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]

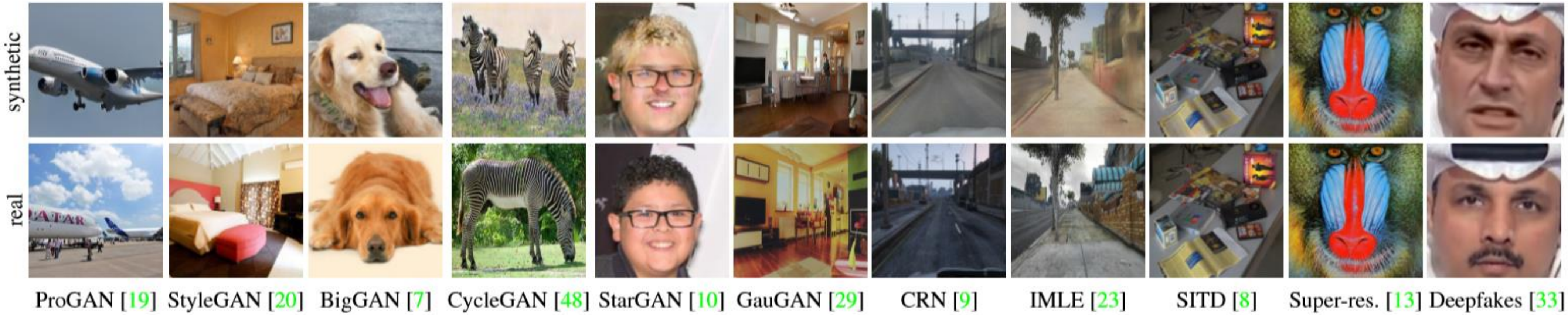


Figure 1: **Are CNN-generated images hard to distinguish from real images?** We show that a classifier trained to detect images generated by only one CNN (ProGAN, far left) can detect those generated by many other models (remaining columns). Our code and models are available at https://peterwang512.github.io/CNNDetection/.

Presented by: Hengtao Guo
08/05/2020

Motivation

1. Recent rapid advances in deep image synthesis techniques, such as Generative Adversarial Networks (GANs), have generated a huge amount of public interest and concern.
   a) People worry that we are entering a world where it will be impossible to tell which images are real
   b) “Deepfake”-style face replacement
   c) Photorealistic synthetic humans
2. This work aims to find a general image forensics approach for detecting CNN-generated imagery.

Summary

1. **Question**: Is it possible to create a “universal” detector for telling apart real images from CNN-generated fake images, regardless of architecture or dataset used?
2. **Proposal**: Collect a dataset consisting of fake images generated by 11 different CNN-based image generator models, test with only one classifier trained on ProGAN.
3. **Conclusion**: A standard image classifier trained on only one specific CNN generator (ProGAN) can generalize surprisingly well to unseen architectures, datasets, and training methods.

Methodology

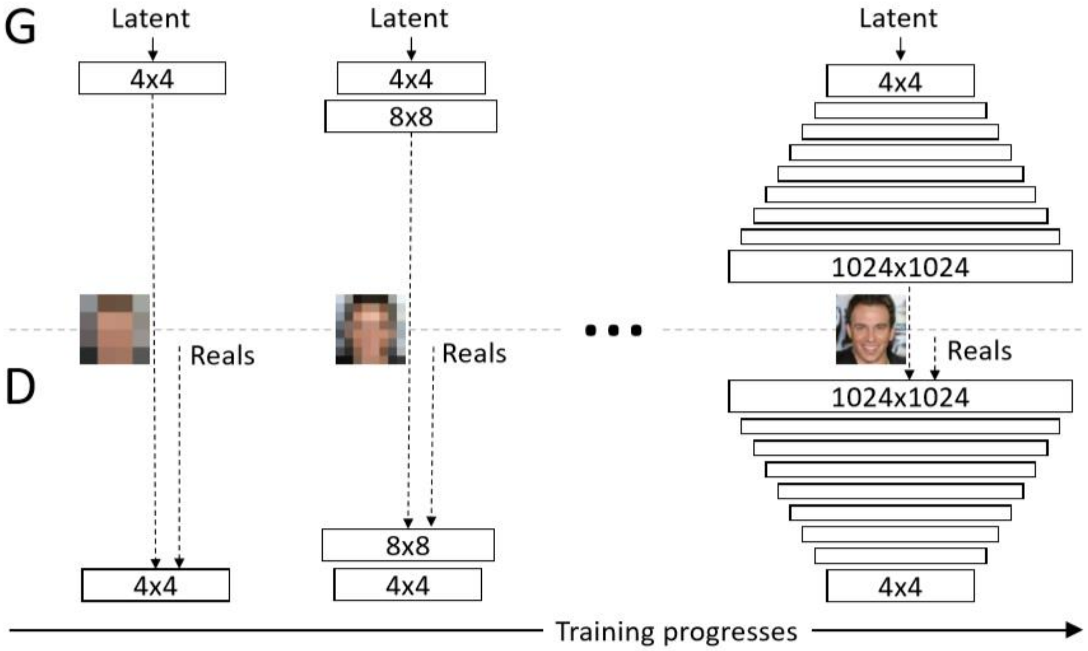
1. The authors create a new dataset of CNN-generated images, the ***ForenSynths*** dataset, consisting of synthesized images from 11 models, that range from unconditional image generation methods, such as StyleGAN, to super-resolution methods, and deepfakes.
2. Each model is trained on a different image dataset appropriate for its specific task.
3. Contributions:
   a) Show that forensics models trained on CNN-generated images exhibit a surprising amount of generalization to other CNN synthesis methods;
   b) Propose a new dataset and evaluation metric for detecting CNN-generated images;
   c) Experimentally analyze the factors that account for cross-model generalization.

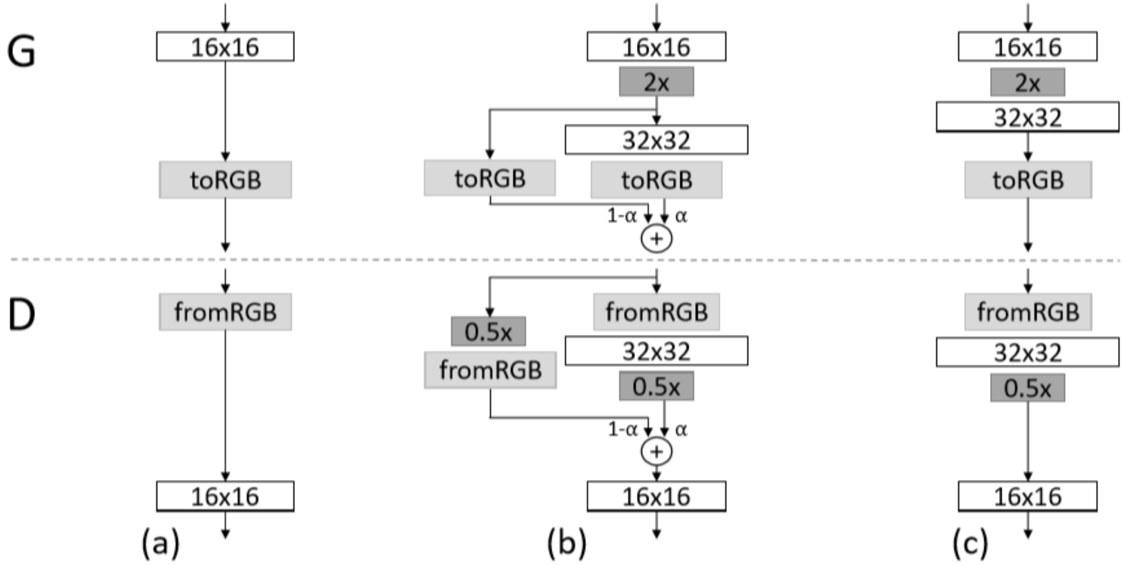
Fake Image Generation Models

1. GANs:
    a) **ProGAN, StyleGAN, BigGAN**
    b) **GauGAN, CycleGAN, StarGAN**
2. Perceptual loss:
    a) Cascaded Refinement Networks (**CRN**)
    b) Implicit Maximum Likelihood Estimation (**IMLE**)
3. Low-level vision
    a) Seeing In The Dark (**SITD**)
    b) Second Order Attention Network (**SAN**)
4. **Deep fakes**

ProGAN

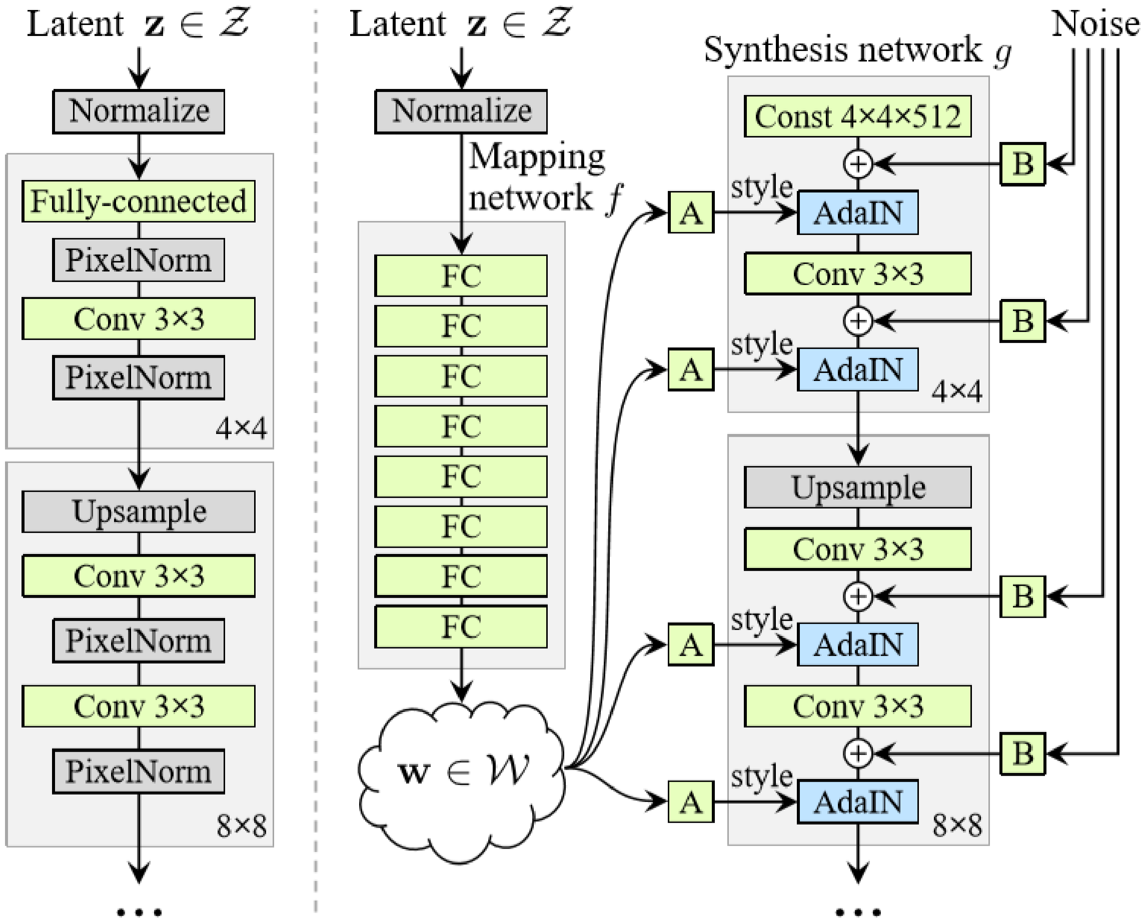
1. Progressive growing of GANs for improved quality, stability, and variation, 2017 arxiv
2. Highlights:
   a) Aims to generate high-resolution human faces (1024)
   b) Model's capacity gradually increases during training
   c) To stabilize the training when increase layers, they propose to use a weighted residual connection (gradually give highres weights from 0 to 1)





StyleGAN

1. A style-based generator architecture for generative adversarial networks. 2019 CVPR
2. Highlights:
   a) Aims to generate high-resolution human faces
   b) Use fully connected layers to give latent vector a non-linear transformation (z to w)
   c) Adaptive instance normalization (AdaIN).

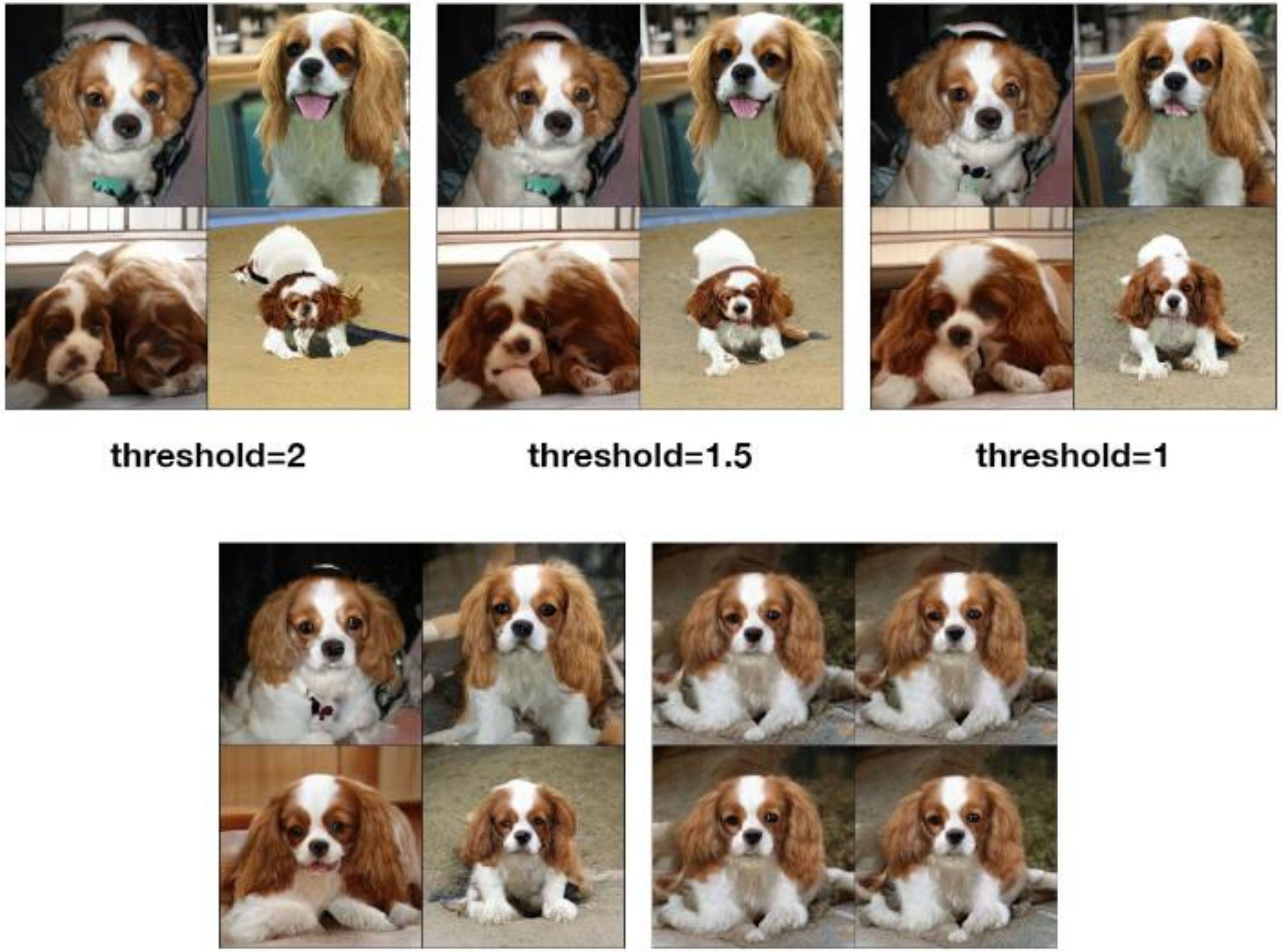


(a) Traditional (b) Style-based generator

$$\mathrm{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i},$$

BigGAN

1. Large scale GAN training for high fidelity natural image synthesis, 2019 ICLR
2. Highlights:
   a) Aims to generate fake image giving a class condition
   b) GANs benefit dramatically from scaling, increasing model parameters and batch size.
   c) A “truncation trick,” a simple sampling technique that allows explicit, fine-grained control of the tradeoff between sample variety and fidelity.



threshold=2

threshold=1.5

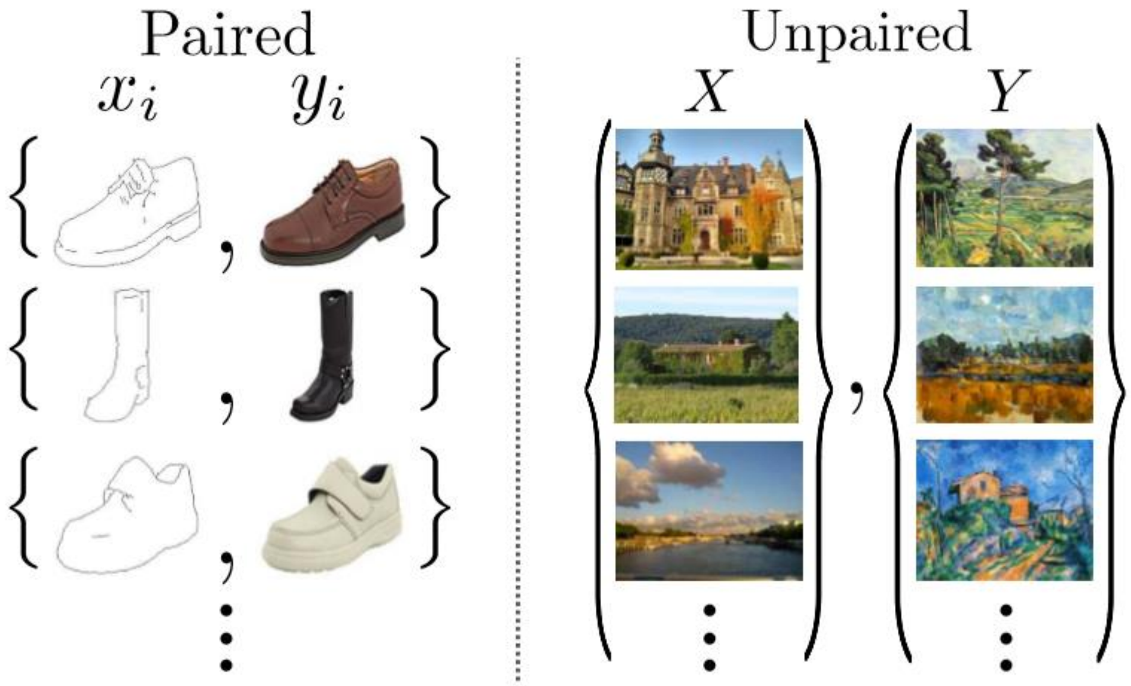
threshold=1

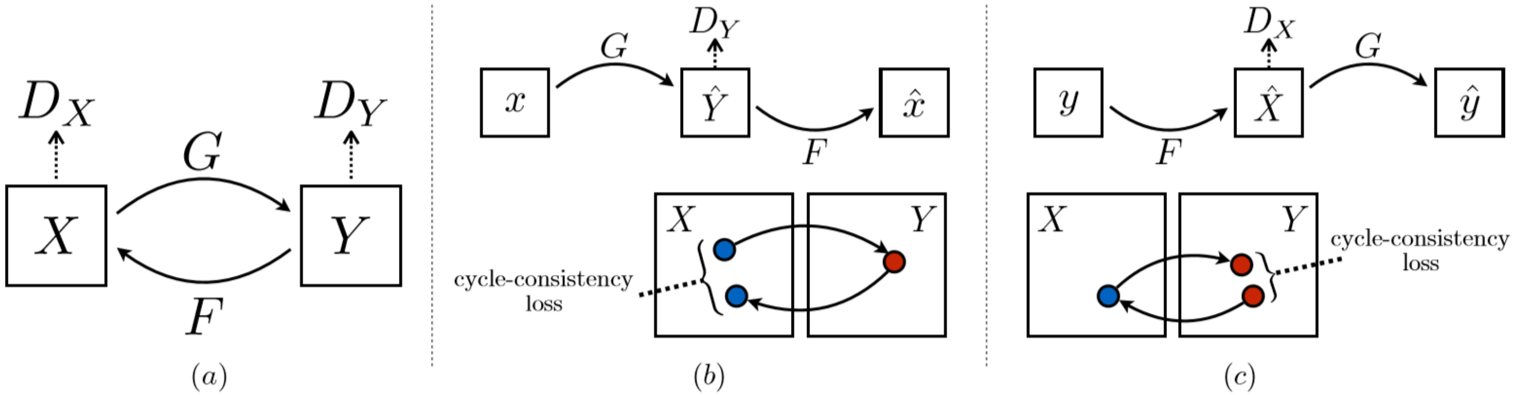
threshold=0.5

threshold=0.04

CycleGAN

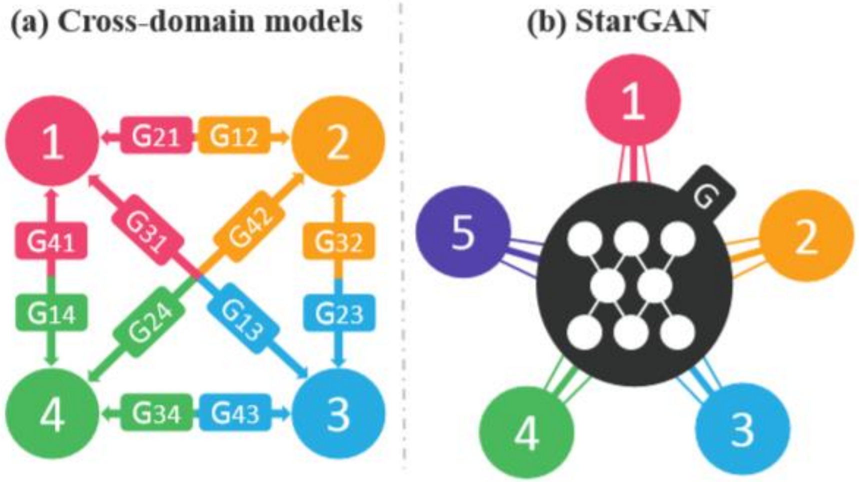
1. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2017 ICCV
2. Highlights:
   a) Image style translation
   b) Unpaired image dataset (two styles)
   c) "Cycle consistency losses": if we translate from one domain to the other and back again we should arrive where we started





StarGAN

1. Unified generative adversarial networks for multi-domain image-to-image translation. 2018 CVPR
2. Highlights:
   a) Capable of learning mappings among multiple domains using a single generator.
   b) D learns to distinguish between real and fake images and classify the real images to its corresponding domain
   c) G takes in as input both the image and target domain label and generates a fake image.



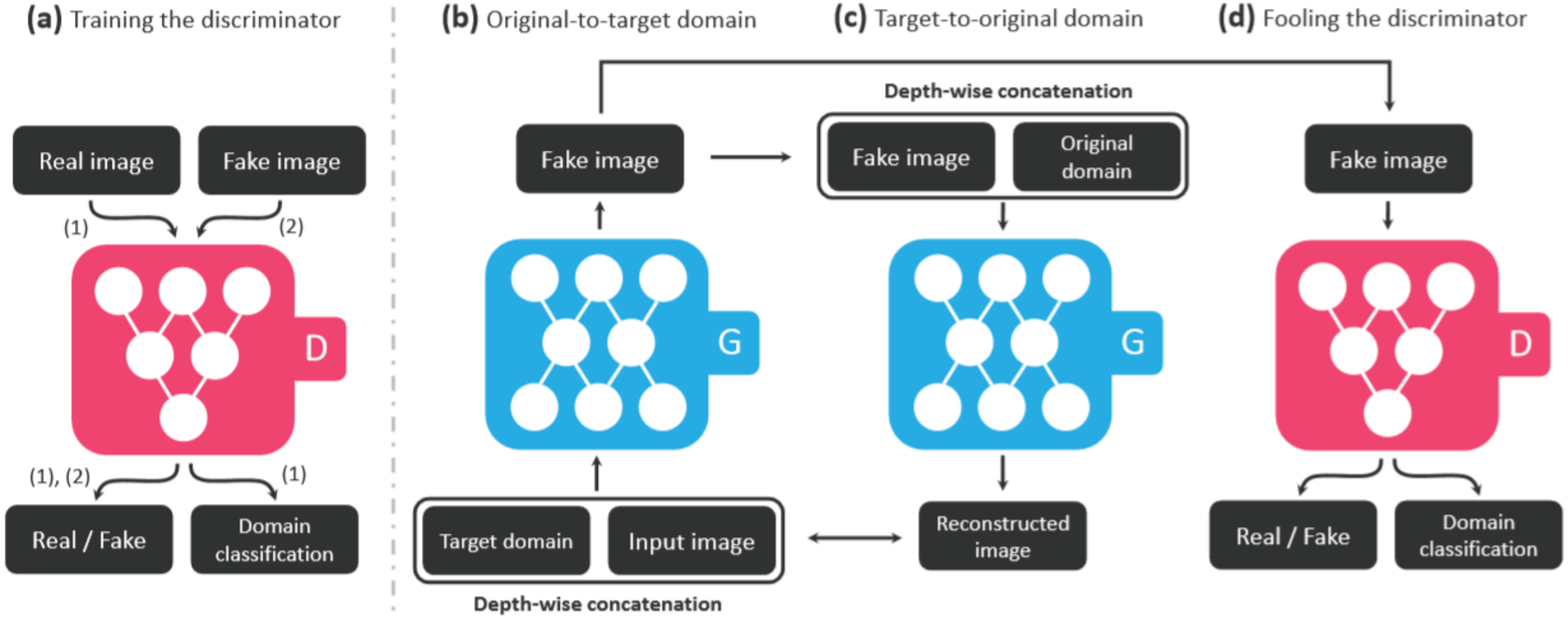
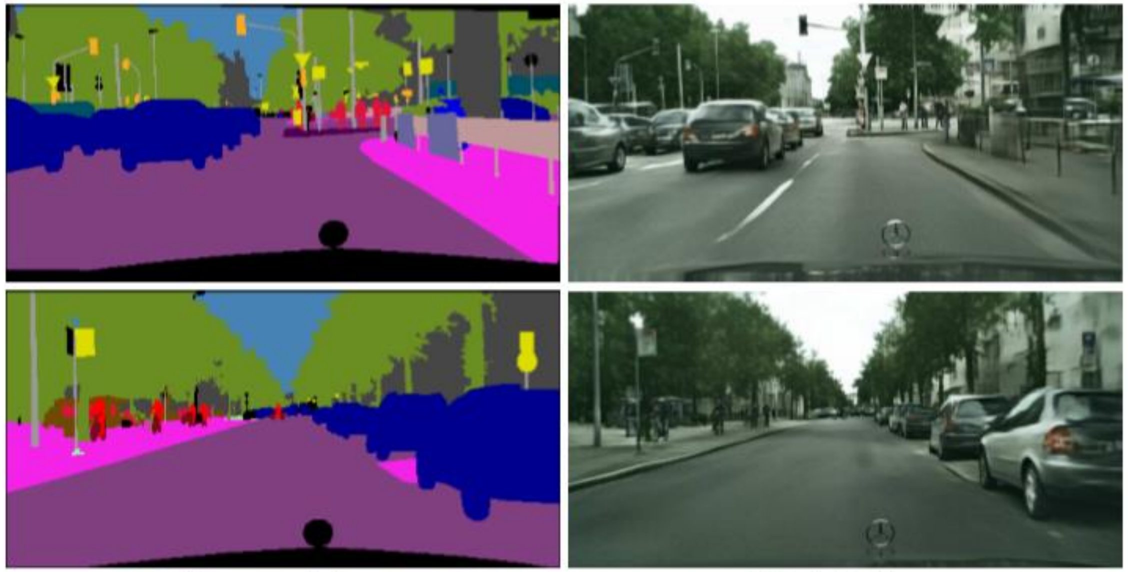


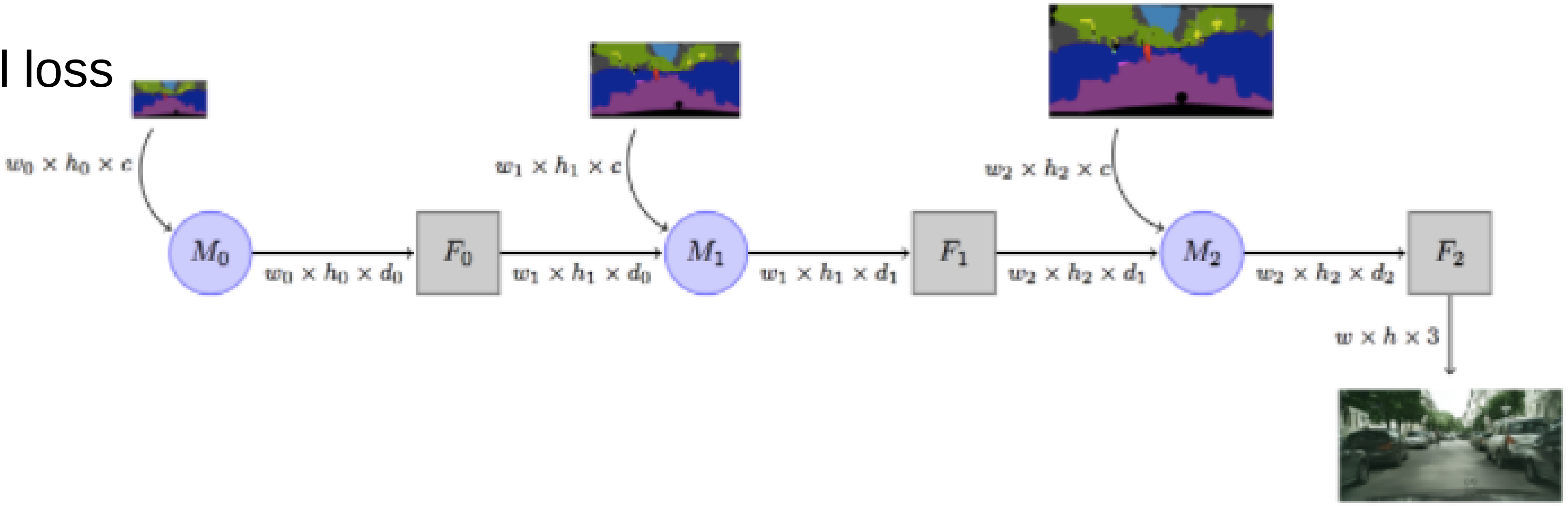
Figure 3. Overview of StarGAN, consisting of two modules, a discriminator D and a generator G. **(a)** D learns to distinguish between real and fake images and classify the real images to its corresponding domain. **(b)** G takes in as input both the image and target domain label and generates an fake image. The target domain label is spatially replicated and concatenated with the input image. **(c)** G tries to reconstruct the original image from the fake image given the original domain label. **(d)** G tries to generate images indistinguishable from real images and classifiable as target domain by D.

CRN

1. Photographic image synthesis with cascaded refinement networks. 2017 ICCV
2. Highlights:
   a) Use masks to generate high-resolution RGB images
   b) Given a pixelwise semantic layout, the presented model synthesizes an image that conforms to this layout.
   c) Trained with perceptual loss



(a) Input semantic layouts (b) Synthesized images


$w_0 \times h_0 \times c$
M_0
$w_0 \times h_0 \times d_0$
F_0
$w_1 \times h_1 \times d_0$
$w_1 \times h_1 \times c$
M_1
$w_1 \times h_1 \times d_1$
F_1
$w_2 \times h_2 \times d_1$
$w_2 \times h_2 \times c$
M_2
$w_2 \times h_2 \times d_2$
F_2
$w \times h \times 3$

SAN

1. Second-order attention network for single image super-resolution. 2019 CVPR
2. Highlights:
   a) propose second-order channel attention (SOCA) mechanism to adaptively rescale features by considering feature statistics higher than first-order.

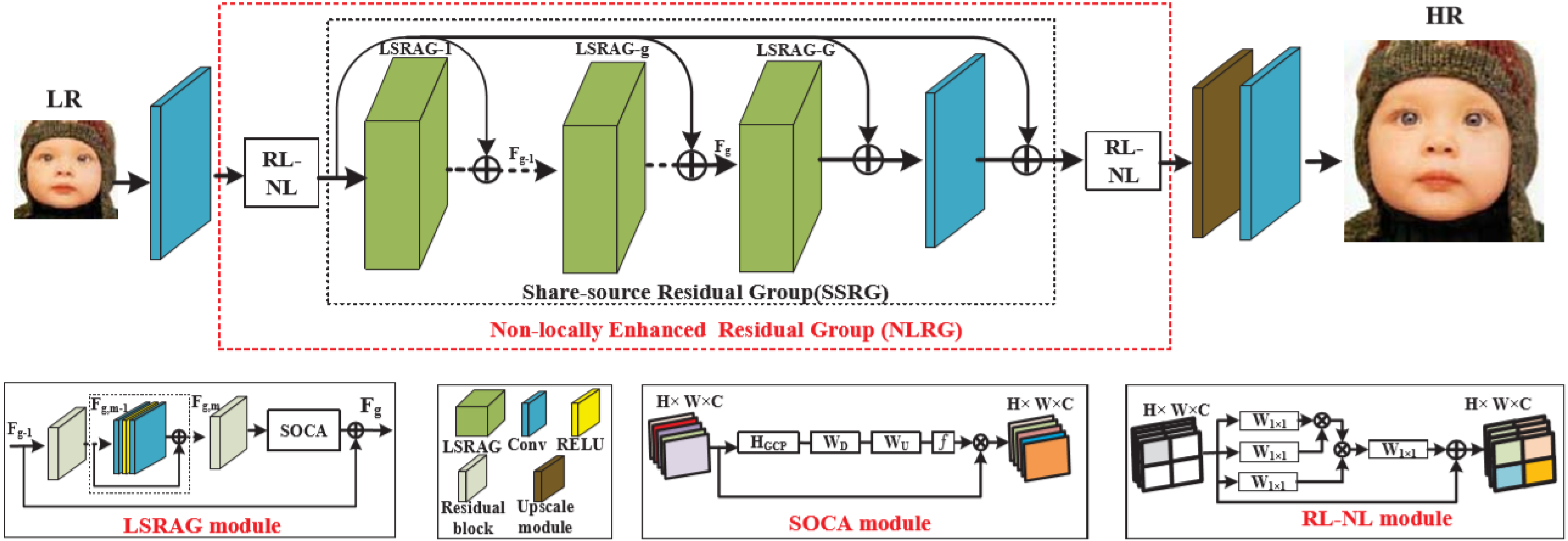


Figure 2. Framework of the proposed second-order attention network (SAN) and its sub-modules.

DeepFake

1. FaceForensics++: Learning to detect manipulated facial images, 2019 ICCV

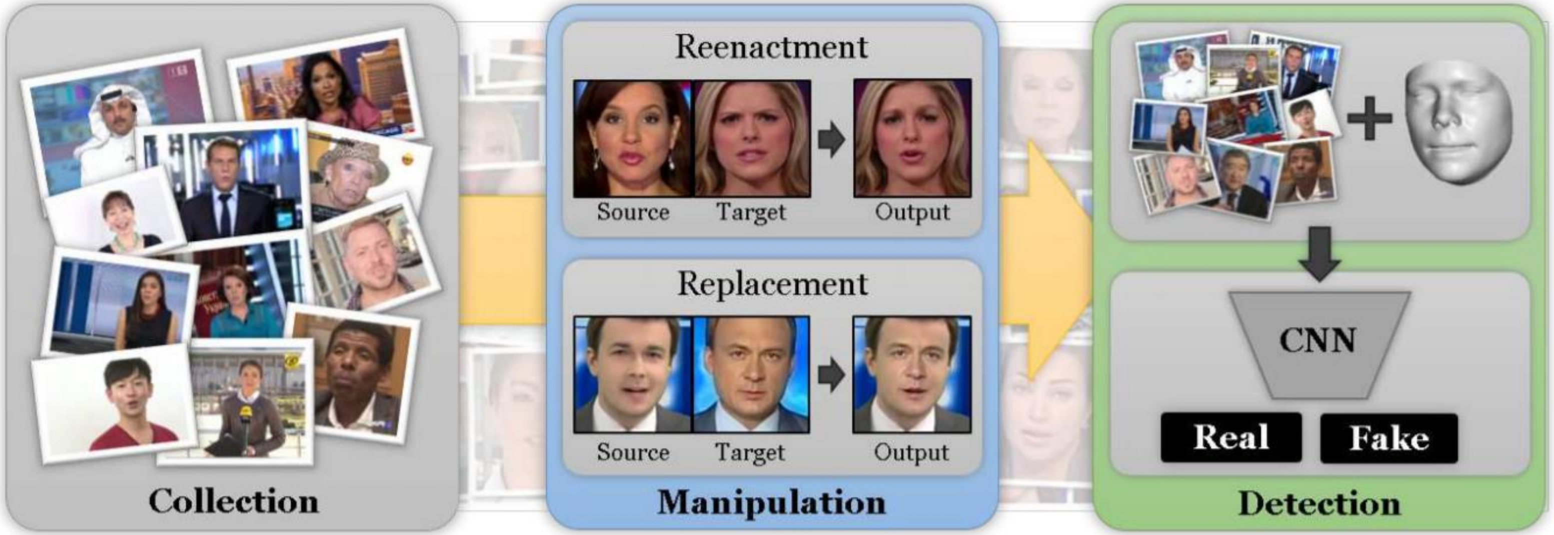
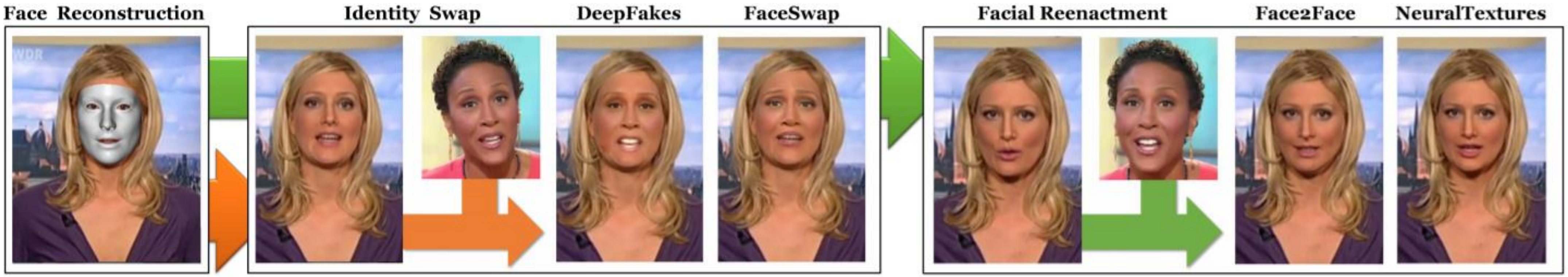


Figure 1: *FaceForensics++* is a dataset of facial forgeries that enables researchers to train deep-learning-based approaches in a supervised fashion. The dataset contains manipulations created with four state-of-the-art methods, namely, *Face2Face*, *FaceSwap*, *DeepFakes*, and *NeuralTextures*.



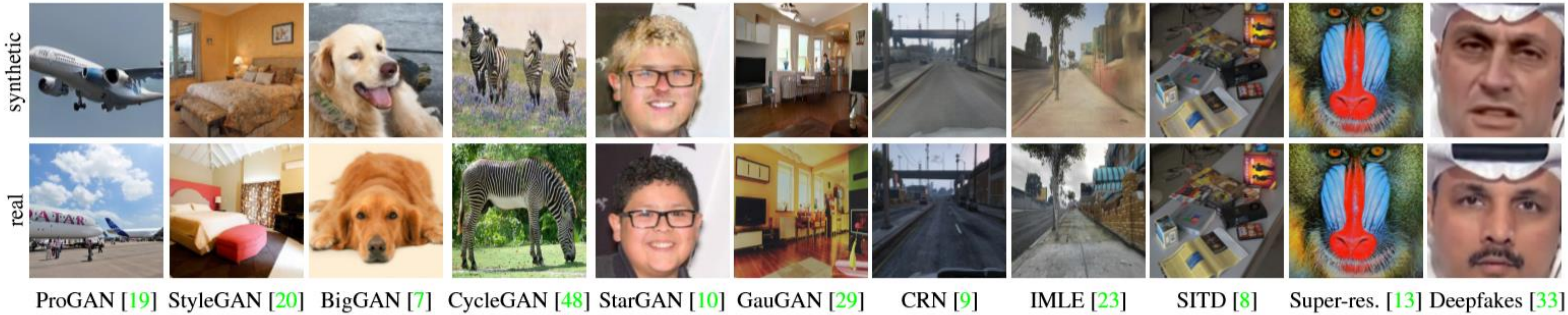


Figure 1: **Are CNN-generated images hard to distinguish from real images?** We show that a classifier trained to detect images generated by only one CNN (ProGAN, far left) can detect those generated by many other models (remaining columns). Our code and models are available at `https://peterwang512.github.io/CNNDetection/`.

Generating Fake Images

1. 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).
2. We collect an **equal number** of real images from each method’s training set.
3. **256×256** resolution is the most commonly shared output size among most off-the-shelf image synthesis models.
4. 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.
5. For models that produce images at **higher** resolution (e.g., ProGAN, StyleGAN, SAN, SITD), we **keep** the images at the same resolution.

Train One Classifier on ProGAN

1. The decision to use a single model for training most closely resembles real world detection problems.
2. We chose ProGAN since it generates **high quality images** and has a simple convolutional network structure.
3. We then create a large-scale dataset that consists **solely of ProGAN-generated images** and real images.
4. 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**.

Train a "real-or-fake" classifier on this ProGAN dataset,
and evaluate how well the model generalizes to other CNN-synthesized images.

Augmentation and Evaluation

1. During training, we simulate image post-processing operations in a variety of ways.
2. All of our models are trained with images that are randomly left-right flipped and cropped to 224 pixels.
3. Augmentation methods:
   a) **No aug**: no augmentation applied
   b) **Gaussian blur**: before cropping, with 50% probability, images are blurred with σ ∼ Uniform[0, 3]
   c) **JPEG**: with 50% probability images are JPEG-ed by two popular libraries, OpenCV [6] and the Python Imaging Library (PIL), with quality ∼ Uniform{30, 31, . . . , 100}
   d) **Blur+JPEG** (0.5): the image is possibly blurred and JPEGed, each with 50% probability
   e) **Blur+JPEG** (0.1): similar to (4a), but with 10% probability.
4. Evaluate our model’s performance on each dataset using average precision (AP)
5. During testing, each image is center-cropped 224x224 without resizing

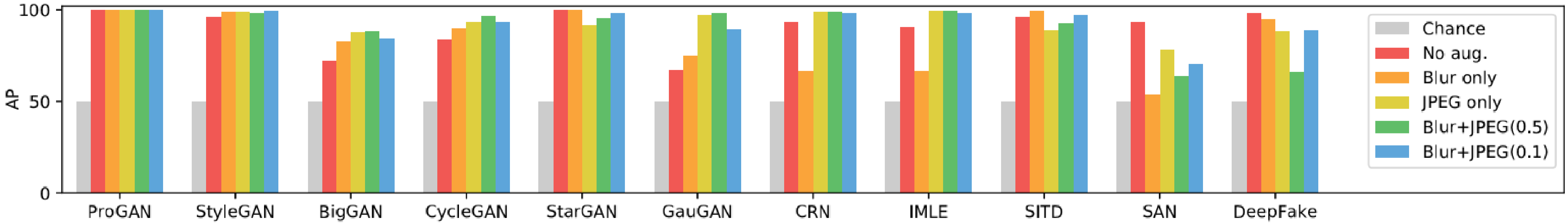


Figure 2: **Effect of augmentation methods.** All detectors are trained on ProGAN, and tested on other generators (AP shown). In general, training with augmentation helps performance. Notable exceptions are super-resolution and DeepFake.

1. Without augmentation, it generalizes extremely well to StyleGAN, which has a similar network structure, but not as well to BigGAN.
2. The performance on BigGAN significantly improves 72.2-88.2 adding augmentations.
3. SAN and DeepFake, where directly training on ProGAN without augmentation performs strongly (93.6 and 98.2, respectively), but augmentation hurts performance.
4. Applying augmentation, but at reduced rate (Blur+JPEG (0.1)), offers a good balance.

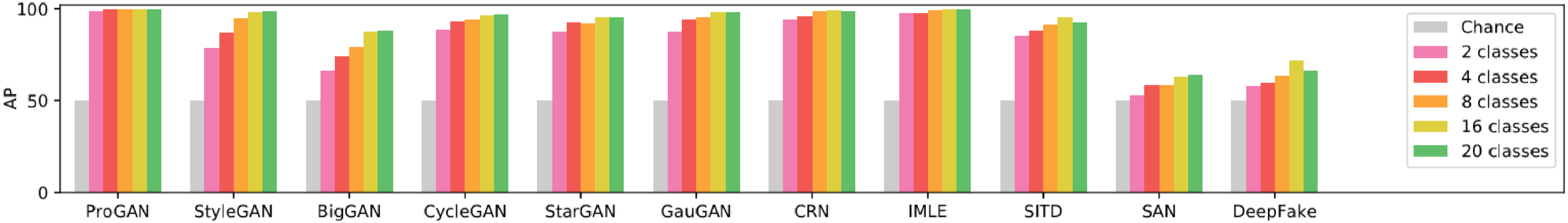


Figure 3: **Effect of dataset diversity.** All detectors are trained on ProGAN, and tested on other generators (AP shown). Training with more classes improves performance. All runs use blur and JPEG augmentation with 50% probability.

1. Image diversity improves performance.
2. 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.
3. 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.
4. This indicates that there may be a training dataset that is “**diverse enough**” for practical generalization

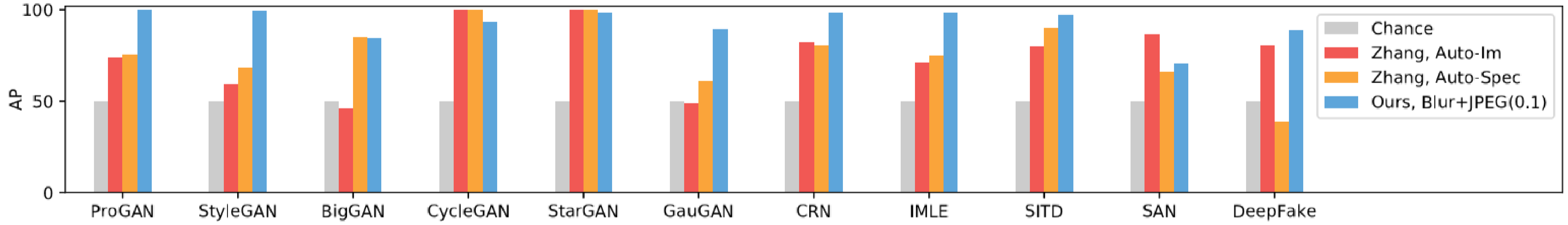
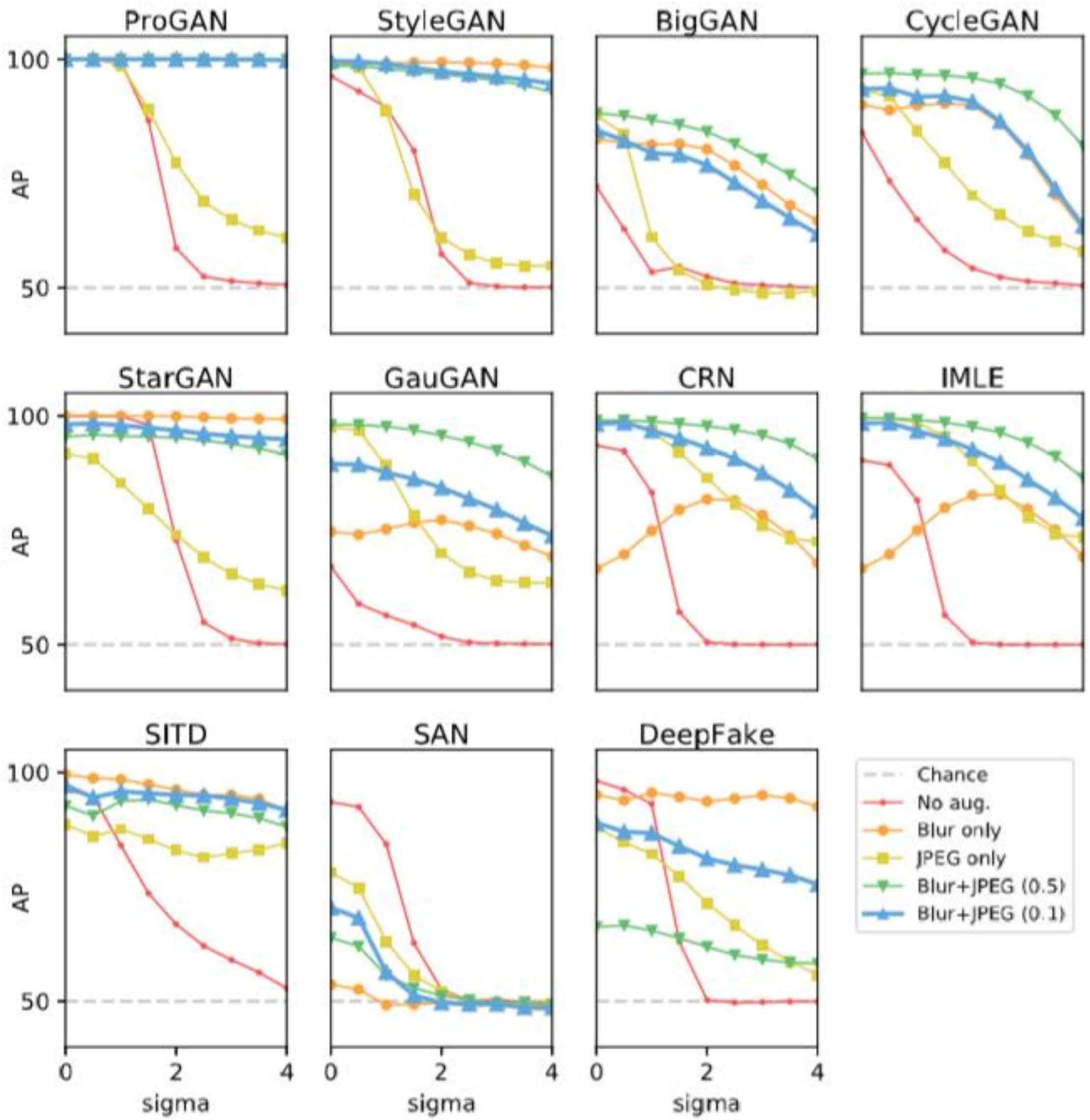


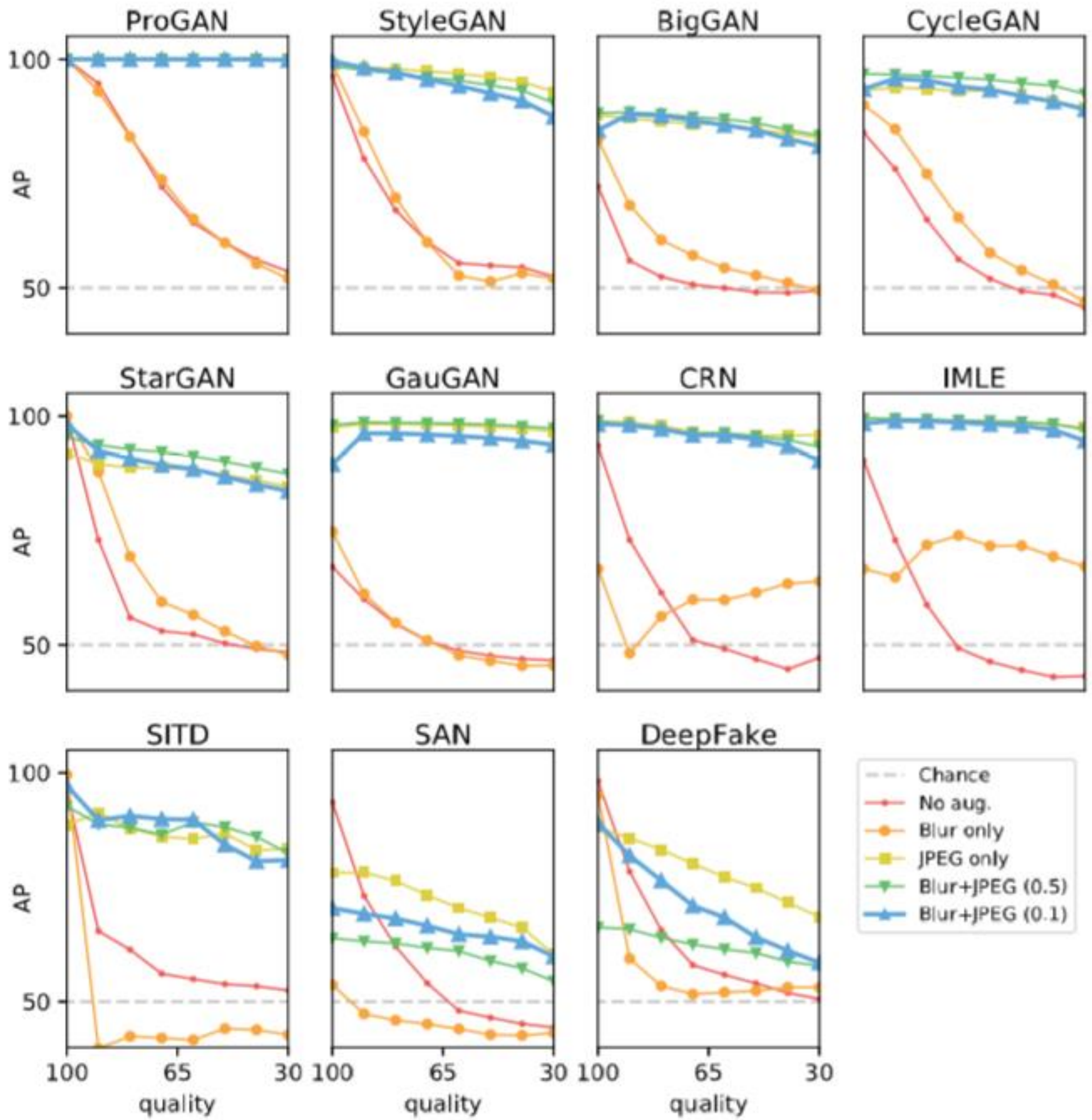
Figure 4: **Model comparison.** Compared to Zhang *et al.* [44], we observe that for the most part, our models generalize better to other architectures. Notable exceptions to this are CycleGAN (which is identical to the training architecture from [44]), StarGAN (where both methods obtain close to 100.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

1. How our generalization performance compares to other proposed forensic methods?
2. Zhang’s model, AutoGAN, is an autoencoder based on CycleGAN’s generator that simulates artifacts resembling that of CycleGAN images
3. 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)

Robustness to Blur



Robustness to JPEG



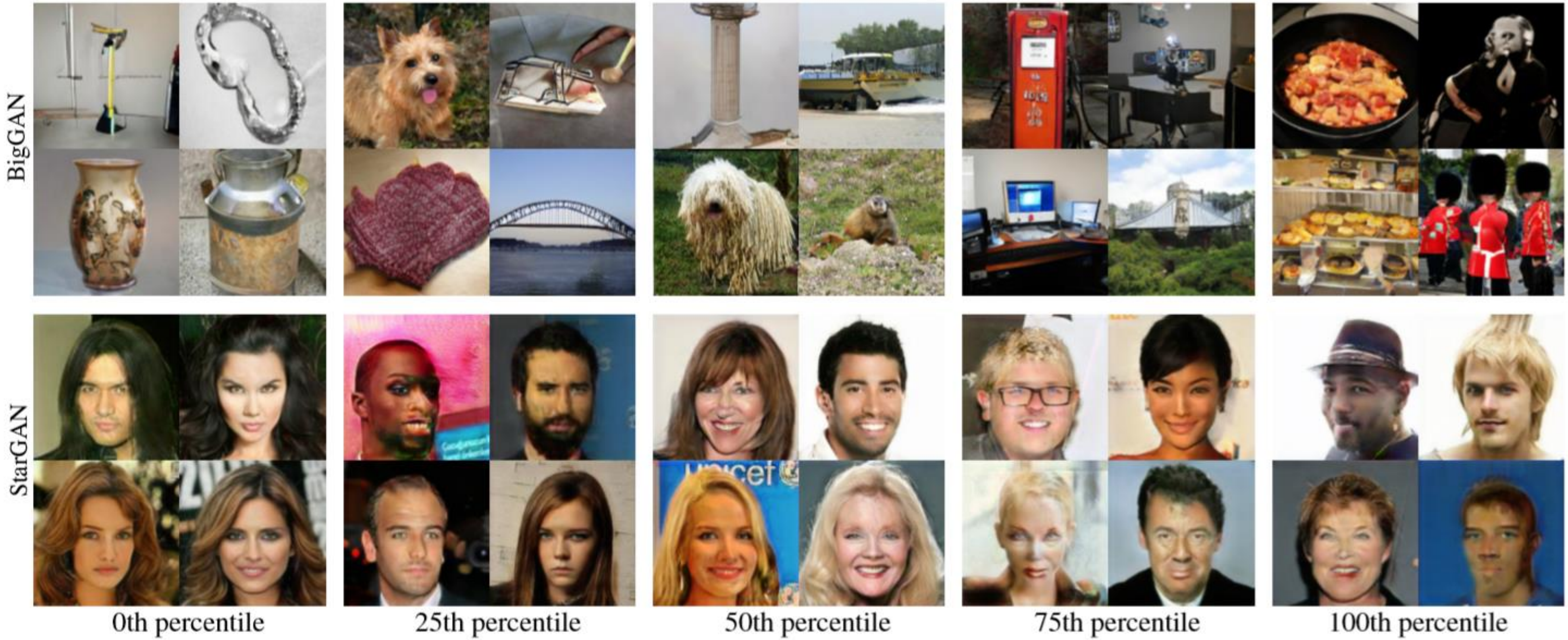


Figure 6: **Does our model's confidence correlate with visual quality?** We have found that for two models, BigGAN and StarGAN, the images on the left (considered more real) tends to look better than the images on the right (considered more fake). However, this does not seem to hold for the other models.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

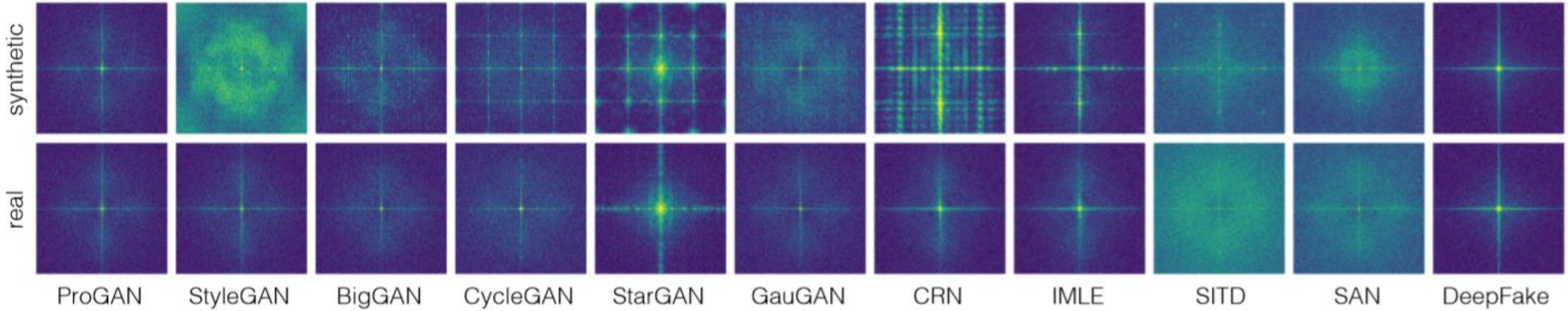


Figure 7: **Frequency analysis on each dataset.** We show the average spectra of each high-pass filtered image, for both the real and fake images, similar to Zhang *et al.* [44]. We observe periodic patterns (dots or lines) in most of the synthetic images, while BigGAN and ProGAN contains relatively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

1. Researchers have shown, recently, that common CNN designs contain artifacts that reduce their representational power.
2. Much of this work has focused on the way networks perform upampling and downsampling.
3. A well-known example of such an artifact is the checkerboard artifact produced by deconvolutional layers.

Conclusions

1. We show that forensics models trained on CNN-generated images exhibit a surprising amount of generalization to other CNN synthesis methods;
2. We find that data augmentation, in the form of common image postprocessing operations, is critical for generalization;
3. We also find that diversity of training images matters; large datasets sampled from CNN synthesis methods lead to better classifiers than those trained on smaller datasets, to a point
4. We propose a new dataset and evaluation metric for detecting CNN-generated images;
5. We experimentally analyze the factors that account for cross-model generalization.