Generative Adversarial Networks

Xiaolong Wang

Previous classes

- Discriminative models for image recognition
- Discriminative models for video understanding

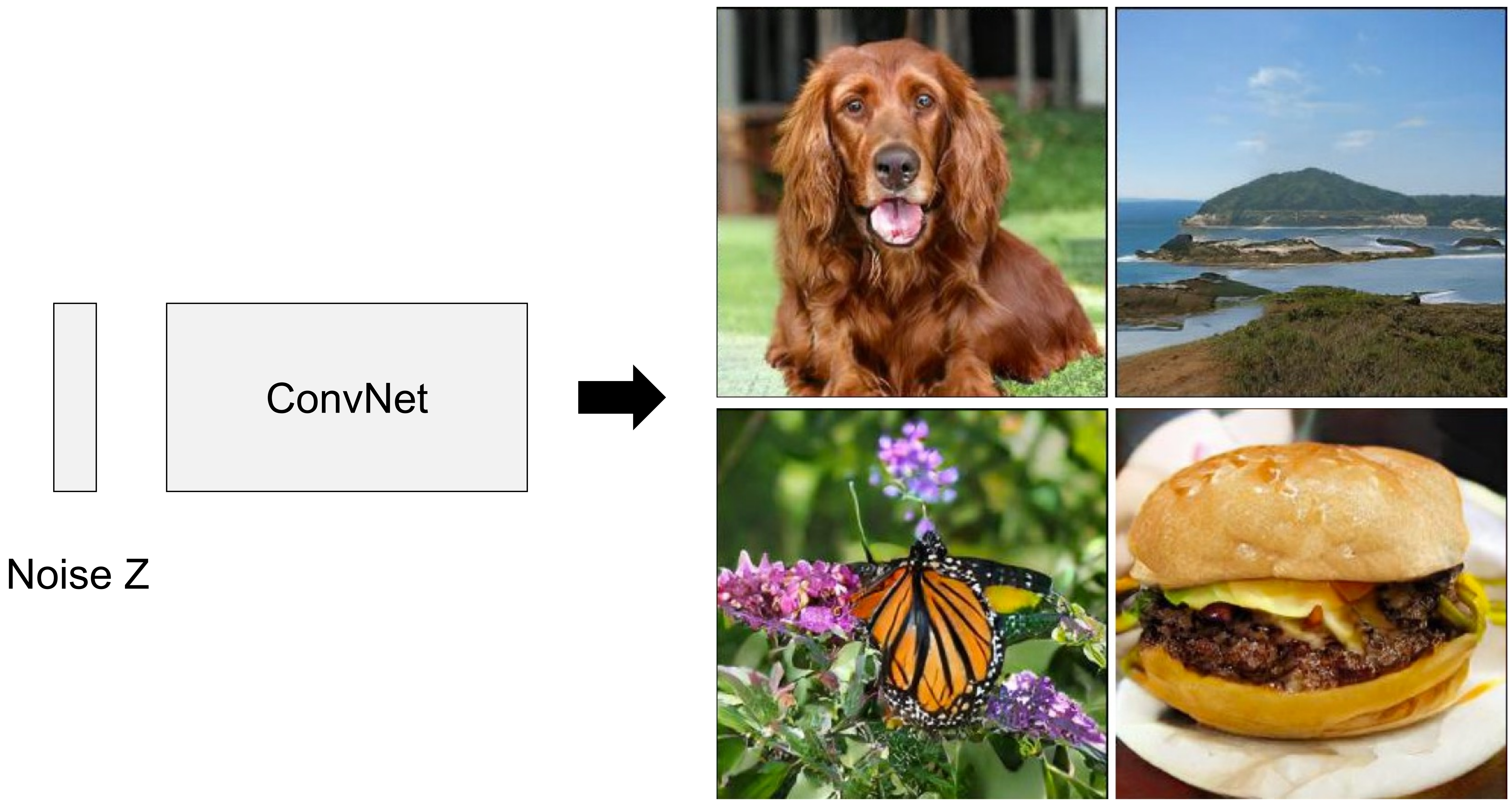
This Class

- Generative Adversarial Networks, DCGAN
- Progressive GAN, StyleGAN
- Evaluating GANs
- Adversarial Examples

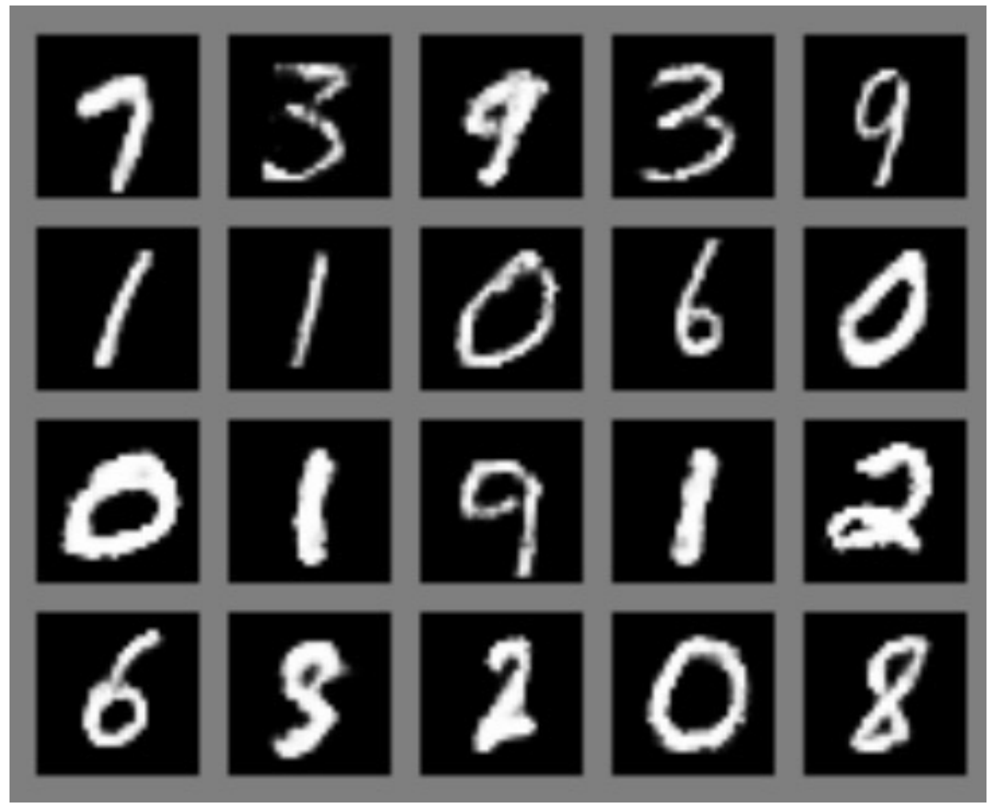
Slides partially from: http://cs231n.stanford.edu/ https://slazebni.cs.illinois.edu/fall20/

Generative Adversarial Networks

Generative Adversarial Networks

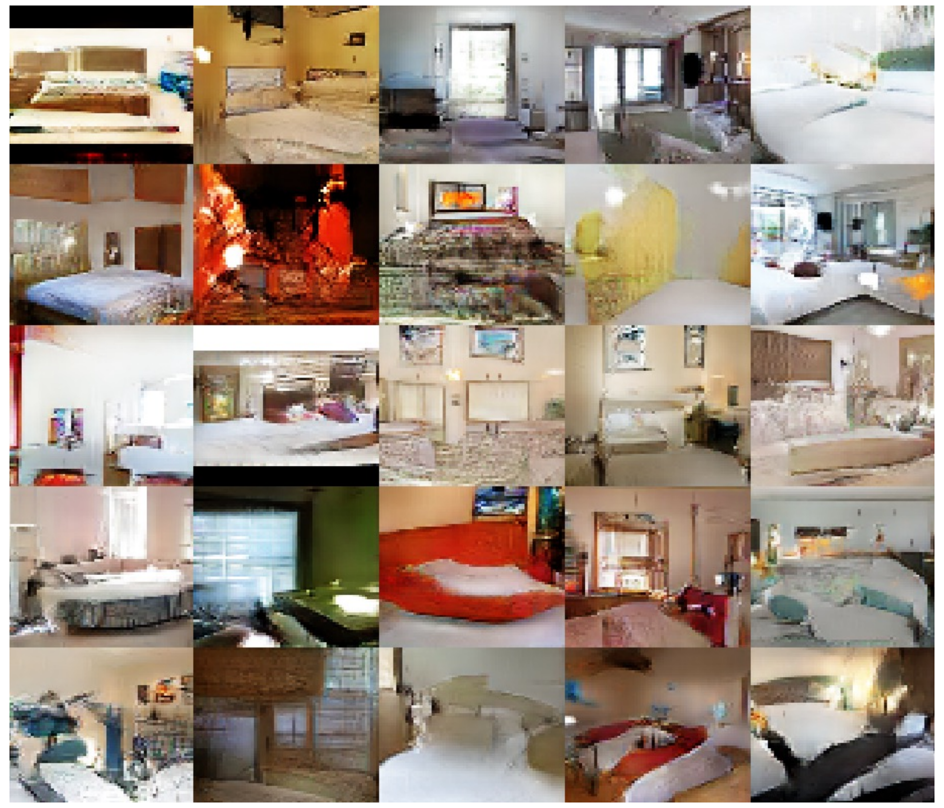


Generative Adversarial Networks



Goodfellow et al. 2014

Brock et al. 2019



Radford et al. 2016.

Karras et al. 2019

GANs

- Generator: Takes noise vector as inputs and outputs the image.

- Discriminator: Classify the images as real or fake.

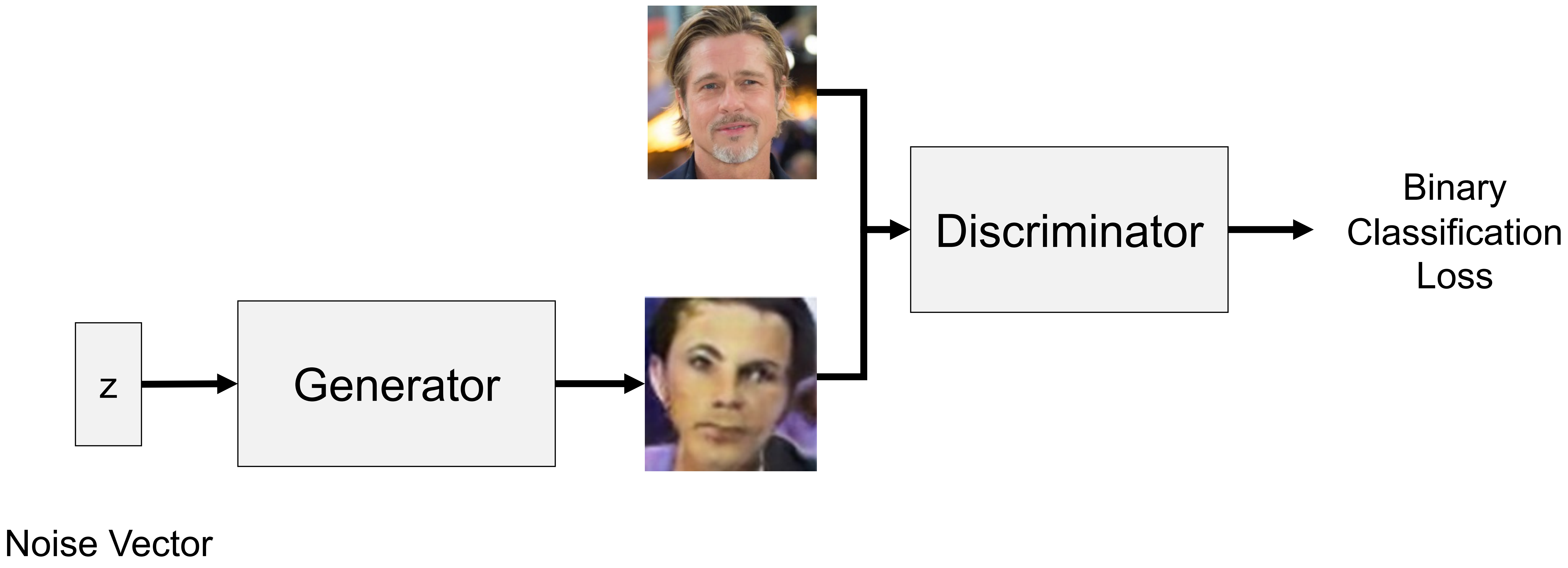
Learning to sample

Training data $x \sim p_{\text{data}}$

Generated samples $x \sim p_{\text{model}}$

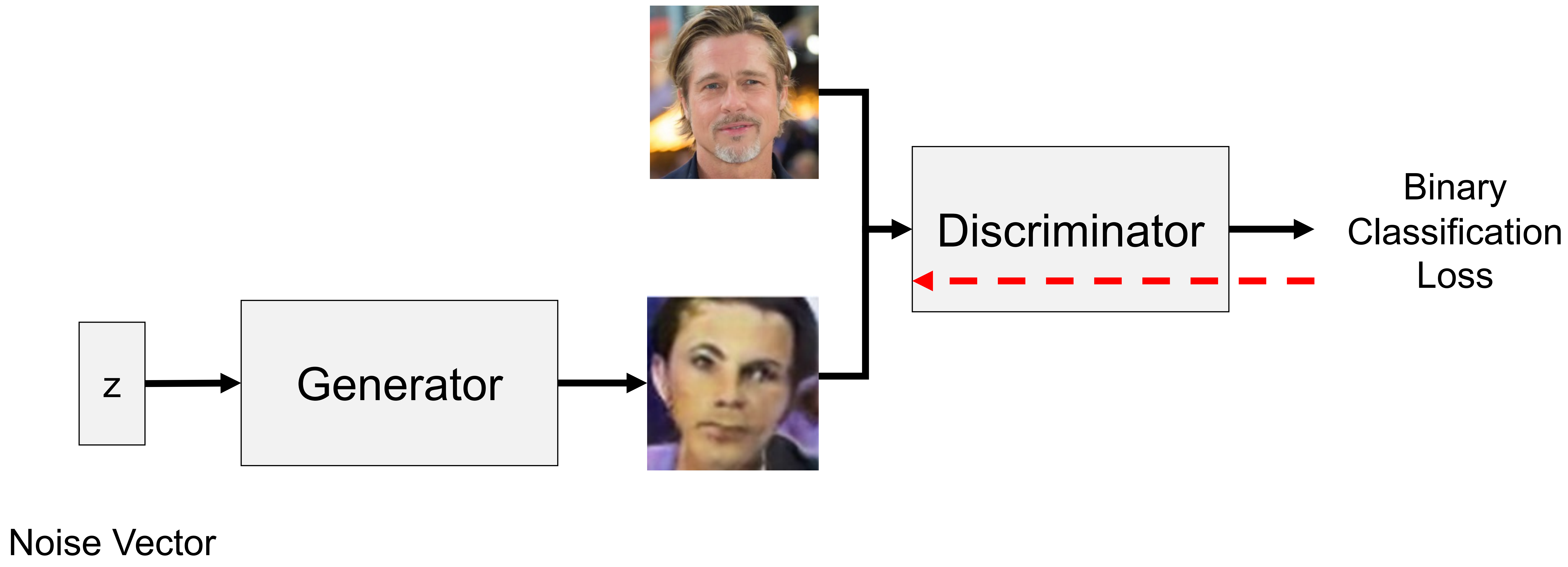
We want to learn p_{model} that matches p_{data}

GANs



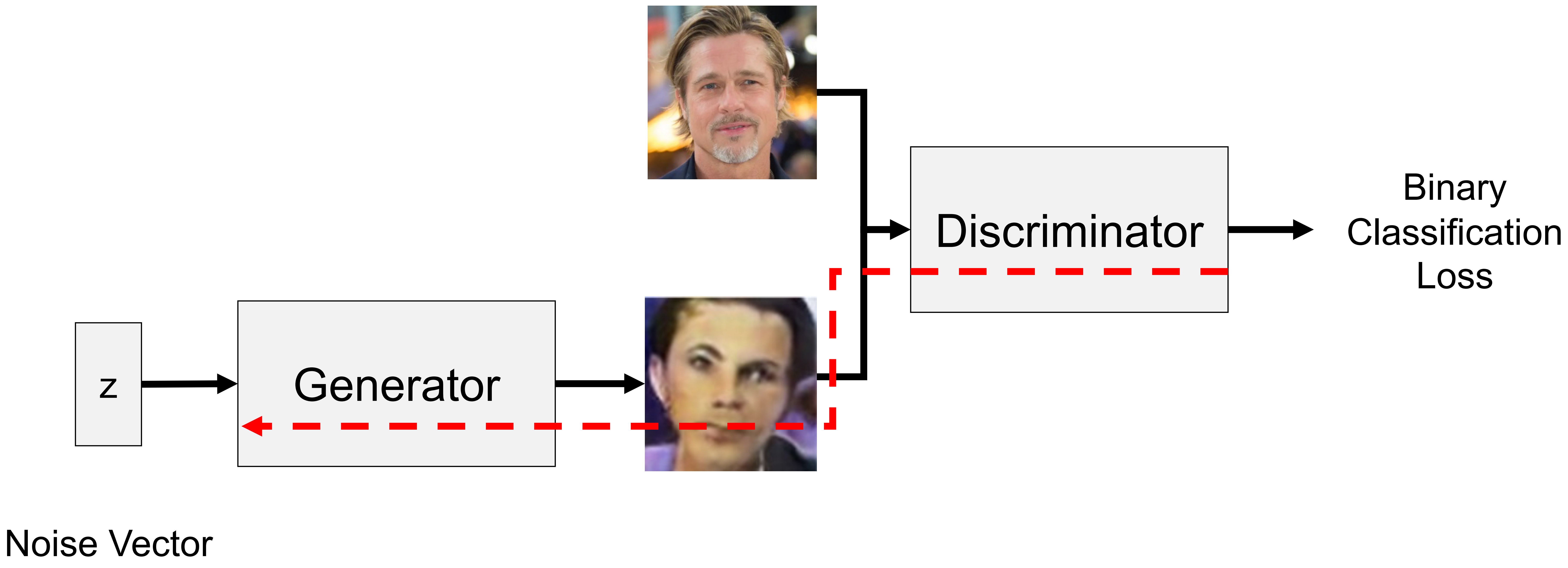
Goodfellow et al., 2014

GANs



Training Discriminator: Minimize the binary classification loss.

GANs



Training Generator: Maximize the binary classification loss.

GAN objective

- The discriminator $D(x)$ should output the probability that the sample x is real
  - That is, we want $D(x)$ to be close to 1 for real data and close to 0 for fake
- Expected conditional log likelihood for real and generated data:
- $\mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{x \sim p_{\text{gen}}} \log\big(1 - D(x)\big)$
- $= \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z \sim p} \log(1 - D(G(z)))$

We seed the generator with noise z drawn from a simple distribution p (Gaussian or uniform)

GAN objective

$$V(G,D) = \mathbb{E}_{x\sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z\sim p} \log(1 - D(G(z)))$$

- The discriminator wants to correctly distinguish real and fake samples:

$$D^* = \arg\max_D V(G,D)$$

- The generator wants to fool the discriminator:

$$G^* = \arg\min_G V(G,D)$$

- Train the generator and discriminator jointly in a *minimax game*

Original GAN results

MNIST digits

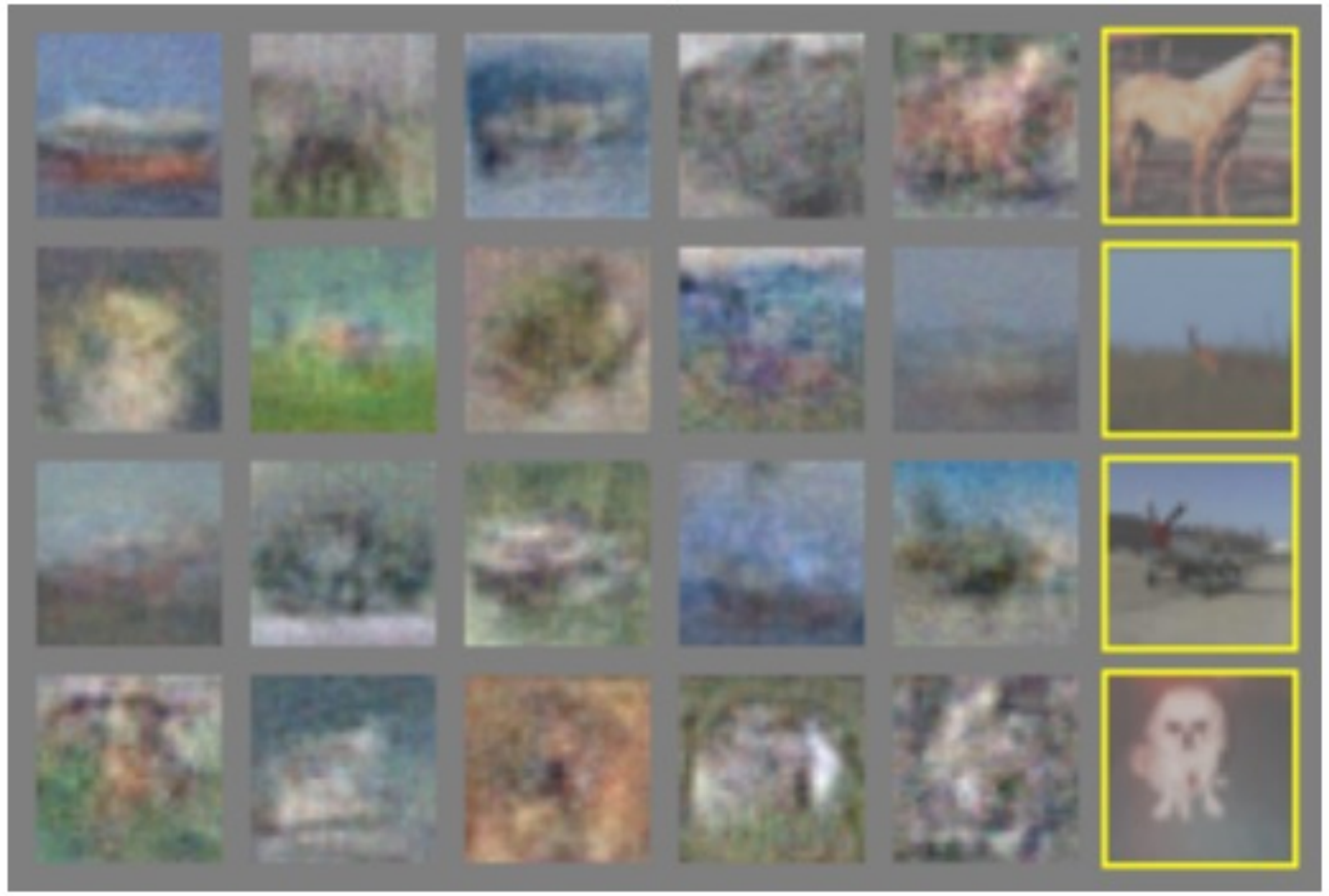
Toronto Face Dataset

Nearest real image
for sample to the left

Goodfellow et al., 2014

Original GAN results

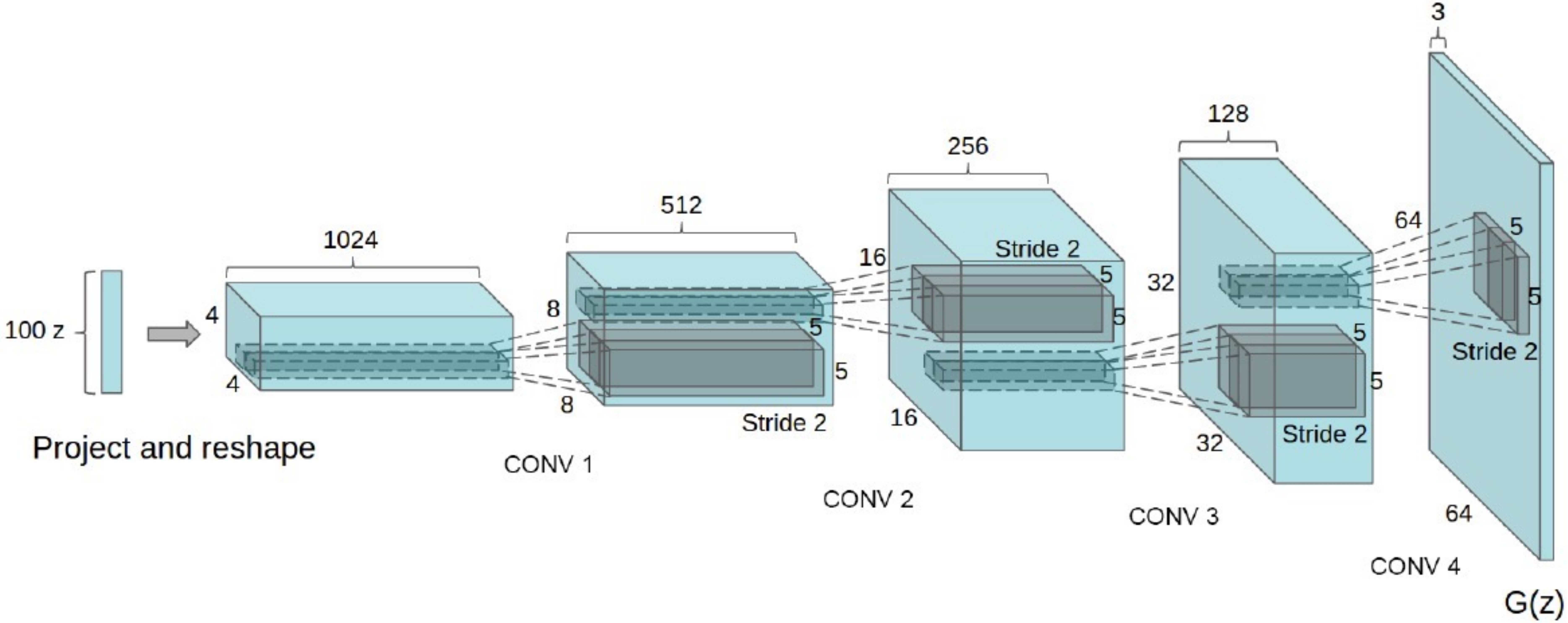
CIFAR-10 (FC networks)



CIFAR-10 (conv networks)

Goodfellow et al., 2014

DCGANs



Radford et al., 2016.

DCGANs

- Early, influential convolutional architecture for generator
- Discriminator architecture:
  - Don’t use pooling, only strided convolutions
  - Use Leaky ReLU activations (sparse gradients cause problems for training)
  - Use only one FC layer before the softmax output
  - Use batch normalization after most layers (in the generator also)

Radford et al., 2016.

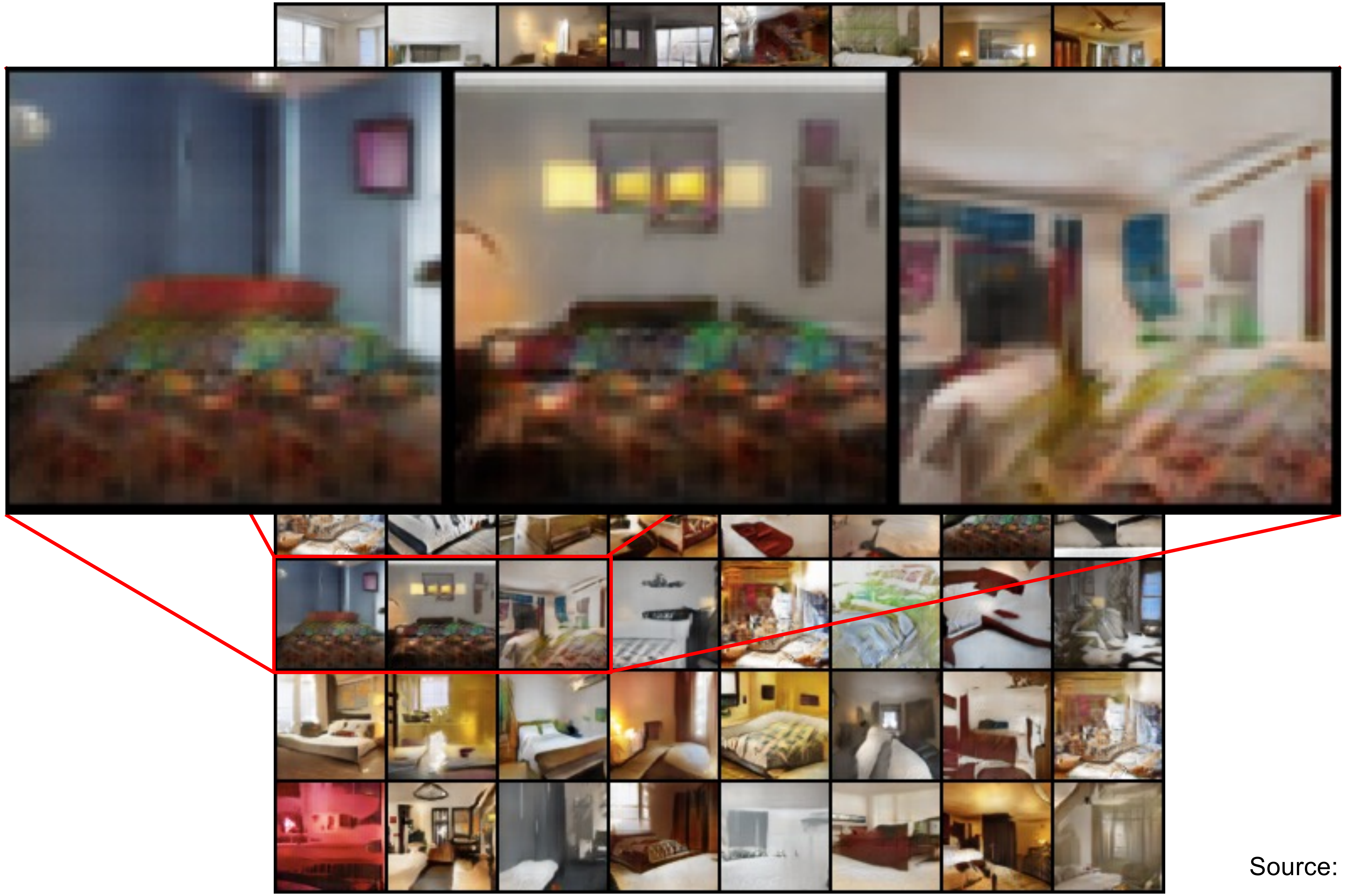
DCGAN results

Generated bedrooms after one epoch

DCGAN results

Generated bedrooms after five epochs

DCGAN results



Source: F. Fleuret

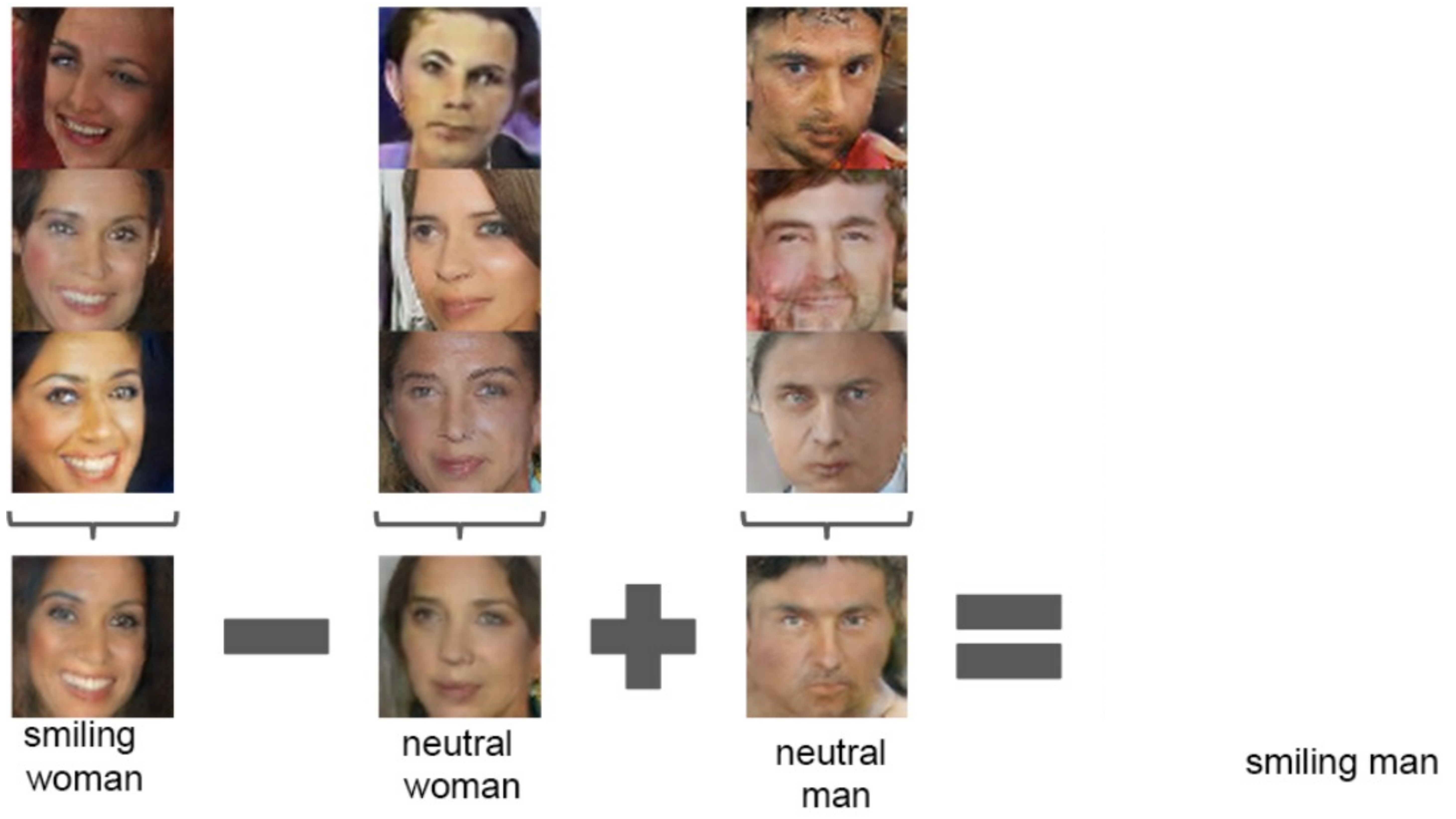
DCGAN results

$$\alpha\, z_0 + (1-\alpha) z_1$$

$z_0 \longleftrightarrow z_1$

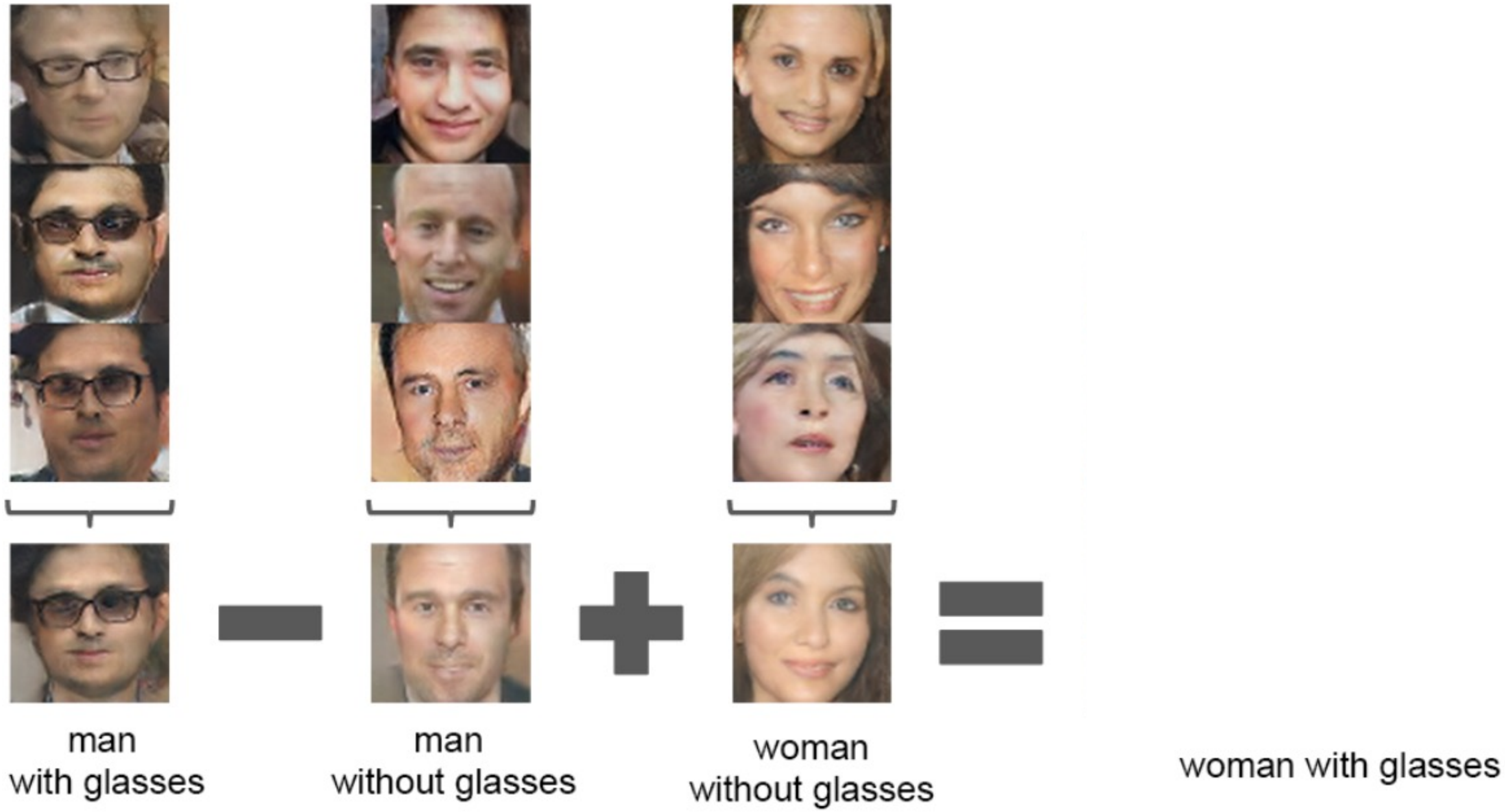
DCGAN results

- Vector arithmetic in the z space



DCGAN results

- Vector arithmetic in the z space



DCGAN results

- Pose transformation by adding a “turn” vector

BigGAN, Progressive GAN, StyleGAN

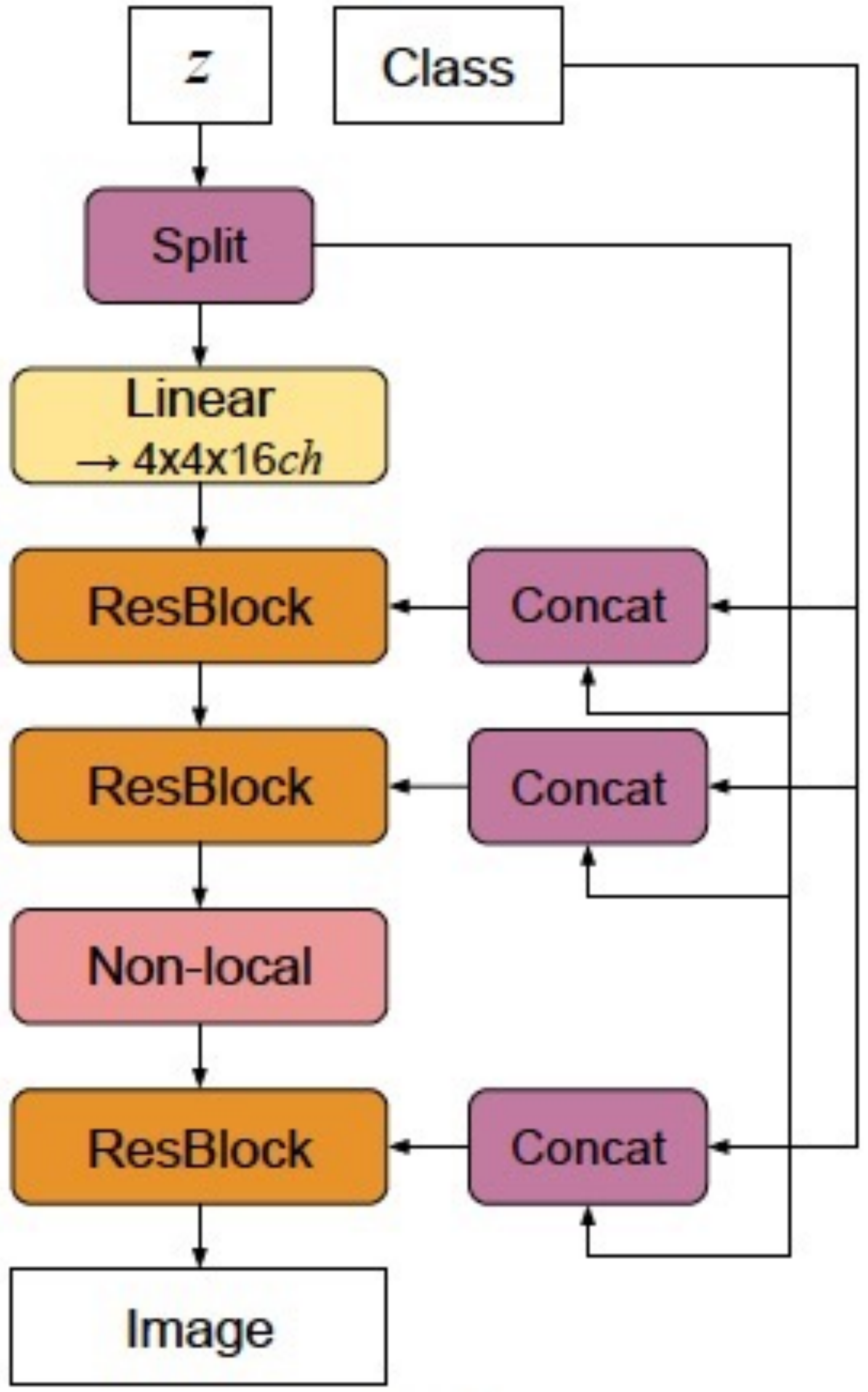
BigGANs

Brock et al. Large Scale GAN Training for High Fidelity Natural Image Synthesis. 2019.

BigGANs

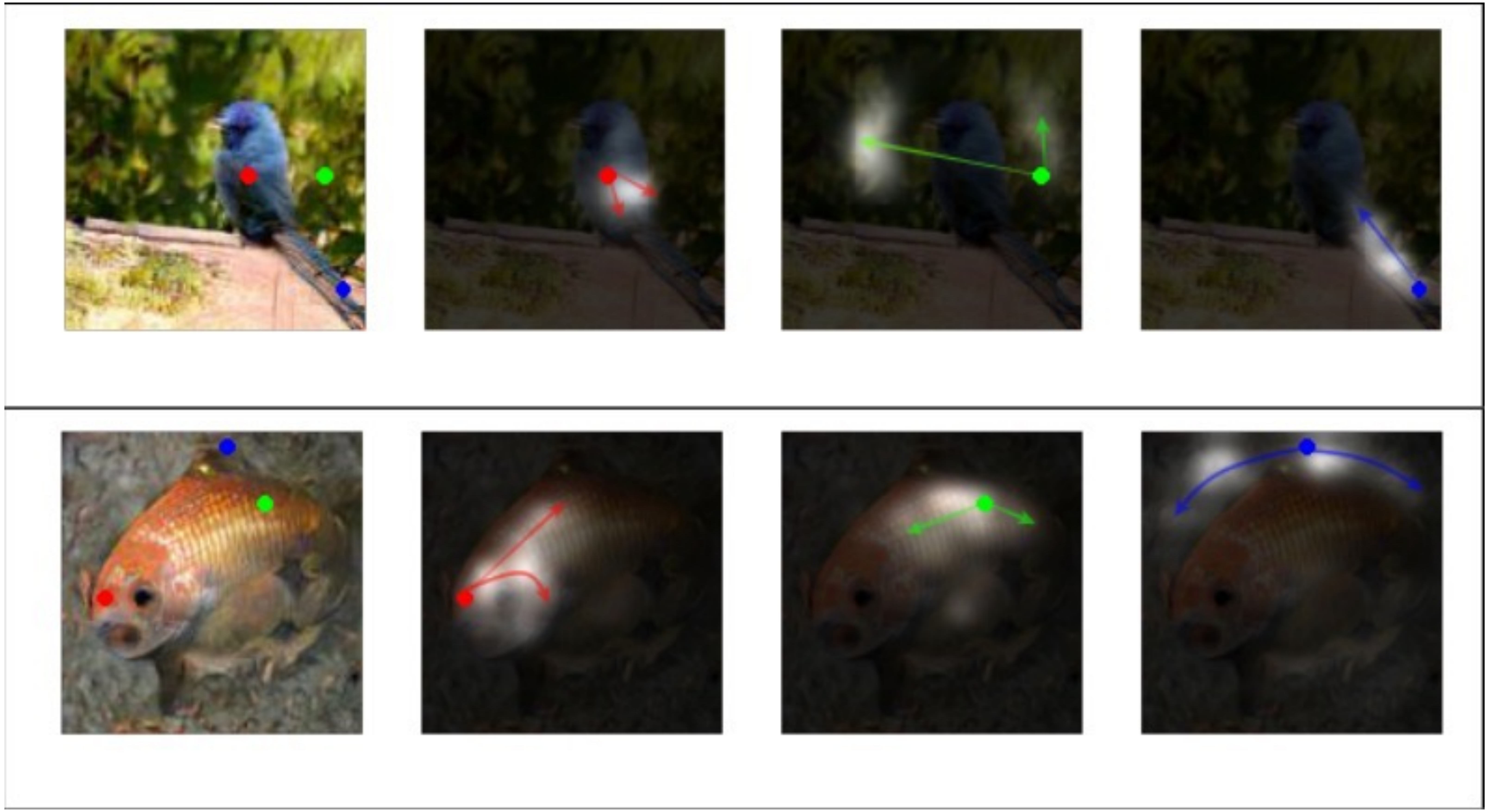
- Large Batch Size: 2048 Images
- Class Conditional Batch Normalization
- Non-local Operator

Conditional Batch Normalization



(a)

Non-local Operator



Zhang et al. Self-Attention Generative Adversarial Networks. 2019.
Wang et al. Non-local Neural Networks. CVPR 2018.

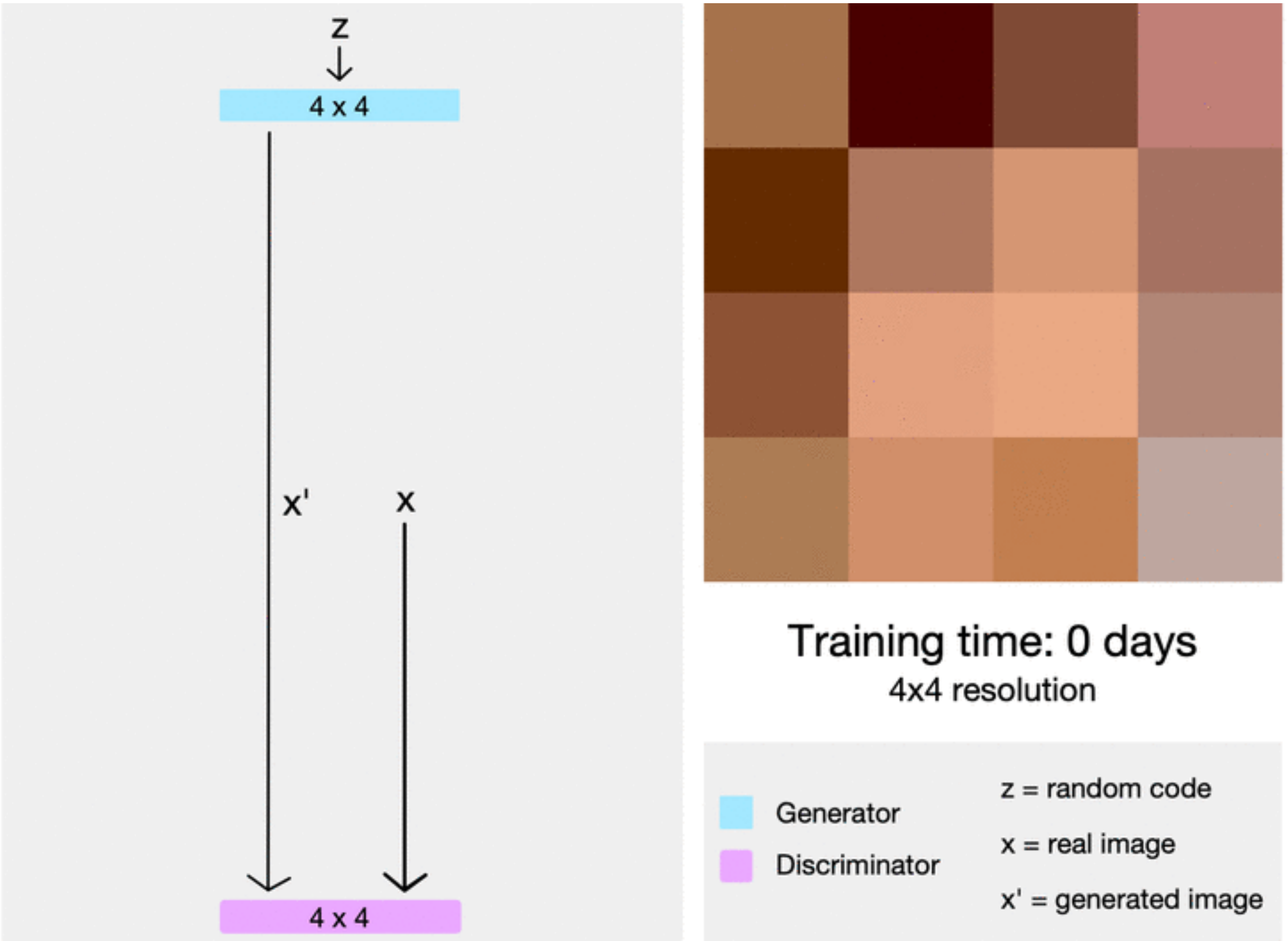
BigGANs Interpolation

Progressive GANs

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

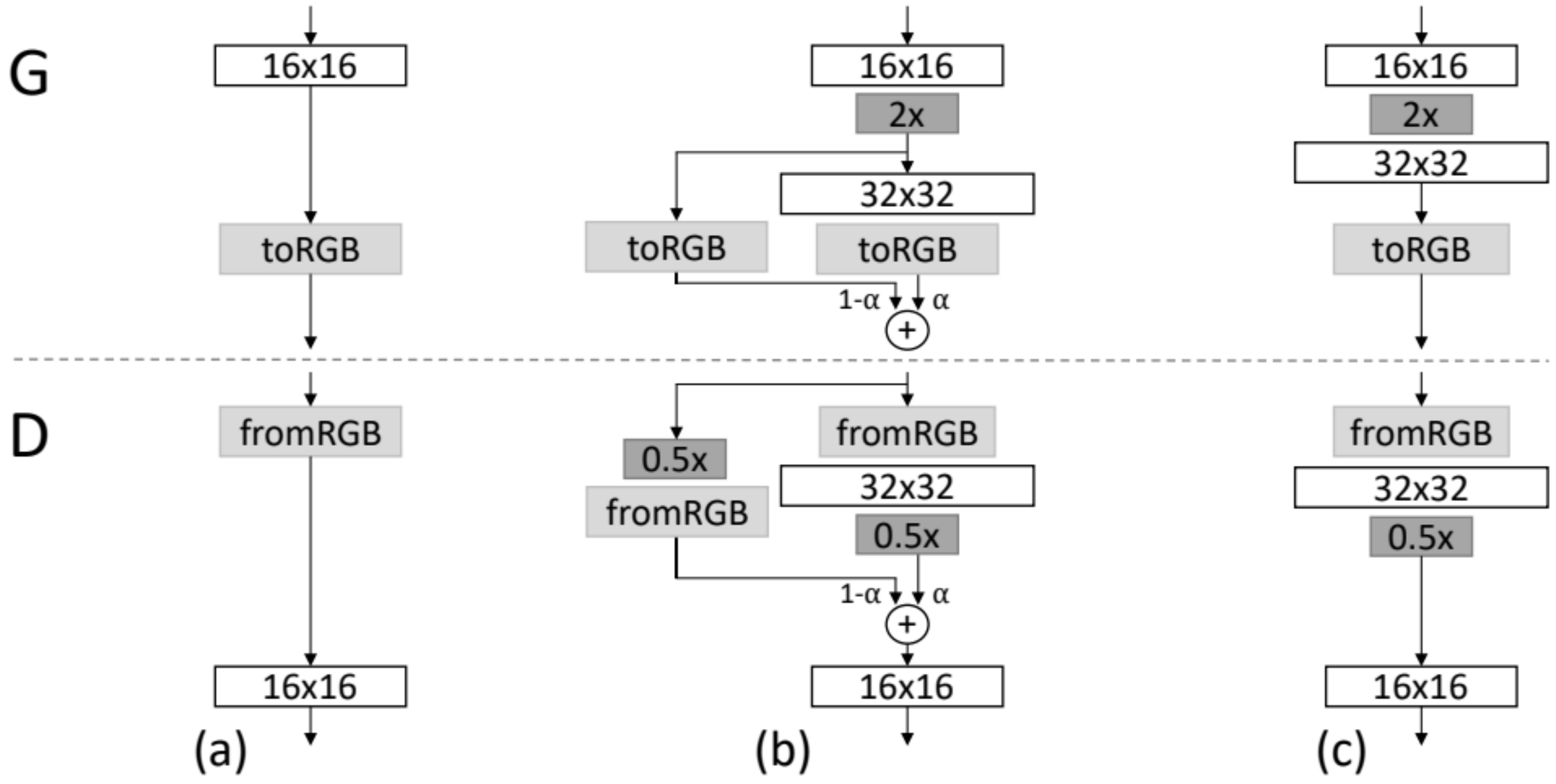
Progressive GANs

- Key idea: train lower-resolution models, gradually add layers corresponding to higher-resolution outputs



Progressive GANs

- Key idea: train lower-resolution models, gradually add layers corresponding to higher-resolution outputs

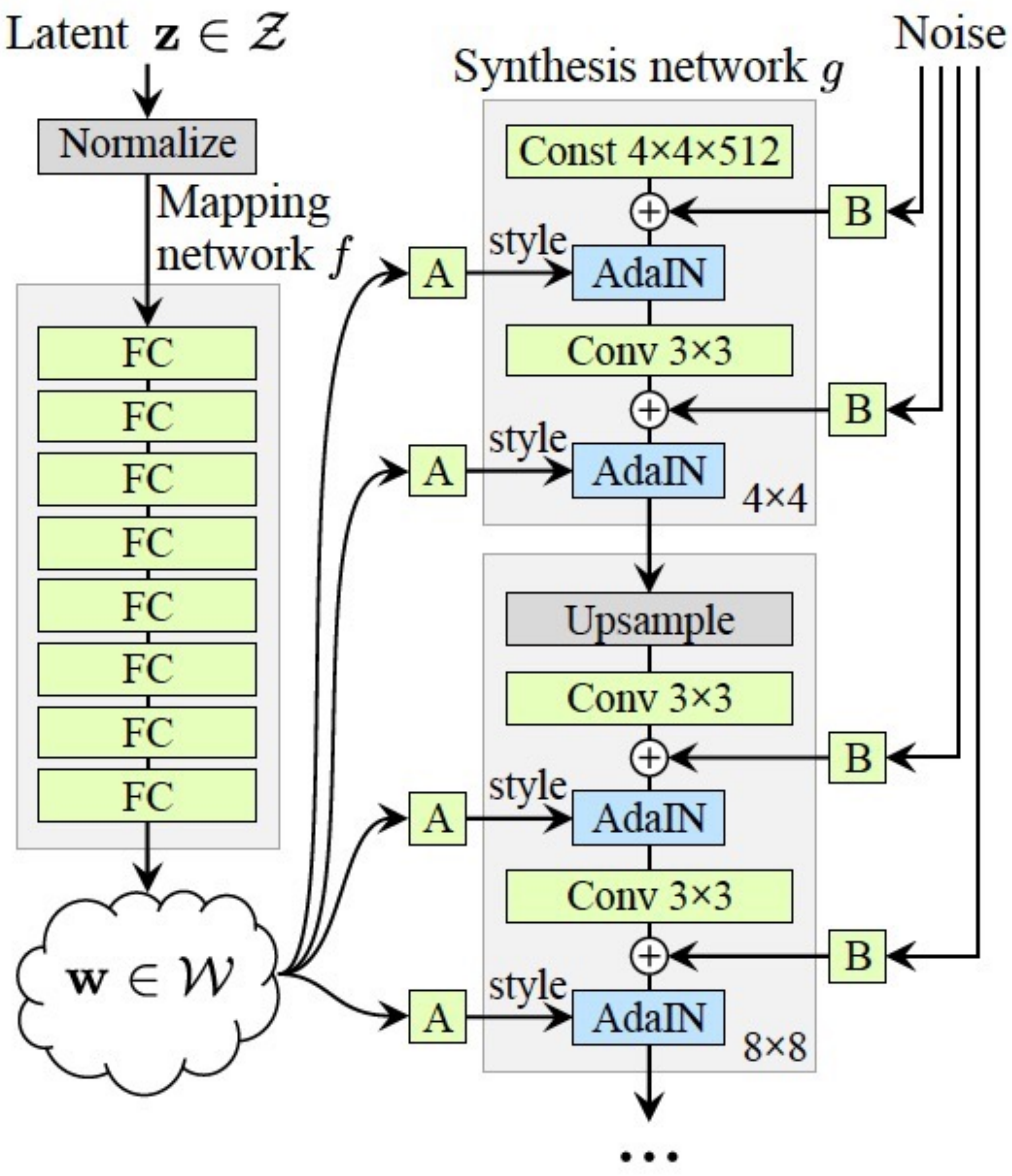


Progressive GANs: Results

256 x 256 results for LSUN categories

POTTEDPLANT HORSE SOFA BUS CHURCHOUTDOOR BICYCLE TVMONITOR

StyleGANs



Before: Conditional BN

$$CBN(x_i, c) = w_{s,c} \frac{x_i - \mathrm{E_B}(x_i)}{\sqrt{\mathrm{Var_B}(x_i)}} + w_{b,c}$$

Here: Adaptive Instance Normalization (AdaIN)

$$\text{AdaIn}(x_i, w) = w_{s,i} \frac{x_i - \mu(x_i)}{\sigma(x_i)} + w_{b,i}$$

Karras et al. A Style-Based Generator Architecture for Generative Adversarial Networks. 2019.

StyleGANs

Mixing styles

“Two sets of images were generated from their respective latent codes (sources A and B); the rest of the images were generated by copying a specified subset of styles from source B and taking the rest from source A.”

Mixing styles

Mixing styles

StyleGAN: Bedrooms

StyleGAN: Cars

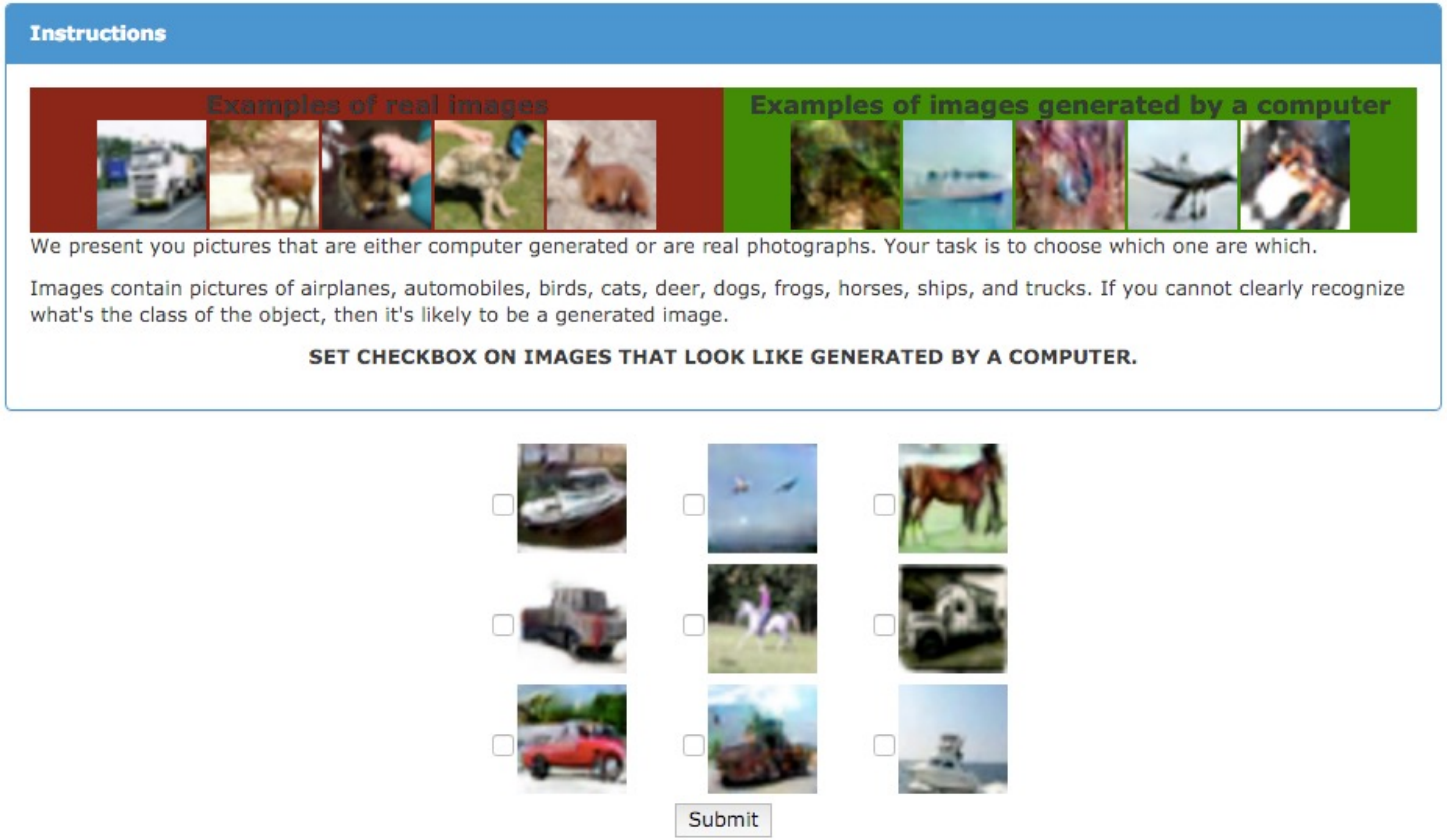
Evaluating GANs

How to evaluate GANs?

- Showing pictures of samples is not enough, especially for simpler datasets like MNIST, CIFAR, faces, bedrooms, etc.
- We cannot directly compute the likelihoods of high-dimensional samples (real or generated), or compare their distributions
- Many GAN approaches claim mainly to improve stability, which is hard to evaluate

GAN evaluation: Human studies

- Example: Turing test



GAN evaluation: Inception score (IS)

- Key idea: generators should produce images with a variety of recognizable object classes
- Defined as $IS(G) = \exp[\mathbb{E}_{x\sim G}\, KL(P(y|x) \parallel P(y))]$ where $P(y|x)$ is the posterior label distribution returned by an image classifier (e.g., InceptionNet) for sample x
  - If x contains a recognizable object, entropy of $P(y|x)$ should be low
  - If generator generates images of diverse objects, the marginal distribution $P(y)$ should have high entropy

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training GANs, NIPS 2016

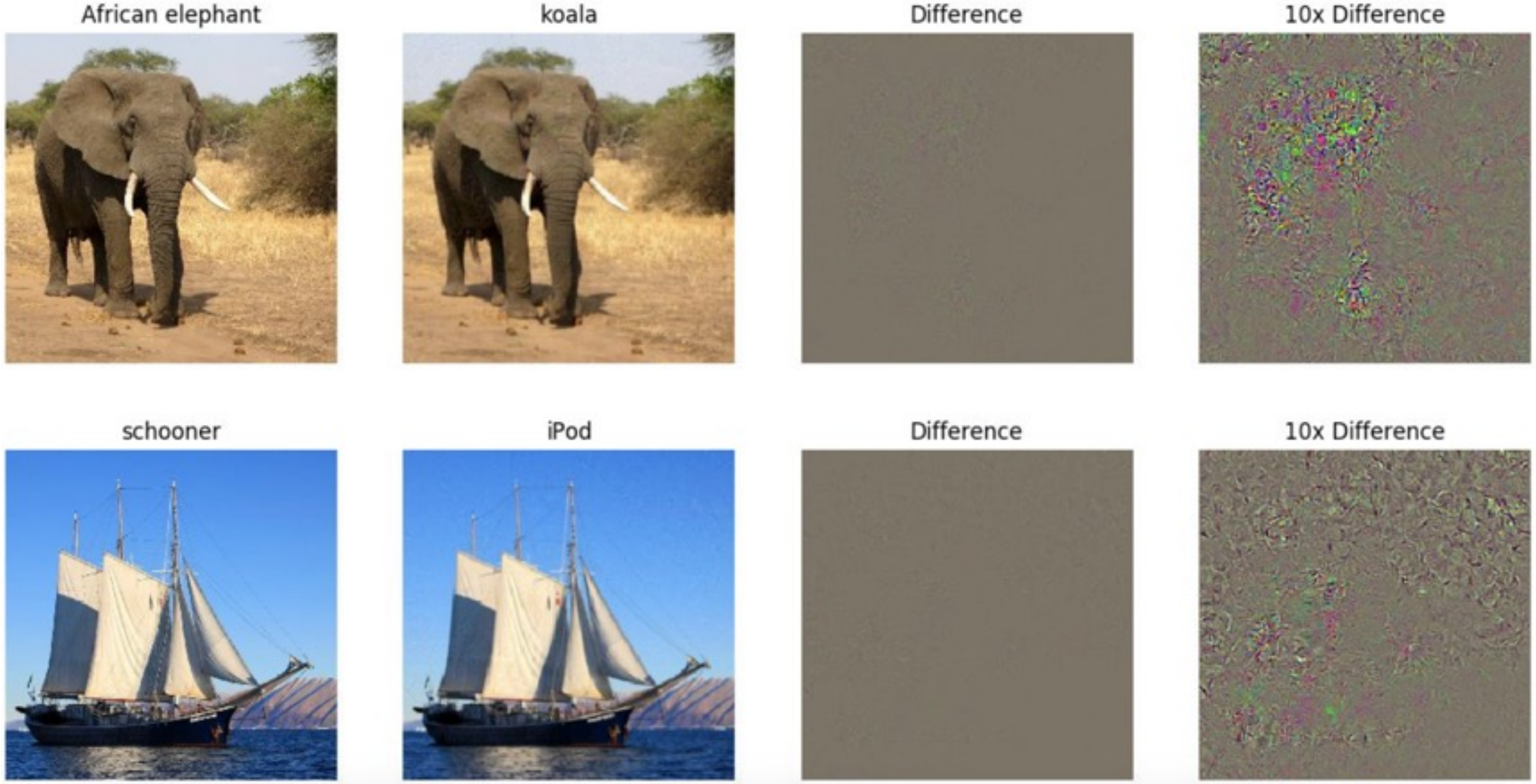
GAN evaluation: Inception score (IS)

- Disadvantages
  - A GAN that simply memorizes the training data (overfitting) or outputs a single image per class (mode dropping) could still score well
  - Is sensitive to network weights, not necessarily valid for generative models not trained on ImageNet, can be gamed (Barratt & Sharma 2018)

Adversarial Examples

Adversarial examples

- We can “fool” a neural network by imperceptibly perturbing an input image so it is misclassified



Finding the smallest adversarial perturbation

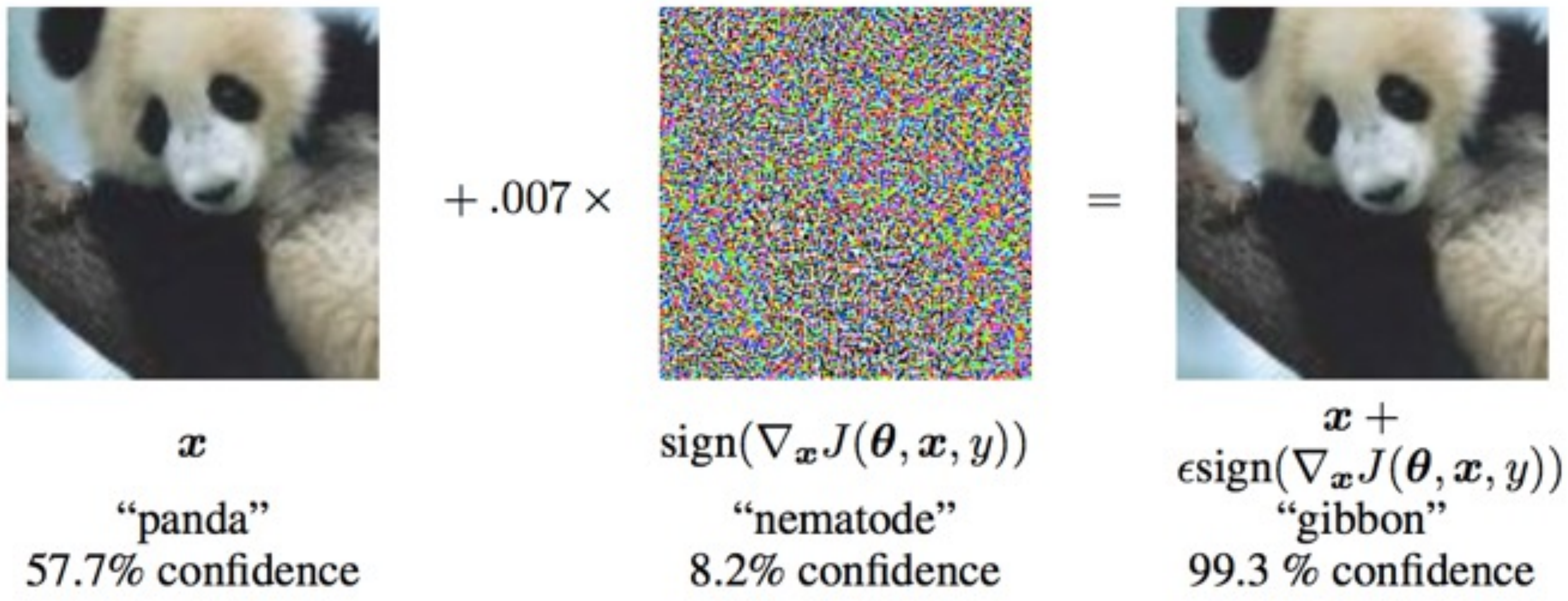
Input | Perturbation x 10 | "Ostrich" | Input | Perturbation x 10 | "Ostrich"

Szegedy, et al., 2014

Generating adversarial examples

- **Fast gradient sign method:** Find the gradient of the loss w.r.t. correct class y^*, take element-wise sign, update in resulting direction:

$$x \leftarrow x + \epsilon \, \mathrm{sgn}\left(\frac{\partial L(x, y^*)}{\partial x}\right)$$



Goodfellow et al., 2015

Defending against adversarial examples

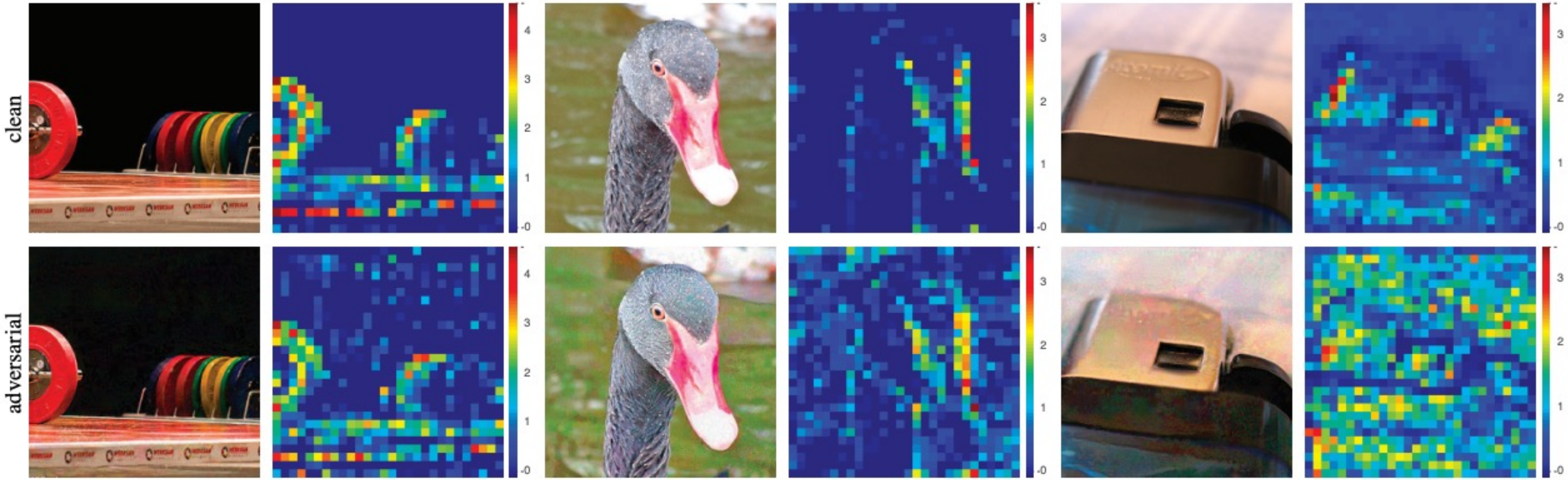


Figure 2. More examples similar to Figure 1. We show feature maps corresponding to clean images (top) and to their adversarial perturbed versions (bottom). The feature maps for each pair of examples are from the same channel of a res_3 block in the same ResNet-50 trained on clean images. The attacker has a maximum perturbation $\epsilon = 16$ in the pixel domain.

C. Xie et al., Feature Denoising for Improving Adversarial Robustness, CVPR 2018

Defending against adversarial examples

- Training with adversarial examples improves the network robustness against adversarial examples

- It does not improve the performance on natural images

Summary

- Generative Adversarial Networks, DCGAN
- Progressive GAN, StyleGAN
- Evaluating GANs
- Adversarial Examples