# Hiding Faces in Plain Sight: Defending DeepFakes by Disrupting Face Detection

Delong Zhu, Yuezun Li, *Member, IEEE*, Baoyuan Wu, *Senior Member, IEEE*, Jiaran Zhou, Zhibo Wang, *Senior Member, IEEE*, and Siwei Lyu, *Fellow, IEEE*

***Abstract*—Face-swapping DeepFakes have become an escalating societal concern, attracting increasing attention in recent years. To counter this, we investigate a new proactive defense framework to prevent individuals from being victimized in DeepFake videos. The core idea of this framework is to contaminate the inputs of DeepFake models by disrupting face detectors, based on the observation that face detectors are commonly used to automatically extract victim faces in most DeepFake techniques. Once the face detectors malfunction, the faces will not be correctly extracted, thereby impairing the training or synthesis stages of DeepFake models. To achieve this, we describe a strategy named *FacePoison*, which fools face detectors by adding dedicated adversarial perturbations to video frames. Building upon this, we introduce *VideoFacePoison*, an extended strategy that can efficiently propagate FacePoison across video frames instead of applying it individually to each frame, thus significantly reducing the computational overhead while retaining favorable attack performance. This framework is validated on five face detectors, and extensive experiments against eleven different DeepFake models demonstrate the effectiveness of disrupting face detectors to hinder DeepFake generation.**

***Index Terms*—DeepFake defense, multimedia forensics, face detection.**

Received 28 November 2024; revised 11 June 2025; accepted 18 July 2025. Date of publication 31 July 2025; date of current version 4 November 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62402464, in part by the Shandong Natural Science Foundation under Grant ZR2024QF035, in part by the China Postdoctoral Science Foundation under Grant 2021TQ0314, Grant 2021M703036, in part by the Sanya Science and Technology Special Fund under Grant 2022KJCX92. The work of Jiaran Zhou was supported by the National Natural Science Foundation of China under Grant 62102380, and in part by the Shandong Natural Science Foundation under Grant ZR2021QF095, Grant ZR2024MF083. The work of Baoyuan Wu was supported by Guangdong Basic and Applied Basic Research Foundation under Grant 2024B1515020095, in part by the National Natural Science Foundation of China under Grant 62076213, and in part by the Shenzhen Science and Technology Program under Grant RCYX20210609103057050. *(Corresponding author: Yuezun Li.)*

Delong Zhu, Yuezun Li, and Jiaran Zhou are with the School of Computer Science and Technology, Ocean University of China, Qingdao 266005, China (e-mail: zhudelong@stu.ouc.edu.cn; liyuezun@ouc.edu.cn; zhoujiaran@ouc.edu.cn).

Baoyuan Wu is with the School of Data Science, Chinese University of Hong Kong, Shenzhen 518172, China (e-mail: wubaoyuan@cuhk.edu.cn).

Zhibo Wang is with the School of Cyber Science and Technology, Zhejiang University, Hangzhou 310027, China, and also with the ZJU-Hangzhou Global Scientific and Technological Innovation Center, Hangzhou 311215, China (e-mail: zhibowang@zju.edu.cn).

Siwei Lyu is with University at Buffalo, Getzville, NY 14068 USA (e-mail: siweilyu@buffalo.edu).

The source code is publicly available at: https://github.com/OUC-VAS/FacePoison.

Digital Object Identifier 10.1109/TDSC.2025.3592230

## I. Introduction

RECENT advances in deep learning and the availability of a vast volume of online personal images and videos have drastically improved the synthesis of highly realistic human faces [1], [2], [3]. DeepFake is one of the most prevalent face forgery techniques that has drawn increasing attention recently [4], [5], [6], [7], [8], [9][1]. Primarily, DeepFake can swap the face with a synthesized target face while retaining the same facial attributes such as facial expression and orientation (Fig. 1). While there are interesting and creative applications of DeepFakes, they can also be weaponized to create illusions of a person's presence and activities that do not occur in reality, leading to serious political, social, financial, and legal consequences [17], [18].

Foreseeing this threat, many forensic methods aiming to detect DeepFake faces have been proposed recently [20], [21], [22], [23], [24], [25], [26], [27], [28]. However, given the speed and reach of the propagation of online media, even the currently best forensic method will largely operate in a postmortem fashion, applicable only after the fake face images or videos emerge. In this work, we aim to develop *proactive* approaches to protect individuals from becoming the victims of such attacks. Our solution is to add specially designed patterns known as adversarial perturbations that are imperceptible to human eyes but can result in face detection failures. The key idea is that: *DeepFake models, whether during training or inference, need many standard faces, i.e., face sets, collected using automatic face detection methods. In the training phase, a large number of standard faces are required as training sets. In the inference phase, standard faces also need to be extracted from each frame of the input videos. Our method aims to "pollute" this face set to disrupt the corresponding process* (see Fig. 2).

Our study, dubbed as *FacePoison*, focuses on adversarial attacks to the deep neural network (DNN)-based face detectors e.g., [29], [30], which have demonstrated superior performance over non-DNN approaches and exhibit greater robustness to variations in pose, expression, and occlusion. Specifically, we

[1]In this paper, the term DeepFake specifically refers to faces-swapping face forgery techniques, as originally defined in [10], where specific local facial content is replaced by newly synthesized content while retaining high visual quality. It is worth noting that the scope of DeepFake has recently expanded to encompass a wide range of AI-generated media, including full image synthesis and face editing techniques [11], [12], [13], [14], [15], [16]. Considering that face-swapping DeepFakes remain the most prevalent form encountered in online content, our work focuses on this category.

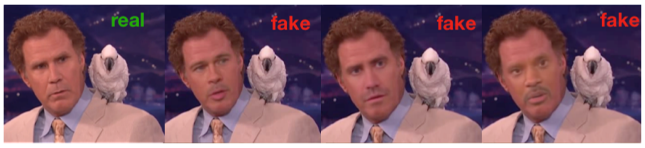

Fig. 1. Examples of DeepFake, which involves replacing the original faces with synthesized faces while keeping the same facial expressions. These examples are from [19].

adapt mainstream adversarial attack methods for this task, with dedicated modifications on objectives, attack places, and attack processes. We comprehensively evaluate their effectiveness in disrupting face detection and the defense ability against DeepFake models. This study has practical implications and could be deployed as a privacy-preserving service on social media platforms, or as a standalone tool for users to preprocess their images before uploading them online.

Moreover, considering that users frequently upload videos to social platforms, we explore a strategy called *VideoFacePoison*, an extension of FacePoison designed for effective video protection. Since the adjacent frames in a video have strong temporal connections, we hypothesize that the adversarial perturbations across these frames also share temporal consistency. Based on this insight, we develop an optical flow-based strategy to propagate adversarial perturbations from one frame to adjacent frames, instead of calculating FacePoison on all frames.

Extensive experimental evaluations are performed on five mainstream face detectors on public face detection datasets, showing the effectiveness of our method in disturbing face detectors. We then apply our method to obstruct eleven well-known DeepFake models. The results demonstrate that our method can effectively degrade the visual quality of DeepFake faces.

*It is important to highlight that the proposed framework is not a replacement but a complement to existing DeepFake detection methods.* The contribution of this paper is summarized as follows:

1) We describe a new proactive defense framework called *FacePoison* to obstruct DeepFake generation by disrupting DNN-based face detectors.
2) We tailor and apply several mainstream adversarial attack techniques to this context, incorporating task-specific modifications, and thoroughly assess their effectiveness under various conditions.
3) In light of the widespread sharing of personal videos online, we describe *VideoFacePoison*, which leverages temporal coherence within videos to propagate the FacePoison from a single frame to adjacent ones. Compared to applying FacePoison to all frames, VideoFacePoison offers an option to reduce computational overhead while retaining favorable attack performance.
4) We conduct extensive experiments on five mainstream DNN-based face detection methods and eleven DeepFake models, demonstrating the effectiveness of our method in obstructing DeepFake generation in both training and testing.

This work extends our preliminary study presented in the `ICME conference paper [31]` in the following aspects: 1) We adapt multiple mainstream adversarial attack methods to this task and comprehensively evaluate their feasibility and effectiveness; 2) We introduce VideoFacePoison, utilizing a novel optical flow-based strategy to propagate FacePoison across video frames; 3) We expand our experiments to include eleven well-known DeepFake models, covering defense scenarios in both training and testing phases.

## II. Background and Related Works

### A. Deepfake

DeepFake is a recent AI-based face forgery technique that can be used to create realistic impersonation videos. There are many variants of DeepFake versions to date, e.g., using different encoders, different decoders, or different loss functions [4], [5], [32], [33], [34]. These techniques share a common pipeline: Given an original video, the faces of source individuals are first cropped out using face detectors. Then these cropped faces are sent into a DeepFake model (e.g., auto-encoders [35], GANs [36] or Diffusion models [37]) to synthesize new blended faces. These synthesized faces belong to the source individuals but have the same facial attributes, such as facial expressions and head orientations, as in the original video. Nowadays, with the thriving of generative models, DeepFake has not been limited to the technique of face-swap and has become a general term for all AI-based face forgery techniques, e.g., face editing [16], full face generation [2], face inpainting [38], etc. This paper still focuses on defending the face-swap DeepFakes.

### B. Face Detection

Using DNNs in face detection has become mainstream with their high performance and robustness regarding variations in pose, expression, and occlusion. There has been a plethora of DNN-based face detectors e.g., [29], [30], [39], [40], [41]. Regardless of the idiosyncrasies of different detectors, they all follow a similar workflow, which predicts the location and confidence score of the potential candidate regions corresponding to faces in an end-to-end fashion. The prohibitive cost of searching optimal network structures and architectures makes the choice of the backbone network limited to four well-tested DNN models, namely, the VGG network [42], ResNet [43], MobileNet [44], and ShuffleNet [45] as reported on the leader board of the WIDER challenge [46].

### C. Adversarial Perturbations

Adversarial Perturbations are intentionally designed noises that are imperceptible to human observers yet can seriously reduce the performance of deep neural networks when added to the input image. Many methods [47], [48], [49], [50], [51], [52], [53], [54], [55], [56] have been proposed to impair image classifiers by adding adversarial perturbations on the entire image. The following briefly overviews several classic attack methods relevant to this work.

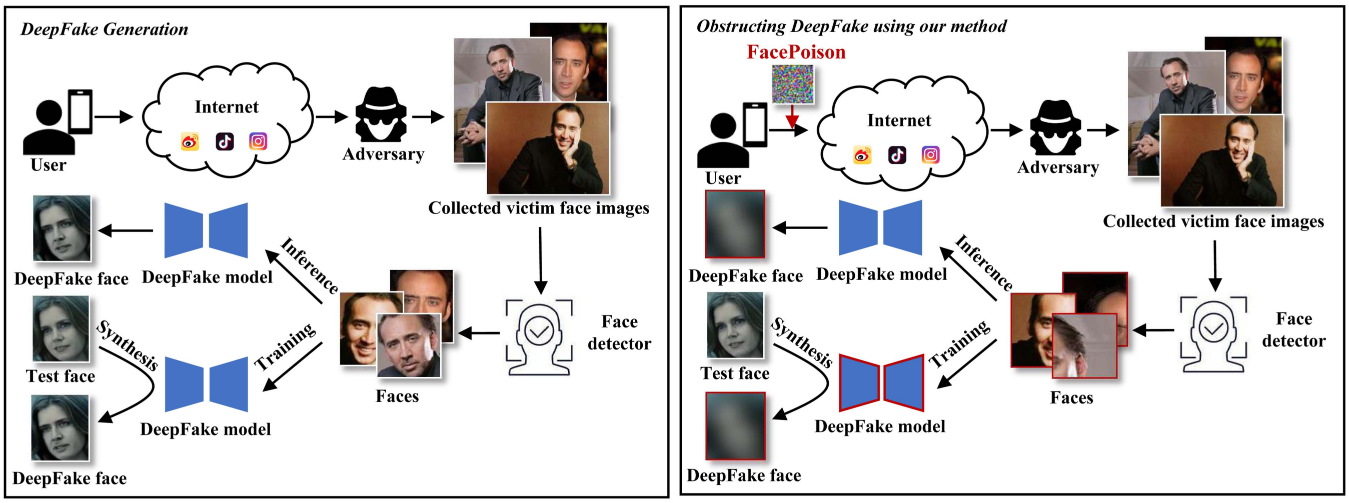

Fig. 2. Overview of FacePoison. The left part shows the typical generation process of DeepFake conducted by adversaries, covering the inference and training phase of DeepFake models. The right part shows how our method obstructs the DeepFake generation. The rationale is that our method disrupts face detection, leading to incorrect face detection results. It can pollute the input faces during either inference or training, ultimately hindering the DeepFake generation process.

*FGSM:* Fast Gradient Sign Method (FGSM) [47] is the method that can update the input image within a distortion bound in one step based on the sign of the gradient, calculated by a designed objective. Let $x, y$ be the input image and its ground-truth label, and $x^{adv}$ be the generated adversarial image. The update process to obtain $x^{adv}$ can be written as

$$x^{adv} = x + \alpha \cdot sign(\nabla_x \mathcal{L}(x, y)), \ s.t. \ ||x^{adv} - x||_\infty \leq \epsilon, \tag{1}$$

where $\mathcal{L}(\cdot, \cdot)$ is the designed objective function, $\nabla_x$ is the gradient with respect to image $x$, $\alpha$ is the step size and $sign$ is a function that retains the sign of values. $\epsilon$ is the distortion bound. Then the obtained image is truncated to a valid range [0,255].

*BIM:* Basic Iterative Method (BIM) [48] is extended from FGSM by using iterative steps. In each iteration $t$, the adversarial image $x^{adv}$ is updated by a step $\alpha$. This process is iterated until the distortion bound $\epsilon$ is reached, as

$$\begin{aligned} x_0^{adv} &= x, \\ x_{t+1}^{adv} &= x_t^{adv} + \alpha \cdot sign(\nabla_x \mathcal{L}(x_t^{adv}, y)). \end{aligned} \tag{2}$$

There are two options for the selection of $\alpha$. The first is to set $\alpha = \epsilon / T$, where $T$ is the total number of iterations, and clip $x_t^{adv}$ to a valid range in the end. The other one is to set $\alpha = \epsilon$ and clip $x_t^{adv}$ at each iteration. Since BIM goes iteratively, it is more likely to find a better optimal solution than FGSM.

*MIM:* Momentum Iterative Fast Gradient Sign Method (MIM) [50] improves the BIM method by considering the momentum when determining the direction of descent, as

$$\begin{aligned} x_0^{adv} &= x, g_0 = 0, \\ g_{t+1} &= \mu \cdot g_t + \frac{\nabla_x \mathcal{L}(x_t^{adv}, y)}{||\nabla_x \mathcal{L}(x_t^{adv}, y)||_1}, \\ x_{t+1}^{adv} &= x_t^{adv} + \alpha \cdot sign(g_{t+1}). \end{aligned} \tag{3}$$

*DIM:* Diverse Inputs Iterative Fast Gradient Sign Method (DIM) [55] improves the diversity of input images by adding random transformations. Denote the random transformation operation as $\mathcal{T}$. The formulation of DIM is similar to MIM as

$$\begin{aligned} g_{t+1} &= \mu \cdot g_t + \frac{\nabla_x \mathcal{L}(\mathcal{T}(x_t^{adv}), y)}{||\nabla_x \mathcal{L}(\mathcal{T}(x_t^{adv}), y)||_1}, \\ x_{t+1}^{adv} &= x_t^{adv} + \alpha \cdot sign(g_{t+1}). \end{aligned} \tag{4}$$

At each iteration, the input image will be transformed by $\mathcal{T}$.

*NIM:* Nesterov Iterative Fast Gradient Sign Method (NIM) [56] integrates Nesterov Accelerated Gradient (NAG) into gradient-based iterative attacks. Compared to MI-FGSM, NI-FGSM makes a step forward in the accumulated gradient direction before each iteration and then proceeds with the update, as

$$\begin{aligned} x_t^{nes} &= x_t^{adv} + \alpha \cdot \mu \cdot g_t, \\ g_{t+1} &= \mu \cdot g_t + \frac{\nabla_x \mathcal{L}(x_t^{nes}, y)}{||\nabla_x \mathcal{L}(x_t^{nes}, y)||_1}, \\ x_{t+1}^{adv} &= x_t^{adv} + \alpha \cdot sign(g_{t+1}). \end{aligned} \tag{5}$$

The existing studies mainly focus on attacking classifiers but pay less attention to the vulnerability of face detectors.

## D. DeepFake Defense

DeepFake detection is the classic passive defense solution to combat DeepFakes. Since the DeepFake is generated by generative models, the early forensics methods utilized traditional clues (e.g., PRNU [57], lighting and shadow inconsistency [58]) are not effective. As such, many dedicated DeepFake detection methods have been proposed recently, e.g., [21], [26], [59], [60], [61], [62]. These methods can fall into many categories, in terms of the way they seek the forgery traces, ranging from using the biological signals [21], [59], [63], color signals [64],

[65], [66], generation artifacts signals [22], [60] to the meticulously designed architectures and training schemes [20], [23], [24], [25], [27], [28] Most of these methods are based on Convolutional Neural Networks (CNNs) [42], [43] and Vision Transformers (ViT) [67], [68]. They can achieve favorable, even perfect, performance on public datasets. However, they are used to lagging behind the generation of DeepFakes, unable to cut off the broadcasting of DeepFakes at the first time.

Different from DeepFake detection methods, the proactive DeepFake defense methods aim to obstruct the generation of DeepFakes. To achieve this goal, adversarial perturbations are usually utilized to malfunction the synthesis process of generative models, such as attacking the latent representations of VAEs [69], attacking the image-to-image translation models in autonomous driving [70], or targeting the generative models including GANs to disrupt the visual quality of synthesized faces [71], [72], [73], [74]. These methods focus on the generative model itself and design suitable adversarial perturbations to disrupt its original behavior. In this paper, we shift the attention from directly disrupting the DeepFake model to the data preparation step, disrupting the face detection to corrupt extracted faces.

## III. FacePoison: Disrupting Face Detection

### A. Threat Model

*1) Defender's Capacity:* The users are allowed to process their photos before uploading them online. Once the photos have been uploaded, they can hardly control where their photos will go, and whether the attackers have collected their photos to train DeepFake models. The users can also hardly intervene or access the training and testing process of the DeepFake model, e.g., the training configuration, and even the model architectures. Once the DeepFake model finishes training, the users can not intervene in the synthesis (inference) process, as this model takes the target face (not from this user) as input and outputs the victim's face (from this user). Thus, the users can not protect or are not responsible for the protection of the photos of other individuals. The described defender's capacity is the minimal requirement, being able to be applied in real-world scenarios: *Users protect themselves from being the victim of DeepFakes by only manipulating their photos before uploading them online.*

*2) Defender's Goals:* By manipulating the user's photos, the faces can not be extracted by DNN-based face detectors, which contaminates the input data of the DeepFake model, subsequently obstructing their regular training or testing. Moreover, this manipulation should be imperceptible as much as possible, avoiding disturbing the original semantic content of photos.

### B. Problem Formulation

Denote $\mathcal{D}_x = \{x_i\}_{i=1}^N$ as the set of users' photos collected by attackers. Denote $\mathcal{F}$ as the face detector and $d_i = \mathcal{F}(x_i)$ denotes the detected face given image $x_i$. For simplicity, assume that only one face exists in an image. Denote $\mathcal{D}_d = \{d_i\}_{i=1}^N$ as the set of extracted faces using face detector $\mathcal{F}$ and $\mathcal{D}_g = \{g_i\}_{i=1}^N$ as the set of ground truth faces. Note that $\mathcal{D}_d$ is used to train the DeepFake model or the input faces for DeepFake faces generation. Our goal is to disrupt face detection and contaminate $\mathcal{D}_d$ with incorrect faces, by manipulating the images $x_i$ in $\mathcal{D}_x$ with minimal distortion, i.e., enlarging the error between $d_i$ and $g_i$. If attackers attempt to train a DeepFake model using the corrupted $\mathcal{D}_d$, the training process will malfunction. Similarly, if attackers use a pre-trained DeepFake model for face synthesis, the contaminated $\mathcal{D}_d$ will also impair the inference process.

Denote $\theta$ as the parameters of the face detector $\mathcal{F}$. Let $x$ be the clean image containing the victim's faces. We omit the subscript $i$ for simplicity. Disrupting face detection is to find the adversarial image $x^{adv}$ that can fool $\mathcal{F}$, while it is visually similar to the clean image $x$. Let $\mathcal{L}$ be the objective function to disrupt the face detector. This problem can be formulated as

$$\arg\min_{x^{adv}} \ \mathcal{L}(x^{adv}, x; \theta), \quad s.t. \ ||x^{adv} - x||_\infty \leq \epsilon, \tag{6}$$

where $\epsilon$ is the bound of distortion. Minimizing this equation can find an adversarial image to fool the face detector.

### C. Adapting Adversarial Attacks to Face Detectors

Existing adversarial attack methods mainly focus on classifiers, which cannot be directly used to disrupt face detectors. Therefore, we adapt the mainstream adversarial attack methods reviewed in Section II-C to fit this task. Specifically, we introduce the following modifications:

*1) Attacking Intermediate Features:* Instead of focusing on the final outputs, we target the intermediate features based on these intuitions: (1) Friendliness: Face detectors use a variety of detection heads, but their base networks are typically limited to a small number of common architectures. Targeting intermediate features ensures compatibility across different detectors. (2) Effectiveness: Intermediate features contain the important information that determines the results, thus disrupting features can naturally lead to incorrect detection. (3) Transferability: Features are more general and versatile compared to final logits, which are often overfitted to specific architectures. Thus, attacking features enhance transferability across different detectors.

Concretely, our method attacks multiple layers of the base network. Given a clean image $x$, the feature at $i$-th layer is denoted as $h_i = \mathcal{F}_i(x)$ and $\mathcal{H} = \{h_i\}_{i=1}^K$ denotes the feature set obtained from $K$ feature layers. Our method aims to mislead the face detector $\mathcal{F}$ by disturbing $\mathcal{H}$. The overview of our method is illustrated in Fig. 3.

*2) Attack Objectives:* For the conventional adversarial attacks, the objective function relates to the task, which can be easily formulated given the training samples with their corresponding annotations (e.g., labels for classification task). However, since the features are not visually understandable, how to design objectives to disturb features is important.

In general, the elements in features have different impacts on the results. Disturbing the high-impact elements can be more effective in disrupting the results. Inspired by [75], [76], we design an importance-guided map for each scale. The larger value in this map represents the greater importance of corresponding feature elements. Denote $\mathcal{H}' = \{h'_i\}_{i=1}^K$ as the corresponding set given the adversarial image $x^{adv}$. Our objective function can be

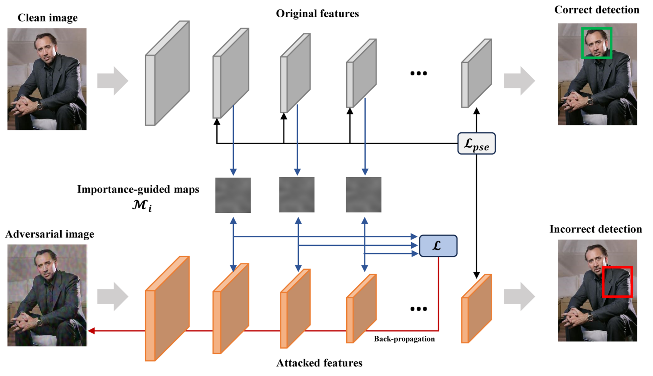

Fig. 3. Overview of our method on disrupting face detection. Our method attacks multiple intermediate features with the instruction of importance-guided maps, amplifying the disturbance on key elements indicated by these maps. See text for details.

defined as

$$\mathcal{L}(x^{adv}, x; \theta) = \sum_{i \in \phi} \alpha_i \cdot (\mathcal{M}_i \cdot h'_i), \tag{7}$$

where $\phi \subseteq \{1, 2, \ldots, K\}$ is a subset of all feature layer indexes, $\alpha_i$ is the weight factor for each scale, $\mathcal{M}_i$ is the importance-guided map for each scale. Minimizing this equation can decrease the response of important feature elements while increasing the response of unimportant feature elements, effectively disrupting the normal distribution of features.

One straightforward way to decide whether a feature element is important is to back-propagate the gradients from the task-related objective of training the face detectors to the features $\mathcal{H}'$. However, the objective of training face detectors likely varies due to their different architectures and training schemes, hindering the generalization of attacks, as the type of face detectors needs to be known in advance. As such, we propose a pseudo-objective that considers feature differences to replace the task-related objective, disentangling the relationship between the attack and its original task-related objective. Specifically, we select the last feature layer $h_K$ from the clean feature set $\mathcal{H} = \{h_i\}_{i=1}^K$ as a reference. Then we calculate the distance between this clean feature layer $h_K$ and the attacked feature layer $h'_K$ to approximate the error in the task-related objective. We use the last feature layer for the pseudo-objective as the deep layer has more concentrated, high-level semantic information than the shallow layer, which can better depict the task-related objective. We use cosine similarity to measure the feature difference as

$$\mathcal{L}_{pse}(x^{adv}, x; \theta) = \frac{h_K \cdot {h'_K}^\top}{||h_K|| \cdot ||h'_K||}. \tag{8}$$

For a feature layer $h'_i$ that is going to be attacked, we can obtain the importance-guided map $\mathcal{M}_i$ by back-propagating the gradient of $\mathcal{L}_{pse}$ to the $i$-th layer as

$$\mathcal{M}_i = \frac{\partial \mathcal{L}_{pse}(x^{adv}, x; \theta)}{\partial h'_i}. \tag{9}$$

Since $\mathcal{M}_i$ is composed of the gradients, it may have noise specific to the model architecture. To reduce these noises, we apply random masking on the input image for $m$ times and calculate the importance-guided map $\mathcal{M}'_i$ for each time. The random masking is to set the pixel value of the input image to zero with the probability $p$. Then we average these importance-guided maps and normalize the averaged map as the final $\mathcal{M}_i$.

## IV. VideoFacePoison: Propagating FacePoison Across Frames in Video

Based on FacePoison, we describe VideoFacePoison to protect videos by propagating adversarial perturbations across frames. While videos can be protected by directly applying perturbations to each frame, this process is computationally expensive. By leveraging the temporal correlations between adjacent frames, we are inspired to improve the efficiency of video protection.

*Hypothesis 1:* The temporal relationships across adjacent frames can also be represented on adversarial perturbations. Therefore, the adversarial perturbations created on one frame can be propagated to adjacent frames while retaining their effectiveness.

This hypothesis stems from the observation that the adversarial perturbations are highly related to image content [77] and can be estimated from nearby frames using the optical flow algorithms. Thus, we utilize optical flow algorithms to estimate and propagate adversarial perturbations.

*3) Revisit of Optical Flow:* Basically, optical flow indicates the displacement of objects among adjacent frames. Optical flow algorithms assume that the intensity of pixels in an object remains unchanged among adjacent frames. Denote the intensity of a pixel at location $(x, y)$ and time $t$ as $I(x, y, t)$. We reuse the notation $x, t$ here to illustrate optical flow better. Denote $\Delta x$ and $\Delta y$ are the displacements during time $\Delta t$. Under the above assumption, an equation can be written as $I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)$. Then we can obtain $0 = \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t$ using Taylor expansion. Many methods exist to calculate optical flow, e.g., Gunnar Farneback [78], FlowNet [79], etc.

*4) Method Introduction:* Denote $\mathcal{X} = \{x_{(v)}\}_{v=0}^V$ as a video, and $v$ is the time index of the frame in this video. The optical flow map from the $v$-th frame to the $v'$-th frame can be denoted as $\mathcal{H}_{v \to v'}$. Given the FacePoison at $v$-th frame, we can estimate the FacePoison for the adjacent $v'$-th frame by mapping the perturbations based on $\mathcal{H}_{v \to v'}$ as

$$x_{(v')}^{adv} = \mathcal{P}(x_{(v)}^{adv} - x_{(v)}, \mathcal{H}_{v \to v'}) + x_{(v')}, \tag{10}$$

where $\mathcal{P}$ is a mapping function that can map the pixels in the current frame to adjacent frames based on the optical flow in the forward direction. As the optical flow algorithms are not perfect, they can also introduce artifacts. To reduce the artifacts, we also consider the optical flow of backward direction as $\mathcal{H}_{v' \to v}$, and can obtain the prediction at $v'$-th frame by reversing the $\mathcal{H}_{v' \to v}$ as

$$x_{(v')}^{adv'} = \mathcal{P}(x_{(v)}^{adv} - x_{(v)}, -\mathcal{H}_{v' \to v}) + x_{(v')}. \tag{11}$$

Then we average these two predictions as the FacePoison for the $v'$-th frame. This propagation will stop after a certain time in case

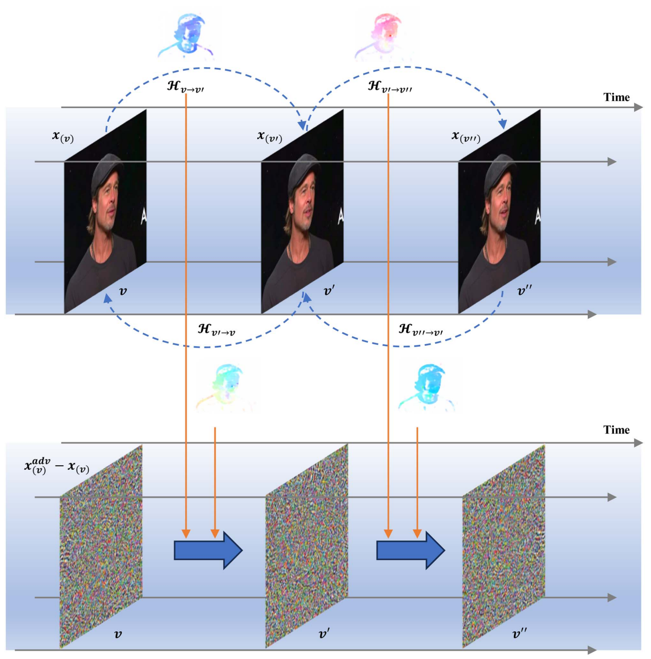

Fig. 4. Overview of the VideoFacePoison method on disrupting face detection in a video. Our method can propagate the FacePoison on a certain frame ($x_{(v)}$ in this example) to adjacent frames based on the optical flow algorithms. See text for details.

the faces have an apparent displacement, and then we recalculate the FacePoison and propagate it again. This procedure will be repeated until all frames are processed. The overview of our VideoFacePoison method is shown in Fig. 4.

## V. Experiments

### A. Experimental Settings

*1) Datasets:* We employ WIDER [46], a widely used face detection dataset, to demonstrate the efficacy of our methods. This dataset contains plenty of faces with location annotations. Moreover, to evaluate the efficacy in obstructing DeepFake generation, we conduct validations of our method using four standard DeepFake datasets, including FaceForensics++ (FF++) [80], Celeb-DF [19], DeepFake Detection Challenge (DFDC) dataset [81] and preview version of DFDC (DFDCP) dataset [82].

*2) Face Detectors:* We consider five mainstream DNN-based face detectors in experiments, which are **RetinaFace** [39], **YOLO5Face** [40], **PyramidBox** [30], **S3FD** [29], and **DSFD** [41], respectively. RetinaFace is built upon MobileNet-0.25 [44]. YOLO5Face is redesigned from the YOLOV5 object detector [83], which is based on ShuffleNetv2 [45]. PyramidBox, S3FD, and DSFD utilize VGG16 as their basenetworks. All of these face detectors are trained under their default settings.

*3) Adapted Adversarial Attack Methods:* We adapt the adversarial attack methods BIM [48], DIM [55], MIM [50], and NIM [56] into this task, using the proposed modifications. We denote these adapted methods as **Ada-BIM**, **Ada-DIM**, **Ada-MIM**, and **Ada-NIM**, respectively. Moreover, inspired by [75], [84], we incorporate operations of spectrum transformation and gradient averaging operations into DIM to improve the robustness. We term this incorporation as **Ada-DIM++**. Note that spectrum transformation increases data diversity by randomly modifying the frequency components of the input, while gradient averaging helps stabilize the attack direction.

*4) DeepFake Models:* To fully demonstrate the effectiveness of our method, we evaluate our method on **eleven** modern DeepFake models, including four representative academic methods: **SimSwap** (ACM MM'20) [6], **InfoSwap** (CVPR'21) [7], **MobileFaceSwap** (AAAI'22) [8], **BlendFace** (ICCV'23) [9], five variants from the open-source tool *FaceSwap* [4] (namely **Origin**, **DFaker**, **IAE**, **LightWeight**, and **DFLH**) and two additional models from [19] (namely **CDFv1** and **CDFv2**). SimSwap[2] adopts an auto-encoder structure with an Identity Injection Module to preserve identity features during synthesis. InfoSwap[3] introduces an Informative Identity Bottleneck and Adaptive Information Integrator to decouple and integrate identity information. MobileFaceSwap[4] is a lightweight model that leverages knowledge distillation to achieve high-quality face swapping with reduced computational cost. BlendFace[5] separates identity and attribute features using a dedicated face recognition module and identity encoder for better face synthesis. The FaceSwap variants (Origin, DFaker, IAE, LightWeight, DFLH) share a typical encoder-decoder framework, with variations in synthesis resolution, encoder/decoder architectures, and training strategies. Detailed configurations are available in the official repository. Similarly, CDFv1 and CDFv2 are designed following FaceSwap with improvement on synthesis size and spatial-temporal alignment. We also note that *DeepFaceLab* [85] is another open-source tool for face swapping. Since this repository shares a similar generation pipeline to FaceSwap, we use FaceSwap for demonstration for simplicity.

*5) Evaluation Metrics:* Since our goal is to obstruct the DeepFake generation by disrupting face detection, we evaluate our method from two perspectives: whether the face detection is disrupted and whether the DeepFake generation is obstructed. We use F1-score to evaluate the attacking performance of our method, which is calculated as

$$\text{F1-score} = 2 \cdot \frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}.$$

F1-score considers both the precision and recall, which is more comprehensive than the Average Precision (AP) metric (See Section V-D for more details). The lower score of these metrics means the faces are not well detected. To evaluate the performance of DeepFake obstruction, we employ the SSIM score to measure the visual quality of the generated DeepFake. The larger score indicates that the quality of DeepFake faces is better.

*6) Implementation Details:* The experiments are conducted on Ubuntu 22.04 with one Nvidia 3060 GPU. The number of feature layers is set to $K = 3$. We target specific intermediate feature layers for each face detector architecture to conduct the

[2]https://github.com/neuralchen/SimSwap
[3]https://github.com/GGGHSL/InfoSwap-master
[4]https://github.com/Seanseattle/MobileFaceSwap
[5]https://github.com/mapooon/BlendFace

TABLE I
F1-Score (%) of Different Methods on WIDER Dataset

| Methods | RF | YF | PB | S3FD | DSFD |
|---|---|---|---|---|---|
| None | 94.0 | 98.4 | 98.7 | 98.0 | 98.6 |
| Random | 93.8 | 98.3 | 98.7 | 98.2 | 98.3 |
| Ada-BIM | 1.4 | 0.0 | 1.7 | 1.6 | 0.3 |
| Ada-DIM | 9.7 | 0.8 | 18.8 | 53.0 | 41.2 |
| Ada-MIM | 1.5 | 0.0 | 3.5 | 13.3 | 3.8 |
| Ada-NIM | 1.6 | 0.0 | 3.5 | 13.6 | 3.6 |
| Ada-DIM++ | 0.6 | 0.3 | 0.4 | 3.9 | 5.0 |

RF, YF, and PB denote RetinaFace, YOLO5Face, and PyramidBox, respectively.

attack. For PyramidBox, we target the feature outputs following the ReLU activations after conv3_3 and conv4_3, and the convolutional output of conv5_3. These correspond to the 15th, 22nd, and 28th feature layers in the VGG16 backbone, respectively. For S3FD and DSFD, we target three feature layers: conv2_2 (7th), pool3 (16th), and a high-level context convolutional layer (31st). For YOLO5Face, we target the feature outputs after the final SiLU activations in the first three ShuffleNetV2 blocks of the backbone. For RetinaFace, we target the feature outputs after the final LeakyReLU activations in the three stages (stage1, stage2, and stage3) of its MobileNet-0.25 backbone. The parameter settings for FacePoison are as follows: $\alpha_1 = 0.2, \alpha_2 = 0.3, \alpha_3 = 0.5, p = 0.9, m = 30$. In Ada-DIM++, we introduce spectrum transformation and gradient averaging. The standard deviation is set to $\sigma = 16$, the fluctuation range to $\mu = 0.5$, the number of spectrum transformations per iteration to $n = 10$, and the gradient averaging window to $o = 4$. The maximum iteration number is 10, and the bound of distortion is set to $\epsilon = 8$. Note that the bound of distortion in attacking general image classification is usually set to 16, see [48], [50], [55]. Given the higher sensitivity of face images, we restrict the bound to 8 to maintain imperceptibility. For VideoFacePoison, we utilize Gunnar Farneback [78] to calculate optical flow.

## B. Results of Disrupting Face Detection

*1) Results:* Table I shows the F1-score (%) of different methods attacking face detectors on the WIDER dataset. RF, YF, and PB denote RetinaFace, YOLO5Face, and PyramidBox, respectively. None denotes that no noises are added to images. By adding random noises, the performance of these methods merely drops, which indicates their robustness against random noises. It can be seen that the adapted methods Ada-BIM, Ada-DIM, Ada-MIM, Ada-NIM, and Ada-DIM++ can effectively disrupt face detection results. Several visual examples are illustrated in Fig. 5.

*2) Ablation Study:* Using Ada-DIM++ as an example, we study the effect of attacking multiple layers on the WIDER dataset. Specifically, we conduct experiments by attacking various numbers of layers. The results are shown in Table II. Here, L1, L2, and L3 denote the first, second, and third layers attacked by our method. The top three rows show the performance when only a single layer is attacked, while the bottom row corresponds to our method of attacking multiple layers. We can observe that disturbing all three layers achieves the best performance

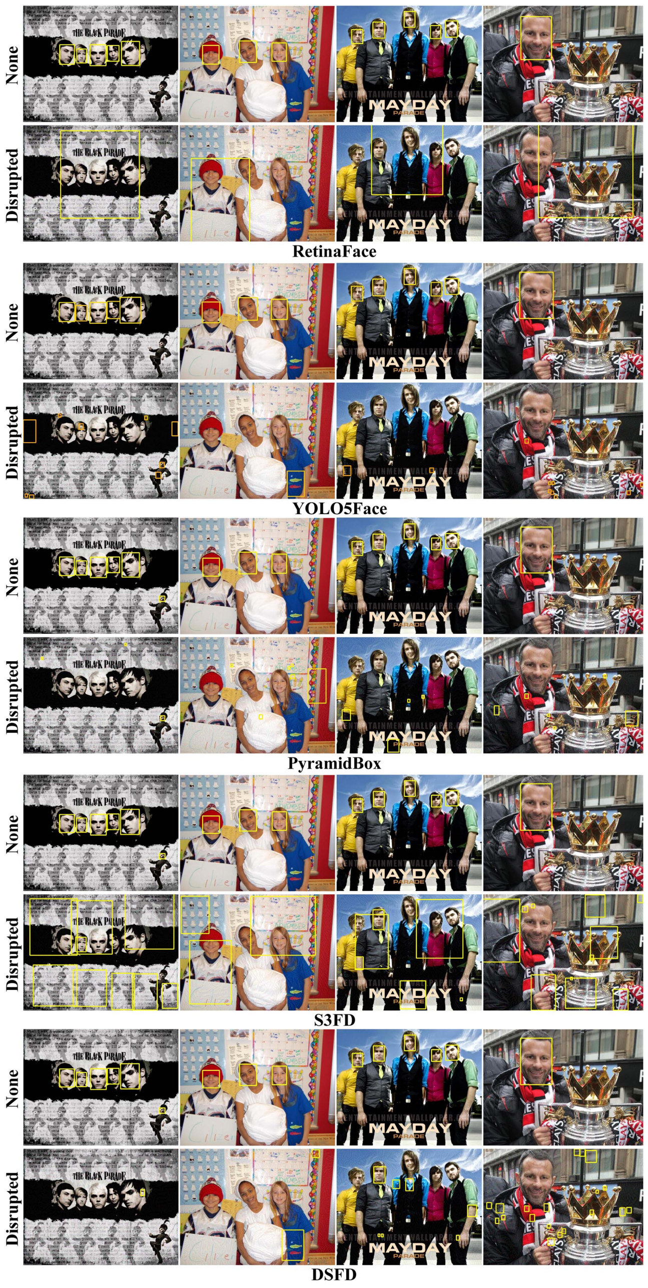

Fig. 5. Visual examples of using different adaptations to disrupt face detection. For each face detector, the top row corresponds to the original results and the bottom row is the results using our method.

compared to single layer attacks, demonstrating the efficacy of targeting multiple layers.

*3) Transferability:* We further explore the transferability of our method, using Ada-DIM++ as an example. Transferability refers to the ability to attack one face detector (target) using adversarial perturbations generated from another face detector (source). The results are shown in Table III, where "S" and "T" denote source and target face detectors, respectively. Our method demonstrates transferability in most cases. For example, using DSFD to attack RetinaFace can notably reduce the performance from 94.0% to 70.6% while attacking YOLO5Face degrades

TABLE II
EFFECT OF ATTACKING MULTIPLE LAYERS

| | RF | YF | PB | S3FD | DSFD |
|---|---|---|---|---|---|
| L1 | 61.9 | 57.2 | 87.2 | 98.3 | 98.5 |
| L2 | 0.7 | 0.7 | 15.7 | 67.2 | 84.8 |
| L3 | 0.7 | 0.8 | 0.5 | 3.8 | 9.1 |
| L1,L2,L3 | 0.6 | 0.3 | 0.4 | 3.9 | 5.0 |

TABLE III
TRANSFERABILITY OF OUR METHOD ON DISRUPTING FACE DETECTION

| S↓, T→ | RF | YF | PB | S3FD | DSFD |
|---|---|---|---|---|---|
| RF | 0.6 | 89.8 | 97.6 | 96.4 | 98.4 |
| YF | 84.2 | 0.3 | 97.1 | 97.0 | 98.3 |
| PB | 76.8 | 72.2 | 0.4 | 8.3 | 2.9 |
| S3FD | 66.9 | 77.3 | 20.9 | 3.9 | 16.5 |
| DSFD | 70.6 | 70.3 | 11.3 | 9.4 | 5.0 |

"S" and "T" denote source and target face detectors, respectively.

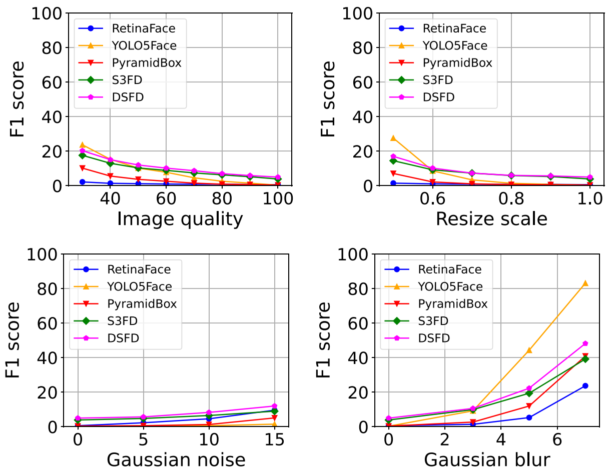

Fig. 6. Performance of different face detectors under different image quality, resize scale, Gaussian noise, and Gaussian blurring.

its performance from 98.4% to 70.3%. The transferability is particularly pronounced among PyramidBox, S3FD, and DSFD, greatly decreasing the performance below 21%. However, when RetinaFace or YOLO5Face is used as the source to attack the other three detectors, the transferability is ineffective. We hypothesize that this is due to the lightweight nature of these source detectors compared to the target detectors, which hinders knowledge transfer, thus limiting the effectiveness of adversarial perturbations.

*4) Robustness:* Since the social platform usually processes the uploaded images, we study the robustness of our method in this part, including the resistance to image compression, resizing, Gaussian noise, and Gaussian blurring. 1) Image compression: We use OpenCV to control image quality, varying the compression factor from $[30, 100]$, where a larger value indicates higher quality. Fig. 6 (top left) shows the results of our method confronting different image compression operations. We can see that the F1-score of these face detectors is still around 10% at quality 50, indicating resilience to image compression. 2) Resizing: To counter adversarial perturbations, transformation-based defenses such as random resizing and padding [86] are commonly employed. We randomly resize input images with a ratio in $[0.5, 1]$. As shown in Fig. 6 (top right), the F1-score decreases only slightly as the resizing ratio decreases, demonstrating a degree of robustness. 3) Gaussian noise: We add random Gaussian noise with standard deviations of 5, 10, and 15 to simulate varying levels of distortion. Fig. 6 (bottom left) shows that our method remains effective even at higher noise levels. 4) Gaussian blur: Robustness to Gaussian blur is evaluated using different kernel sizes. As shown in Fig. 6 (bottom right), our method maintains good performance with kernel sizes of 3 and 5, further highlighting its robustness.

TABLE IV
THE PERFORMANCE OF DIFFERENT FACE DETECTORS UNDER VARIOUS COUNTERMEASURES

| | RF | YF | PB | S3FD | DSFD |
|---|---|---|---|---|---|
| None | 94.0 | 98.4 | 98.7 | 98.0 | 98.6 |
| Ada-DIM++ | 0.6 | 0.3 | 0.4 | 3.9 | 5.0 |
| NRP | 54.5 | 72.5 | 86.2 | 78.5 | 81.7 |
| Facebook | 0.8 | 1.1 | 0.6 | 5.0 | 5.9 |
| Twitter | 0.8 | 0.7 | 0.6 | 4.6 | 5.4 |

To further study the robustness, we apply countermeasures such as adaptive attacks and platform-level defenses on our method. Specifically, we employ a representative adaptive attack *NRP*, an adversarially trained purifier proposed in [87] to eliminate our perturbations. For platform-level defenses, we select Facebook and Twitter as our target social media platforms. Due to upload limitations on these platforms, we follow the offline processing strategy described in [73]. The detailed experimental results are shown in Table IV. None indicates original detection performance without any perturbations. Ada-DIM++ is our method for disrupting face detectors. The results show that NRP partially mitigates the effect of our method, recovering some functionality of face detectors. Nevertheless, our method still has an effect under this defense. In contrast, the processing in social platforms has a negligible effect on our method, demonstrating its robustness in real-world usage scenarios.

*5) Results of VideoFacePoison:* Since the WIDER dataset is image-based, we use the videos in the Celeb-DF dataset to validate our method. Specifically, we select three identities (id0, id1, id2), each with 10 videos. Since these videos do not have face annotations, we use the results of face detectors as the ground truth. For each video, we first calculate the FacePoison on the first frame and then propagate this perturbation across frames. As the subject in a video moves very smoothly with minimal changes between adjacent frames, we evaluate the performance of our method with an interval of 5 frames. After a certain propagation length, we recalculate FacePoison on a new frame and repeat the process until all frames are processed.

Table V shows the performance of VideoFacePoison. FP-all denotes that we apply FacePoison on every frame. FP-fixed means we apply the FacePoison of the first frame to all other frames. FP-forward represents that we only consider the optical flow from the current frame to the next frame using (10). It is important to note that CADVG [88] also uses optical flow in adversarial perturbation generation, but it is originally designed to attack DeepFake detectors, not face detectors. Unlike

TABLE V
F1-Score (%) of VideoFacePoison and Its Variants

| | RF | YF | PB | S3FD | DSFD | Avg. |
|---|---|---|---|---|---|---|
| FP-all | 0.0 | 0.4 | 1.0 | 2.6 | 23.4 | 5.48 |
| FP-fixed | 47.4 | 28.1 | 39.1 | 33.1 | 51.8 | 39.9 |
| FP-forward | 46.7 | 55.8 | 22.9 | 20.2 | 44.2 | 38.0 |
| CADVG [88] | 56.1 | 69.0 | 0.7 | 63.1 | 0.0 | 37.8 |
| VideoFacePoison | 33.6 | 59.0 | 19.6 | 15.0 | 45.5 | 34.5 |

FP-all denotes applying the FacePoison on every frame. FP-fixed denotes applying the FacePoison of the first frame to all other frames. FP-forward represents only considering the optical flow in the forward direction.

TABLE VI
The Efficiency (Seconds Per Frame) of VideoFacePoison and Other Methods Across Different Face Detectors

| | RF | YF | PB | S3FD | DSFD | Avg. |
|---|---|---|---|---|---|---|
| FP-all | 1.18 | 1.60 | 13.13 | 10.53 | 10.92 | 7.47 |
| CADVG | 0.36 | 0.39 | 2.11 | 1.32 | 2.07 | 1.25 |
| VideoFacePoison | 0.21 | 0.19 | 0.36 | 0.28 | 0.29 | 0.27 |

our method, it requires creating adversarial perturbations for every video frame, using optical flow as a restriction loss to smooth perturbations across adjacent frames. For a comprehensive comparison, we adapt this method to our task. Since the authors have not released the source code, we implement the algorithm rigorously following their description in the paper. It can be seen that using FP-all can achieve the best performance compared to others if the time cost is not considered. Since the subject in the video does not have dramatic movement, FP-fixed also has a decent effect on detection disruption. In comparison, FP-forward propagates the adversarial perturbations to adjacent frames instead of using the fixed pattern, it surpasses the FP-fixed on average. Our method, which incorporates both forward and backward optical flow for propagation, largely outperforms FP-forward on average. Compared to CADVG, our method can achieve better performance, even though it involves additional adversarial optimization on every frame. Note that on YOLO5Face, the optical flow propagation is less effective. This is likely due to the base architecture, YOLOv5, which relies on complex feature fusion and multi-scale processing when predicting detection boxes, which requires more precise adversarial perturbations. Consequently, propagated perturbations may lose effectiveness. The similar trend is also observed in CADVG. Despite this, the average results demonstrate that our optical flow-based propagation strategy remains effective for disrupting face detection, aligning with our hypothesis that adversarial perturbations strongly correlate with the semantic content of the images.

Moreover, we evaluate the computational efficiency of the proposed VideoFacePoison method. The results are shown in Table VI. It can be seen that the improved VideoFacePoison method is significantly more efficient than FP-all, due to the newly proposed optical flow strategy, which enables effective propagation of perturbations across video frames. Additionally, when compared to CADVG, our method still achieves faster runtime, demonstrating superior computational efficiency.

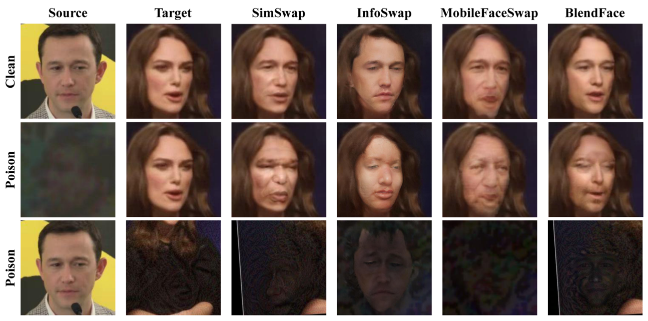

Fig. 7. Visual examples of our method obstructing the inference phase of DeepFake models. The first row shows the effect of results using clean images. The second and third rows show the results corresponding to AO (source image is polluted while target image is original) and OA (source image is original while target image is polluted) scenarios. It can be seen that the polluted image can not extract the face correctly, thus notably degrading the visual quality of generated faces.

### C. Results of Obstructing DeepFake Generation

*1) Obstructing Inference:* We validate the effectiveness of our method by obstructing the inference process of all DeepFake models on FF++, Celeb-DF, DFDC, and DFDCP datasets with all face detectors. For each dataset, we randomly select three identities (id0, id1, id2) and make three pairs of identities. Then we contaminate one identity in each pair using our method and randomly select a detection box as the input face. We use an empty image (all 0 pixels) as the input face for the cases that contain no detection boxes.

For the DeepFake models SimSwap, InfoSwap, MobileFaceSwap, and BlendSwap, we use their officially released weights and generate faces with default settings. The performance of obstructing inference is shown in Table VII. Since these models require both source and target identity as input, we design two scenarios to fully demonstrate the efficacy of our method. The first is "AO", where the source identity is polluted while the target identity is original. The second is "OA", which represents the opposite scenario. "Random" represents a baseline where random Gaussian noise is added instead of adversarial perturbations, with equivalent distortion strength. The synthesis quality is assessed using SSIM, where higher scores indicate better visual quality. The results show that random noise can hardly affect the synthesis quality, whereas our method achieves favorable performance under both scenarios, notably degrading the visual quality of DeepFake output. Fig. 7 shows examples of our method obstructing these DeepFake models under AO and OA scenarios.

Note that in the inference phase, DeepFake models such as Origin, IAE, LightWeight, DFaker, DFLH, CDFv1, and CDFv2 require the same identity used in the training phase. Therefore, we infer three variants of each model using the corresponding images of all identity pairs on the Celeb-DF dataset. Table VIII shows corresponding results, revealing that our method greatly reduces the visual quality of DeepFake faces.

*2) Obstructing Training:* To validate the effectiveness of obstructing training, we apply our method to pollute the training faces of one identity while leaving the other one untouched.

TABLE VII
OBSTRUCTING INFERENCE: SSIM SCORE (%) OF SYNTHESIZED FACES BY OBSTRUCTING THE INFERENCE PHASE OF DEEPFAKE MODELS USING DIFFERENT FACE DETECTORS ON DIFFERENT DATASETS

| Dataset | DF model | Method | RetinaFace | | YOLO5Face | | PyramidBox | | S3FD | | DSFD | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | AO | OA | AO | OA | AO | OA | AO | OA | AO | OA |
| FF++ | SimSwap | Random | 99.8 | 88.4 | 99.8 | 88.5 | 99.8 | 88.8 | 99.8 | 87.7 | 99.8 | 88.2 |
| | | Ours | 71.1 | 10.0 | 39.1 | 24.7 | 83.7 | 24.8 | 88.7 | 15.6 | 83.9 | 7.5 |
| | InfoSwap | Random | 98.3 | 88.7 | 98.3 | 88.6 | 98.4 | 89.2 | 98.4 | 88.3 | 98.4 | 88.6 |
| | | Ours | 66.0 | 18.8 | 36.8 | 33.0 | 78.2 | 39.4 | 84.0 | 27.3 | 79.4 | 12.9 |
| | MobileFaceSwap | Random | 99.9 | 88.9 | 99.9 | 89.1 | 99.9 | 89.3 | 99.9 | 88.3 | 99.9 | 88.8 |
| | | Ours | 72.8 | 11.2 | 39.8 | 25.1 | 85.3 | 26.7 | 90.6 | 16.9 | 84.8 | 8.0 |
| | BlendSwap | Random | 99.7 | 87.2 | 99.7 | 87.4 | 99.6 | 87.7 | 99.7 | 86.5 | 99.7 | 87.0 |
| | | Ours | 75.1 | 12.4 | 40.6 | 25.2 | 88.6 | 27.1 | 94.2 | 18.8 | 85.5 | 8.8 |
| CDF-v2 | SimSwap | Random | 99.8 | 88.9 | 99.7 | 88.3 | 99.6 | 88.7 | 99.7 | 88.8 | 99.6 | 88.6 |
| | | Ours | 74.0 | 14.1 | 47.5 | 18.7 | 82.6 | 28.3 | 88.9 | 27.6 | 66.8 | 33.4 |
| | InfoSwap | Random | 98.6 | 92.2 | 98.5 | 91.8 | 98.4 | 92.0 | 98.5 | 92.2 | 98.4 | 92.1 |
| | | Ours | 70.3 | 28.6 | 44.9 | 28.5 | 77.9 | 43.6 | 84.1 | 43.6 | 63.2 | 48.5 |
| | MobileFaceSwap | Random | 99.8 | 89.7 | 99.9 | 89.0 | 99.8 | 89.4 | 99.8 | 89.5 | 99.8 | 89.4 |
| | | Ours | 76.4 | 15.3 | 49.2 | 19.2 | 85.2 | 29.5 | 91.7 | 29.0 | 68.8 | 35.0 |
| | BlendSwap | Random | 99.7 | 88.4 | 99.7 | 87.8 | 99.6 | 88.2 | 99.6 | 88.3 | 99.6 | 88.1 |
| | | Ours | 78.7 | 18.4 | 50.2 | 20.1 | 87.1 | 31.0 | 93.8 | 30.3 | 70.0 | 36.1 |
| DFDC | SimSwap | Random | 99.6 | 90.7 | 99.6 | 90.0 | 99.7 | 90.2 | 99.6 | 90.4 | 99.6 | 90.3 |
| | | Ours | 26.8 | 18.8 | 24.6 | 28.9 | 35.2 | 35.9 | 92.0 | 23.0 | 77.2 | 34.7 |
| | InfoSwap | Random | 98.5 | 94.0 | 98.5 | 93.5 | 98.6 | 93.7 | 98.5 | 93.8 | 98.4 | 93.7 |
| | | Ours | 26.1 | 30.0 | 23.9 | 39.0 | 34.5 | 49.5 | 89.0 | 35.3 | 74.4 | 48.3 |
| | MobileFaceSwap | Random | 99.8 | 91.4 | 99.7 | 90.7 | 99.8 | 90.8 | 99.7 | 91.1 | 99.7 | 90.9 |
| | | Ours | 27.2 | 19.8 | 25.1 | 29.8 | 35.9 | 36.3 | 93.6 | 23.4 | 78.7 | 35.6 |
| | BlendSwap | Random | 99.7 | 90.0 | 99.7 | 89.3 | 99.7 | 89.4 | 99.7 | 89.7 | 99.7 | 89.6 |
| | | Ours | 27.7 | 20.8 | 25.3 | 30.9 | 36.4 | 38.8 | 95.1 | 24.6 | 79.6 | 36.9 |
| DFDCP | SimSwap | Random | 99.4 | 86.6 | 97.4 | 84.9 | 98.3 | 85.9 | 97.7 | 86.1 | 98.1 | 86.8 |
| | | Ours | 80.5 | 12.2 | 56.4 | 16.3 | 88.5 | 35.0 | 89.2 | 25.1 | 84.4 | 32.2 |
| | InfoSwap | Random | 98.2 | 89.1 | 96.2 | 88.0 | 97.1 | 89.1 | 96.3 | 88.0 | 96.8 | 89.4 |
| | | Ours | 76.3 | 20.7 | 53.5 | 24.3 | 84.3 | 50.7 | 85.3 | 37.3 | 79.6 | 47.2 |
| | MobileFaceSwap | Random | 99.6 | 87.8 | 97.0 | 86.3 | 99.0 | 87.5 | 97.7 | 86.7 | 98.0 | 88.2 |
| | | Ours | 82.8 | 13.5 | 58.8 | 16.5 | 90.3 | 36.4 | 91.8 | 25.7 | 86.2 | 34.6 |
| | BlendSwap | Random | 99.4 | 86.1 | 97.2 | 84.8 | 98.4 | 85.7 | 97.3 | 85.6 | 97.8 | 85.8 |
| | | Ours | 84.8 | 14.3 | 59.5 | 18.2 | 91.8 | 37.8 | 93.9 | 26.9 | 87.4 | 34.5 |
| Avg. | SimSwap | Random | 99.7 | 88.7 | 99.6 | 87.9 | 99.4 | 88.4 | 99.2 | 88.3 | 99.3 | 88.5 |
| | | Ours | 63.1 | 13.8 | 41.9 | 22.2 | 72.5 | 31.0 | 89.7 | 22.8 | 78.1 | 27.0 |
| | InfoSwap | Random | 98.4 | 91.0 | 97.9 | 90.5 | 98.1 | 91.0 | 97.9 | 90.6 | 97.0 | 91.0 |
| | | Ours | 59.7 | 24.5 | 39.8 | 31.2 | 68.7 | 45.8 | 85.6 | 35.9 | 74.2 | 39.2 |
| | MobileFaceSwap | Random | 99.8 | 89.4 | 99.1 | 88.8 | 99.6 | 89.3 | 99.3 | 88.9 | 99.4 | 89.3 |
| | | Ours | 64.8 | 17.5 | 43.2 | 22.7 | 74.2 | 32.2 | 91.9 | 23.8 | 79.6 | 28.3 |
| | BlendSwap | Random | 99.6 | 87.9 | 99.1 | 87.3 | 99.3 | 87.8 | 99.1 | 87.5 | 99.2 | 87.6 |
| | | Ours | 66.6 | 16.5 | 43.9 | 23.6 | 76.0 | 33.7 | 94.3 | 25.2 | 80.6 | 29.1 |

AO: the source identity is polluted while the target identity is original. OA: the source identity is original while the target identity is polluted.

TABLE VIII
OBSTRUCTING INFERENCE: SSIM SCORE (%) OF SYNTHESIZED FACES BY OBSTRUCTING THE INFERENCE PHASE OF DEEPFAKE MODELS USING DIFFERENT FACE DETECTORS ON THE CELEB-DF DATASET

| DF model | Method | RF | YF | PB | S3FD | DSFD |
|---|---|---|---|---|---|---|
| Origin | Random | 96.5 | 96.4 | 97.1 | 96.1 | 97.1 |
| | Ours | 11.6 | 11.4 | 19.6 | 19.0 | 23.5 |
| IAE | Random | 96.3 | 96.2 | 96.9 | 95.8 | 96.9 |
| | Ours | 11.8 | 11.3 | 19.3 | 18.7 | 23.3 |
| LightWeight | Random | 96.4 | 96.2 | 97.0 | 95.9 | 97.0 |
| | Ours | 11.2 | 11.0 | 19.0 | 18.0 | 23.0 |
| DFaker | Random | 96.1 | 95.9 | 96.7 | 95.7 | 96.5 |
| | Ours | 16.7 | 17.7 | 29.5 | 28.8 | 31.2 |
| DFLH | Random | 96.0 | 95.9 | 96.5 | 95.5 | 96.4 |
| | Ours | 17.1 | 17.0 | 28.6 | 27.3 | 30.4 |
| CDFv1 | Random | 96.5 | 96.3 | 97.0 | 96.0 | 96.9 |
| | Ours | 17.9 | 17.7 | 30.5 | 28.7 | 31.7 |
| CDFv2 | Random | 96.9 | 96.8 | 97.3 | 96.7 | 97.3 |
| | Ours | 27.1 | 24.1 | 41.8 | 40.4 | 41.9 |

Following a similar setup as in the previous section, we select three identities (id0, id1, id2) and train DeepFake models using these identity pairs. Our method is validated using all face detectors. For each detector, we train $7 \times 3 \times 3 = 63$ models: seven DeepFake models trained individually three times, corresponding to three identity pairs, with each pair having three variants—clean training data, training data with random noise, and training data polluted by our method. Since our method is validated on five face detectors, a total of $\mathbf{63 \times 5 = 315}$ DeepFake models are trained.

After training, we feed correct face images into obstructed DeepFake models to see whether the synthesized faces are significantly disrupted. Table IX shows the SSIM score of synthesized faces after using our method. We can observe that the random noise can hardly affect the synthesis quality, but our method can greatly reduce the SSIM score, demonstrating the efficacy of our method in obstructing the training of DeepFake models. Fig. 8 shows visual examples. It can be seen that our method can effectively degrade the visual quality of DeepFake faces.

Remarkably, our method reduces the SSIM scores to single digits for models trained with the YOLO5Face detector. This is because our method fully disrupts this face detector, resulting

TABLE IX
OBSTRUCTING TRAINING: SSIM SCORE (%) OF SYNTHESIZED FACES USING OUR METHOD ON THE CELEB-DF DATASET

| DF model | Method | RF | YF | PB | S3FD | DSFD |
|---|---|---|---|---|---|---|
| Origin | Random | 90.8 | 90.5 | 90.8 | 90.6 | 90.7 |
| | Ours | 62.9 | 70.4 | 75.7 | 76.7 | 83.9 |
| IAE | Random | 92.2 | 92.1 | 92.1 | 92.1 | 92.3 |
| | Ours | 48.5 | 1.7 | 80.3 | 82.5 | 86.6 |
| LightWeight | Random | 92.3 | 92.4 | 92.4 | 92.3 | 92.2 |
| | Ours | 71.4 | 77.0 | 80.9 | 81.9 | 86.9 |
| DFaker | Random | 89.3 | 89.6 | 89.9 | 89.7 | 89.3 |
| | Ours | 41.9 | 2.2 | 70.6 | 72.5 | 56.7 |
| DFLH | Random | 93.2 | 92.9 | 92.9 | 92.8 | 92.9 |
| | Ours | 49.4 | 2.4 | 80.4 | 81.9 | 87.1 |
| CDFv1 | Random | 88.7 | 88.9 | 88.7 | 88.7 | 88.9 |
| | Ours | 42.9 | 25.1 | 73.4 | 74.1 | 81.5 |
| CDFv2 | Random | 91.0 | 90.6 | 90.8 | 90.6 | 91.2 |
| | Ours | 48.9 | 3.2 | 79.7 | 80.5 | 85.0 |

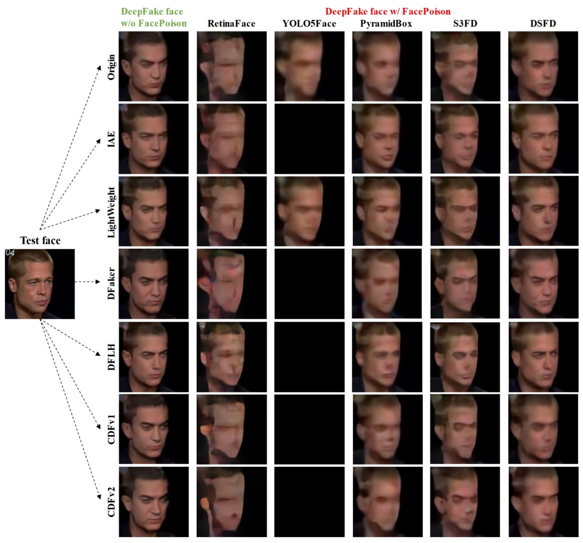

Fig. 8. Visual examples of DeepFake faces. The first column shows DeepFake faces without our method. The other right columns show DeepFake faces with our method.

TABLE X
OBSTRUCTING TRAINING: SSIM SCORE (%) OF SYNTHESIZED FACES USING PYRAMIDBOX FACE DETECTOR ON OTHER IDENTITIES

| Origin, PB | (id2, id4, id10) | (id12, id23, id5) | (id6, id9, id16) |
|---|---|---|---|
| Random | 92.4 | 90.1 | 90.8 |
| Ours | 74.8 | 73.6 | 71.8 |

in no detection boxes appearing. Consequently, no areas are extracted as training faces, leaving the corresponding regions as all-zero pixels. Therefore, given an input face, this DeepFake model can only generate images with all-zero pixels, as shown in Fig. 8.

Training a single model is time-consuming, typically taking around, making it very difficult to validate all identities in Celeb-DF. To verify the generalizability of our method to different identity selections, we randomly select three additional sets of identities: (id2, id4, id10), (id12, id23, id5), and (id6, id9, id16). Using the PyramidBox face detector, we train the Origin DeepFake model with these identity sets. The results are shown in Table X, which reveals that show that identity selection does not affect the effectiveness of our method.

TABLE XI
OBSTRUCTING TRAINING: SSIM SCORE (%) OF SYNTHESIZED FACES USING OUR METHOD ON FF++ DATASET

| DF model | Random | Ours |
|---|---|---|
| Origin | 81.7 | 32.7 |
| IAE | 79.5 | 49.1 |
| LightWeight | 82.8 | 36.7 |
| DFaker | 84.3 | 38.8 |
| DFLH | 84.6 | 50.8 |
| CDFv1 | 79.2 | 42.6 |
| CDFv2 | 85.1 | 58.8 |

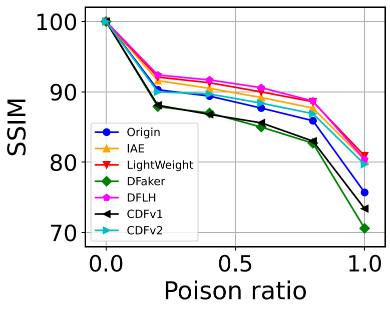

Fig. 9. Effect of different poison ratios used for polluting training faces. 0 denotes all training samples are clean and 1 denotes our method is applied to all training faces.

Moreover, we perform our method on FF++ datasets for a comprehensive evaluation. We randomly select three identities, extract faces using the PyramidBox detector, and train all DeepFake models following the same protocol. The performance is shown in Table XI, demonstrating the effectiveness of our method in disrupting the generative quality.

*3) Effect of Poison Ratio:* This part investigates the effect of the poison ratio of training faces, i.e., the relationship between the synthesis quality and the ratio of incorrect faces in training. Specifically, we change the poison ratio in a range of $[0, 1]$, where 0 denotes our method is not applied and 1 denotes our method is applied to all training faces. Fig. 9 illustrates the curve of the SSIM score with different poison ratios. It can be seen that the training faces do not need to be fully disrupted by our method, in order to reduce the synthesis quality, e.g., the SSIM score is still around 90% at poison ratio 0.2.

*4) Adapted Comparisons:* For a comprehensive evaluation, we incorporate two representative works, CMUA [74] and DF-RAP [73], as baselines and assess their performance of obstructing inference on the Celeb-DF dataset. Note that both methods are designed to directly disrupt DeepFake output rather than interfere with the face detection process, thus not fully aligning with our setting. For a fair comparison, we follow their default setting and adapt them to the extracted face images. For CMUA, we use the universal perturbation provided in their official repository. For DF-RAP, we utilize the pretrained generator they released. As shown in Table XII, CMUA and DF-RAP exhibit unsatisfactory effectiveness in disrupting the synthesized results when applied to DeepFake models outside of those used in their original studies. This indicates their limited generalizability, in contrast to the broader applicability demonstrated by our defense strategy.

TABLE XII
ADAPTED COMPARISONS WITH OTHERS

| DF model | Method | RetinaFace | | YOLO5Face | | PyramidBox | | S3FD | | DSFD | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | AO | OA | AO | OA | AO | OA | AO | OA | AO | OA |
| SimSwap | CMUA | 99.6 | 87.9 | 99.5 | 87.8 | 99.5 | 87.8 | 99.4 | 87.8 | 99.5 | 87.9 |
| | DF-RAP | 99.7 | 91.0 | 99.6 | 91.0 | 99.6 | 91.0 | 99.6 | 91.0 | 99.7 | 91.0 |
| | Ours | 74.0 | 14.1 | 47.5 | 18.7 | 82.6 | 28.3 | 88.9 | 27.6 | 66.8 | 33.4 |
| InfoSwap | CMUA | 98.6 | 94.6 | 99.5 | 94.6 | 98.4 | 94.5 | 98.4 | 94.5 | 98.5 | 94.6 |
| | DF-RAP | 98.6 | 97.0 | 99.7 | 97.0 | 98.5 | 96.9 | 98.5 | 96.9 | 98.6 | 97.0 |
| | Ours | 70.3 | 28.6 | 44.9 | 28.5 | 77.9 | 43.6 | 84.1 | 43.6 | 63.2 | 48.5 |
| MobileFaceSwap | CMUA | 99.8 | 88.6 | 98.5 | 88.6 | 99.7 | 88.6 | 99.7 | 88.5 | 99.8 | 88.6 |
| | DF-RAP | 99.9 | 92.5 | 98.6 | 92.4 | 99.8 | 92.4 | 99.9 | 92.5 | 99.9 | 92.5 |
| | Ours | 76.4 | 15.3 | 49.2 | 19.2 | 85.2 | 29.5 | 91.7 | 29.0 | 68.8 | 35.0 |
| BlendSwap | CMUA | 99.5 | 76.4 | 99.8 | 76.2 | 99.5 | 76.2 | 99.4 | 76.1 | 99.5 | 76.2 |
| | DF-RAP | 99.6 | 75.7 | 99.9 | 75.7 | 99.6 | 75.7 | 99.6 | 75.7 | 99.6 | 75.7 |
| | Ours | 78.7 | 18.4 | 50.2 | 20.1 | 87.1 | 31.0 | 93.8 | 30.3 | 70.0 | 36.1 |

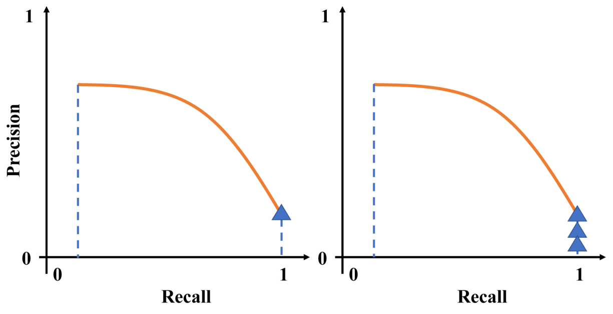

Fig. 10. Comparison of PR curves in different detection scenarios. The left example has a moderate number of candidate proposals (blue triangles), while the right has significantly more. Despite both detecting all ground-truth faces (recall = 1), the excessive redundant proposals in the right example reduce precision. However, the area under the curve (AP) remains unchanged. This illustrates that AP is not sensitive to redundancy. In contrast, our proposed use of F1-score penalizes such redundant detections, providing a more accurate assessment for defense purposes.

### D. Discussion

*1) F1-Score vs Average Precision (AP):* In the context of face detection, candidate proposals refer to all possible face bounding boxes predicted by the detector, each associated with a different confidence score. These proposals are essential for computing the Average Precision (AP) score, a widely used metric in the evaluation of face detection. To calculate the AP, all candidate proposals are first sorted by their confidence scores. Then, precision and recall are computed at various thresholds, where precision indicates the proportion of predicted boxes that are accurate, and recall reflects the proportion of ground-truth faces that are correctly detected. The AP is obtained by plotting the precision-recall curve and calculating the area under this curve. However, this metric is not sensitive to redundant face candidate proposals, see Fig. 10. In face detection, this insensitivity is acceptable: high Average Precision (AP) can still be achieved as long as all faces are correctly detected, even if many redundant proposals exist. But it becomes problematic in our task, where redundant face proposals can significantly contaminate the training face set. Therefore, we use the F1-score to directly measure precision and recall, providing a more appropriate and reliable evaluation metric for our specific objective.

*2) Limitations:* Our method is specifically dedicated to defending face-swap based DeepFake, which relies on extracting the face region as a critical initial step. Therefore, it is not effective for other DeepFake techniques mentioned in related works, such as face editing and entire-face generation, as face detection is not the prerequisite step in their inference or training process. As a potential direction for future work, it would be valuable to explore hybrid defense strategies that not only disrupt the face detection pipeline but also interfere with latent facial attribute representations within generative models, thereby improving the generalizability of proactive defense mechanisms.

Another limitation is that our method focuses on the modern DNN-based face detectors. Traditional face detection tools, such as those based on non-DNN machine learning models (e.g., SVMs in Dlib [89]), are difficult to be affected by our method. These non-DNN models typically have far fewer parameters than DNN models, making it more challenging to create effective adversarial perturbations. In future work, we intend to explore more advanced strategies beyond adversarial perturbation, such as manipulating high-level semantic presentations of facial appearance. This may help bypass architectural constraints and extend defense capabilities across diverse detector types. While our current method targets DNN-based face detectors, it serves as a proof of concept that proactive DeepFake defense via disrupting the face detection pipeline is both feasible and impactful, offering fresh insight for the community of DeepFake defense.

Furthermore, our method may be less effective against adaptive countermeasures employed by forgery makers, as discussed in Section V-B4. It is thus our continuing effort to improve the robustness of the adversarial effectiveness.

## VI. CONCLUSION

DeepFake is becoming a problem that is encroaching on our trust in online media. As DeepFake requires automatic face detection as an indispensable pre-processing step in preparing training data, an effective protection scheme can be obtained by disrupting the face detection methods. In this work, we develop a proactive protection method (FacePoison) to deter bulk reuse of automatically detected faces for the production of DeepFake faces. Our method exploits the sensitivity of DNN-based face detectors and uses adversarial perturbation to contaminate the face sets. This is achieved by a dedicated adaptation of mainstream adversarial attacks to disrupt face detectors. Moreover, we introduce VideoFacePoison, an extended strategy to propagate the

FacePoison across frames. In contrast to applying FacePoison to each frame individually, VideoFacePoison can only calculate adversarial perturbations on a certain set of frames and estimate the perturbations for others, while retaining the effectiveness of disrupting face detection. The experiments are conducted on eleven different DeepFake models with five different face detectors and empirically show the effectiveness of our method in obstructing DeepFakes in both inference and training phases.

*On one hand*, we expect this technology can offer fresh insights for the proactive DeepFake defense research and inspire more effective solutions to break the pipeline of the DeepFake generation, instead of only focusing on disrupting the DeepFake models themselves. *On the other hand*, the current forensics strategies are impossibly perfect for arbitrary scenarios, and these strategies can be hacked with a high probability once they are released. *Nevertheless, they still play a significant role. Even when hacked, these strategies raise the bar for creating forgeries, thereby slowing down the pace of their creation and dissemination.*

## References

[1] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–26.

[2] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4401–4410.

[3] T. Karras et al., "Alias-free generative adversarial networks," in *Proc. Conf. Neural Inf. Process. Syst.*, 2021, pp. 852–863.

[4] "Faceswap github," 2019. [Online]. Available: https://github.com/deepfakes/faceswap

[5] K. Liu, I. Perov, D. Gao, N. Chervoniy, W. Zhou, and W. Zhang, "Deepfacelab: Integrated, flexible and extensible face-swapping framework," *Pattern Recognit.*, vol. 141, 2023, Art. no. 109628.

[6] R. Chen, X. Chen, B. Ni, and Y. Ge, "SimSwap: An efficient framework for high fidelity face swapping," in *Proc. ACM Int. Conf. Multimedia*, 2020, pp. 2003–2011.

[7] G. Gao, H. Huang, C. Fu, Z. Li, and R. He, "Information bottleneck disentanglement for identity swapping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3404–3413.

[8] Z. Xu et al., "Mobilefaceswap: A lightweight framework for video face swapping," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 2973–2981.

[9] K. Shiohara, X. Yang, and T. Taketomi, "Blendface: Re-designing identity encoders for face-swapping," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 7600–7610.

[10] R. Spivak, ""Deepfakes": The newest way to commit one of the oldest crimes," *Georgetown Law Technol. Rev.*, 2018.

[11] S. Luo, C. Yan, C. Hu, and H. Zhao, "DIFF-FOLEY: Synchronized video-to-audio synthesis with latent diffusion models," in *Proc. Conf. Neural Inf. Process. Syst.*, 2024, pp. 48855–48876.

[12] D. Epstein, A. Jabri, B. Poole, A. Efros, and A. Holynski, "Diffusion self-guidance for controllable image generation," in *Proc. Conf. Neural Inf. Process. Syst.*, 2023, pp. 16222–16239.

[13] Y. Gong et al., "ToonTalker: Cross-domain face reenactment," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 7656–7666.

[14] S. Yang et al., "Learning dense correspondence for NeRF-based face reenactment," in *Proc. AAAI Conf. Artif. Intell.*, 2024, pp. 6522–6530.

[15] Z. Huang, K. C. Chan, Y. Jiang, and Z. Liu, "Collaborative diffusion for multi-modal face generation and editing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 6080–6090.

[16] X.-J. Li, D. Zhang, S.-Y. Chen, and F.-L. Liu, "StrokeFaceNeRF: Stroke-based facial appearance editing in neural radiance field," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 7538–7547.

[17] A. Busacca and M. A. Monaca, "Deepfake: Creation, purpose, risks," in *Proc. Innovations Econ. Social Changes Due Artif. Intelligence: State Art*, 2023, pp. 55–68.

[18] M. Tahraoui, C. Krätzer, and J. Dittmann, "Defending informational sovereignty by detecting deepfakes: Risks and opportunities of an AI-based detector for deepfake-based disinformation and illegal activities," in *Proc. Weizenbaum Conf. Practicing Sovereignty: Interv. Open Digit. Futures*, 2023, pp. 142–161.

[19] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for deepfake forensics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.

[20] Z. Yan, Y. Luo, S. Lyu, Q. Liu, and B. Wu, "Transcending forgery specificity with latent space augmentation for generalizable deepfake detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 8984–8994.

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

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

[23] C. Yu et al., "Diff-ID: An explainable identity difference quantification framework for deepfake detection," *IEEE Trans. Dependable Secure Comput.*, vol. 21, no. 5, pp. 5029–5045, Sep./Oct. 2024.

[24] J. Fei, Y. Dai, P. Yu, T. Shen, Z. Xia, and J. Weng, "Learning second order local anomaly for general face forgery detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 20238–20248.

[25] L. Chen, Y. Zhang, Y. Song, L. Liu, and J. Wang, "Self-supervised learning of adversarial example: Towards good generalizations for deepfake detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 18689–18698.

[26] Y. Wang, K. Yu, C. Chen, X. Hu, and S. Peng, "Dynamic graph learning with content-guided spatial-frequency relation reasoning for deepfake detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 7278–7287.

[27] Z. Ba et al., "Exposing the deception: Uncovering more forgery clues for deepfake detection," in *Proc. AAAI Conf. Artif. Intell.*, 2024, pp. 719–728.

[28] X. Cui, Y. Li, A. Luo, J. Zhou, and J. Dong, "Forensics adapter: Adapting clip for generalizable face forgery detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 19207–19217.

[29] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "S3FD: Single shot scale-invariant face detector," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 192–201.

[30] X. Tang, D. K. Du, Z. He, and J. Liu, "Pyramidbox: A context-assisted single shot face detector," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 812–828.

[31] Y. Li, J. Zhou, and S. Lyu, "Face poison: Obstructing deepfakes by disrupting face detection," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2023, pp. 1223–1228.

[32] "FakeApp," 2019. [Online]. Available: https://www.malavida.com/en/soft/fakeapp/

[33] "github DFaker," 2018. [Online]. Available: https://github.com/dfaker/df

[34] "faceswap-GAN github," https://github.com/shaoanlu/faceswap-GAN

[35] J.-Y. Zhu et al., "Toward multimodal image-to-image translation," in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 465–476.

[36] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Conf. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[37] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Conf. Neural Inf. Process. Syst.*, 2020, pp. 6840–6851.

[38] S. Sola and D. Gera, "Unmasking your expression: Expression-conditioned GAN for masked face inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 5908–5916.

[39] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, "RetinaFace: Single-shot multi-level face localisation in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5202–5211.

[40] D. Qi, W. Tan, Q. Yao, and J. Liu, "Yolo5face: Why reinventing a face detector," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 228–244.

[41] J. Li et al., "DSFD: Dual shot face detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5060–5069.

[42] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

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

[44] A. G. Howard, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv: 1704.04861*.

[45] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6848–6856.
[46] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5525–5533.
[47] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–11.
[48] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Proc. Artif. Intell. Saf. Secur.*, 2018, pp. 99–112.
[49] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2574–2582.
[50] Y. Dong et al., "Boosting adversarial attacks with momentum," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9185–9193.
[51] L. Bu, Z. Zhao, Y. Duan, and F. Song, "Taking care of the discretization problem: A comprehensive study of the discretization problem and a black-box adversarial attack in discrete integer domain," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 3200–3217, Sep./Oct. 2022.
[52] A. Agarwal, R. Singh, M. Vatsa, and N. Ratha, "Image transformation-based defense against adversarial perturbation on deep learning models," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2106–2121, Sep./Oct. 2021.
[53] Z. Wang, M. Song, S. Zheng, Z. Zhang, Y. Song, and Q. Wang, "Invisible adversarial attack against deep neural networks: An adaptive penalization approach," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 3, pp. 1474–1488, May/Jun. 2021.
[54] M. Shen, C. Li, H. Yu, Q. Li, L. Zhu, and K. Xu, "Decision-based query efficient adversarial attack via adaptive boundary learning," *IEEE Trans. Dependable Secure Comput.*, vol. 21, no. 4, pp. 1740–1753, Jul./Aug. 2024.
[55] C. Xie et al., "Improving transferability of adversarial examples with input diversity," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 563–579.
[56] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–12.
[57] G. Chierchia, G. Poggi, C. Sansone, and L. Verdoliva, "A bayesian-MRF approach for PRNU-based image forgery detection," *IEEE Trans. Inf. Forensics Secur.*, vol. 9, no. 4, pp. 554–567, Apr. 2014.
[58] H. Farid, "Photo fakery and forensics," *Adv. Comput.*, 2009.
[59] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2019, pp. 8261–8265.
[60] L. Li et al., "Face X-ray for more general face forgery detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5000–5009.
[61] R. Han, X. Wang, N. Bai, Q. Wang, Z. Liu, and J. Xue, "FCD-Net: Learning to detect multiple types of homologous deepfake face images," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 2653–2666, 2023.
[62] C. Miao, Z. Tan, Q. Chu, H. Liu, H. Hu, and N. Yu, "F 2 trans: High-frequency fine-grained transformer for face forgery detection," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 1039–1051, 2023.
[63] H. Guo, S. Hu, X. Wang, M.-C. Chang, and S. Lyu, "Robust attentive deep neural network for detecting GAN-generated faces," *IEEE Access*, vol. 10, pp. 32574–32583, 2022.
[64] S. McCloskey and M. Albright, "Detecting GAN-generated imagery using color cues," 2018, *arXiv: 1812.08247*.
[65] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *Proc. IEEE Winter Appl. Comput. Vis. Workshops*, 2019, pp. 83–92.
[66] H. Li, B. Li, S. Tan, and J. Huang, "Identification of deep network generated images using disparities in color components," *Signal Process.*, vol. 174, 2020, Art. no. 107616.
[67] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 9992–10002.
[68] K. Han et al., "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 87–110, Jan. 2023.
[69] J. Kos, I. Fischer, and D. Song, "Adversarial examples for generative models," in *Proc. IEEE Secur. Privacy Workshops*, 2018, pp. 36–42.
[70] L. Wang, W. Cho, and K.-J. Yoon, "Deceiving image-to-image translation networks for autonomous driving with adversarial perturbations," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1421–1428, Apr. 2020.
[71] J. Guan, Y. Zhao, Z. Xu, C. Meng, K. Xu, and Y. Zhao, "Adversarial robust safeguard for evading deep facial manipulation," in *Proc. AAAI Conf. Artif. Intell.*, 2024, pp. 118–126.
[72] Y. Zhu et al., "Information-containing adversarial perturbation for combating facial manipulation systems," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 2046–2059, 2023.
[73] Z. Qu, Z. Xi, W. Lu, X. Luo, Q. Wang, and B. Li, "DF-RAP: A robust adversarial perturbation for defending against deepfakes in real-world social network scenarios," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 3943–3957, 2024.
[74] H. Huang et al., "CMUA-watermark: A cross-model universal adversarial watermark for combating deepfakes," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 989–997.
[75] Z. Wang, H. Guo, Z. Zhang, W. Liu, Z. Qin, and K. Ren, "Feature importance-aware transferable adversarial attacks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 7619–7628.
[76] X. He, Y. Li, H. Qu, and J. Dong, "Improving transferable adversarial attack via feature-momentum," *Comput. Secur.*, vol. 128, 2023, Art. no. 103135.
[77] T. Chakraborty, U. Trehan, K. Mallat, and J.-L. Dugelay, "Generalizing adversarial explanations with Grad-CAM," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 186–192.
[78] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Proc. Scand. Conf. Image Anal.*, 2003, pp. 363–370.
[79] A. Dosovitskiy et al., "FlowNet: Learning optical flow with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2758–2766.
[80] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics : Learning to detect manipulated facial images," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1–11.
[81] B. Dolhansky et al., "The deepfake detection challenge (DFDC) dataset," 2020, *arXiv: 2006.07397*.
[82] B. Dolhansky, "The deepfake detection challenge (DFDC) pre view dataset," 2019, *arXiv: 1910.08854*.
[83] "YOLOv5 github," 2020. [Online]. Available: https://github.com/ultralytics/yolov5
[84] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, no. 1, pp. 90–93, Jan. 1974.
[85] "github DeepFaceLab," 2018. [Online]. Available: https://github.com/iperov/DeepFaceLab
[86] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–16.
[87] M. Naseer, S. Khan, M. Hayat, F. S. Khan, and F. Porikli, "A self-supervised approach for adversarial robustness," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 259–268.
[88] H. Liu et al., "Coherent adversarial deepfake video generation," *Signal Process.*, vol. 203, 2023, Art. no. 108790.
[89] D. E. King, "Dlib-ml: A machine learning toolkit," *J. Mach. Learn. Res.*, 2009, pp. 1755–1758.

**Delong Zhu** received the BE degree in computer science and technology from Qingdao Agricultural University, in 2023. He is currently working toward the ME degree in secrecy science and technology with the Ocean University of China. His research interests include focuses on AI security and adversarial attack.

**Yuezun Li** (Member, IEEE) received the BS degree in software engineering from Shandong University, in 2012, the MS degree in computer science, in 2015, and the PhD degree in computer science from the University at Albany–SUNY, in 2020. He was a senior research scientist with the Department of Computer Science and Engineering, University at Buffalo–SUNY. He is currently a lecturer with the Center on Artificial Intelligence, Ocean University of China. His research interests include computer vision and multimedia forensics. He has been recognized on Stanford's 2024 list of the top 2% of scientists worldwide and received the 2024 ACM Qingdao Rising Star Award.

**Baoyuan Wu** (Senior Member, IEEE) is a Tenured associate professor with the School of Data Science, Chinese University of Hong Kong, Shenzhen, Guangdong, 518172, P.R. China. His research interests include trustworthy and generative AI. He has published more than 100 top-tier conference and journal papers. He is currently serving as an associate editor of *IEEE Transactions on Information Forensics and Security and Neurocomputing*, Organizing Chair of PRCV 2022, and Area Chairs of several top-tier conferences. He received the “2023 Young Researcher Award” of The Chinese University of Hong Kong, Shenzhen.

**Jiaran Zhou** received the BEng and PhD degrees in computer science and technology from Shandong University, in 2012 and 2020, respectively. She is currently a lecturer with the Center on Artificial Intelligence, Ocean University of China. Her research interests include computer graphics, geometry processing, and artificial Intelligence security.

**Zhibo Wang** (Senior Member, IEEE) received the BE degree in automation from Zhejiang University, China, in 2007, and the PhD degree in electrical engineering and computer science from University of Tennessee, Knoxville, USA, in 2014. He is currently a professor with the School of Cyber Science and Technology, Zhejiang University, China. His current research interests include Internet of Things, AI security, and data security & privacy. He is a senior member of ACM.

**Siwei Lyu** (Fellow, IEEE) received the BS and MS degrees in computer science and information science from Peking University, Beijing, China, in 1997 and 2000 respectively, and the PhD degree in computer science from Dartmouth College, Hanover, NH, USA, in 2005. He is a SUNY Distinguished professor with the Department of Computer Science and Engineering, University at Buffalo, State University of New York at Buffalo, Buffalo, NY, USA. His research interests include digital media forensics, computer vision, and machine learning. He is a fellow of IAPR, and AAIA, and a distinguished member of ACM.