FrePGAN: Robust Deepfake Detection Using Frequency-level Perturbations

Yonghyun Jeong[1], **Doyeon Kim**[1], **Youngmin Ro**[1], **Jongwon Choi**[2*]

[1]Samsung SDS, Seoul, Korea
[2]Department of Advanced Imaging, Chung-Ang University, Seoul, Korea
{yhyun.jeong, dy31.kim, youngmin.ro}@samsung.com , choijw@cau.ac.kr

Abstract

Various deepfake detectors have been proposed, but challenges still exist to detect images of unknown categories or GAN models outside of the training settings. Such issues arise from the overfitting issue, which we discover from our own analysis and the previous studies to originate from the frequency-level artifacts in generated images. We find that ignoring the frequency-level artifacts can improve the detector's generalization across various GAN models, but it can reduce the model's performance for the trained GAN models. Thus, we design a framework to generalize the deepfake detector for both the known and unseen GAN models. Our framework generates the frequency-level perturbation maps to make the generated images indistinguishable from the real images. By updating the deepfake detector along with the training of the perturbation generator, our model is trained to detect the frequency-level artifacts at the initial iterations and consider the image-level irregularities at the last iterations. For experiments, we design new test scenarios varying from the training settings in GAN models, color manipulations, and object categories. Numerous experiments validate the state-of-the-art performance of our deepfake detector.

Introduction

The recent rise of Generative Adversarial Networks (GAN) (Goodfellow et al. 2014; Karras et al. 2018; Karras, Laine, and Aila 2019; Karras et al. 2020) has allowed the easy and extensive generation of highly realistic fake images, as known as deepfakes. Unfortunately, the risk of malicious abuse of deepfakes also rises with such an advancement (Nguyen et al. 2019), and the importance of detecting deepfakes has become crucial. The target range of deepfakes has broadened from swapping the face of the celebrity on the body of pornography to spreading misinformation in social media as fake news, and even alluring victims to transfer money for scams (Tolosana et al. 2020; Nguyen et al. 2019). To solve this issue, the tech giants and the academia have joined together for 'Deepfake Detection Challenge' (Dolhansky et al. 2019, 2020) to promote the current issues and encourage fellow researchers to tackle this problem.

As confirmed by several previous studies (Chen et al. 2021), the CNN-based generative models are known to have limitations in reconstructing the high-frequency components. However, as shown in Fig. 1, although the frequency-level artifacts are effective to detect the generated images for the specific GAN models, it is easy for the detectors to be overfitted to the training settings, due to the unique appearances of the frequency-level artifacts varying by the CNN structures and training categories. Thus, it can be analyzed that the frequency-level artifacts are effective to detect the generated images from the known GAN models, but the key to the generalization of deepfake detectors is to reduce the effect of frequency-level artifacts during training.

Based on intuition, we propose a novel framework com-

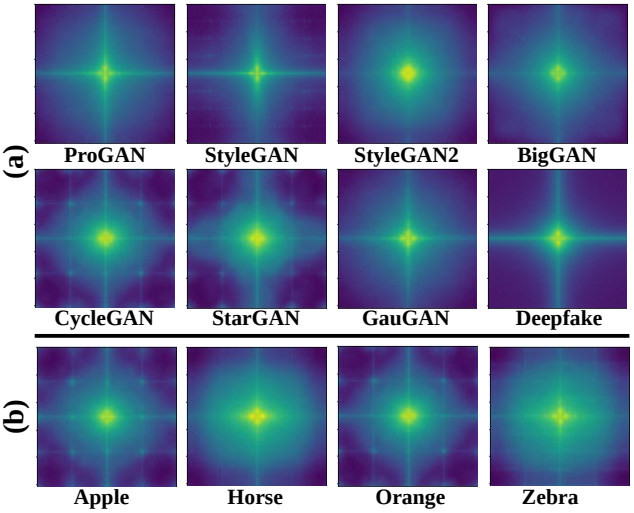

Figure 1: **Various Patterns of Frequency Artifacts.** The frequency-level artifacts can be extracted by averaging the frequency-level spectrum of the generated images. The appearance of artifacts is evident but uniquely vary by the type of GAN model or object category. Thus, we can analyze that the artifacts are easily detected by their evident appearances but can cause overfitting to the training settings due to their uniqueness. Thus, the artifacts should be ignored for generalized detection, but they are still useful for detection of specific GAN models.

*Corresponding author

posed of two modules: the Frequency Perturbation GAN (FrePGAN) and the deepfake classifier. FrePGAN contains the frequency-level perturbation generator and the perturbation discriminator, which cooperate to adversarially generate perturbation maps added onto both the real and fake images for reduced differences in the frequency-level. Then, the perturbed images are fed into the deepfake classifier to distinguish the fake images from the real ones in the pixel-level. To train the deepfake detector to utilize both the frequency-level artifacts and the general pixel-level irregularity, we update the frequency-level perturbation generator and the deepfake classifiers alternatively. To validate the performance of our model, we conduct numerous experiments using multiple deepfake datasets. Including the benchmark evaluations, we have designed three types of distinct test settings using unknown categories, models, and manipulations unused during training. Our model achieves state-of-the-art performance in both the known and unseen settings.

Our paper makes the following contributions:

- We develop FrePGAN to generate the frequency-level perturbation maps to ignore the domain-specific artifacts in fake images.
- The perturbation maps obtained from FrePGAN are added to the given input images, which can reduce the effect of domain-specific artifacts and improve the generalization ability of the deepfake detector.
- FrePGAN and the deepfake classifier are updated alternatively to train the deepfake classifier to consider both the frequency-level artifact and the general feature.
- Our model achieves superior results compared to the state-of-the-art models and robust detection performance of generated images in the known and unseen domains.

Related Work

The previous work can be categorized into the physiological feature-based, image-based, and frequency-based detection.

Physiological Feature-based Detection

With the rise of realistic human deepfakes, most studies focus on the temporal properties, such as facial features (Agarwal et al. 2019; Matern, Riess, and Stamminger 2019; Li and Lyu 2019; Montserrat et al. 2020), incoherent head poses (Yang, Li, and Lyu 2019), and lack of eye-blinking (Li, Chang, and Lyu 2018). (Rossler et al. 2019; Dolhansky et al. 2019; Li et al. 2020) provide large-scale datasets and evaluate various image forensics for face manipulations. However, since most of these methods focus on the face only, they can be ineffective in non-facial domains.

Image-based Detection

To expand the detection range, some studies take images as input data. Tralic et al.analyze the inconsistencies in blocking artifacts generated during JPEG compression (Tralic, Petrovic, and Grgic 2012). Ferrara et al.explore the demosaicing artifacts generated in manipulated images due to a color filter array (Ferrara et al. 2012) but the artifacts can disappear during resizing. Thus, some focus on the deviations in lighting conditions to detect manipulations (Carvalho, Farid, and Kee 2015; Peng et al. 2016). Also, (Bayar and Stamm 2016) suggest learning the prediction error filters for generalization but it struggles with post-processing methods used to manipulated regions. Thus, Cozzolino et al. propose an adaptable neural network to new target domains using a few training samples (Cozzolino et al. 2018). Wang et al. use RGB images to distinguish cross-model manipulations, such as blurring and JPEG (Wang et al. 2020). Also, (Guarnera, Giudice, and Battiato 2020) explore the hidden traces by analyzing the last computational layer to predict real and fake and the most probable technique used. Recently, Zhao et al. suggests a multi-attention network to attend different local parts for the artifacts and aggregate the high and low features for classification (Zhao et al. 2021).

Frequency-based Detection

Some analyze the spectral traces in the frequency domain, as (Kirchner 2008) suggests using the frequency artifacts with the variance of prediction residue. Also, (Huang et al. 2017) employ Fast Fourier Transform and singular value decomposition to identify copy-move manipulations. (Marra et al. 2019) suggest a GAN-specific detection using the artificial fingerprints in the frequency domain, and (Bappy et al. 2019) propose a manipulation localization architecture using spatial maps and frequency domain correlation. Also, (Frank et al. 2020) analyze the frequency artifacts using Discrete Cosine Transform, while (Zhang, Karaman, and Chang 2019) exploit the artifacts induced by the up-sampler of GANs. Others (Durall, Keuper, and Keuper 2020; Durall et al. 2019) exploit the spectral distortions via azimuthal integration, while (Jeong et al. 2021) adopt the bilateral high-pass filters for generalized detection. Recently, (He et al. 2021) propose to re-synthesize testing images and extract visual cues for flexible detection.

Deepfake Detection Framework

We design a generalized deepfake detector containing the Frequency Perturbation GAN (FrePGAN) and the deepfake classifier. FrePGAN generates the frequency-level perturbation maps for the deepfake detector to ignore the frequency-level artifacts. To reduce the effect of the frequency-level artifacts, both real and fake images are added with the generated perturbation maps of FrePGAN, respectively. The deepfake classifier is designed to distinguish between the real and fake images. The visual illustration for the overall architecture is shown in Fig. 2.

Training of Deepfake Detection Framework

Though FrePGAN and the deepfake classifier can be trained in a sequence, we purposefully train both networks in one iteration for comprehensive training of various properties of the perturbation maps. Also, through the alternating update, we can enhance the generalization of the deepfake classifier by expanding the variety of its input data. At the initial updates, FrePGAN fails to generate the proper perturbation map to ignore the effect of frequency-level artifacts, so the deepfake classifier is trained to distinguish the fake images

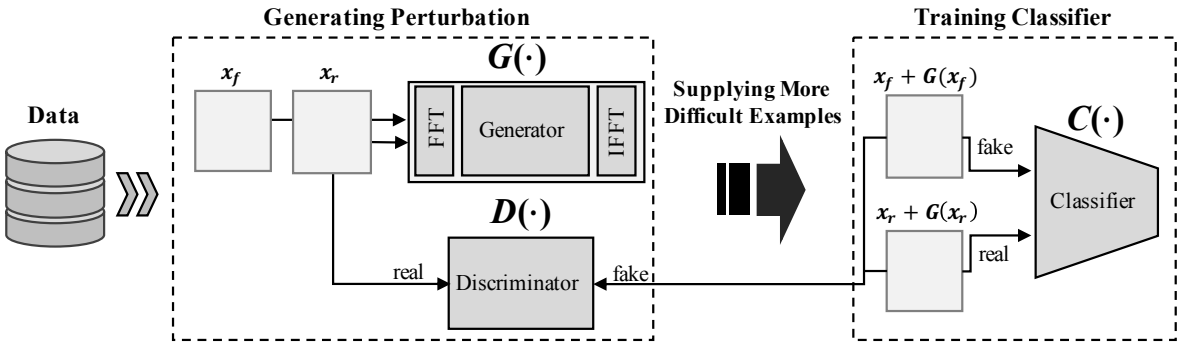

Figure 2: **Overall framework.** Consisted of FrePGAN for generating the frequency-level perturbations and the deepfake classifier for distinguishing the real and fake, the framework allows detecting fake images with the balanced effect of domain-specific frequency-level artifacts and general image-level irregularity in the generated images.

from the real images by using the artifact that is easy to be detected. On contrary, when FrePGAN is sufficiently trained to generate the perturbation maps confusing the real and fake images, the deepfake classifier needs to extract the new feature that works generally across the various types of GAN models. As a result, the alternating updates can make the deepfake classifier consider the frequency-level artifacts and the general features simultaneously.

Frequency Perturbation GAN

To train FrePGAN, we build a novel architecture composed of two major parts: the perturbation map generator trained by the perturbation generation losses, and the perturbation discriminator. The input of FrePGAN is an image $x \in \mathbb{R}^{w \times h \times c}$ where w, h, and c present its width, height, and number of channels, respectively. Each and every input image is labeled by y either as *real* ($y = 0$) or *generated* ($y = 1$), representing either actually captured in the real world or generated by GAN. We define the real and generated images as $x_r \equiv x_{y=0}$ and $x_f \equiv x_{y=1}$ respectively, and thus x would be one of x_r or x_f. The perturbation map generator and the perturbation discriminator are denoted as $G(\bullet)$ and $D(\bullet)$, respectively.

Perturbation Map Generator As in Fig. 3, the real and fake images can be easily distinguished when transformed into the frequency domain. Also, it can be observed that the frequency-level artifacts mainly locate at the high-frequency components. Thus, by adding the frequency-level perturbations, we can reduce the effect of domain-specific artifacts.

To ignore the frequency-level artifacts, the perturbation should be generated in the frequency domain as well. Thus, we utilize the frequency map transformed from the original image as the input of the perturbation map generator. The perturbation map generator contains three modules: frequency-level transformer, frequency-level generator, and the inverse frequency-level transformer.

First, the frequency-level transformer converts the input image into the frequency map by employing Fast Fourier Transform (FFT) (Cooley, Lewis, and Welch 1969), as denoted by $\widetilde{x} = \mathcal{F}(x)$ where $\widetilde{x} \in \mathbb{R}^{w \times h \times 2c}$ is the frequency map transformed from x and $\mathcal{F}(\bullet)$ represents the operation

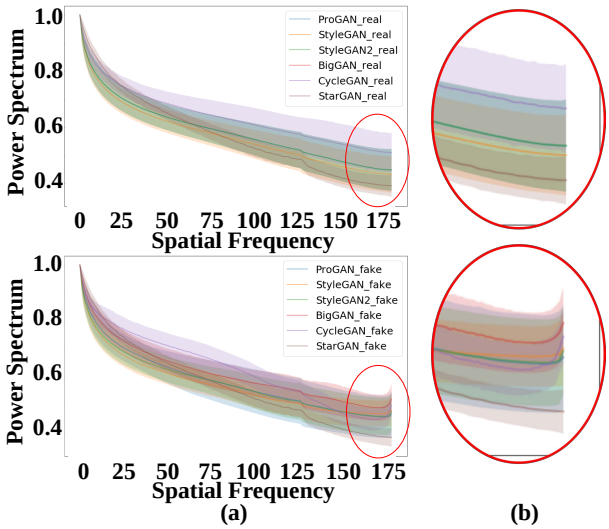

Figure 3: **Comparison in power spectra of real and fake data** We average the power spectrum from (a) the entire training data and (b) the generated data, respectively. For every GAN model of the graph, the fake images suffer from the dramatic increment of the high-frequency components.

of FFT. The number of channels of $\widetilde{x}$ becomes doubled because each image channel is separated into two channels for the real and imaginary parts of the frequency-map.

Then, the frequency-level generator receives $\widetilde{x}$ to generate a frequency map with the same size of $\widetilde{x}$. The scheme of the generator is similar to those of image-to-image translation GANs (Isola et al. 2017; Zhu et al. 2017; Kim et al. 2017; Choi et al. 2018, 2020; Liu et al. 2019), which contain the encoder and decoder. Thus, when the frequency-level generator is denoted by H, and $\widetilde{\mathbf{z}} = H(\widetilde{x})$ where $\widetilde{\mathbf{z}} \in \mathbb{R}^{w \times h \times 2c}$ is the output from the frequency-level generator.

Lastly, the generated map $\widetilde{\mathbf{z}}$ is transformed into a pixel-level perturbation map. The overall operation of the perturbation map generator is derived with a given input of x as:

$$G(x) = \mathcal{F}^{-1}(H(\mathcal{F}(x))), \tag{1}$$

where $\mathcal{F}^{-1}(\bullet)$ means the inverse FFT operation. We need to remark that the final output from the perturbation map generator is shaped by $G(x) \in \mathbb{R}^{w \times h \times c}$, which is shaped by the same size of input $x \in \mathbb{R}^{w \times h \times c}$.

Perturbation Discriminator To enhance the effect of the generated perturbation maps, we add the perturbation discriminator to adversarially train the perturbation map generator. The overall architecture follows the conventional GAN discriminator (Radford, Metz, and Chintala 2015) that down-samples the input features by the consecutive convolution layers and performs binary classification at the last convolution layer. By the last fully connected layer, the perturbation discriminator distinguishes the output of the perturbation map generator from the original image. Thus, for an input of x, the target prediction of the perturbation discriminator is a probability that can be represented by $D(x_r) = 0$ and $D(G(x)) = 1$.

Training of FrePGAN The two compositions of FrePGAN are adversarially trained to generate the perturbation maps from the input images. Thus, when real images are given to FrePGAN, the empty perturbation maps should be ideally acquired after the generator, due to the absence of frequency-level artifacts. In contrast, when the perturbation maps are added to the fake images, the distribution of the added images should be difficult to distinguish from that of real images.

At every iteration, two training steps alternate, updating the perturbation map generator and the perturbation discriminator, respectively. The perturbation map generator is updated by minimizing the perturbation generation loss ($\mathcal{L}_G$) while trying to maximize the discriminator loss ($\mathcal{L}_D$) for the update of the perturbation discriminator. Thus, the overall training of FrePGAN can be defined as:

$$\widehat{G}, \widehat{D} = \arg_{G,D} \min_G \max_D \mathcal{L}_G + \mathcal{L}_D. \tag{2}$$

$\mathcal{L}_G$ has the generative adversarial loss ($\mathcal{L}_{adv}$) and the compression loss ($\mathcal{L}_{com}$) to compress the magnitude of perturbation maps. At every mini-batch update, G is first updated to minimize the following loss:

$$\mathcal{L}_G = \lambda \mathcal{L}_{adv} + (1 - \lambda)\mathcal{L}_{com}, \tag{3}$$

where λ is a hyperparameter to tune the scales of $\mathcal{L}_{adv}$ and $\mathcal{L}_{com}$. In this work, we use $\lambda = 0.5$.

We employ $\mathcal{L}_{adv}$ for the perturbation map generator to generate the perturbation maps added to the images to be indistinguishable from the real images by the perturbation discriminator. Since x_f is also the sample generated from other GAN models, x_f is improper to be considered as the real sample for the adversarial training of FrePGAN. Thus,

$$\mathcal{L}_{adv} = \mathbb{E}_{x \sim \mathbf{X}}[\log (1 - D (G (x)))], \tag{4}$$

where $\mathbf{X}$ represents the batch sets of images.

If only the generative adversarial loss is considered during training, FrePGAN would not be able to preserve even the distribution of real images by adding a large magnitude of perturbation maps. Since the purpose of employing FrePGAN is to obtain a similar distribution of real images, we additionally include the compression loss $\mathcal{L}_{com}$ in the perturbation generation loss. Thus,

$$\mathcal{L}_{com} = \mathbb{E}_{x \sim \mathbf{X}} \left[\|G(x)\|_2^2 \right] \tag{5}$$

According to the adversarial training, the perturbation discriminator is trained to distinguish the images reconstructed by the perturbation map generator ($G(x)$) from the real images (x_r). Thus, $\mathcal{L}_D$ can be defined as:

$$\mathcal{L}_D = \mathbb{E}_{x_r \sim \mathbf{X}_r}[\log (D (x_r))] + \mathbb{E}_{x \sim \mathbf{X}}[\log (1 - D (x + G (x)))]. \tag{6}$$

As a result, by alternating the generative adversarial loss and the discriminator loss, the perturbation map generator can generate high-quality perturbation maps.

Algorithm 1: Training the deepfake detection model

```
G, D ← random initial parameters
C ← pre-trained parameters
epoch = 0
repeat
    (X, Y) ← batch sampled from dataset
    //Forward Propagation
    Estimate L_adv, L_com, L_D, L_C by Eq. 4, 5, 6, 8
    //Update parameters according to gradients
    Update G by arg_G min_G L_G
    Update D by arg_D max_D L_D
    Update C by arg_C min_C L_C
    if No remaining data then
        epoch ← epoch + 1
    end if
until epoch = 20
```

Deepfake Classifier

The deepfake classifier is a network to distinguish whether the input image is the generated fake one or not. Thus, the overall framework of the deepfake classifier is a conventional classification network using ResNet-50 (Li and Lyu 2019) to predict the binary label for deepfake detection (Frank et al. 2020; Wang et al. 2020). We denote the deepfake classifier as $C(\bullet)$.

Input of Deepfake Classifier Since the deepfake classifier detects the presence of the informative features upon the frequency-level artifacts in the input image, the images with the generated perturbation maps should be inserted into the deepfake classifier instead of the raw images. The input image of the deepfake classifier is defined as:

$$A_G(x) = x + G(x). \tag{7}$$

Training of Deepfake Classifier The training loss of the deepfake classifier is built by the cross-entropy loss as:

$$\begin{aligned} \mathcal{L}_C = \mathbb{E}_{(x,y) \sim (\mathbf{X},\mathbf{Y})} \big[& y \log (C(A_G(x))) \\ & + (1 - y) \log (1 - C(A_G(x))) \big], \end{aligned} \tag{8}$$

where $\mathbf{Y}$ is the set of real and fake labels paired with the respective samples of $\mathbf{X}$. Then, the deepfake classifier can be trained as follows: $\widehat{C} = \arg_C \min_C \mathcal{L}_C$.

The training procedure of our overall framework is presented in Algorithm 1.

Model	Original		Hue		Brightness		Saturation		Gamma		Contrast		Blur		Rotation	
	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.
Wang(2020)	99.9	**100.**	73.9	81.3	61.8	74.7	74.3	84.4	70.2	83.2	66.6	79.7	45.3	49.1	71.3	80.4
Frank (2020)	99.6	99.4	85.5	97.2	84.2	97.2	91.2	98.0	85.4	97.4	84.3	96.7	53.8	90.3	99.6	99.4
Durall (2020)	99.7	99.3	**98.6**	97.9	93.6	88.8	98.6	97.9	97.2	95.3	94.8	91.0	50.0	53.2	98.4	97.6
Jeong (2021)	99.8	**100.**	85.0	92.6	89.9	90.8	96.9	99.6	**99.7**	**100.**	90.8	91.6	67.1	99.2	99.0	100.
Ours	**100.**	**100.**	95.0	**99.7**	**99.5**	**100**.	**100.**	**100.**	85.5	98.6	**98.8**	**100.**	**98.5**	**98.5**	**100.**	**100.**

Table 1: **Comparison of cross-manipulation performance.**

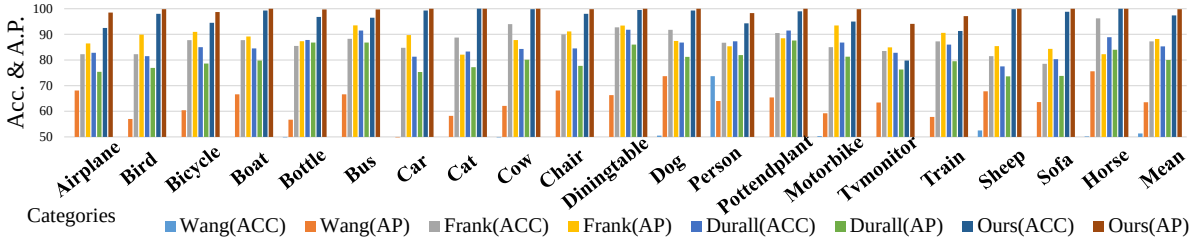

Figure 4: **Comparison of performance in unknown categories.**

Training	Resolutions									
Model	1024×1024		512×512		256×256		128×128		64×64	
	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	AP.	Acc.	A.P.
Wang (2020)	99.9	**100.**	97.6	97.3	66.1	74.4	62.6	69.4	50.4	54.9
Frank (2020)	99.6	99.4	92.2	90.2	90.5	86.0	91.3	86.9	89.7	85.1
Durall (2020)	99.7	99.3	85.1	79.0	80.0	73.7	77.2	70.9	77.9	71.7
Jeong (2021)	99.8	**100.**	97.9	99.9	97.8	99.8	89.4	96.9	59.7	62.2
Ours	**100.**	**100.**	**100.**	**100.**	**100.**	**100.**	**98.0**	**99.9**	**95.9**	**99.4**

Table 2: **Testing results with variance in resolutions.**

Deepfake Image Prediction

After the training of our overall framework, we predict whether the new test image is labeled as real or fake by utilizing the perturbation map generator of FrePGAN ($\widehat{G}$) and the deepfake classifier ($\widehat{C}$). For the new test image x', we acquire the perturbed images by estimating $A_{\widehat{G}}(x') = x' + \widehat{G}(x')$. Then, $A_{\widehat{G}}(x')$ is fed into the deepfake classifier, which results in the final prediction as: $\widehat{C}(A_{\widehat{G}}(x'))$.

Implementation Details

We employ the architecture of VGG model (Simonyan and Zisserman 2014) for the perturbation map generator and the discriminator of DCGAN (Radford, Metz, and Chintala 2015) for the perturbation discriminator. In addition, we utilize ResNet (Li and Lyu 2019) pre-trained by ImageNet (Russakovsky et al. 2015) for the deepfake classifier. We use Adam (Kingma and Ba 2014) to train the perturbation map generator and the perturbation discriminator with the learning rate of 10^{-4} and 10^{-1}, respectively. Also, the deepfake classifier is trained by Adam (Kingma and Ba 2014) with the learning rate of 10^{-4}. The batch size of the optimizer is always set to 16, and the input image size is resized to 256×256 when the image sizes vary. The number of epochs is set to 20.

Experimental Results

We conduct experiments to confirm the performance of the deepfake detector in the known domain and unseen domain.

Dataset

We conduct experiments based on the same trainset and testset of the experimental data of Wang et al. (Wang et al. 2020). The trainset contains 20 objects of Progan (Karras et al. 2018). The testset consists of FFHQ (Karras, Laine, and Aila 2019) and LSUN (Yu et al. 2015) to train ProGAN (Karras et al. 2018), StyleGAN (Karras, Laine, and Aila 2019), and StyleGAN2 (Karras et al. 2020), and employs Imagenet (Russakovsky et al. 2015) to train BigGAN (Brock, Donahue, and Simonyan 2019) and CycleGAN (Zhu et al. 2017). Also, we use CelebA (Liu et al. 2015) for training StarGAN (Choi et al. 2018), and COCO (Lin et al. 2014) for training GauGAN (Park et al. 2019). Lastly, we utilize Deepfake dataset (Rossler et al. 2019), which is a combination of various videos collected online with partially generated images reconstructed by face-swapping models.

Also, to test the model's performance in various manipulation techniques and resizing, we employ the face data of ProGAN (Karras et al. 2018) dataset in $1,024 \times 1,024$ resolution. For the experiments with unknown categories and unknown models, we utilize the horse data of ProGAN (Karras et al. 2018) dataset in 256×256 resolution.

Deepfake Detection Performance

The deepfake detection performance is tested by the four types of experiments: manipulated face images, resized face images, unseen categories, and unseen models. We utilize two evaluation metrics of the average precision score (A.P.) and accuracy (Acc.) as represented by (Wang et al. 2020; Durall, Keuper, and Keuper 2020; Frank et al. 2020). To validate the effectiveness of the proposed deepfake detector, we

Model	Training settings		Test Models																	
	Input	# class	ProGAN		StyleGAN		StyleGAN2		BigGAN		CycleGAN		StarGAN		GauGAN		Deepfake		Mean	
			Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.	Acc.	A.P.
Wang (2020)	Image	1	50.4	63.8	50.4	79.3	68.2	**94.7**	50.2	61.3	50.0	52.9	50.0	48.2	50.3	67.6	50.1	51.5	52.5	64.9
Frank (2020)	Freq	1	78.9	77.9	69.4	64.8	67.4	64.0	62.3	58.6	67.4	65.4	60.5	59.5	67.5	69.1	52.4	47.3	65.7	63.3
Durall (2020)	Freq	1	85.1	79.5	59.2	55.2	70.4	63.8	57.0	53.9	66.7	61.4	**99.8**	99.6	58.7	54.8	53.0	51.9	68.7	65.0
Jeong (2021)	Freq	1	82.5	81.4	68.0	62.8	68.8	63.6	**67.0**	**62.5**	**75.5**	**74.2**	90.1	90.1	**73.6**	**92.1**	51.6	49.9	72.1	72.1
Our	Image	1	**95.5**	**99.4**	**80.6**	**90.6**	**77.4**	93.0	63.5	60.5	59.4	59.9	99.6	**100.**	53.0	49.1	**70.4**	**81.5**	**74.9**	**79.3**
Wang (2020)	Image	2	64.6	92.7	52.8	82.8	75.7	**96.6**	51.6	70.5	58.6	81.5	51.2	74.3	53.6	86.6	50.6	51.5	57.3	79.6
Frank (2020)	Freq	2	85.7	81.3	73.1	68.5	75.0	70.9	76.9	70.8	**86.5**	80.8	85.0	77.0	67.3	65.3	50.1	55.3	75.0	71.2
Durall (2020)	Freq	2	79.0	73.9	63.6	58.8	67.3	62.1	69.5	62.9	65.4	60.8	**99.4**	99.4	67.0	63.0	50.5	50.2	70.2	66.4
Jeong (2021)	Freq	2	87.4	87.4	71.6	74.1	**77.0**	81.1	**82.6**	**80.6**	86.0	**86.6**	93.8	80.8	**75.3**	**88.2**	53.7	54.0	**78.4**	79.1
Our	Image	2	**99.0**	**99.9**	**80.8**	**92.0**	72.2	94.0	66.0	61.8	69.1	70.3	98.5	**100.**	53.1	51.0	**62.2**	**80.6**	75.1	**81.2**
Wang(2020)	Image	4	91.4	99.4	63.8	**91.4**	76.4	97.5	52.9	73.3	72.7	**88.6**	63.8	90.8	63.9	**92.2**	51.7	62.3	67.1	86.9
Frank (2020)	Freq	4	90.3	85.2	74.5	72.0	73.1	71.4	**88.7**	**86.0**	75.5	71.2	99.5	99.5	69.2	77.4	60.7	49.1	78.9	76.5
Durall (2020)	Freq	4	81.1	74.4	54.4	52.6	66.8	62.0	60.1	56.3	69.0	64.0	98.1	98.1	61.9	57.4	50.2	50.0	67.7	64.4
Jeong (2021)	Freq	4	90.7	86.2	76.9	75.1	76.2	74.7	84.9	81.7	**81.9**	78.9	94.4	94.4	**69.5**	78.1	54.4	54.6	78.6	77.9
Our	Image	4	**99.0**	**99.9**	**80.7**	89.6	**84.1**	**98.6**	69.2	71.1	71.1	74.4	**99.9**	**100.**	60.3	71.7	**70.9**	**91.9**	**79.4**	**87.2**

Table 3: **Comparison of cross-model performance.**

Ablation settings			Self		Category		Model		Manipulation		Mean	
Generator	L_{com}	L_{adv}	Acc.	A.P.	mAcc.	mA.P.	mAcc.	mA.P.	mAcc.	mA.P.	Acc.	A.P.
Freq		✓	99.0	100.	90.9	98.2	69.4	76.1	90.9	98.4	87.6	93.2
Freq	✓		100.	100.	91.0	98.5	67.3	74.4	91.3	99.1	87.4	93.0
Img	✓	✓	99.8	100.	92.9	99.2	68.7	77.4	92.4	99.6	88.5	94.1
Freq	✓	✓	100.	100.	95.5	99.4	74.9	79.3	96.8	99.5	91.8	94.6

Table 4: **Ablation Test with Various Settings.**

select the image-based (Wang et al. 2020) and frequency-based state-of-the-art models (Frank et al. 2020; Durall, Keuper, and Keuper 2020; Jeong et al. 2021) for comparison.

Deepfake Detection of Manipulated Face Images We conduct various image manipulation experiments using the face data of ProGAN (Karras et al. 2018) in $1,024 \times 1,024$ resolution. To test the performance with unknown manipulations, we add 7 various changes in images, such as adjusting the hue, brightness, saturation, gamma, contrast, blurriness, and image rotation. As shown in Table 1, ours is the most robust model achieving superior performance in image manipulation experiments.

Deepfake Detection of Resized Face Images To test the model with the previous detectors' chronic issue of significant performance decline with resizing, we conduct experiments with the face data of ProGAN (Karras et al. 2018) dataset by gradually reducing the image sizes with five different resolutions from $1,024 \times 1,024$ to 64×64. Based on the experimental results of the resizing performance of the models as shown in Table 2, we can confirm that our model outperforms all other models when tested with the five cases of resized resolutions. Furthermore, even when the image resolution is reduced, our model maintains 100% performance from $1,024 \times 1,024$ to 256×256. When reduced to 128×128 and 64×64, our model's performance slightly declines but maintains at least 97.8%, proving the best performance compared to the existing model.

Deepfake Detection of Unknown Categories As shown in Figure 4, we conduct various experiments using the three classifiers to analyze the performance of the models with 20 unknown categories. We compare our model's performance to the previous state-of-the-art models (Wang et al. 2020; Durall, Keuper, and Keuper 2020; Frank et al. 2020). The experimental results verify that ours is the most robust model in all categories, even when the number of training classes increases and the type of inputs varies. The variety in the testing environment shows that each component in our model plays an important role to detect not only the test category but also all categories.

Deepfake Detection of Unknown GAN Models To expand the testing scope, we compare the performance of our model with 8 different generative models, including ProGAN (Karras et al. 2018), StyleGAN (Karras, Laine, and Aila 2019), StyleGAN2 (Karras et al. 2020), BigGAN (Brock, Donahue, and Simonyan 2019), CycleGAN (Zhu et al. 2017), StarGAN (Choi et al. 2018), GauGAN (Park et al. 2019), and Deepfake (Rossler et al. 2019). As shown in Table 3, we make changes to the training settings and conduct experiments to detect the GAN models. First, we train the models with one type of category and test with all GAN models. Then, to add variety, we increase the number of training categories to two and four. Interestingly, even when trained with only one type of category, our model achieves excellent performance similar to the case when trained with four classes. The results show that ours achieves the highest performance in both Acc. and A.P in ProGAN (Karras et al. 2018), StyleGAN (Karras, Laine, and Aila 2019), GauGAN (Park et al. 2019), and Deepfake (Rossler et al. 2019). Also, in StyleGAN2 (Karras et al. 2020), BigGAN (Brock, Donahue, and Simonyan 2019), CycleGAN (Zhu et al. 2017), and StarGAN (Choi et al. 2018), our model achieves the best performance in either Acc. or A.P. Our performance rises with the number of

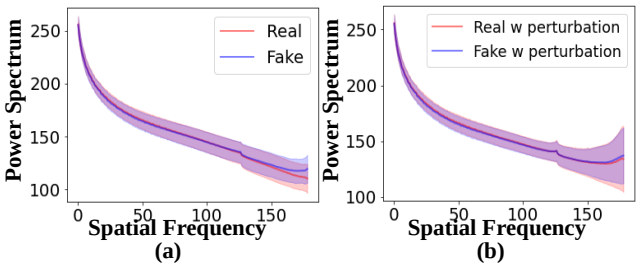

Figure 5: **Addition of Perturbation Maps.** (a) shows the averaged power spectrum of real and fake images, respectively. (b) compares the averaged power spectrum of real with that of fake. While the real and fake images can be distinguished by the high-frequency components in the power spectrum, the difference is nearly removed after the addition of the generated perturbation maps.

training categories, even when tested with the partially generated images in Deepfake dataset.

Ablation Study

To validate the effectiveness of the components in the proposed framework, we also test several variants. We modify four settings of the proposed framework, which include the composition of the perturbation generation loss and the data input types of the generator of FrePGAN. The training dataset for the ablation tests is the horse images generated by ProGAN (Karras et al. 2018). The testing environments for cross-category and cross-model experiments are the same as the previous section, and the manipulation experiment is conducted by manipulating the horse test images with the same methods of cross-manipulation experiments.

The results of the ablation tests are provided in Table 4. The best performance of $(91.8,\ 94.6)$ can be obtained by utilizing the entire framework with the frequency-level generator, the adversarial learning mechanism, and the compression loss. Interestingly, the simultaneous usage of the compression loss and the adversarial loss makes synergy to improve the generalization of deepfake classifier. This result shows that the quality of perturbation is important for the overall performance. Also, when we replace our frequency-level generator with the pixel-level generator without the frequency transformers, the performance drops by far due to the limited quality of the generated perturbation maps.

Visualization of Perturbation Maps

To visualize the effect of the perturbation maps, we conduct two experiments. First, as shown in Fig. 5, we obtain the power spectra of real and fake images. Then, due to the frequency-level artifacts, the high-frequency components of real and fake become distinct to distinguish between them. However, after adding the perturbation maps, the difference is reduced by far in the power spectra, which validates that the proposed framework successfully generates high-quality perturbation maps to make the real and fake similar.

Second, as shown in Fig. 1, from one real image, we obtain both the 1D and 2D power spectrum. Interestingly, even

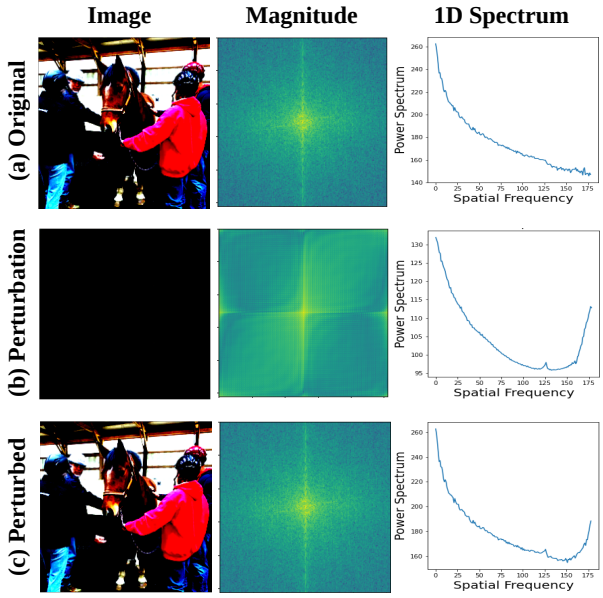

Figure 6: **Effect of Perturbation Map.** With the compression loss, the magnitude of perturbation maps is negligent in the pixel-level domain as shown in (b). However, the perturbation map increments the high-frequency components as shown in (c).

after the addition of perturbation maps, the pixel-level image and its 2D power spectrum are almost preserved. However, when we estimate the 1D power spectrum, we can find that the high-frequency components are magnified, which results in the 1D power spectrum similar to that of fake images containing the frequency-level artifacts.

Conclusion

It has become highly important to develop a robust, generalized deepfake detector, which is not limited to the training settings. Numerous experiments validate that our framework achieves a generalized detection robust in various testing scenarios including the unknown categories, GAN models, manipulations, and resizing. Trained with the perturbation generation loss and compression loss, our newly proposed FrePGAN generates perturbations to reduce the effects of domain-specific artifacts in generated images. Also, our framework shows the effectiveness of the alternate updates of the deepfake classifier and the perturbation generator, which is validated to be helpful for the improved generalization of deepfake detectors.

Acknowledgement

This work was supported by Samsung SDS and Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government(MSIT) (2021-0-01341, Artificial Intelligence Graduate School Program(Chung-Ang University); 2021-0-01778, Development of Human Image Synthesis and Discrimination Technology Below the Perceptual Threshold; 2021-0-02067, Next Generation AI for Multi-purpose Video Search).

References

Agarwal, S.; Farid, H.; Gu, Y.; He, M.; Nagano, K.; and Li, H. 2019. Protecting World Leaders Against Deep Fakes. In *CVPR Workshops*.

Bappy, J. H.; Simons, C.; Nataraj, L.; Manjunath, B.; and Roy-Chowdhury, A. K. 2019. Hybrid lstm and encoder–decoder architecture for detection of image forgeries. *IEEE TIP*.

Bayar, B.; and Stamm, M. C. 2016. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *ACM Workshop on Information Hiding and Multimedia Security*.

Brock, A.; Donahue, J.; and Simonyan, K. 2019. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *ICLR*.

Carvalho, T.; Farid, H.; and Kee, E. R. 2015. Exposing photo manipulation from user-guided 3d lighting analysis. In *Media Watermarking, Security, and Forensics 2015*. International Society for Optics and Photonics.

Chen, Y.; Li, G.; Jin, C.; Liu, S.; and Li, T. 2021. SSD-GAN: Measuring the Realness in the Spatial and Spectral Domains. In *AAAI*.

Choi, Y.; Choi, M.; Kim, M.; Ha, J.-W.; Kim, S.; and Choo, J. 2018. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. In *CVPR*.

Choi, Y.; Uh, Y.; Yoo, J.; and Ha, J.-W. 2020. StarGAN v2: Diverse Image Synthesis for Multiple Domains. In *CVPR*.

Cooley, J. W.; Lewis, P. A.; and Welch, P. D. 1969. The fast fourier transform and its applications. *IEEE Transactions on Education*.

Cozzolino, D.; Thies, J.; Rössler, A.; Riess, C.; Nießner, M.; and Verdoliva, L. 2018. ForensicTransfer: Weakly-supervised Domain Adaptation for Forgery Detection. *arXiv*.

Dolhansky, B.; Bitton, J.; Pflaum, B.; Lu, J.; Howes, R.; Wang, M.; and Ferrer, C. C. 2020. The DeepFake Detection Challenge Dataset. *arXiv preprint arXiv:2006.07397*.

Dolhansky, B.; Howes, R.; Pflaum, B.; Baram, N.; and Ferrer, C. C. 2019. The Deepfake Detection Challenge (DFDC) Preview Dataset. *arXiv preprint arXiv:1910.08854*.

Durall, R.; Keuper, M.; and Keuper, J. 2020. Watch your Up-Convolution: CNN Based Generative Deep Neural Networks are Failing to Reproduce Spectral Distributions. In *CVPR*.

Durall, R.; Keuper, M.; Pfreundt, F.-J.; and Keuper, J. 2019. Unmasking DeepFakes with simple Features. *arXiv preprint arXiv:1911.00686*.

Ferrara, P.; Bianchi, T.; De Rosa, A.; and Piva, A. 2012. Image forgery localization via fine-grained analysis of CFA artifacts. *IEEE Transactions on Information Forensics and Security*, 7(5): 1566–1577.

Frank, J.; Eisenhofer, T.; Schönherr, L.; Fischer, A.; Kolossa, D.; and Holz, T. 2020. Leveraging Frequency Analysis for Deep Fake Image Recognition.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NeurIPS*.

Guarnera, L.; Giudice, O.; and Battiato, S. 2020. Deepfake detection by analyzing convolutional traces. In *CVPR Workshops*.

He, Y.; Yu, N.; Keuper, M.; and Fritz, M. 2021. Beyond the Spectrum: Detecting Deepfakes via Re-Synthesis. *arXiv preprint arXiv:2105.14376*.

Huang, D.-Y.; Huang, C.-N.; Hu, W.-C.; and Chou, C.-H. 2017. Robustness of copy-move forgery detection under high JPEG compression artifacts. *Multimedia Tools and Applications*.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*.

Jeong, Y.; Kim, D.; Min, S.; Joe, S.; Gwon, Y.; and Choi, J. 2021. BiHPF: Bilateral High-Pass Filters for Robust Deepfake Detection. *arXiv preprint arXiv:2109.00911*.

Karras, T.; Aila, T.; Laine, S.; and Lehtinen, J. 2018. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *ICLR*.

Karras, T.; Laine, S.; and Aila, T. 2019. A style-based generator architecture for generative adversarial networks. In *CVPR*.

Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2020. Analyzing and Improving the Image Quality of StyleGAN. *CVPR*.

Kim, T.; Cha, M.; Kim, H.; Lee, J. K.; and Kim, J. 2017. Learning to discover cross-domain relations with generative adversarial networks. In *ICML*.

Kingma, D.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. *ICLR*.

Kirchner, M. 2008. Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue. In *ACM workshop on Multimedia and security*.

Li, Y.; Chang, M.; and Lyu, S. 2018. In Ictu Oculi: Exposing AI Created Fake Videos by Detecting Eye Blinking. In *WIFS*.

Li, Y.; and Lyu, S. 2019. Exposing DeepFake Videos By Detecting Face Warping Artifacts. In *CVPR Workshops*.

Li, Y.; Sun, P.; Qi, H.; and Lyu, S. 2020. Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics. In *CVPR*.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *ECCV*.

Liu, M.-Y.; Huang, X.; Mallya, A.; Karras, T.; Aila, T.; Lehtinen, J.; and Kautz, J. 2019. Few-shot unsupervised image-to-image translation. In *ICCV*.

Liu, Z.; Luo, P.; Wang, X.; and Tang, X. 2015. Deep Learning Face Attributes in the Wild. In *ICCV*.

Marra, F.; Gragnaniello, D.; Verdoliva, L.; and Poggi, G. 2019. Do gans leave artificial fingerprints? In *CMIPR*. IEEE.

Matern, F.; Riess, C.; and Stamminger, M. 2019. Exploiting visual artifacts to expose deepfakes and face manipulations. In *IWACV Workshops*. IEEE.

Montserrat, D. M.; Hao, H.; Yarlagadda, S.; Baireddy, S.; Shao, R.; Horváth, J.; Bartusiak, E.; Yang, J.; Güera, D.; Zhu, F.; et al. 2020. Deepfakes Detection with Automatic Face Weighting. *arXiv preprint arXiv:2004.12027*.

Nguyen, T. T.; Nguyen, C. M.; Nguyen, D. T.; Nguyen, D. T.; and Nahavandi, S. 2019. Deep Learning for Deepfakes Creation and Detection. *arXiv preprint arXiv:1909.11573*.

Park, T.; Liu, M.-Y.; Wang, T.-C.; and Zhu, J.-Y. 2019. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*.

Peng, B.; Wang, W.; Dong, J.; and Tan, T. 2016. Optimized 3D lighting environment estimation for image forgery detection. *IEEE Transactions on Information Forensics and Security*, 479–494.

Radford, A.; Metz, L.; and Chintala, S. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Rossler, A.; Cozzolino, D.; Verdoliva, L.; Riess, C.; Thies, J.; and Nießner, M. 2019. Faceforensics++: Learning to detect manipulated facial images. In *ICCV*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. ImageNet Large Scale Visual Recognition Challenge. *IJCV*.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Tolosana, R.; Vera-Rodriguez, R.; Fierrez, J.; Morales, A.; and Ortega-Garcia, J. 2020. Deepfakes and beyond: A survey of face manipulation and fake detection. *arXiv preprint arXiv:2001.00179*.

Tralic, D.; Petrovic, J.; and Grgic, S. 2012. JPEG image tampering detection using blocking artifacts. In *International Conference on Systems, Signals and Image Processing*, 5–8. IEEE.

Wang, S.-Y.; Wang, O.; Zhang, R.; Owens, A.; and Efros, A. A. 2020. CNN-generated images are surprisingly easy to spot...for now. In *CVPR*.

Yang, X.; Li, Y.; and Lyu, S. 2019. Exposing Deep Fakes Using Inconsistent Head Poses. In *IEEE International Conference on Acoustics, Speech and Signal Processing*.

Yu, F.; Zhang, Y.; Song, S.; Seff, A.; and Xiao, J. 2015. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365*.

Zhang, X.; Karaman, S.; and Chang, S. 2019. Detecting and Simulating Artifacts in GAN Fake Images. In *WFIS*.

Zhao, H.; Zhou, W.; Chen, D.; Wei, T.; Zhang, W.; and Yu, N. 2021. Multi-attentional deepfake detection. In *CVPR*.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In *ICCV*.