Explore and Enhance the Generalization of Anomaly DeepFake Detection

Yiting Wang$^{1,2\dagger[0009-0002-9407-669X]}$, Shen Chen$^{2\dagger[0000-0001-9140-194X]}$, Taiping Yao$^{2[0000-0002-2359-1523]}$, Lizhuang Ma$^{1*[0000-0003-1653-4341]}$, Zhizhong Zhang$^{1[0000-0001-6905-4478]}$, and Xin Tan$^{1[0000-0001-9346-1196]}$

[1] East China Normal University, China
[2] Tencent YouTu Lab, China
`51215901088@stu.ecnu.edu.cn`
`{kobeschen,taipingyao}@tencent.com`
`{lzma,zzzhang,xtan}@cs.ecnu.edu.cn`

Abstract. In recent years, Anomaly DeepFake Detection (ADFD) has made significant breakthroughs in terms of generalization when meeting various unknown tampers. These detection methods primarily enhance generalization by constructing pseudo-fake samples, which involve three main steps: mask generation, source-target preprocessing, and blending. In this paper, we conducted a systematic analysis of some core factors in these steps. Based on the aforementioned observations at the mask generation step, we find that previous ADFD methods have limitations as they only consider specific tampering types, which is not representative of real-world scenarios, and generate noise samples that closely resemble real samples, causing confusion and hindering generalization. To alleviate these issues, we propose our new method, which consists of the Boundary Blur Mask Generator (BBMG) and the Noise Refinement Strategy (NRS) modules. BBMG leverages the inherent characteristics of boundary blur to simulate a comprehensive range of tampering techniques, enabling a more realistic representation of real-world scenarios. In conjunction with BBMG, the NRS module effectively mitigates the influence of noise samples. Extensive ablation experiments and comparative evaluations demonstrate the effectiveness of our method.

Keywords: Anormaly DeepFake Detection · Pseudo-fake · Noise Strategy · DeepFake Detection.

1 Introduction

With the rapid advancements in artificial intelligence and deep learning, the creation of fake facial content has become efficient. However, the misuse of such

[*] *Corresponding author
†Equal Contribution. This work was done when Yiting Wang was a research intern at Tencent Youtu Lab.

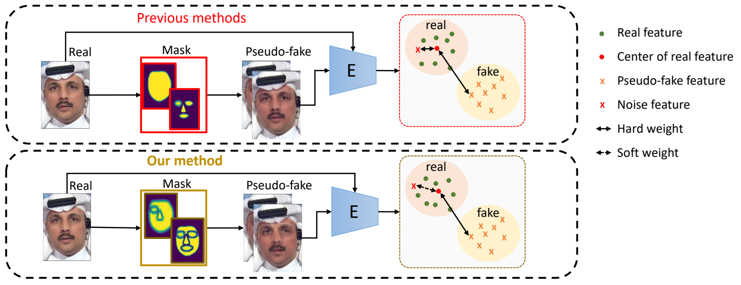

Fig. 1. Comparison of the previous method and our method. Previous methods focused on either local or global tampering. In contrast, our method considers both local and global tampering and employs a noise refinement strategy to mitigate the impact of noisy samples.

technology can lead to harmful consequences, including the spread of false information, social engineering attacks, political manipulation, reputation damage, and financial fraud. Therefore, the emergence of deepfake detection technology is crucial in order to discern between genuine and artificially generated fake faces, thereby safeguarding societal security and order.

Prior arts, such as Face X-ray [17], SBI [29], SLADD [2] have greatly enhanced the generalization capability of deepfake detection. These methods primarily generate pseudo-fake faces through three typical steps: mask generation, source-target preprocessing, and blending. As shown in the top left part of Figure 1, most of them are primarily trained using only real face data, hence we refer to this kind of approach as Anomaly DeepFake Detection (ADFD). Building upon the breakthrough generalization capability of ADFD methods, we investigate several key factors that influence the performance of these methods. Through experiments, we make the following observations: *1) Local tampering based on facial features, which involves making subtle changes to facial features (mouth, nose, eyes, etc.), is less effective compared to global tampering, and simple fusion of both does not improve generalization. 2) The generalization of the model is sensitive to the choice of boundary blur processing methods.*

During our exploration, we found that previous methods in ADFD have focused on specific tampering techniques, neglecting the importance of simulating a comprehensive range of tampering scenarios. However, tampering can occur in both local manipulations (e.g., subtle changes in facial features) and global manipulations (e.g., complete face replacement) specifically related to facial images. Therefore, it is crucial to develop a solution that can effectively handle both types of tampering challenges related to facial manipulation to ensure robust detection of deepfake anomalies. Another key challenge is that previous research has neglected the impact of noise samples introduced during the source-target preprocessing step on the quality of generated facial images. As depicted in the upper-right portion of Figure 1, these noise samples closely resemble real facial images and can significantly affect the model's performance. The similarity between noise samples and genuine facial images can introduce confusion and hinder the model's ability to generalize and make reliable predictions.

In light of the aforementioned challenges, we propose a new method, including Boundary Blur Mask Generator (BBMG) and Noise Refinement Strategy (NRS), as shown in Figure 1. BBMG introduces boundary blur tampering traces at both global and local levels to simulate attacks that are more closely aligned with real-world scenarios. However, the randomness in the data augmentation process of BBMG can still produce pseudo-fake samples that resemble real samples and introduce noise. To alleviate this, we propose the NRS method, where we construct a real feature memory to find the center of the real distribution and select noise samples based on feature similarity. We assign lower weights to the noise samples to suppress their interference with the model's generalization capability. We systematically validate the feasibility of these two methods, demonstrating that our proposed approaches significantly improve the generalization of detection models, achieving state-of-the-art results. The main contributions of our work are summarized as follows:

- We systematically explore a series of Anomaly DeepFake Detection (ADFD) methods, analyzing core factors in the generation of pseudo-fake samples. We find that the design of mask region size and the boundary blurring operation have significant impacts on the generalization capability.
- Based on the above explorations, we propose a new method, which consists of two parts: Boundary Blur Mask Generator and Noise Refinement Strategy. Boundary Blur Mask Generator effectively simulates both local and global tampering techniques, thereby improving model generalization. Besides, Noise Refinement Strategy mitigates the impact of noise samples in ADFD methods.
- Through extensive ablation experiments, we demonstrate that our proposed methods for deepfake detection achieve state-of-the-art performance. These contributions significantly advance the field of deepfake detection and provide valuable insights into improving the generalization capability of ADFD.

2 Related Work

2.1 Conventional DeepFake Detection.

In the early stages, deepfake detection algorithms relied on manual prior knowledge. With the introduction of deep learning, algorithms started focusing on spatial information and neural network design. For instance, compact network Mesonet [1], capsule network [24], autoencoder [8], recurrent convolutional networks [12, 27], and attentional networks [34]. Some methods have also investigated the utilization of frequency domain information [10, 16, 21, 22, 25, 28, 32], leveraging the characteristics of frequency domain to effectively capture forgery traces. Additionally, other methods have utilized temporal information to enhance the model's ability to discriminate forgeries, such as local mouth motion [13, 35], Facial Action [30] and temporal consistency [11, 20, 33]. While these methods have shown good performance in detecting known tempering, many of them struggle to generalize well to detect deepfakes created by unknown temperings, resulting in poor generalization.

2.2 Anomaly DeepFake Detection

To improve the generalization of deepfake detection, several researches have proposed Anomaly DeepFake Detection(ADFD) [2, 17, 29, 36] that mainly use real training data to improve model's discriminative ability towards unknown tampering methods. OC-FakeDect [14] proposed a one-class anomaly detection approach using a one-class Variational Autoencoder (VAE) trained solely on real face images to detect Deepfakes. Some other methods tried to synthesize pseudo-fake data, encouraging models to learn generalizable features for deepfake detection. Face-Xray [17] introduced blending the altered face into an existing background image to simulate forgery traces and generate synthetic training data. PCL [36] proposed using an inconsistency image generator (I2G) to synthesize forgery data and detect whether an image is forged based on patch consistency. SLADD [2] focused on local feature tampering using facial landmarks and employs adversarial learning to generate more sophisticated and novel forgery configurations, thereby improving detector performance. SBI [29] generated fake faces by blending pseudo source and target images from single pristine images, achieving higher generalization compared to previous methods.

3 Approach

3.1 Overview

In this section, we review previous Anomaly DeepFake Detection (ADFD) methods and explore influencing factors on pseudo-fake sample generation. Based on these insights, we introduce our innovative methods, including the Boundary Blur Mask Generator (BBMG) and Noise Refinement Strategy (NRS), as depicted in Figure 3.

3.2 Review and Exploration of ADFD

Previous ADFD methods [2, 17, 29, 36] generate pseudo-fake faces D_{pf} from real faces D_r typically through blending, and then learn to distinguish real and fake samples through feature extractor $E(I;\Theta)$ and classifier $\mathscr{F}(I;\omega)$. The generating process can be mainly divided into the following three steps, as shown in Figure 2:

1. **Mask Generation.** To obtain the tampered regions, it is necessary to generate corresponding masks M as the areas of manipulation. SBI, Face X-ray, and SLADD propose a variety of mask generation methods, among which Face X-ray and SBI are based on the global mask, and SLADD is based on the local mask of the facial features, such as left eye, right eye, nose, and

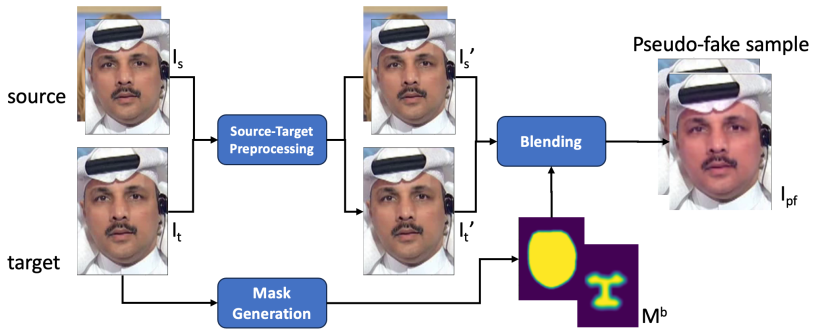

Fig. 2. The Pipeline of pseudo-fake face generation in ADFD.

mouth. Typically, various augmentations such as boundary-blurring are employed to obtain the processed mask M^b. This approach helps to make the mask softer, resulting in a more realistic simulation of tampering.

2. **Source-Target Preprocessing.** This part primarily involves preprocessing the source images I_s and target images I_t before tampering. For Methods blending two different faces, such as Face X-ray, SLADD, alignment adjustment, and color correction operations are typically performed to make the source face better fit the target. In the case of methods like SBI, where I_t and I_s are derived from the same image, color transform, and frequency transform are applied to accentuate the differences between I_t and I_s in this step. After the preprocessing step, the processed versions of I_s and I_t are obtained, denoted as I_t' and I_s', respectively.
3. **Blending.** This part generally blends the processed source image I_s' and the processed target image I_t' to generate a pseudo-fake sample I_{pf} based on the generated mask M^b. The fusion formula for getting I_{pf} is as follows:

$$I_{pf} = M^b \cdot I_s' + (1 - M^b) \cdot I_t'. \tag{1}$$

Different ADFD methods offer various parameters for these three steps, which may affect the generalization capability of the model on unseen tampering scenarios. Therefore, we conducted experiments to explore the impact of these parameters and identify key factors that contribute to the improvement of model generalization. Comparative experiments are conducted based on a common baseline (SBI), and the specific experimental procedures and results can be found in Section 4.2.

Based on our exploration of mask generation, we draw the following conclusions: *1) Local tampering based on facial features is less effective compared to global tampering, and simple fusion of both does not improve generalization. 2) The model's generalization is susceptible to the influence of boundary blur processing methods.*

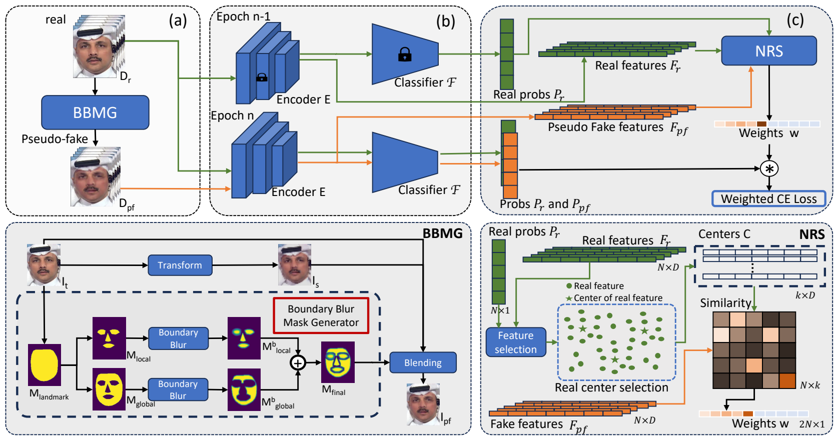

Fig. 3. Overview of our method. The overall process can be divided into three steps. (a) We generate pseudo-fake training data using our proposed Boundary Blur Mask Generator. (b) The real face image D_r and the pseudo-fake face image D_{pf} are fed into the encoder and classifier, obtaining the features and scores for each image. (c) We calculate the weighted CE loss using the Noise Refinement Strategy, which is used to update the network.

3.3 Boundary Blur Mask Generator

In this part, we propose a Boundary Blur Mask Generator (BBMG), corresponding to Figure 3. According to our previous exploration, existing methods have difficulty in covering all types of tampering. Moreover, we observed that random fusion methods did not improve generalization, possibly due to the limited effectiveness of local masks in generating tampering traces. To cover a wider range of tampering types and increase the visibility of tampering traces, we propose to design a method that can simultaneously simulate both local and global tampering. Our exploration revealed the significant impact of the boundary-blurring operation on model generalization. Therefore, we propose our Boundary Blurring Mask Generator (BBMG), which allows for simple modification of the mask to learn both local and global tampering. This approach enhances tampering traces and improves the generalization of the model.

First, we select the facial region of the input image as the mask ($M_{landmark}$) based on landmarks. Then, we further localize the facial features, i.e. left eye, right eye, nose, and mouth, and randomly select some of these features. After that, selected features are extracted from the global mask, resulting in two masks: a local mask (M_{local}) containing only the selected features and a global mask (M_{global}) with some features removed. The calculation of M_{global} is as follows:

$$M_{global}(i,j) = M_{landmark}(i,j) - M_{local}(i,j), \tag{2}$$

where (i, j) represents the pixel position in the mask. Next, we apply Boundary Blur to both M_{local} and M_{global}. Considering the widespread use of Gaussian Blur, we also employ Gaussian Blur to blur the boundaries. After applying Gaussian Boundary Blur, we obtain blurred masks M^b_{local} and M^b_{global}. Finally, we merge the two blurred masks by taking the maximum value at each pixel position, resulting in a final mask M_{final} with both local and global Boundary Blur characteristics. Calculations are as follows:

$$M_{final}(i,j) = max(M^b_{global}(i,j), M^b_{local}(i,j)). \tag{3}$$

Through these steps, we successfully generate a mask with both local and global Boundary Blur characteristics. Following SBI, we initialize source image I_s and target image I_t by copying input image I, then randomly apply image transformations to either of them to get a processed source image I'_s and a processed target image I'_t. We then blend the processed source and target images based on the generated mask M_{final} using the blending operation, resulting in a pseudo-fake face image. This pseudo-fake image is treated as a fake sample in the training set and used to train the model. The calculation of the pseudo-fake face image is as follows:

$$I_{pf} = M_{final} \cdot I'_s + (1 - M_{final}) \cdot I'_t. \tag{4}$$

Our proposed BBMG method generates a mask that incorporates both local and global tampering traces, and experimental results have shown that this mask-generation method can further improve the model's generalization.

3.4 Noise Refinement Strategy

Since the use of random data augmentation to generate corresponding source or target images introduces noise and makes the pseudo-fake samples too similar to real samples in BBMG method, we propose Noise Refinement Strategy to reduce the impact of noise samples on the model's generalization, corresponding to Figure 3. We measure the distance between N pseudo-fake sample features $F_{pf} = E(I_{pf}, \Theta) \in \mathbb{R}^{N\times D}$ and the centers $C \in \mathbb{R}^{k\times D}$ of real sample features $F_r = E(I_r, \Theta) \in \mathbb{R}^{N\times D}$ to identify pseudo-fake samples that are too close to the center of the real sample distribution. These samples are considered noise samples and assigned lower loss weights to reduce their influence on the model's learning, thereby improving the model's generalization.

First, we need to determine the centers of the real sample distribution. Considering that real samples will dynamically change during training, in order to obtain the latest distribution of features for real samples, we use the encoder trained in the last epoch to get the real features $F_r = E(I_r, \Theta) \in \mathbb{R}^{N\times D}$. To obtain a more accurate center of the real feature distribution, we select the top k_s correctly predicted real features, i.e. features with the highest probability value $P_r \in \mathbb{R}^{N\times 1}$, as the memory of real features. These features in memory are then clustered using k-means clustering to obtain k center features $C = \{c_1, c_2, ..., c_k\} \in \mathbb{R}^{k\times D}$, which serve as the clustering centers of the real samples and are used as references for selecting noise samples in the current epoch.

Once we have obtained the distribution centers $C \in \mathbb{R}^{k \times D}$ of the real features, we can filter out noise samples by measuring the feature distances. For the pseudo-fake samples features $F_{pf} = E(I_{pf}, \varTheta) \in \mathbb{R}^{N \times D}$, we calculate the cosine distance between the pseudo-fake features and each center point to determine the distance, selecting the minimum distance as the final noise distance score $Dist(F_{pf}, C)$. Samples with noise distance scores below a threshold value λ are considered noise samples. As the discriminative ability of the model is relatively poor in the early stages, we introduce the epoch parameter β when calculating weights of loss $w(F, C)$ to gradually reduce the importance of the noise samples as the epoch increases. The weights of loss can be calculated as:

$$w(F,C) = \begin{cases} \alpha_{pf}(F,C) \ , Dist(F,C) < \lambda \wedge F \in F_{pf} \\ \qquad\quad 1 \ , Dist(F,C) \geq \lambda \vee F \in F_r , \end{cases} \tag{5}$$

$$\alpha_{pf}(F_{pf}, C) = Dist(F_{pf}, C) * (epoch/\beta) + 1 * (1 - epoch/\beta), \tag{6}$$

The cross-entropy loss is represented as:

$$p(I) = \mathscr{F}(E(I, \varTheta), \omega), \tag{7}$$

$$L_{ce}(I, y) = -(y \log(p(I)) + (1 - y) \log(1 - p(I))), \tag{8}$$

where y represents the label of the image I. Combining Equ. 5 and Equ. 8, we obtain the final loss function as follow:

$$L = 1/N * \sum_{I \in D_r, D_{pf}} (w(E(I, \varTheta), C) * L_{ce}(I, y)), \tag{9}$$

where N represents the number of training samples.

3.5 Algorithm

The training pipeline of the proposed algorithm can be roughly divided into two stages. In the early stage, before the T_kth epoch, we focus on learning the differences between real and generated samples. For each step, we first generate fake samples using the Boundary Blur Mask Generator. Then, we feed the generated fake samples and real samples from the training set into the Encoder $E(I; \varTheta)$ and the Classifier $\mathscr{F}(F; \omega)$, updating the weights using a simple Cross-Entropy (CE) loss to learn the distribution of real and fake samples. After reaching the T_kth epoch, the model has acquired some discriminative ability between real and fake samples. At this stage, we introduce the Weighted Loss Noise Refinement Strategy. This strategy assigns lower loss weights to noise samples to reduce their impact on learning real-face features. The specific training strategy is shown in Algorithm 1.

Algorithm 1 Algorithm of our proposed method

Input: Real Training Images D_r
Parameter: Encoder E with Θ, Classifier $\mathscr{F}$ with ω, epoch T_k and T_{max}, fixed λ,β,k, k_s, batch size B
Output: the model parameter $W = \{\Theta, \omega\}$

```
1:  for t=1,2,...,$T_{max}$ do
2:    shuffle training set $D_r$.
3:    for n=1,...,$|D_r|/B$ do
4:      Fetch n-th mini-batch $D_r^n$ from $D_r$.
5:      Read preprocessed landmark $L_r^n$.
6:      Obtain pseudo-fake data $D_{pf}^n$ by BBMG using $D_r^n$ and $L_r^n$.
7:      $p_r^n = \mathscr{F}(E(I_r, \Theta), \omega), \forall I_r \in D_r^n$.
8:      $p_{pf}^n = \mathscr{F}(E(I_{pf}, \Theta), \omega), \forall I_{pf} \in D_{pf}^n$.
9:      $F_r^n = E(I_r, \Theta), \forall I_r \in D_r^n$.
10:     $F_{pf}^n = E(I_{pf}, \Theta), \forall I_{pf} \in D_{pf}^n$.
11:     if $t >= T_k$ then
12:       Calculate loss $L$ by NRS using $\lambda$,$\beta$,$C$ and $F_{pf}^n$.
13:     else
14:       Calculate cross-entropy loss $L$.
15:     end if
16:     Update $\{\Theta, \omega\}$;
17:   end for
18:   if $t >= T_k - 1$ then
19:     select $k_s$ real features $F_{rm}$ from $F_r = \{F_r^1, F_r^2, ..., F_r^{|D_r|/B}\}$ according to $P_r = \{p_r^1, p_r^2, ..., p_r^{|D_r|/B}\}$.
20:     Get k centers $C = c_1, c_2, ...c_k$ using Kmeans by $F_{rm}$
21:   end if
22: end for
23: return $\{\Theta, \omega\}$
```

4 Experiments

4.1 Experiments Setting

Dataset. We trained our model on the widely used FaceForensics++ (FF++) dataset [26], and evaluated its generalization performance on the Celeb-DF-v2 (CDF) [19], DeepFake Detection Challenged Preview (DFDCP) [7], DeepFake Detection Challenge (DFDC) [6], and FFIW-10K (FFIW) [38] datasets.
Comparison. We compare our method with previous methods includes DSP-FWA [18], Face X-ray+BI [17], LRL [3], FRDM [22], PCL+I2G [36], SBI [29] and SLADD [2]. At the video-level, we compare our method with previous approaches using the receiver operating characteristic curve (AUC). Frame-level predictions are typically averaged over video frames.
Data preprocessing. Following the preprocessing method in SBI, during the testing phase, we used the 81 facial landmarks shape predictor from Dlib[15] and RetinaFace[5] to extract facial landmarks and bounding boxes for each frame. The facial region was randomly cropped with margins ranging from 4% to 20%.

Table 1. Implementation details of the three steps of pseudo-fake face generation in the ADFD approach.

Step	Part	Parameters
Mask Generation	Local and global mask regions	Local(SLADD), Global(SBI, landmark augmentation library)
	Parameter settings of mask	Mask LineType(LINE_88, LINE_AA), Mask DataType(Uint8, Float32), Mask Value(1, 255)
	Boundary-blurring operations	Blurring Operation(Gaussian, Average, Median), Blurring Degree(Gaussian Blur Kernel Size)
Source-Target Preprocessing	Transform methods	RGBShift, HueSaturation, RandomBrightnessContrast, etc.
	Transform Objects	Source Transform, Target Transform
Blending	Blending types	Dynamic Blending, Alpha Blending

During inference, we only use RetinaFace to detect the facial region and fixed the cropping margin at 12.5%.
Implementation Details. We used the EfficientNet-b4 [31] pre-trained on ImageNet [4] as the encoder and trained it for 100 epochs using the SAM [9] optimizer. The learning rate was set to 0.001, and the batch size was 32, including 16 real faces and corresponding 16 generated pseudo-fake faces. During training, we only used the real samples from the FF++ and extracted 8 frames from each video as the training set. It is worth noting that our BBMG and NRS modules are training strategies applied during the training phase. During inference, we only utilize the forward propagation of the EfficientNet model to obtain the prediction results. Therefore, our proposed method does not introduce any additional parameters or increase the inference FLOPs. For data augmentation, we applied techniques including ImageCompression, RGBShift, HueSaturationValue, and RandomBrightnessContrast. The hyperparameters used in our algorithm, as shown in Algorithm 1, are $\lambda = 0.5, \beta = 200, k = 5, k_s = 500$. During testing, we extracted 32 frames from each video and selected the maximum predicted value among all the faces in each frame as the prediction score for that frame. The average of all frame prediction values was taken as the confidence score for the video. To ensure fairness, for videos where no faces were detected, we set the confidence score to 0.5.

4.2 Exploration Experiments of ADFD Methods

In this section, we provide an experimental exploration of Anomaly DeepFake Detection methods, with a focus on the Mask Generation, Source-Target Preprocessing, and Blending step, as details in Table 1.

Exploration of Mask Generation Exploration of Local and Global Mask Regions. We summarize the mask generation strategies into three types,

Table 2. AUC comparison of different mask generation strategies based on SBI.

Name	Test Set AUC(%)						
	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
SBI w/ global mask	99.01	80.31	70.40	82.37	79.58	82.33	78.17
SBI w/ LAL	**99.56**	**91.23**	**72.10**	**86.25**	83.68	**86.56**	**83.32**
SBI w/ local mask	95.89	73.99	61.70	60.89	72.08	72.91	67.17
SBI w/ LAL and local mask	99.29	88.73	71.70	81.55	**84.49**	85.15	81.62

Table 3. AUC comparison of different parameter settings of the mask.

Parameters				Test Set AUC(%)						
LineType	DataType	Value	Size	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
LINE_8	uint8	1	facehull	99.01	80.31	70.40	82.37	79.58	82.33	78.17
LINE_8	uint8	255	facehull	99.01	80.31	70.40	82.37	79.58	82.33	78.17
LINE_AA	uint8	1	facehull	99.05	79.67	70.37	83.19	79.88	82.43	78.28
LINE_8	float32	1	facehull	99.36	89.74	71.52	84.05	85.76	86.09	82.77
LINE_AA	float32	255	facehull	99.36	88.23	71.46	84.54	85.41	85.80	82.41
LINE_AA	float32	255	LAL	99.56	91.23	72.10	86.25	83.68	86.56	83.32

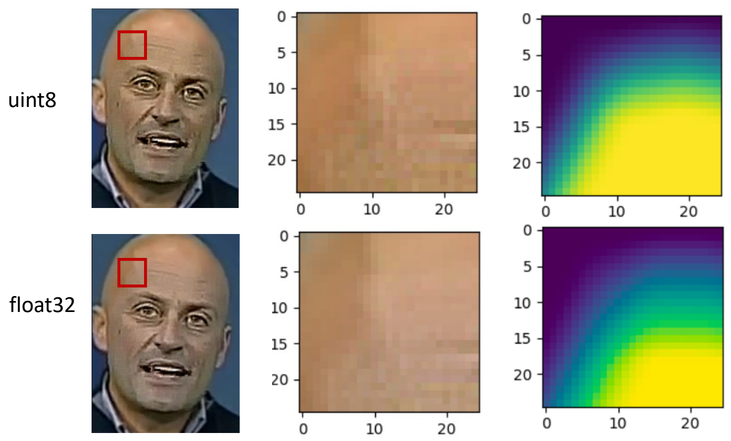

Fig. 4. Comparison of the mask with different data types. We visualize the blurred mask with uint8 and float32 data types respectively. Mask with float32 data type shows slightly higher blurring at the boundaries.

i.e. global mask used in the source code of SBI, the global landmark augmentation library (LAL) from Face X-ray, and the local mask proposed by the SLADD. We conducted ablation experiments on these strategies and the corresponding combinations, the results are shown in Table 2. From Table 2, we can observe several interesting phenomena: 1) Despite both being based on landmark-based

https://github.com/mapooon/SelfBlendedImages
https://github.com/AlgoHunt/Face-Xray

global masks, the mask generation method based on SBI performs much worse in terms of generalization compared to the method based on LAL. 2) The local mask generation method based on SLADD ignores the global tampering patterns and produces images that are too close to the real images, resulting in low generalization performance. 3) By randomly selecting between the LAL and local masks, the results are still lower than those obtained using the global LAL tampering. The above experiments illustrate that different mask strategies have a significant impact on generalisability.

Exploration of Parameter Settings of Mask. To further explore the factors at the heart of the performance differences between the global mask and LAL, we identify four main areas of distinction: line drawing method (LineType), mask data type (DataType), mask value range (Value), and mask size selection(Size). We conduct experiments by adjusting these parameters, and the results are shown in Table 3. Through this exploration, we find that the core influencing parameter causing this difference lies in the mask DataType. Simply changing the mask DataType from uint8 to float32 leads to a 4.6% improvement in average generalization performance (Avg w/o FF++). Then, we visualize the masks with uint8 and float32 data types, as shown in Figure 4. We observe that after applying Gaussian blur, the float32 data type shows slightly higher blurring at the boundaries. Therefore, we can preliminarily conclude that the boundary-blurring approach has a significant impact on the model's generalization.

Exploration of Boundary-blurring Operation. We further investigate the extent to which boundary blurring affects generalization in the Exploration of Boundary-blurring Operation. Firstly, we explored three different boundary operations: Gaussian, Average, and Median. The results are shown in Table 4. The experiment revealed that there was not much difference in the results between the Gaussian method and the Average method, while the Median method yielded significantly lower performance. The main difference between the Median method and the other two methods is that Gaussian and Average blur the boundaries of the mask, while the Median method does not. Hence, we hypothesized that the blurring of boundaries has a substantial impact on the model's generalization. To validate this hypothesis, we further selected Gaussian blur and fixed the regions in the mask without any blurring, i.e. regions with a mask value of 1, to solely examine the effect of boundary-blurring degree on the model's generalization, which is shown in Figure 5. The experiment confirmed that different boundary blurring degrees, i.e., different sizes of the Gaussian blur kernel, have a significant difference in the model's generalization. Therefore, we further prove that the generalization of ADFD methods is sensitive to boundary-blurring operations. The maximum difference in average generalization can reach up to 9.64% (from 62.39% to 72.03%).

Exploration of Source-Target Preprocessing Exploration of Transform Methods. In the Source-Target Preprocessing step, most methods employ augmentation techniques such as transformations to help generate pseudo-fake samples that facilitate model learning. Taking SBI as an example, we categorize

Table 4. AUC comparison of different boundary-blurring operations.

	Test Set AUC(%)						
Name	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
Gaussion	99.37	88.45	71.57	84.79	83.19	85.47	82.00
Average	99.31	89.41	71.80	84.67	87.02	86.44	83.23
Median	96.16	83.08	63.52	83.63	73.95	80.07	76.05

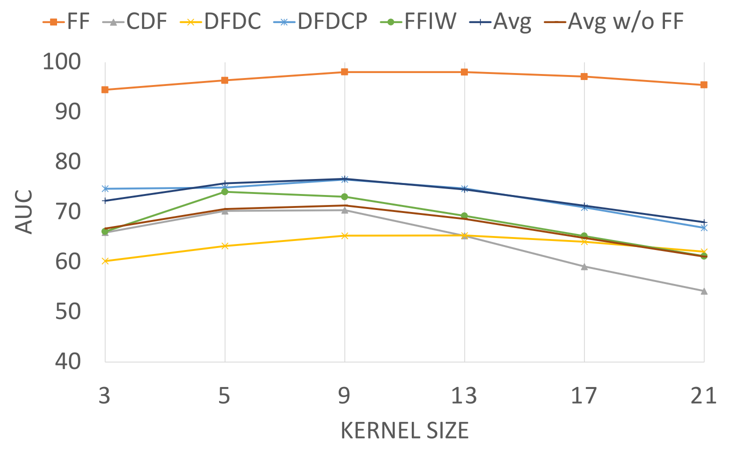

Fig. 5. AUC comparison of different boundary-blurring degrees. We visualize the experiments with different Gaussian kernel sizes with fixed regions in the mask.

different transformation methods into three types: color transform, frequency transform, and affine transform. For color transformation, we select data augmentation techniques related to image color, including RGBShift, HueSaturationValue, and RandomBrightnessContrast. Frequency transform primarily involves random downscaling or sharpening. Affine transform mainly involves random translation, scaling, and elastic deformation. According to the results shown in Table 5, we observed a significant improvement in model generalization with affine transform, achieving an average generalization improvement of 10.15%. This indicates the importance of affine transformations in enhancing model generalization. On the other hand, the effectiveness of frequency transforms varied across different datasets, which could be attributed to the variations in the types of forgery present in each dataset. Overall, combining all three transformation methods yielded the best results in terms of average improvement.

Exploration of Transform Objects. The Source-Target Preprocessing step primarily involves applying various transform operations to two objects, i.e. source images and target images. We explored different combinations of transform objects to investigate their impact on model generalization, as shown in Table 6. We observed that performing transforms on both the source and target

Table 5. AUC comparison of different transform methods.

Transform Method			Test Set AUC(%)						
Color	Frequency	Affine	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
✓	✓	✓	99.56	**91.23**	**72.10**	86.25	83.68	**86.56**	**83.32**
-	✓	✓	99.54	90.81	69.70	82.05	82.71	84.96	81.32
✓	-	✓	91.19	71.79	66.99	**86.72**	**89.81**	81.30	78.83
✓	✓	-	**99.62**	80.50	61.47	78.26	73.66	78.70	73.47

Table 6. AUC comparison of different transform objects.

	Test Set AUC(%)						
Name	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
source transform	99.19	74.09	65.73	84.82	**86.06**	81.98	77.68
target transform	98.10	90.76	66.16	77.65	64.90	79.51	74.87
target and source transform	98.89	89.19	71.95	**89.57**	80.83	86.09	82.89
target/source transform	**99.56**	**91.23**	**72.10**	86.25	83.68	**86.56**	**83.32**

Table 7. AUC comparison of different blending types.

	Test Set AUC(%)						
Name	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
dynamic blend	99.56	91.23	72.10	86.25	83.68	86.56	83.32
alpha blend	98.61	87.02	67.83	80.80	81.25	83.10	79.23

simultaneously ("target and source transform") or randomly selecting one with a probability of 0.5 ("target/source transform") allows for a greater variety of tampering simulations. Consequently, these approaches exhibit stronger generalization compared to solely applying transforms to either the source or target image.

Exploration of Blending Exploration of Blending Types. We experimented with two blending types: dynamic blend and alpha blend. Dynamic blend refers to a blending method where the blending ratio μ between two images is dynamically adjusted, i.e. we sample μ from $\{0.25, 0.5, 0.75, 1, 1, 1\}$ following SBI. On the other hand, alpha blend is a traditional blending technique that uses a fixed blending ratio $\mu = 1$. The experimental results for these two blending types are shown in Table 7. Compared to alpha blend, dynamic blend provides more variations and can simulate a wider range of tampering scenarios. Besides, the dynamic blend enables a seamless and natural transition between the source and target images, making it more closely resemble real-world attack scenarios. Therefore, dynamic blend exhibits superior generalization compared to alpha blend.

Table 8. Cross-dataset evaluation on CDF, DFDC, DFDCP, and FFIW. The results of prior methods are directly cited from SBI and the original paper, and their subsequences for fair comparisons. Bold and underline represent the first and second highest results, respectively. * denotes results reproduced by our own.

	Training Set		Test Set AUC(%)				
Method	Real	Fake	CDF	DFDC	DFDCP	FFIW	Avg
DSP-FWA[18]	✓	✓	69.30	-	-	-	69.30
Face X-ray + BI[17]	✓		-	-	71.15	-	71.15
Face X-ray + BI[17]	✓	✓	-	-	80.92	-	80.92
Two-branch[23]	✓	✓	76.65	-	-	-	76.65
DAM[38]	✓	✓	75.30	-	72.80	-	74.05
LipForensics[13]	✓	✓	82.40	-	-	-	82.40
FTCN[37]	✓	✓	86.90	71.00	74.00	74.47	76.59
LRL[3]	✓	✓	78.26	-	76.53	-	77.40
FRDM[22]	✓	✓	79.40	-	79.70	-	79.55
PCL+I2G[36]	✓		90.03	67.52	74.37		77.31
SBI[29]	✓		91.23*	72.10*	86.25*	83.68*	83.32*
SLADD[2]	✓	✓	79.70	-	76.00	-	77.85
Ours	✓		**91.37**	**72.98**	85.89	**87.77**	**84.50**

Table 9. Cross-manipulation validation on FF++.

	Test Set AUC(%)				
Method	DF	F2F	FS	NT	FF++
Face X-ray + BI[17]	99.17	98.57	98.21	98.13	98.52
PCL+I2G[36]	**100.00**	98.93	99.86	97.63	99.11
SLADD[2]	-	-	-	-	98.40
SBI[29]	99.99*	99.77*	**99.90***	**98.57***	**99.56***
Ours	99.98	**99.83**	99.89	98.38	99.52

4.3 Comparison Experiments

Cross-Dataset Evaluation. We conducted Cross-Dataset Evaluation to demonstrate the generalization of our method. The model was trained on FF++ and evaluated on other datasets, *i.e.* CDF, DFDC, DFDCP, and FFIW. From Table 8, our method basically outperformed all methods, with an average AUC of 84.50%. Furthermore, it shows improvements over the SBI on other datasets such as CDF, DFDC, and FFIW. Particularly on the FFIW dataset, our method achieves a significant improvement of 4.09% (from 83.6% to 87.77%) compared to the second-ranked method SBI, surpassing other methods by a large margin.

Cross-Manipulation Evaluation. Following the evaluation protocol used in previous methods, we tested our method on the FF+ with different tampering techniques, *i.e.* DeepFake (DF), Face2Face (F2F), FaceSwap (FS), and NeturalTextures (NT) to validate its generalization across various manipulations, as shown in Table 9. The experimental results demonstrate that our method performs on par with the SOTA for each manipulation method. Therefore, our

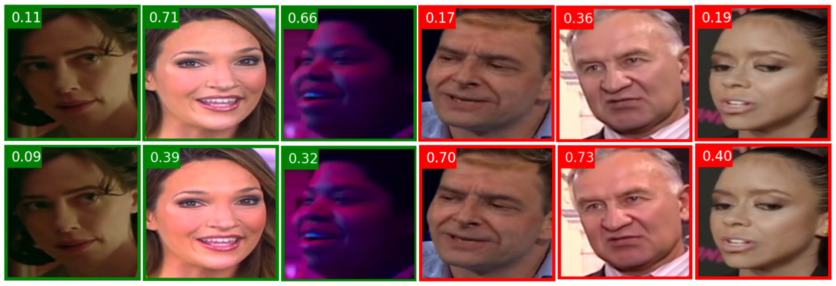

Fig. 6. Qualitative Analysis. We compare our method (second row) with the baseline (first row) by showing a cross-section of visual examples of real (Green) and fake (Red) faces with different anomaly scores.

method not only improves cross-dataset generalization but also maintains accuracy on the FF++ (cross-manipulation).

Qualitative Analysis. We compare our method with the baseline by showing a cross-section of visual examples of real and fake samples with different anomaly scores. This analysis helps us understand the strengths and limitations of our method. According to Figure 6, it can be observed that our method has improved the generalization of the model in distinguishing real faces, particularly in real facial images with noticeable contrast (corresponding to the second column in the figure) and color deviations (third column). This demonstrates the effectiveness of our NRS method in suppressing noise and enhancing generalization. In addition to the improvement in generalization for real faces, our method also exhibits significant enhancements in the accuracy of detecting fake faces, both in cases of full-face tampering (fourth column) and localized tampering (mouth area of figures in the fifth column). These results provide further evidence of the effectiveness of BBMG method. However, admittedly, there is still room for improvement when dealing with highly realistic fake face images (corresponding to the sixth column in the figure).

Robustness Study. To evaluate the robustness of our method, we applied different image distortion methods to raw images from all test datasets. Our robustness testing is conducted using weights trained on raw images without any fine-tuning. The result is shown in Table 10. The distortion types we considered include 1) "Resize": the process of downsampling the original image to an $s \times s$ size and then upsampling it back to the original image size, simulating the compression process during image transmission, 2) Gaussian blurring with a kernel size of k, and 3) JPEG compression with a quality factor q. The experimental results demonstrated that our method consistently outperformed the baseline in terms of generalization, regardless of the distortion method applied. Moreover, in most cases, our method exhibited superior performance on the FF++ dataset compared to the baseline. These findings confirm the robustness of our proposed method in handling various image distortions.

Table 10. Robustness analysis of the proposed method.

Operations	Baseline		Ours	
	FF++	Avg	FF++	Avg
Raw	99.56	83.32	99.52	84.50
Resize(s=128)	98.00	81.17	98.16	82.26
Resize(s=256)	99.29	83.20	99.29	84.34
GaussianBlur(k=3)	99.35	80.34	99.32	82.09
GaussianBlur(k=5)	98.81	79.50	98.90	81.33
JPEGCompress(q=50)	92.48	81.95	93.35	83.04
JPEGCompress(q=100)	99.48	81.71	99.51	82.53

Table 11. Ablation study of the proposed method.

Methods		Test Set AUC(%)				
BBMG	NRS	CDF	DFDC	DFDCP	FFIW	Avg
-	-	91.23	72.10	86.25	83.68	83.32
✓	-	91.37	72.57	85.66	86.31	83.98
-	✓	**91.38**	72.80	**86.50**	86.18	84.22
✓	✓	91.37	**72.98**	85.89	**87.77**	**84.50**

Table 12. Ablation study of Boundary Blur Mask Generator with different types of masks. We experimented with three forms of masks, which are visualized in Figure 7.

	Test Set AUC(%)						
Method	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
1	**99.64**	91.30	71.96	84.40	83.79	86.22	82.86
2	99.60	90.90	72.04	84.37	85.76	86.53	83.27
3	99.55	**91.37**	**72.57**	**85.66**	**86.31**	**87.09**	**83.98**

Ablation Study of the proposed method. In Table 11, we conducted an ablation study on the Boundary Blur Mask Generator (BBMG) and the Noise Refinement Strategy (NRS). From the experimental results, we observed that the inclusion of both methods improved performance on most datasets and the average AUC. Furthermore, the fusion of these two methods resulted in a further improvement in the average AUC. This confirms that both Boundary Blur Mask Generator and Noise Refinement Strategy contribute significantly to enhancing the model's generalization capabilities.

Ablation Study of Boundary Blur Mask Generator. To provide a clearer representation of the mask effects, we excluded the Weighted Loss Noise Refinement Strategy in this experiment and solely utilized our Boundary Blur Mask Generator. We experimented with three types of masks generated by the Boundary Blur Mask Generator. These three masks are 1) Using only the result of M^b_{global}(first column); 2)Narrowing M^b_{local}; and 3)The method mentioned in our method. The visual results are shown in Figure 7. The experimental results in Table 12 indicate that the first type of mask achieved the best performance

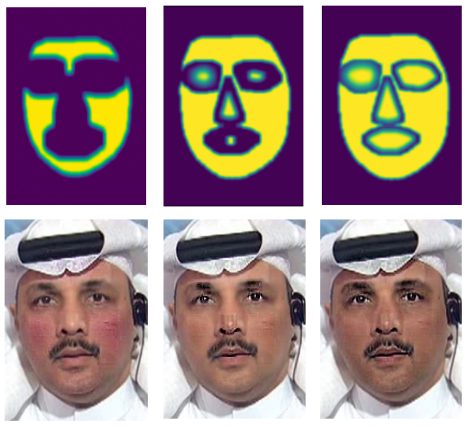

Fig. 7. Different masks in BBMG. We experimented with three forms of masks: using only the result of M_{global}^{b} (first column), the result of scaling M_{local}^{b} (second column), and the method mentioned in our method (third column).

Table 13. Ablation study of Noise Refinement Strategy.

Parameters		Test Set AUC(%)						
Filter	Distance	FF++	CDF	DFDC	DFDCP	FFIW	Avg	Avg w/o FF++
random	cosine	99.51	90.91	72.38	85.24	86.96	87.00	83.87
random	L2	99.52	90.85	72.39	85.37	87.12	87.05	83.93
predicts	cosine	99.52	91.37	72.98	85.89	87.77	87.51	84.50
predicts	L2	99.52	91.40	73.06	85.90	87.00	87.38	84.34

on the FF++ [26], exhibiting the highest accuracy. On the other hand, the third method (as mentioned in our approach) showed the most pronounced local tampering artifacts in the visual results and demonstrated the highest generalization performance across different datasets.

Ablation Study of Noise Refinement Strategy. We conducted various experiments with different Noise Refinement Strategies, as shown in Table 13. The "Filter" column refers to the method used to filter real face samples. In this part, we selected 500 samples from the training set as input for k-means clustering to calculate the cluster centers representing the distribution of real faces. The "random" strategy involved randomly sampling 500 real samples, while the "predicts" strategy involved selecting the 500 real samples with the highest predicted probabilities based on the model's predictions from the previous epoch. The "Distance" column indicates the distance metric used to measure the distance between generated fake samples and real samples. Specifically, we compared the cosine distance and L2 distance. The experimental results revealed that the cosine distance metric outperformed the L2 distance metric. Additionally, the strategy of selecting samples based on predicted probabilities showed improvement when using either the L2 or cosine distance metric.

5 Conclusion

In this paper, we first systematically analyze a range of existing anomalous deep forgery detection methods and identify the core factors that have a relatively large impact on generalisability, such as mask selection as well as boundary processing. In addition, we point out that previous work generates some pseudo-noise samples. To address these issues, we propose a new framework that contains Boundary Blur Mask Generator and Noise Refinement Strategy. Extensive experiments have fully demonstrated the excellent generalization capability of our approach, achieving state-of-the-art performance in deep forgery detection.

Acknowledgments This work was supported in part by the National Natural Science Foundation of China under Grants 62222602, 62302167, 62176224, 62106075, 61972157 and U23A20343, in part by Science and Technology Commission under Grant 21511100700, in part by CCF-Tencent Rhino-Bird Young Faculty Open Research Fund under Grant RAGR20230121.

References

1. Afchar, D., Nozick, V., Yamagishi, J., Echizen, I.: Mesonet: a compact facial video forgery detection network. In: 2018 IEEE international workshop on information forensics and security (WIFS). pp. 1–7. IEEE (2018)
2. Chen, L., Zhang, Y., Song, Y., Liu, L., Wang, J.: Self-supervised learning of adversarial example: Towards good generalizations for deepfake detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 18710–18719 (2022)
3. Chen, S., Yao, T., Chen, Y., Ding, S., Li, J., Ji, R.: Local relation learning for face forgery detection. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 1081–1088 (2021)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
5. Deng, J., Guo, J., Zhou, Y., Yu, J., Kotsia, I., Zafeiriou, S.: Retinaface: Single-stage dense face localisation in the wild. arXiv preprint arXiv:1905.00641 (2019)
6. Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., Ferrer, C.C.: The deepfake detection challenge (dfdc) dataset. arXiv preprint arXiv:2006.07397 (2020)
7. Dolhansky, B., Howes, R., Pflaum, B., Baram, N., Ferrer, C.C.: The deepfake detection challenge (dfdc) preview dataset. arXiv preprint arXiv:1910.08854 (2019)
8. Du, M., Pentyala, S., Li, Y., Hu, X.: Towards generalizable deepfake detection with locality-aware autoencoder. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 325–334 (2020)
9. Foret, P., Kleiner, A., Mobahi, H., Neyshabur, B.: Sharpness-aware minimization for efficiently improving generalization. arXiv preprint arXiv:2010.01412 (2020)
10. Frank, J., Eisenhofer, T., Schönherr, L., Fischer, A., Kolossa, D., Holz, T.: Leveraging frequency analysis for deep fake image recognition. In: International conference on machine learning. pp. 3247–3258. PMLR (2020)

11. Guan, J., Zhou, H., Hong, Z., Ding, E., Wang, J., Quan, C., Zhao, Y.: Delving into sequential patches for deepfake detection. Advances in Neural Information Processing Systems **35**, 4517–4530 (2022)
12. Güera, D., Delp, E.J.: Deepfake video detection using recurrent neural networks. In: 2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS). pp. 1–6. IEEE (2018)
13. Haliassos, A., Vougioukas, K., Petridis, S., Pantic, M.: Lips don't lie: A generalisable and robust approach to face forgery detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5039–5049 (2021)
14. Khalid, H., Woo, S.S.: Oc-fakedect: Classifying deepfakes using one-class variational autoencoder. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. pp. 656–657 (2020)
15. King, D.E.: Dlib-ml: A machine learning toolkit. The Journal of Machine Learning Research **10**, 1755–1758 (2009)
16. Li, J., Xie, H., Li, J., Wang, Z., Zhang, Y.: Frequency-aware discriminative feature learning supervised by single-center loss for face forgery detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6458–6467 (2021)
17. 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)
18. Li, Y., Lyu, S.: Exposing deepfake videos by detecting face warping artifacts. arXiv preprint arXiv:1811.00656 (2018)
19. Li, Y., Yang, X., Sun, P., Qi, H., Lyu, S.: Celeb-df: A large-scale challenging dataset for deepfake forensics. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3207–3216 (2020)
20. Liu, B., Liu, B., Ding, M., Zhu, T., Yu, X.: Ti2net: Temporal identity inconsistency network for deepfake detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 4691–4700 (2023)
21. Liu, H., Li, X., Zhou, W., Chen, Y., He, Y., Xue, H., Zhang, W., Yu, N.: Spatial-phase shallow learning: rethinking face forgery detection in frequency domain. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 772–781 (2021)
22. Luo, Y., Zhang, Y., Yan, J., Liu, W.: Generalizing face forgery detection with high-frequency features. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16317–16326 (2021)
23. Masi, I., Killekar, A., Mascarenhas, R.M., Gurudatt, S.P., AbdAlmageed, W.: Two-branch recurrent network for isolating deepfakes in videos. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16. pp. 667–684. Springer (2020)
24. Nguyen, H., Yamagishi, J., Echizen, I.: Use of a capsule network to detect fake images and videos. arXiv preprint arXiv:1910.12467 (2019)
25. Qian, Y., Yin, G., Sheng, L., Chen, Z., Shao, J.: Thinking in frequency: Face forgery detection by mining frequency-aware clues. In: European conference on computer vision. pp. 86–103. Springer (2020)
26. 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)

27. Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., Natarajan, P.: Recurrent convolutional strategies for face manipulation detection in videos. Interfaces (GUI) **3**(1), 80–87 (2019)
28. Shao, R., Wu, T., Liu, Z.: Detecting and recovering sequential deepfake manipulation. In: European Conference on Computer Vision. pp. 712–728. Springer (2022)
29. 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)
30. Tan, L., Wang, Y., Wang, J., Yang, L., Chen, X., Guo, Y.: Deepfake video detection via facial action dependencies estimation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 5276–5284 (2023)
31. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
32. Wei, J., Wang, S., Huang, Q.: F^3net: fusion, feedback and focus for salient object detection. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 12321–12328 (2020)
33. Yu, Y., Zhao, X., Ni, R., Yang, S., Zhao, Y., Kot, A.C.: Augmented multi-scale spatiotemporal inconsistency magnifier for generalized deepfake detection. IEEE Transactions on Multimedia (99), 1–13 (2023)
34. 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)
35. Zhao, H., Zhou, W., Chen, D., Zhang, W., Yu, N.: Self-supervised transformer for deepfake detection. arXiv preprint arXiv:2203.01265 (2022)
36. Zhao, T., Xu, X., Xu, M., Ding, H., Xiong, Y., Xia, W.: Learning self-consistency for deepfake detection. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15023–15033 (2021)
37. Zheng, Y., Bao, J., Chen, D., Zeng, M., Wen, F.: Exploring temporal coherence for more general video face forgery detection. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15044–15054 (2021)
38. Zhou, T., Wang, W., Liang, Z., Shen, J.: Face forensics in the wild. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5778–5788 (2021)