Table 2: The decoder of the proposed method consists of four layers of convolution and unpooling, followed by a fifth level of convolution for the fusion of the inner and outer face information for image reconstruction.

Layer Name	Kernel	Input	Output
Input	-	14x14x512	14x14x512
Conv1	3x3, 5 filters	14x14x512	14x14x512
UnPool1	2x2, stride 2	14x14x512	28x28x512
Conv2	3x3, 3 filters	28x28x512	28x28x256
UnPool2	2x2, stride 2	28x28x256	56x56x256
Conv3	3x3, 3 filters	56x56x256	56x56x128
UnPool3	2x2, stride 2	56x56x128	112x112x128
Conv4	3x3, 2 filters	112x112x128	112x112x64
UnPool4	2x2, stride 2	112x112x64	224x224x64
Conv5	3x3, 3 filters	224x224x64	224x224x3

Inspired by cosine similarity, we utilize cosine distance between x_{source} and x_{target} as the metric function and normalize it to [0,1], as shown in Equation (4).

$$D\left(x_{target}, x_{source}\right) = 0.5 * \left(1 - cos(x_{target}, x_{source})\right) \tag{4}$$

In addition, we adopt the traditional cross-entropy loss function as the loss of the classifier, which is expressed by Equation (5). Among them, $\hat{y}$ is represented by the distance in Equation (4), and y refers to the ground truth label.

$$L_{cls}(y, \hat{y}) = -y \log \hat{y} + (1 - y) \log(1 - \hat{y}) \tag{5}$$

Moreover, to ensure that the encoder can extract enough detailed information, we design a decoder for image reconstruction and construct a reconstruction loss function as a constraint.

Pixel loss is commonly used in the field of image reconstruction, where the traditional approach is to use mean squared error (MSE) loss as a constraint term. However, MSE loss averages over all pixels, assuming that each pixel in the image has equal learning capability. In contrast, perceptual loss is a commonly used loss function in deep learning-based image style transfer methods. Compared to traditional MSE loss, perceptual loss focuses more on the perceptual quality of the image, aligning with the human visual perception of image quality and better capturing the learning of pixels that impact visual perception. Therefore, to ensure that the reconstructed image does not alter the visual perception of the input image by the human eye, we employ both pixel loss and perceptual loss as reconstruction loss.

Among them, the reconstruction loss $L_{rec}(x, \hat{x})$ consists of two parts, namely perceptual loss $L_{per}(x, \hat{x})$ and pixel loss $L_{pix}(x, \hat{x})$, as shown in Equation (8). Here, we use the pre-trained VGG16 as a measure of perceptual loss, as shown in Equation (7), where i represents the i_{th} pixel in the image x, and i belongs to [1, n], n is the total number of pixels in x. And $\hat{x}$ represents the reconstructed image. We adopt the mean square error loss function for the pixel loss, as shown in Equation (6).

$$L_{pix}(x, \hat{x}) = \|x - \hat{x}\|_2^2 \tag{6}$$

$$L_{per}(x,\hat{x}) = \frac{1}{n}\sum_{i=1}^{n}\left(vgg16\left(x_i\right) - vgg16\left(\hat{x}_i\right)\right)^2 \tag{7}$$

$$L_{rec}(x,\hat{x}) = L_{pix}(x,\hat{x}) + L_{per}(x,\hat{x}) \tag{8}$$

Finally, the total loss L consists of three parts, as shown in Equation (9).

$$L = L_{cls}(y, D\left(x_{target}, x_{source}\right)) + L_{pix}(x,\hat{x}) + L_{per}(x,\hat{x}) \tag{9}$$

4 Experimental Analysis and Results

4.1 Datasets

In this paper, we aim to study the cross-method generalization of the deepfake detection model, that is, whether the model trained based on a specific forgery method can obtain good results on the fake content synthesized by other forgery methods. Based on the above purpose, in this paper, we choose the **FaceForensics++** [13] dataset, which contains multiple forgery methods, to confirm the efficiency of our proposed method. This large-scale dataset aims to advance the development of deepfake detection and defence technologies. It contains 5,000 video clips covering a variety of deepfake techniques, including facial synthesis and expression exchanges. The dataset includes five forgery methods: Deepfakes, Face2Face, FaceSwap, FaceShifter, and NeuralTextures. Each synthetic forgery method contains 1000 fake videos. In particular:

(1) **Deepfakes** is formed by replacing the faces in the source video or images with the faces in the target sequence. The logic is based on two autoencoders trained to reconstruct the source and target faces separately.
(2) **Face2Face** is a facial reenactment system that transfers the expression of a source video to a target video while maintaining the identity of the target.
(3) **FaceSwap** is a graphic-based approach which transfers the face region from a source video to a target video. This approach extracts face regions and fits 3D templates based on sparsely detected facial landmarks.
(4) **FaceShifter** is a face swap algorithm based on a two-stage method with high realism and occlusion awareness. The first stage generates high-realistic replacement faces, and the second stage deals with face occlusion, the self-supervised improvement of unnatural face regions through the constraints of multiple loss functions.
(5) **NeuralTextures** uses raw video data to learn the neural texture of the target person to achieve the purpose of facial reenactment, and only the facial expressions corresponding to the mouth area are modified in this dataset.

4.2 Experimental Setup

In this paper, we divide the training set and test set according to 75% and 25%. The specific settings are as follows:

Table 3: Accuracy (ACC) results based on different τ of FF++ dataset.

Train Set (FF++)	Test Set (ACC-%)						Average
	Deepfakes	Face2Face	FaceSwap	FaceShifter	NeuralTextures	FF++	
τ=0.1	50.56	50.56	50.53	50.56	50.44	50.54	50.53
τ=0.2	54.39	54.08	53.39	54.48	52.98	53.92	53.87
τ=0.3	60.43	58.49	56.46	60.19	55.96	58.57	58.35
τ=0.4	65.19	61.28	57.64	64.40	57.00	61.58	61.18
τ=0.5	67.91	63.05	57.98	67.03	57.36	63.06	62.73
τ=0.6	67.99	61.84	56.97	67.29	56.17	62.19	62.08
τ=0.7	65.77	60.56	54.34	64.99	55.49	60.88	60.34
τ=0.8	60.08	56.76	51.87	58.03	53.06	56.86	56.11
τ=0.9	52.57	51.14	50.08	50.78	50.23	51.48	51.04

– For the intra-method experiments, we use the single forge method as the training set and synthesized data based on the same method as the test set. For example, we choose 750 Deepfakes videos as the training set and the remaining 250 Deepfakes videos as the testing set. We arranged other fake datasets in the same way.
– For the cross-method generalization experiments, we use the single forge method as the training set and the other forge method as the testing set. To ensure that the test set does not overlap with the training set, we choose the same test video names in the cross-method experiments as in the intra-method experiments. In other words, the test set here has the same test individuals as the above test set, and the difference is the synthesis method. For example, we choose the same 750 Deepfakes videos, which are the same in intra-method experiments, as the training set, and 250 Face2Face videos, where the video name is the same as the 250 Deepfakes videos in the intra-method experiments, as the testing set. The rest of the dataset is arranged in the same way.

In addition, for each video, we use the face detector MTCNN for face cropping to obtain 32 frames of face images. After the dataset preprocessing operation, we have 24000 images (750x32) in each experiment as the training set and 8000 images (250x32) as the test set. For the acquisition of facial landmarks and masks, we use the dlib package. All experiments are based on the Pytorch framework and implemented on NVIDIA GeForce RTX 2080.

All experiments in this paper use the same parameter settings for a fair comparison. Specifically, we use the stochastic gradient descent (SGD) optimizer when training the model. The learning rate is 0.001, each model is trained for ten epochs, and the batch size is 4.

In order to compare our method with SOTA, we selected four methods. The first one, Xception [3], is an architecture based on Inception, where the traditional modules have been replaced with depthwise separable convolutions. DSP-FWA [9], where convolutional neural networks (CNNs) are used to capture the artifacts originating from the warping process necessary to adapt the new face to the source image. EfficientNetB4, taken from a wider study, is currently one of the best networks in the field of deepfake detection. Moreover, EfficientNetB4ATTST [2] integrates attention mechanism and siamese training on the basis of EfficientNetB4 and is committed to improving the generalization ability of the model.

Ablation experiments on parameter To correctly set the parameters of the proposed method, we conducted comparative experiments on the threshold τ to explore which value is the most reasonable (Table 3). For the parametric ablation experiments, we consider all fake methods in the FF++ dataset. In particular, we extract the five deepfake techniques in the FF++ dataset at the same proportion. Precisely, we extract 150 videos for each forgery method (a total of 750 videos) and 750 real videos as the training set and 50 videos for each forgery method (whole 150 videos) and 150 real videos as the test set. We adopted the accuracy rate to measure the performance. The experimental results show that when $\tau = 0.5$, the average accuracy rate is maximum, equal to 62.73%.

4.3 Results

We first verify the intra-method performance of the proposed approach, which means training and testing on the same deepfake technique. Related experimental results are shown in Table 4, highlighted in blue.

Table 4: Intra-method and Cross-method accuracy based on different training sets in FF++.

Train Set	Model	Test Set (ACC %)					cross-method
		Deepfakes	Face2Face	FaceSwap	FaceShifter	NeuralTextures	Average
Deepfakes	Xception [3]	98.38	51.25	49.87	52.03	52.92	51.52
	DSP-FWA [9]	89.79	51.73	52.03	54.94	53.17	52.97
	EfficientNetB4 [2]	99.06	52.83	49.77	54.53	55.69	53.21
	EfficientNetB4ATTST [2]	73.42	52.57	48.51	53.74	57.29	53.03
	Ours(τ=0.5)	93.97	54.81	48.04	57.61	64.91	**56.34**
Face2Face	Xception	56.28	98.17	50.73	49.74	51.43	52.04
	DSP-FWA	55.70	82.98	51.63	50.76	51.80	52.47
	EfficientNetB4	58.98	98.91	51.28	50.02	51.75	53.01
	EfficientNetB4ATTST	54.26	65.04	53.66	47.98	53.56	52.37
	Ours(τ=0.5)	56.57	85.49	53.96	52.00	53.78	**54.08**
FaceSwap	Xception	51.22	52.67	97.72	50.14	49.42	50.86
	DSP-FWA	61.13	51.52	70.56	52.94	50.59	**54.05**
	EfficientNetB4	50.65	53.16	98.03	50.27	50.09	51.04
	EfficientNetB4ATTST	47.53	53.02	69.71	48.08	47.33	48.99
	Ours(τ=0.5)	55.38	52.42	58.77	53.72	50.89	53.10
FaceShifter	Xception	51.51	48.73	49.16	96.06	51.18	50.14
	DSP-FWA	51.54	49.88	49.69	98.06	50.34	50.36
	EfficientNetB4	50.52	49.75	49.94	98.23	50.09	50.08
	EfficientNetB4ATTST	53.71	48.48	49.33	69.41	52.57	51.02
	Ours(τ=0.5)	60.67	50.40	50.01	91.28	51.82	**53.22**
NeuralTextures	Xception	67.06	61.21	48.16	54.81	89.50	57.88
	DSP-FWA	62.36	59.18	50.18	55.79	87.61	56.88
	EfficientNetB4	72.10	59.49	48.61	58.99	93.80	59.80
	EfficientNetB4ATTST	59.22	53.14	48.79	52.33	62.08	53.37
	Ours(τ=0.5)	82.43	61.17	45.20	57.94	83.13	**61.69**

The intra-method performance of the proposed method is not competitive with the analyzed SOTA. It never performs better than the best one, i.e. EfficientNetB4, and outperforms DSP-FWA only in the tests on Deepfakes and Face2Face. However, the performance is generally good on face swap generation techniques, especially on FaceShifter and Deepfakes (Table 4). A dramatic performance drop occurs when training and testing on FaceSwap images. However, this drop could be explained by the

fact that FaceSwap is based on 3D template synthesis instead of 2D, as in the case of FaceShifter and Deepfakes.

The general good behavior on face swap techniques is because the proposed method bases its functioning on the inconsistencies between the inner and outer face. This observation agrees with the reported results, where it is worth remarking that the face swap technique modifies the inner face, thus increasing the differences with the outer face.

We believe the low intra-method performance is due to the global description obtained from the difference between the target component and the source component. In other words, the proposed method pays more attention to global information that allows for increasing the cross-methods accuracy.

Cross-method experiments allow us to evaluate whether the proposed method can correctly classify unknown types of deepfakes. As shown in Table 4, we conduct cross-method testing based on different training sets. Specifically, we use each forgery method in the FF++ dataset as the training set and the non-overlapping data based on the remaining methods as test sets for cross-validation. The specific data construction method is shown in Section 4.2.

As seen from Table 4, the proposed method obtains the highest average accuracy compared to the SOTA methods. This aspect is crucial because deepfake detection is an arms race problem, and new manipulations are created frequently.

Furthermore, we notice that the training method with a greater generalization is NeuralTexture. This is expected as this reenactment method is recent and produces realistic deepfakes. The most significant increase over SOTA is obtained by training on Deepfakes. In particular, from Table 4, we can see that it reaches 56.34% accuracy, which is 3.13% higher than that of EfficientNetB4. Furthermore, for the FaceShifter method, the average cross-method performance of our approach is 53.22%, which is 2.86% better than DSP-FWA, and 2.14% better than EfficientNetB4ATTST.

The cross-method findings show that the proposed method's main contribution is connected to the classification of face swap-type samples, but it also helps the classification of reenacted images. For example, we have reported in Fig. 3 an input image of low quality. It can be observed by looking at each column that the deep fake probability is high in most cases (columns 4-8), thus independent of the training set. On the other hand, the probability related to the real image leads to the correct classification in 4 cases out of 5.

5 Conclusions

In this paper, we described a novel general-purpose deepfake detection framework. Specifically, we exploited facial landmark detection algorithms to extract masks by assuming that the input image combines source and target images. Furthermore, we segment the face image into inner and outer faces and construct different encoders for feature extraction. The inconsistency is used to discriminate real from fake, and a reconstruction loss is proposed to force the two encoders to obtain richer detailed information.

The experimental findings demonstrate that the proposed technique is promising regarding generalization ability. However, the loss function must be improved to get more details on the differences between inner and outer faces. This will be addressed in future works.

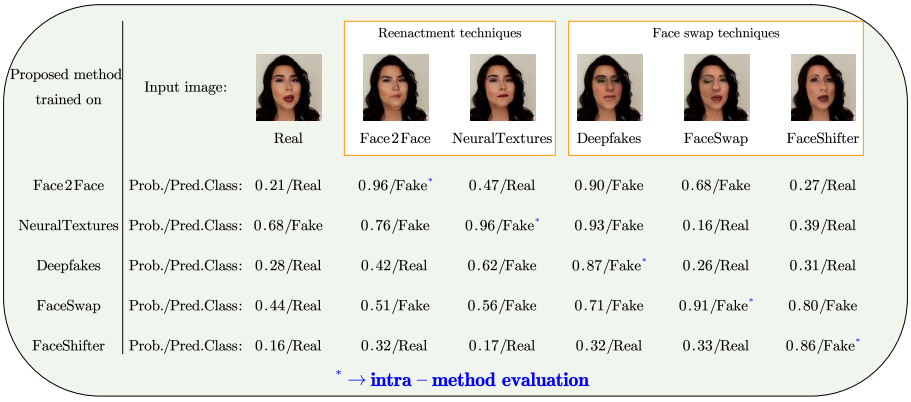

Fig. 3: Classification of input samples obtained with various types of manipulation with models trained on FF++ deepfake types.

Acknowledgment

This work is partially supported by China Scholarship Council (No.202206290093), by SERICS (PE00000014) under the Italian Ministry of University and Research (MUR) National Recovery and Resilience Plan funded by the European Union - NextGenerationEU and within the PRIN2017 - BullyBuster - A framework for bullying and cyberbullying action detection by computer vision and artificial intelligence methods and algorithms (CUP: F74I19000370001). The project has been included in the Global Top 100 list of AI projects addressing the 17 United Nations Strategic Development Goals by the International Research Center for Artificial Intelligence under the auspices of UNESCO.

References

1. Bekci, B., Akhtar, Z., Ekenel, H.K.: Cross-dataset face manipulation detection. In: 2020 28th Signal Processing and Communications Applications Conference (SIU). pp. 1–4. IEEE (2020)
2. Bonettini, N., Cannas, E.D., Mandelli, S., Bondi, L., Bestagini, P., Tubaro, S.: Video face manipulation detection through ensemble of cnns. In: 2020 25th international conference on pattern recognition (ICPR). pp. 5012–5019. IEEE (2021)
3. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1251–1258 (2017)
4. Chugh, K., Gupta, P., Dhall, A., Subramanian, R.: Not made for each other-audio-visual dissonance-based deepfake detection and localization. In: Proceedings of the 28th ACM international conference on multimedia. pp. 439–447 (2020)
5. Ciftci, U.A., Demir, I., Yin, L.: How do the hearts of deep fakes beat? deep fake source detection via interpreting residuals with biological signals. In: 2020 IEEE international joint conference on biometrics (IJCB). pp. 1–10. IEEE (2020)

6. Concas, S., Perelli, G., Marcialis, G.L., Puglisi, G.: Tensor-based deepfake detection in scaled and compressed images. In: 2022 IEEE International Conference on Image Processing (ICIP). pp. 3121–3125. IEEE (2022)
7. Kim, M., Tariq, S., Woo, S.S.: Fretal: Generalizing deepfake detection using knowledge distillation and representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1001–1012 (2021)
8. Li, L., Bao, J., Zhang, T., Yang, H., Chen, D., Wen, F., Guo, B.: Face x-ray for more general face forgery detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5001–5010 (2020)
9. Li, Y., Lyu, S.: Exposing deepfake videos by detecting face warping artifacts. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2019)
10. Lyu, S.: Deepfake detection: Current challenges and next steps. In: 2020 IEEE international conference on multimedia & expo workshops (ICMEW). pp. 1–6. IEEE (2020)
11. Nadimpalli, A.V., Rattani, A.: On improving cross-dataset generalization of deepfake detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 91–99 (2022)
12. Ramachandran, S., Nadimpalli, A.V., Rattani, A.: An experimental evaluation on deepfake detection using deep face recognition. In: 2021 International Carnahan Conference on Security Technology (ICCST). pp. 1–6. IEEE (2021)
13. Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., Nießner, M.: Faceforensics++: Learning to detect manipulated facial images. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1–11 (2019)
14. Shiohara, K., Yamasaki, T.: Detecting deepfakes with self-blended images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18720–18729 (2022)
15. Taeb, M., Chi, H.: Comparison of deepfake detection techniques through deep learning. Journal of Cybersecurity and Privacy **2**(1), 89–106 (2022)
16. Zhao, H., Zhou, W., Chen, D., Wei, T., Zhang, W., Yu, N.: Multi-attentional deepfake detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2185–2194 (2021)