CNN-generated images are surprisingly easy to spot... for now

Sheng-Yu Wang[1] Oliver Wang[2] Richard Zhang[2] Andrew Owens[1,3] Alexei A. Efros[1]
UC Berkeley[1] Adobe Research[2] University of Michigan[3]

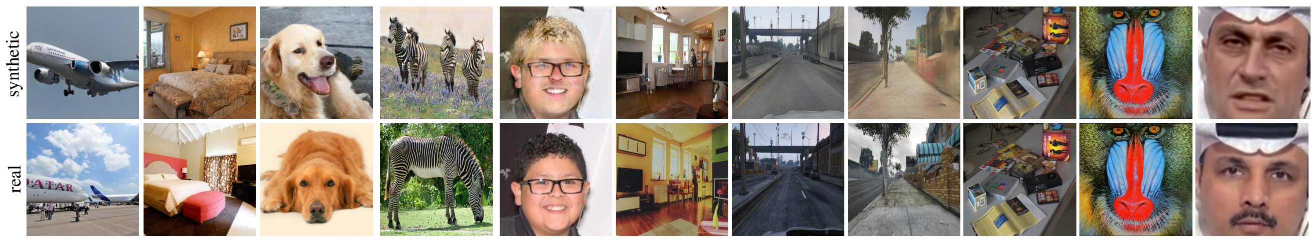

Figure 1: **Are CNN-generated images hard to distinguish from real images?** We show that a classifier trained to detect images generated by only one CNN (ProGAN, far left) can detect those generated by many other models (remaining columns). Our code and models are available at `https://peterwang512.github.io/CNNDetection/`.

Abstract

In this work we ask whether it is possible to create a "universal" detector for telling apart real images from these generated by a CNN, regardless of architecture or dataset used. To test this, we collect a dataset consisting of fake images generated by 11 different CNN-based image generator models, chosen to span the space of commonly used architectures today (ProGAN, StyleGAN, BigGAN, CycleGAN, StarGAN, GauGAN, DeepFakes, cascaded refinement networks, implicit maximum likelihood estimation, second-order attention super-resolution, seeing-in-the-dark). We demonstrate that, with careful pre- and post-processing and data augmentation, a standard image classifier trained on only one *specific CNN generator (ProGAN) is able to generalize surprisingly well to unseen architectures, datasets, and training methods (including the just released StyleGAN2 [21]). Our findings suggest the intriguing possibility that today's CNN-generated images share some common systematic flaws, preventing them from achieving realistic image synthesis.*

1. Introduction

Recent rapid advances in deep image synthesis techniques, such as Generative Adversarial Networks (GANs), have generated a huge amount of public interest and concern, as people worry that we are entering a world where it will be impossible to tell which images are real and which are fake [14]. This issue has started to play a significant role in global politics; in one case a video of the president of Gabon that was claimed by opposition to be fake was one factor leading to a failed coup d'etat*. Much of this concern has been directed at specific manipulation techniques, such as "deepfake"-style face replacement [2], and photorealistic synthetic humans [20]. However, these methods represent only two *instances* of a broader set of techniques: image synthesis via convolutional neural networks (CNNs). Our goal in this work is to find a general image forensics approach for detecting CNN-generated imagery.

Detecting whether an image was generated by a specific synthesis technique is relatively straightforward — just train a classifier on a dataset consisting of real images and images synthesized by the technique in question. However, such an approach will likely be tied to the dataset used in image generation (*e.g*. faces), and, due to dataset bias [35], might not generalize when tested on new data (*e.g*. cars). Even worse, the technique-specific detector is likely to soon become ineffective as generation methods evolve and the technique it was trained on becomes obsolete.

It is natural, therefore, to ask whether today's CNN-generated images contain common artifacts, *e.g*., some kind of detectable CNN *fingerprints*, that would allow a classifier to generalize to an entire family of generation methods, rather than a single one. Unfortunately, prior work has reported generalization to be a significant problem for

*`https://www.motherjones.com/politics/2019/03/deepfake-gabon-ali-bongo/`

image forensics approaches. For example, several recent works [44, 12, 37] observe that that classifiers trained on images produced by one GAN architecture perform poorly when tested on others, and in many cases they also fail to generalize when only the dataset (and not the architecture or task) is changed [44]. This makes sense, as image generation methods are highly varied: they use different datasets, network architectures, loss functions, and image pre-processing.

In this paper, we show that, contrary to this current understanding, classifiers trained to detect CNN-generated images *can* exhibit a surprising amount of generalization ability across datasets, architectures, and tasks. We follow conventions and train our classifiers in a straightforward manner, by generating a large number of fake images using a *single* CNN model (we use ProGAN, a high-performing unconditional GAN model [19]), and train a binary classifier to detect fakes, using the model's real training images as negative examples.

To evaluate our model, we create a new dataset of CNN-generated images, the *ForenSynths* dataset, consisting of synthesized images from 11 models, that range from from unconditional image generation methods, such as StyleGAN [20], to super-resolution methods [13], and deepfakes [33]. Each model is trained on a different image dataset appropriate for its specific task. We have also continued evaluating our detector on models that were released after our paper was originally written, finding that it works out-of-the-box on the very recent unconditional GAN, StyleGAN2 [21].

Underneath the apparent simplicity of this approach, we have found that there are a number of subtle challenges which we study through a set of experiments and a new dataset of trained image generation models. We find that data augmentation, in the form of common image post-processing operations, is critical for generalization, even when the target images are not post-processed themselves. We also find that diversity of training images matters; large datasets sampled from CNN synthesis methods lead to classifiers that outperform those trained on smaller datasets, to a point. Finally, it is critical to examine the effect of post-processing on the model's generalization ability which often occur downstream of image creation (*e.g*., during storage and distribution). We show that when the correct steps are taken, classifiers are indeed robust to common operations such as JPEG compression, blurring, and resizing.

In summary, our main contributions are: 1) we show that forensics models trained on CNN-generated images exhibit a surprising amount of generalization to other CNN synthesis methods; 2) we propose a new dataset and evaluation metric for detecting CNN-generated images; 3) we experimentally analyze the factors that account for cross-model generalization.

2. Related work

Detecting CNN-based Manipulations Several recent works have addressed the problem of detecting images generated by CNNs. Rössler *et al*. [33] evaluated methods for detecting face manipulation techniques, including CNN-based face and mouth replacement methods. While they showed that simple classifiers could detect fakes generated *by the same* model, they did not study generalization between models or datasets. Marra *et al*. [24] likewise showed that simple classifiers can detect images created by an image translation network [17], but did not consider cross-model transfer.

Recently, Cozzolino *et al*. [12] found that forensics classifiers transferred poorly between models, often obtaining near-chance performance. They propose a new representation learning method, based on autoencoders, to improve transfer performance in zero- and low-shot training regimes for a variety of generation methods. While their ultimate goal is similar to ours, they take an orthogonal approach. They focus on new learning methods for improving transfer learning, and apply them to a diverse assortment of models (including both CNN and non-CNN). In contrast, we empirically study the performance of simple "baseline" classifiers under different training and testing conditions for CNN-based image generation. Zhang *et al*. [44] finds that classifiers generalize poorly between GAN models. They propose a method called AutoGAN for generating images that contain the upsampling artifacts common in GAN architectures, and test it on two types of GANs. Other work has proposed to detect GAN images using hand-crafted co-occurrence features [26], or by anomaly detection models built on pretrained face detectors [37]. Researchers have also proposed methods for identifying which, of several, known GANs generated a given image [25, 41].

Image forensics Researchers have proposed a variety of methods for detecting more traditional manipulation techniques, such as those made by image editing tools. Early work focused on hand-crafted cues [14] such as compression artifacts [3], resampling [31], or physical scene constraints [27]. More recently, researchers have applied learning-based methods to these problems [45, 16, 11, 32, 38]. This line of work has found, like us, that simple, supervised classifiers are often effective at detecting manipulations [45, 38].

Artifacts from CNN-based Generators Researchers have shown, recently, that common CNN designs contain artifacts that reduce their representational power. Much of this work has focused on the way networks perform upampling and downsampling. A well-known example of such an artifact is the checkerboard artifact produced by deconvolutional layers [28]. Azulay and Weiss [4] showed convolutional networks ignore the classical sampling theorem and that strided convolutions therefore reduce translation invariance, and Zhang [43] improved translation invariance by reducing aliasing in these layers. Very recently, Bau *et al*. [5] suggested that GANs have limited generation capacity, and analyzed the image structures that a pretrained GAN is unable to produce.

3. A dataset of CNN-based generation models

To study the transferability of classifiers trained to detect CNN-generated images, we collected a dataset of images created from a variety of CNN models.

3.1. Generation models

Our dataset contains 11 synthesis models. We chose methods that span a variety of CNN architectures, datasets, and losses. All of these models have an upsampling-convolutional structure (*i.e*. they generate images by a series convolution and upsampling operations) since this is by far the most common design for generative CNNs. Examples of their synthesized images can be found in Figure 1. The statistics of each dataset are listed in Table 1. Details of the data collection process are provided in the supplemental material.

GANs We include three state-of-the-art unconditional GANs: ProGAN [19], StyleGAN [20], BigGAN [7], trained on either the LSUN [40] or ImageNet [34] datasets. The network structures and training procedures for these models contain significant differences. ProGAN and StyleGAN train a different network for each category; StyleGAN injects large, per-pixel noise into the model to introduce high frequency detail. BigGAN has a monolithic, class-conditional structure, is trained on very large batch sizes, and uses self-attention layers [42, 39].

We also include three conditional GANs: the state-of-the-art image-to-image translation method GauGAN [29], and the popular unpaired image-to-image translation methods CycleGAN [48] and StarGAN [10].

Perceptual loss We consider models that directly optimize a perceptual loss [18], with no adversarial training. This includes Cascaded Refinement Networks (CRN) [9], which synthesizes images in a coarse-to-fine manner, and the recent Implicit Maximum Likelihood Estimation (IMLE) conditional image translation model [23].

Low-level vision We include the Seeing In The Dark (SITD) model [8], which approximates long-exposure photography under low light conditions from short-exposure raw camera input using a high-resolution fully convolutional network. We also use a state-of-the-art super-resolution model, the Second Order Attention Network (SAN) [13].

Family	Method	Image Source	# Images
Unconditional GAN	ProGAN [19]	LSUN	8.0k
	StyleGAN [20]	LSUN	12.0k
	BigGAN [7]	ImageNet	4.0k
Conditional GAN	CycleGAN [48]	Style/object transfer	2.6k
	StarGAN [10]	CelebA	4.0k
	GauGAN [29]	COCO	10.0k
Perceptual loss	CRN [9]	GTA	12.8k
	IMLE [23]	GTA	12.8k
Low-level vision	SITD [8]	Raw camera	360
	SAN [13]	Standard SR benchmark	440
Deepfake	FaceForensics++ [33]	Videos of faces	5.4k

Table 1: **Generation models.** We evaluate forensic classifiers on a variety of CNN-based image generation methods.

Deep fakes We also evaluate our model on the face replacement images provided in the FaceForensics++ benchmark of Rössler *et al*. [33], which used the publicly available *faceswap* tool [1]. While "deepfake" is often used as a general term, we take inspiration from the convention in [33] and refer to this specific model as *DeepFake*. This model uses an autoencoder to generate faces, and images undergo extensive post-processing steps, including Poisson image blending [30] with real content. We note that our main goal is to detect images directly output by CNN decoders, while DeepFake serves as an out-of-distribution test case. Following [33], we use cropped faces.

3.2. Generating fake images

We collect images from the models, taking care to match the pre-processing operations performed by each (*e.g*. resizing and cropping). For each dataset, we collect fake images by generating them from the model without applying additional post-processing (or we download the officially released generated images if they are available). We collect an equal number of real images from each method's training set. To make the distribution of the real and fake images as close as possible, real images are pre-processed according to the pipeline prescribed by each method.

Since 256×256 resolution is the most commonly shared output size among most off-the-shelf image synthesis models (*e.g*., CycleGAN, StarGAN, ProGAN LSUN, GauGAN COCO, IMLE, *etc*.), we used this resolution for our dataset. For models that produce images at lower resolutions, (*e.g*., DeepFake), we rescale the images using bilinear interpolation to 256 on the shorter side with the same aspect ratio, and for models that produce images at higher resolution (*e.g*., ProGAN, StyleGAN, SAN, SITD), we keep the images at the same resolution. Despite these cases being slightly different from our training scheme, we observe that our model is still able to detect fake images under these categories. For all datasets, we make our real/fake prediction from 224×224 crops (random-crop at training time and center-crop at testing time).

Family	Name	Training settings					Individual test generators											Total
		Train	Input	No. Class	Augments		Pro-GAN	Style-GAN	Big-GAN	Cycle-GAN	Star-GAN	Gau-GAN	CRN	IMLE	SITD	SAN	Deep-Fake	**mAP**
					Blur	JPEG												
Zhang et al. [44]	Cyc-Im	CycleGAN	RGB	–			84.3	65.7	55.1	100.	99.2	79.9	74.5	90.6	67.8	82.9	53.2	77.6
	Cyc-Spec	CycleGAN	Spec	–			51.4	52.7	79.6	100.	**100.**	70.8	64.7	71.3	92.2	78.5	44.5	73.2
	Auto-Im	AutoGAN	RGB	–			73.8	60.1	46.1	99.9	**100.**	49.0	82.5	71.0	80.1	86.7	80.8	75.5
	Auto-Spec	AutoGAN	Spec	–			75.6	68.6	84.9	**100.**	**100.**	61.0	80.8	75.3	89.9	66.1	39.0	76.5
Ours	2-class	ProGAN	RGB	2	✓	✓	98.8	78.3	66.4	88.7	87.3	87.4	94.0	97.3	85.2	52.9	58.1	81.3
	4-class	ProGAN	RGB	4	✓	✓	99.8	87.0	74.0	93.2	92.3	94.1	95.8	97.5	87.8	58.5	59.6	85.4
	8-class	ProGAN	RGB	8	✓	✓	99.9	94.2	78.9	94.3	91.9	95.4	98.9	99.4	91.2	58.6	63.8	87.9
	16-class	ProGAN	RGB	16	✓	✓	100.	98.2	87.7	96.4	95.5	**98.1**	**99.0**	**99.7**	95.3	63.1	71.9	91.4
	No aug	ProGAN	RGB	20			100.	96.3	72.2	84.0	**100.**	67.0	93.5	90.3	96.2	**93.6**	**98.2**	90.1
	Blur only	ProGAN	RGB	20	✓		100.	99.0	82.5	90.1	**100.**	74.7	66.6	66.7	**99.6**	53.7	95.1	84.4
	JPEG only	ProGAN	RGB	20		✓	100.	99.0	87.8	93.2	91.8	97.5	**99.0**	99.5	88.7	78.1	88.1	**93.0**
	Blur+JPEG (0.5)	ProGAN	RGB	20	✓	✓	100.	98.5	**88.2**	96.8	95.4	**98.1**	98.9	99.5	92.7	63.9	66.3	90.8
	Blur+JPEG (0.1)	ProGAN	RGB	20	†	†	100.	**99.6**	84.5	93.5	98.2	89.5	98.2	98.4	97.2	70.5	89.0	92.6

Table 2: **Cross-generator generalization results.** We show the average precision (AP) of various classifiers from baseline Zhang et al. [44] and ours, tested across 11 generators. Symbols ✓ and † mean the augmentation is applied with 50% or 10% probability, respectively, at training. Chance is 50% and best possible performance is 100%. When test generators are used in training, we show those results in gray (as they are not testing generalization). Values in black show cross-generator generalization. Amongst those, the highest value is highlighted in **black**. We show ablations with respect to fewer classes in ProGAN and by removing data augmentation. We report the mean AP by averaging the AP scores over all datasets. Subsets are plotted in Figures 2, 3, 4 for comparison.

4. Detecting CNN-synthesized images

Are there common features or artifacts shared across diverse CNN generators? To understand this, we study whether it is possible to train a forensics classifier on images from *one* model that generalize to those of *many* models.

4.1. Training classifiers

While all of these models are useful for *evaluation*, due to limitations in dataset size, not all are well-suited to training a classifier. We take advantage of the fact that the unconditional GAN models in our dataset can synthesize arbitrary numbers of images, and choose one specific model, ProGAN [19] to train the detector on. The decision to use a single model for training most closely resembles real world detection problems, where the diversity or number of models to generalize on are *unknown* at training time. By selecting only a single model to train on, we are computing an *upper bound* on how challenging the task is — jointly training on multiple models would make the generalization problem easier. We chose ProGAN since it generates high quality images and has a simple convolutional network structure.

We then create a large-scale dataset that consists *solely* of ProGAN-generated images and real images. We use 20 models each trained on a different LSUN [40] object category, and generate 36K train images and 200 validation images, each with equal numbers of real and fake images for each model. In total there are 720K images for training and 4K images for validation. To evaluate the choice of the training dataset, we also include a model that is trained *solely* on the BigGAN dataset. We also consider a model that generates training images using the deep image prior [36], rather than a GAN. The details for these models are provided in the supplementary material.

The main idea of our experiments is to train a "real-or-fake" classifier on this ProGAN dataset, and evaluate how well the model generalizes to other CNN-synthesized images. For the choice of classifier, we use ResNet-50 [15] pre-trained with ImageNet, and train it in a binary classification setting. Details of the training procedure are provided in the supplemental material.

Data augmentation During training, we simulate image post-processing operations in a variety of ways. All of our models are trained with images that are randomly left-right flipped and cropped to 224 pixels. We evaluate several additional augmentation variants: **(1) No aug**: no augmentation applied, **(2) Gaussian blur**: before cropping, with 50% probability, images are blurred with $\sigma \sim$ Uniform$[0, 3]$, **(3) JPEG**: with 50% probability images are JPEG-ed by two popular libraries, OpenCV [6] and the Python Imaging Library (PIL), with quality $\sim$ Uniform$\{30, 31, \ldots, 100\}$, **(4a) Blur+JPEG (0.5)**: the image is possibly blurred and JPEG-ed, each with 50% probability, **(4b) Blur+JPEG (0.1)**: similar to (4a), but with 10% probability.

Evaluation Following other recent forensics works [46, 16, 38], we evaluate our model's performance on each dataset using average precision (AP), since it is a threshold-less, ranking-based score that is not sensitive to the base rate of the real and fake images in the dataset. We compute this score for each dataset separately, since we expect it to be dependent on the semantic content of the photos as a whole. To help interpret the threshold-less results, we also conduct experiments on thresholding the model's outputs and computing accuracy, under the assumption that real and fake images are equally likely to appear; the details are in the supplemental material. During testing, each image is

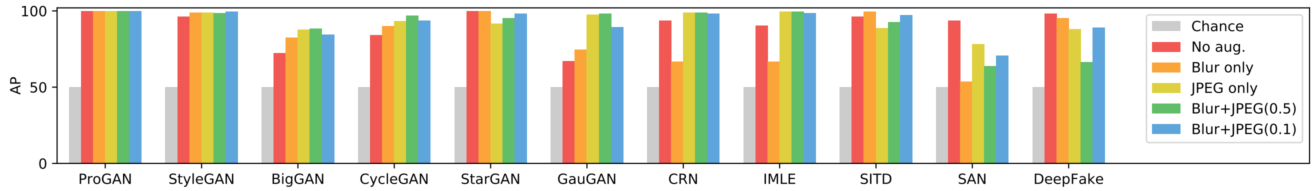

Figure 2: **Effect of augmentation methods.** All detectors are trained on ProGAN, and tested on other generators (AP shown). In general, training with augmentation helps performance. Notable exceptions are super-resolution and DeepFake.

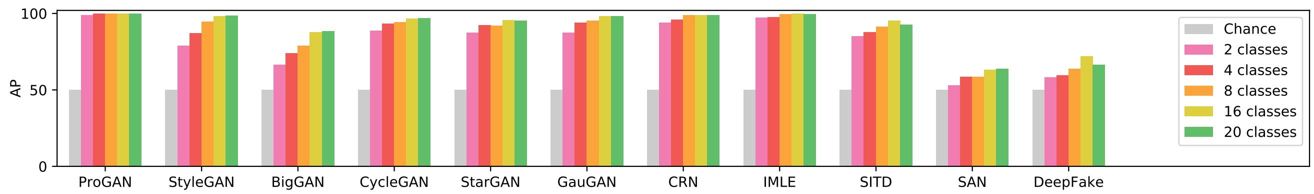

Figure 3: **Effect of dataset diversity.** All detectors are trained on ProGAN, and tested on other generators (AP shown). Training with more classes improves performance. All runs use blur and JPEG augmentation with 50% probability.

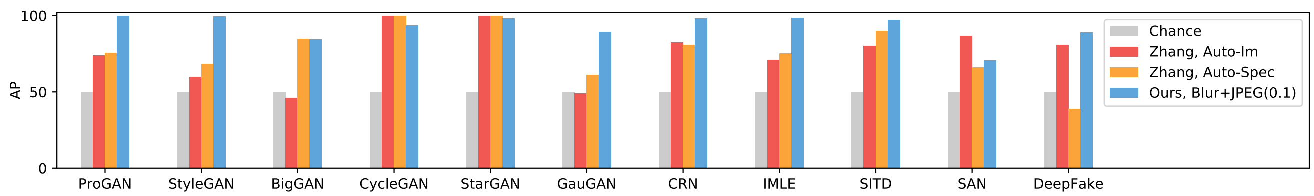

Figure 4: **Model comparison.** Compared to Zhang *et al*. [44], we observe that for the most part, our models generalize better to other architectures. Notable exceptions to this are CycleGAN (which is identical to the training architecture from [44]), StarGAN (where both methods obtain close to 100. AP), and SAN (where applying data augmentation hurts performance).

center-cropped to 224×224 pixels without resizing in order to match the post-processing pipeline used by models during training. No data augmentation is included during testing; instead, we conduct experiments on model robustness under post-processing in Section 4.2.

4.2. Effect of data augmentation

In Table 2, we investigate the generalization ability of training with different augmentation methods. We find that using aggressive data augmentation (in the form of simulated post-processing) provides surprising generalization capabilities, even when such perturbations are not used at test time. Additionally, we observe that these models are significantly more robust to post-processing (Figure 5).

Augmentation (usually) improves generalization To begin, we first evaluate ProGAN-based classifier *without* augmentation, shown in the "no aug" row. As in previous work [33], we find that testing on held-out ProGAN images works well (100.0 AP). We then test how well it generalizes to other unconditional GANs. We find that it generalizes extremely well to StyleGAN, which has a similar network structure, but not as well to BigGAN. When adding augmentations, the performance on BigGAN significantly improves, $72.2 \rightarrow 88.2$. On conditional models (CycleGAN, GauGAN, CRN, and IMLE), performance is similarly improved, $84.0 \rightarrow 96.8$, $67.0 \rightarrow 98.1$, $93.5 \rightarrow 98.9$, $90.3 \rightarrow 99.5$, respectively.

Interestingly, there are two models, SAN and DeepFake, where directly training on ProGAN without augmentation performs strongly (93.6 and 98.2, respectively), but augmentation hurts performance. As SAN is a super-resolution model, only high-frequency components can differentiate between real and fake images. Removing such cues at training time (*e.g*. by blurring) would therefore be likely to reduce performance. As explained in Section 3.1, DeepFake serves as an out-of-distribution test case as images are not generated by CNN architectures alone, but surprisingly our model is able to generalize to this test case. However, it remains challenging to identify clear reasons for the performance deterioration when applying augmentations. Applying augmentation, but at reduced rate **(Blur+JPEG (0.1))**, offers a good balance: DeepFake detection is comparable to the no-augmentation case (89.0), while most other datasets are significantly improved over no augmentation.

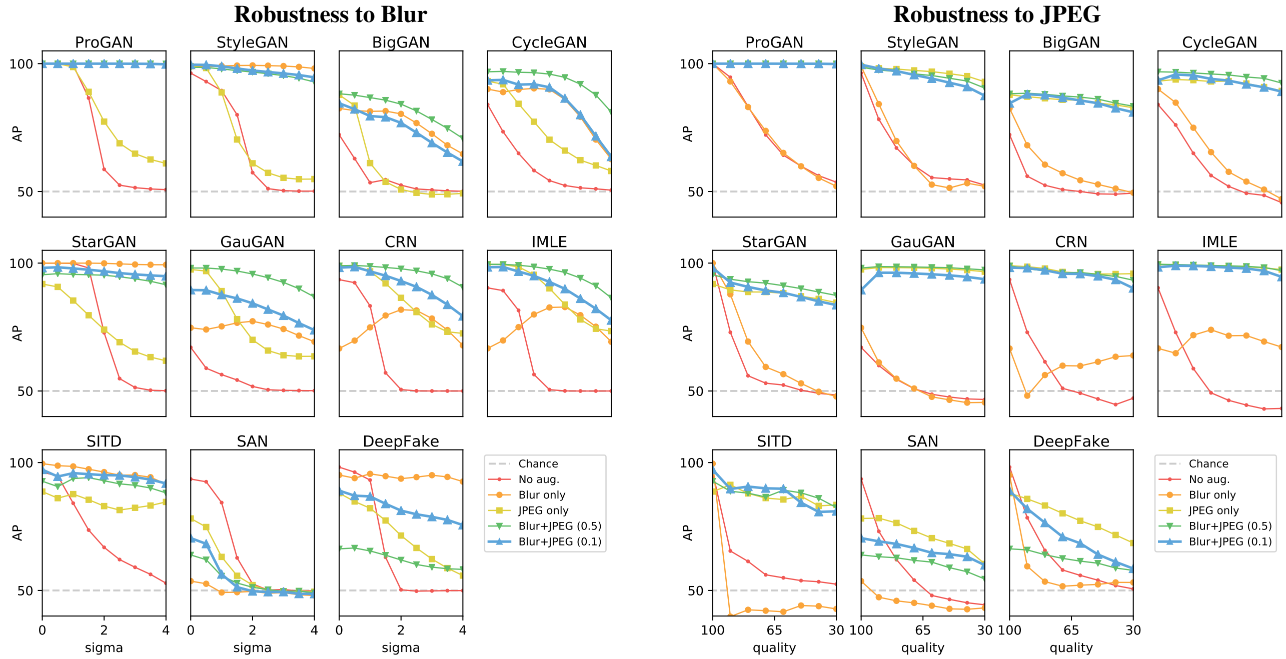

Figure 5: **Robustness.** We show the effect of AP given test-time perturbation to **(left)** Gaussian blurring and **(right)** JPEG. We show classifiers trained on ProGAN, with different augmentations applied during training. Note that in all cases and both perturbations, when training without augmentation (red), performance degrades across all datasets when perturbations are added. In most cases, training with *both* augmentations, performs best or near best. Notable exceptions are for super-resolution (where no augmentation is best), and DeepFake, where training *only* with the perturbation used during testing, rather than both, performs best.

Augmentation improves robustness In many real-world scenarios, images that we would like to evaluate have undergone unknown post-processing operations, such as compression and resizing. We investigated whether CNN-generated images can still be detected, even after these post-processing steps. To test this, we blurred (simulating re-sampling) and JPEG-compressed the real and fake images following the protocol in [38], and evaluated our ability to detect them (Figure 5). On ProGAN (*i.e.* the case where the test distribution matches the training), performance is 100% even when applying augmentation operations, indicating that artifacts may not only be high-frequency, but exist across frequency bands. In terms of cross-generator generalization, the augmented model is most robust to post-processing operations that are included in data augmentation, agreeing with observations from [33, 38, 41, 44]. However, we note that our model *also* gains robustness from augmentation even when testing on *out-of-distribution* CNN models.

4.3. Effect of data diversity

Next, we asked how the diversity of the real and fake images in the training set affects a classifier's generalization ability.

Image diversity improves performance To study this, we varied the number of classes in the dataset used to train our real-or-fake classifier (Figure 3). Specifically, we trained multiple classifiers, each one on a subset of the full training dataset by excluded both real and fake images derived from a specific set of LSUN classes. For all models we use the same augmentation scheme as the **Blur+JPEG (0.5)** model. We found that increasing the training set diversity improves performance, but only up to a point. When the number of classes used increases from 2 to 16, AP consistently improves, but we see diminishing returns. Minimal improvement is observed when increasing from 16 to 20 classes. This indicates that there may be a training dataset that is "diverse enough" for practical generalization.

4.4. Comparison to other models

Next, we asked how our generalization performance compares to other proposed forensic methods. We compare our approach to Zhang *et al.* [44], which is a suite of classifiers trained to detect artifacts generated by a common CNN architecture, which is shared by many image synthesis tasks such as CycleGAN and StarGAN. They introduced AutoGAN, an autoencoder based on CycleGAN's generator that simulates artifacts resembling that of CycleGAN images.

We considered four variations of pretrained models from

Figure 6: **Does our model's confidence correlate with visual quality?** We have found that for two models, BigGAN and StarGAN, the images on the left (considered more real) tends to look better than the images on the right (considered more fake). However, this does not seem to hold for the other models. More examples on each dataset are provided in the supplemental material.

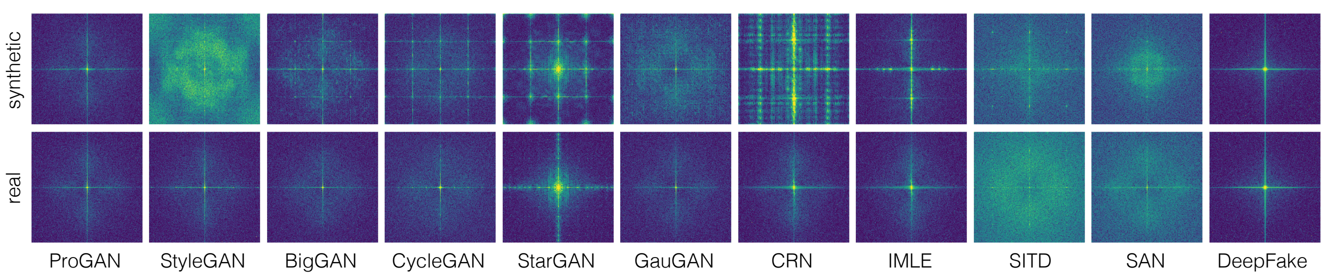

Figure 7: **Frequency analysis on each dataset.** We show the average spectra of each high-pass filtered image, for both the real and fake images, similar to Zhang *et al*. [44]. We observe periodic patterns (dots or lines) in most of the synthetic images, while BigGAN and ProGAN contains relatively few such artifacts.

Zhang *et al*. [44], each trained from one of the two image sources (CycleGAN and AutoGAN), and one of the two image representations (images and spectrum) respectively. All four variants included JPEG and resize data augmentation during training to improve the robustness of each model. We found that our models generalized significantly better to other architectures, except on CycleGAN (which is the model architecture used by [44]), StarGAN (where both methods obtain near 100.0 AP). The comparison results are shown in Table 2 and Figure 4. We also include comparisons to other baseline models in the supplemental material.

4.5. New CNN models

We hope that as new deep synthesis models arrive, our system will detect them out-of-the-box. One such an evaluation scenario has naturally arisen, with the recent release of StyleGAN2 [21], a state-of-the-art unconditional GAN appearing in these proceedings. The StyleGAN2 model makes several changes to StyleGAN, including redesigned normalization, multi-resolution, and regularization methods. In Table 3, we test our detector on publicly available StyleGAN2 generators. We used our **Blur+JPEG (0.1)** model and tested on the *LSUN car*, *cat*, *church*, and *horse* variants. Despite these changes, our technique performs at 99.1% AP. These results reinforce the notion that training on today's generators can generalize well to future generators, given that they use similar underlying building blocks.

4.6. Qualitative Analysis

To understand how the network is able to generalize to unseen CNN models, we study what possible cues the classifier might be using by visualizing its ranking on the "fakeness" over the synthetic dataset. In addition, we analyze the difference between the frequency responses of both real and synthetic images across datasets.

"Fakeness" ranking by the model We study whether

	ProGAN	StyleGAN	StyleGAN2
AP	100.	99.6	99.1

Table 3: **Out-of-the box evaluation on recently released StyleGAN2 [21] model**. We used our **Blur+JPEG (0.1)** model and tested on StyleGAN2. We observed that our model generalizes to detecting StyleGAN2 images. Numbers for ProGAN and StyleGAN are included for comparison.

our model is learning subtle low-level features generated by CNN architectures, or high-level features such as visual quality. Taking the similar approach as previous image realism works [22, 47], we rank synthesized images from each dataset by the model's prediction, and visualize images in the 0^{th}, 25^{th}, 50^{th}, 75^{th}, 100^{th} percentile of the "fakeness" score from our model's output.

In most datasets, we observe little noticeable correlation between the model predictions and the visual quality of the synthesized images. However, there is a weak correlation in the BigGAN and StarGAN datasets; qualitative examples are shown in Figure 6. As the "fakeness" scores are higher, the images tend to contain more visible artifacts which deteriorate the visual quality. This implies that our model might learn to capture perceptual realism under this task. However, since the correlation is not observed in other datasets, it is more likely that the model learns features more towards low-level CNN artifacts. Examples across all datasets are provided in the supplemental material.

Artifacts of CNN image synthesis Inspired by Zhang *et al*. [44], we visualize the average frequency spectra from each dataset to study the artifacts generated by CNNs, as shown in Figure 7. Following prior work, we perform a simple form of high-pass filtering (subtracting the image from its median blurred version) before calculating the Fourier transform, as it provides a more informative visualization [25]. For each dataset, we average over 2000 randomly chosen images (or the entire set, if it is smaller).

We note that there are many interesting patterns visible in these visualizations. While the real image spectra generally look alike (with minor variations due to differences in the datasets), there are distinct patterns visible in images generated by different CNN models. Furthermore, the repeated period patterns in these spectra may be consistent with aliasing artifacts, a cue considered by [44]. Interestingly, the most effective unconditional GANs (BigGAN, ProGAN) contain relatively few such artifacts. Also, DeepFake images does not contain obvious artifacts. We note that DeepFake images have gone through various pre- and post-processing, where the synthesized face region is resized, blended, and compressed with MPEG. These operations perturbs the low-level image statistic, which may cause the frequency patterns to not emerge with this visualization method.

5. Discussion

Despite the alarm that has been raised by the rapidly improving quality of image synthesis methods, our results suggest that today's CNN-generated images retain *detectable fingerprints* that distinguish them from real photos. This allows forensic classifiers to generalize from one model to another without extensive adaptation.

However, this does not mean that the current situation will persist. Due to the difficulties in achieving Nash equilibria, none of the current GAN-based architectures are optimized to convergence, *i.e.* the generator never wins against the discriminator. Were this to change, we would suddenly find ourselves in a situation when synthetic images are completely indistinguishable from real ones.

Even with the current techniques, there remain practical reasons for concern. First, even the best forensics detector will have some trade-off between true detection and false-positive rates. Since a malicious user is typically looking to create a single fake image (rather than a distribution of fakes), they could simply hand-pick the fake image which happens to pass the detection threshold. Second, malicious use of fake imagery is likely be deployed on a social media platform (Facebook, Twitter, YouTube, *etc*.), so the data will undergo a number of often aggressive transformations (compression, resizing, re-sampling, *etc*.). While we demonstrated robustness to some degree of JPEG compression, blurring, and resizing, much more work is needed to evaluate how well the current detectors can cope with these transformations *in-the-wild*. Finally, most documented instances of effective deployment of visual fakes to date have been using classic "shallow" methods, such as Photoshop. We have experimented with running our detector on the face-aware liquify dataset from [38], and found that our method performs at chance on this data. This suggests that shallow methods exhibit fundamentally different behavior than deep methods, and should not be neglected.

We note that detecting fake images is just one small piece of the puzzle of how to combat the threat of visual disinformation. Effective solutions will need to incorporate a wide range of strategies, from technical to social to legal.

Acknowledgements We'd like to thank Jaakko Lehtinen, Taesung Park, Jacob (Minyoung) Huh, Hany Farid, and Matthias Kirchner for helpful discussions. We are grateful to Xu Zhang, Lakshmanan Nataraj, and Davide Cozzolino for significant help with comparisons to [44, 26, 12], respectively. This work was funded, in part, by DARPA MediFor, Adobe gift and grant from the UC Berkeley Center for Long-Term Cybersecurity. The views, opinions and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

[1] Deepfakes faceswap github repository. https://github.com/deepfakes/faceswap. 3

[2] Faceswap. https://faceswap.dev/. 1

[3] Shruti Agarwal and Hany Farid. Photo forensics from jpeg dimples. In *2017 IEEE Workshop on Information Forensics and Security (WIFS)*, 2017. 2

[4] Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *JMLR*, 2019. 2

[5] David Bau, Jun-Yan Zhu, Jonas Wulff, William Peebles, Hendrik Strobelt, Bolei Zhou, and Antonio Torralba. Seeing what a gan cannot generate. In *ICCV*, 2019. 3

[6] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc.", 2008. 4

[7] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 1, 3

[8] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *CVPR*, 2018. 1, 3

[9] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *ICCV*, 2017. 1, 3

[10] 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 *CVPR*, 2018. 1, 3

[11] Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva. Splicebuster: A new blind image splicing detector. In *2015 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2015. 2

[12] Davide Cozzolino, Justus Thies, Andreas Rössler, Christian Riess, Matthias Nießner, and Luisa Verdoliva. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*, 2018. 2, 8

[13] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. Second-order attention network for single image super-resolution. In *CVPR*, 2019. 1, 2, 3

[14] Hany Farid. *Photo forensics*. MIT Press, 2016. 1, 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[16] Minyoung Huh, Andrew Liu, Andrew Owens, and Alexei A Efros. Fighting fake news: Image splice detection via learned self-consistency. In *ECCV*, 2018. 2, 4

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2

[18] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 3

[19] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018. 1, 2, 3, 4

[20] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 2, 3

[21] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 1, 2, 7, 8

[22] Jean-François Lalonde and Alexei A. Efros. Using color compatibility for assessing image realism. In *ICCV*, 2007. 8

[23] Ke Li, Tianhao Zhang, and Jitendra Malik. Diverse image synthesis from semantic layouts via conditional imle. In *ICCV*, 2019. 1, 3

[24] Francesco Marra, Diego Gragnaniello, Davide Cozzolino, and Luisa Verdoliva. Detection of gan-generated fake images over social networks. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2018. 2

[25] Francesco Marra, Diego Gragnaniello, Luisa Verdoliva, and Giovanni Poggi. Do gans leave artificial fingerprints? In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2019. 2, 8

[26] Lakshmanan Nataraj, Tajuddin Manhar Mohammed, BS Manjunath, Shivkumar Chandrasekaran, Arjuna Flenner, Jawadul H Bappy, and Amit K Roy-Chowdhury. Detecting gan generated fake images using co-occurrence matrices. *Electronic Imaging*, 2019. 2, 8

[27] James F O'Brien and Hany Farid. Exposing photo manipulation with inconsistent reflections. *ACM Trans. Graph.*, 2012. 2

[28] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. 2

[29] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 1, 3

[30] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Transactions on graphics (TOG)*, 2003. 3

[31] Alin C Popescu and Hany Farid. Exposing digital forgeries by detecting traces of resampling. *IEEE Transactions on signal processing*, 2005. 2

[32] Yuan Rao and Jiangqun Ni. A deep learning approach to detection of splicing and copy-move forgeries in images. In *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2016. 2

[33] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. FaceForensics++: Learning to detect manipulated facial images. In *ICCV*, 2019. 1, 2, 3, 5, 6

[34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 3

[35] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR*, 2011. 1

[36] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *CVPR*, 2018. 4

[37] Run Wang, Lei Ma, Felix Juefei-Xu, Xiaofei Xie, Jian Wang, and Yang Liu. Fakespotter: A simple baseline for spotting

ai-synthesized fake faces. *arXiv preprint arXiv:1909.06122*, 2019. 2

[38] Sheng-Yu Wang, Oliver Wang, Andrew Owens, Richard Zhang, and Alexei A Efros. Detecting photoshopped faces by scripting photoshop. In *ICCV*, 2019. 2, 4, 6, 8

[39] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3

[40] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 3, 4

[41] Ning Yu, Larry Davis, and Mario Fritz. Attributing fake images to gans: Analyzing fingerprints in generated images. In *ICCV*, 2019. 2, 6

[42] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, 2019. 3

[43] Richard Zhang. Making convolutional networks shift-invariant again. In *ICML*, 2019. 3

[44] Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detecting and simulating artifacts in gan fake images. In *WIFS*, 2019. 2, 4, 5, 6, 7, 8

[45] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016. 2

[46] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Learning rich features for image manipulation detection. In *CVPR*, 2018. 4

[47] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Learning a discriminative model for the perception of realism in composite images. In *ICCV*, 2015. 8

[48] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 1, 3