DeepFake Disrupter: The Detector of DeepFake Is My Friend (Supplementary Material)

Anonymous CVPR submission

Paper ID 7865

In the following, we provide Algorithms Pseudocode, Model Architecture and Training details and Algorithms for the training and evaluation process for the DeepFake Disrupter and additional experiments.

1. Algorithm

Algorithm 1: Training with DeepFake Disrupter

```
Data: real inputs x, target conditions c,
      hyper-parameter ε, C_1, C_2, C_3 and number of
      epochs E, batch size B
Result: Well trained Perturbation generator P(·)
Initialize Perturbation Generator P(·) with weight
  W_p;
Initialize loss weights to W_i = 1, for i = 1, 2, 3,
  where W_i = C_i;
Loading pre-trained DeepFake Generator G with
  weight W_G;
Loading pre-trained DeepFake discriminator D with
  weight W_D;
for epoch = 0 to E do
    for i = 1 to B do
        Compute perturbation η by η = P(x);;
        Compute Adversarial Inputs x̂ = x + η;
        Compute L using Eq.(7);
        Update W_{p(t+1)} ← W_{p(t)} using ∇_W L(t);
        Keep W_G and W_D unchanged after each
          iteration;
        Using GradNorm [1] to update loss item
          weights W_i
    end
end
```

Algorithm 2: Evaluation Detection Outcomes

```
Data: a batch of test data: real images or videos x,
      target conditions c, number of testing data: N
Result: Precision p, Recall r and F-1 score F
Loading pre-trained models P(·), G(·) D(·);
for data in batches do
    Compute η = P(x);
    Compute x_fake = G(x + η, c);
    Compute x_preal = x + η;
end
Pass N real inputs x, N fake inputs x_fake into
  DeepFake Detector D;
Compute
  p = True_Real/(True_Real + False_Real);

  r = True_Real/(True_Real + False_Fake);
F1 = 2 * (P * r)/(p + r);
Pass N perturbed inputs x_preal into D;
Compute
  r = True_Real/(True_Real + False_Fake)
```

2. Model Architecture and Training Details

Perturbation Generator For perturbation generator $P(\cdot)$, We choose U-Net [5]. The U-net architectures can be divided into two sections: The encoding section and the decoding section. We use 2D U-Net for image-based experiment and 3D U-Net for video-based experiment. In the encoding section, we apply contraction blocks consists of 2D or 3D convolution and max-pooling layers to encode the source inputs. In the decoding section, we apply expansion blocks consisting of 2D or 3D transpose convolution as well as normal 2D or 3D convolutions. The center part of U-Net is that each feature map of the encoding section has a shortcut connection with the corresponding feature map in the decoding section.

DeepFake Generator We use StarGAN [2], GANimation [4] and First-Order-Motion Model [8] to illustrate that our

*Equal Contribution

proposed pipeline can be used for different DeepFake manipulation systems. For StarGAN, we use a pretrained generator model in the open-source implementation used by [7]. This generator is trained on the CelebA dataset with seven domains including black hair, blond hair, brown hair, gender(male/female) and aged(young/old). For GANimation, we use its official open-source pretrained generators trained for 37 epochs on the CelebA dataset for 80 action units(AU) based on the Facial Action Unit Coding System [3]. For First-Order Motion Model, we deploy its official open-source framework. The core part of this framework includes motion estimation and image generation. Firstly, a source image and a driving video will be fed into Motion Module, then the Motion module will use a keypoint detector to extract motion representations, after which generating a dense optical flow and occlusion map mapping from the driving video to the source image. Finally, the generation process will provide quality animations by feeding into the source image and Motion Module's outputs.

DeepFake Detector As our work is to test the effectiveness of perturbation generator rather than detection power of deepfake detectors, we use commonly used backbone for various SOTA deepfake detectors, namely Xception, Resnet18 and Resnet50 as our detection architectures. All these models are trained on the FaceForensic++ datasets. In terms of image level detectors, for Xception architecture, we choose the open-source pretrained model from [6]. For Resnet18 and Resnet50, we trained 100 epochs to get a classification accuracy at 96% and 98% respectively. In terms of video level discriminators, we choose 3D Xception and 3D resnet18 and trained 100 epochs to get accuracy at 91% and 95% respectively.

Hyper Parameters There are several hyper parameters. The first one is ϵ introduced in Eq. (1) in the main submission to constrain the scale of perturbation. In order to ensure the perturbation to be human imperceptible, we follow baseline method [7] to set $\epsilon = 0.05$. For hyper parameters C_1, C_2 and C_3 that balance the different loss items, we initialize them to be 1 at the beginning, and then we follow GradNorm [1] algorithm to adaptively update the loss weight items at each iteration. In the GradNorm algorithm, there is a further hyper parameters α that corresponding to the strength of restoring force, and we set $\alpha = 0.1$ for all our experiments. For detailed explanation of the GradNorm algorithm, please refer to the original work.

3. Additional Experiments

Adaptive Attack An attacker knowing the detector challenges the defender. But preparing for the worst could perhaps ensure that the worst will not happen. We thus train an enhanced deepfake generator $G^{'}$ by incorporating the detector loss. We use $G^{'}$ and detector D to train disrupter P.

We choose StarGAN as our attacker, Xception as the detector. In Table 1, if the attacker knows the detector, [20] will have a much lower precision and F1-score (e.g. 0.52 & 0.67 in Disrupting StarGAN), but the performance of our DeepFake Disrupter can remain similar as before.

Disruption Methods	precision	recall	F1-score
Disrupting StarGAN [7]	0.64	0.99	0.78
Disrupting StarGAN [7]-Adaptive	0.52	0.99	0.67
DeepFake Disrupter(Ours)	**0.86**	0.99	**0.92**
DeepFake Disrupter(Ours-Adaptive)	**0.83**	0.99	**0.89**

Table 1. Deepfake detection performance with adaptive attacks.

Ablation on different test size In Table 2, our algorithm can achieve superior results under three larger test sizes evaluated by Xception/Resnet18 for StarGAN/GANimation generators.

Disruption Methods	Xception			Resnet18		
	100	500	1000	100	500	1000
StarGAN [2]	0.72	0.70	0.73	0.56	0.58	0.61
Disrupring StarGAN [7]	0.78	0.77	0.74	0.60	0.56	0.59
DeepFake disrupter (ours)	**0.92**	**0.93**	**0.91**	**0.71**	**0.75**	**0.71**
GANimation [4]	0.74	0.76	0.72	0.55	0.52	0.59
Disrupting GANimation [7]	0.82	0.79	0.83	0.60	0.64	0.58
DeepFake disrupter (ours)	**0.89**	**0.88**	**0.91**	**0.75**	**0.71**	**0.77**

Table 2. F1-score under different test image size 100, 500, 1000

References

[1] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pages 794–803. PMLR, 2018. 1, 2

[2] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018. 1, 2

[3] Rosenberg Ekman. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997. 2

[4] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of the European conference on computer vision (ECCV)*, pages 818–833, 2018. 1, 2

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1

[6] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *Pro-*

ceedings of the IEEE/CVF International Conference on Computer Vision, pages 1–11, 2019. 2

[7] Nataniel Ruiz, Sarah Adel Bargal, and Stan Sclaroff. Disrupting deepfakes: Adversarial attacks against conditional image translation networks and facial manipulation systems. In *European Conference on Computer Vision*, pages 236–251. Springer, 2020. 2

[8] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. *arXiv preprint arXiv:2003.00196*, 2020. 1