CS409 : Neural Networks (Semester II - 2021/22)

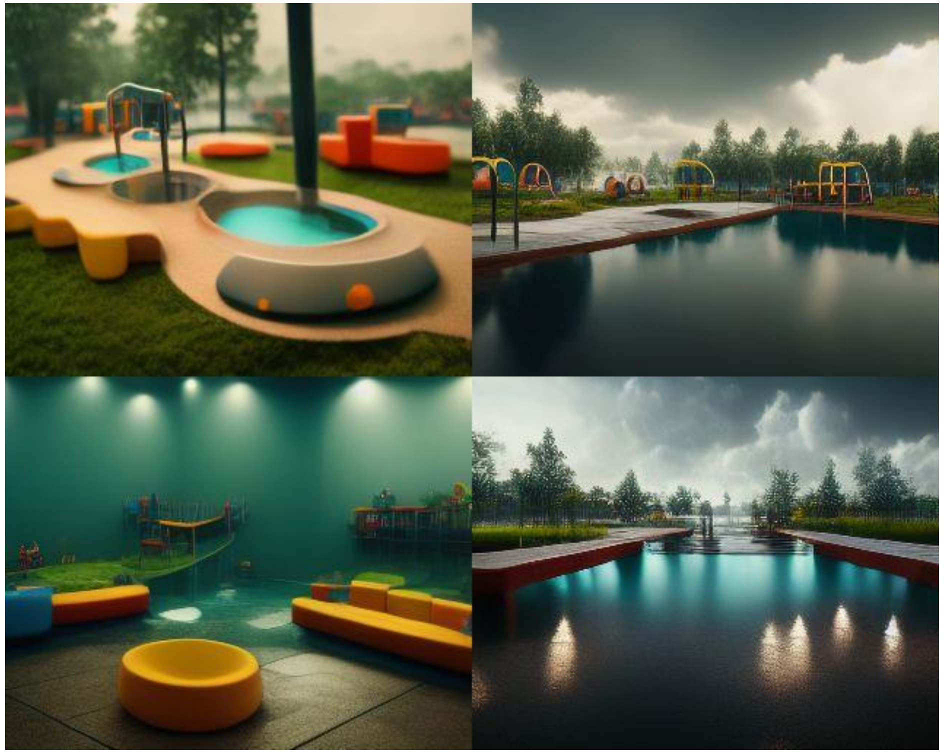
Unit 8: Generative Adversarial Networks (GANs)

Dr. Ruwan Nawarathna
Department of Statistics & Computer Science
Faculty of Science
University of Peradeniya

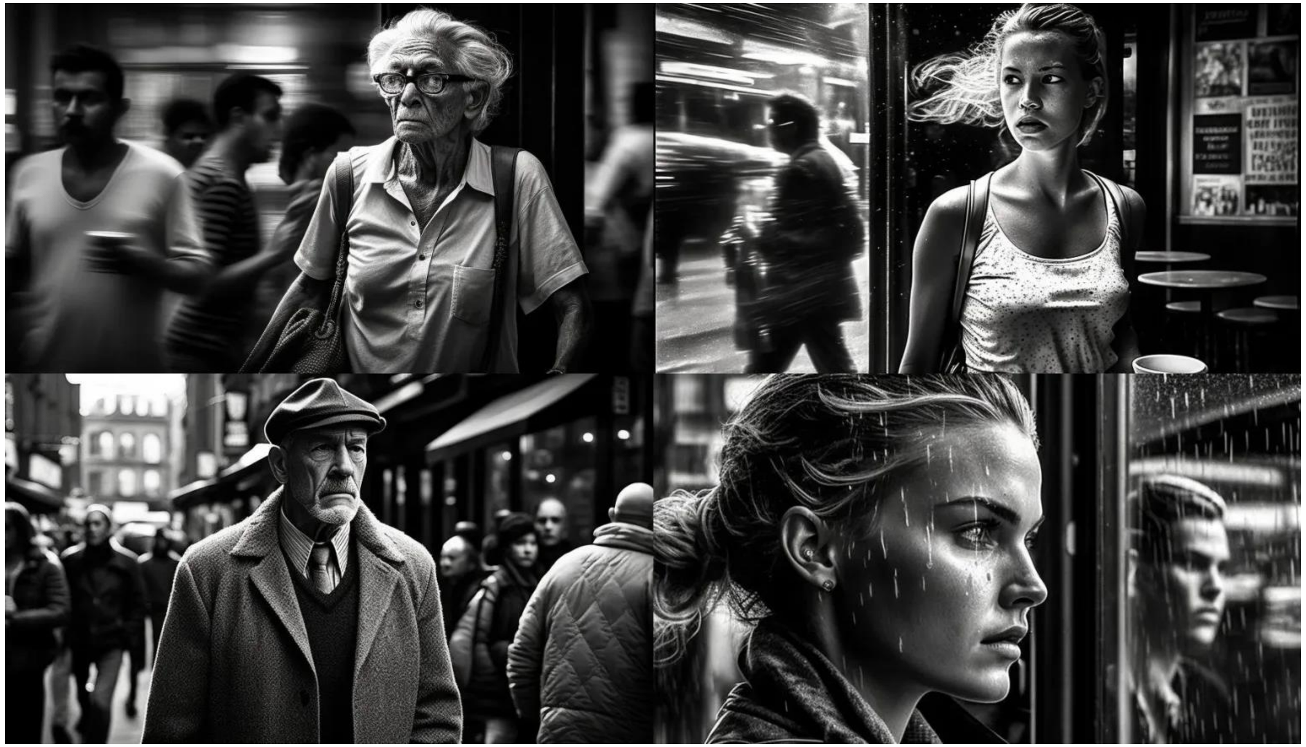
Generative Adversarial Networks (GANs)

- The original purpose is to generate new data
- Classically for generating new images, but applicable to wide range of domains
- Learns the training set distribution and can generate new images that have never been seen before

Midjourney Prompts



modern kids play area landscape architecture, water play area, floating kids, seating areas, perspective view, rainy weather, biopunk, cinematic photo, highly detailed, cinematic lighting, ultra-detailed, ultrarealistic, photorealism, 8k, octane render, --ar 16:12

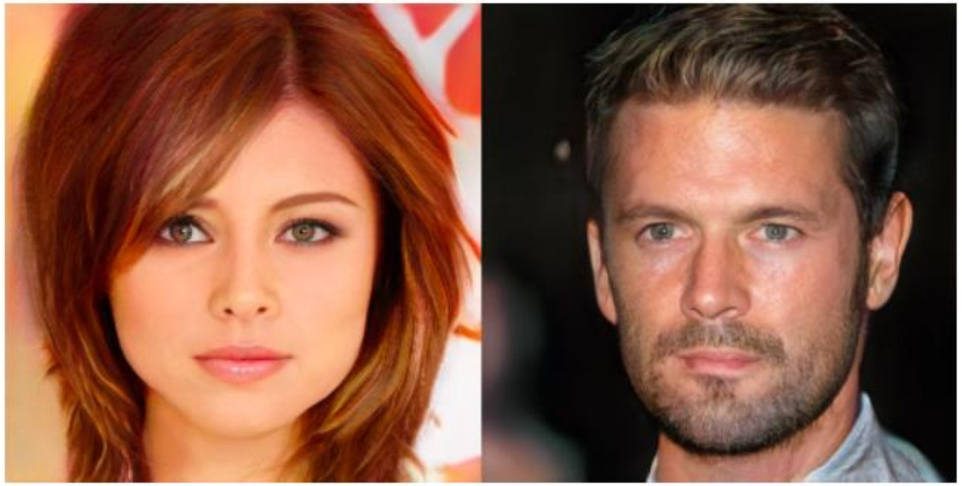


“Candid moments,” “Urban landscapes,” “Street life,” “Stories in motion,” “Street portraits.”

Generative Adversarial Networks (GANs)

- Given training data, generate new samples from same distribution.

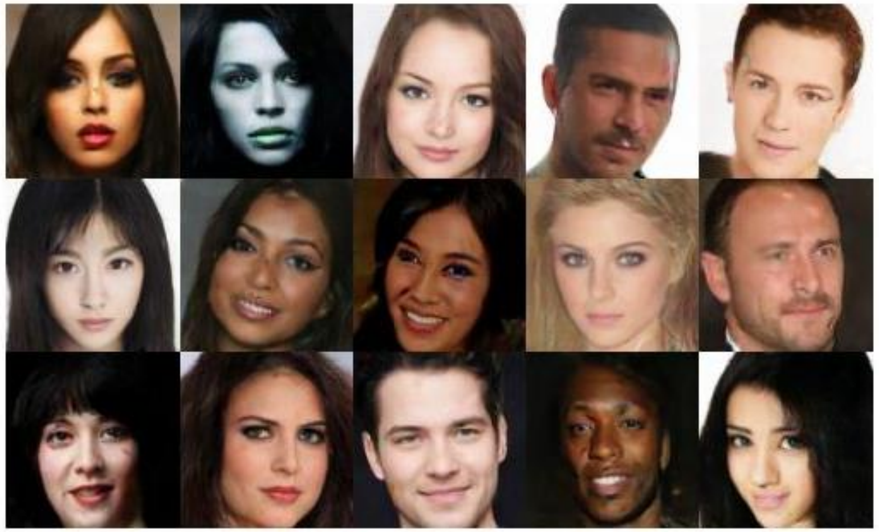
Training Data Sample Generator
(CelebA)



Generated celebrity images

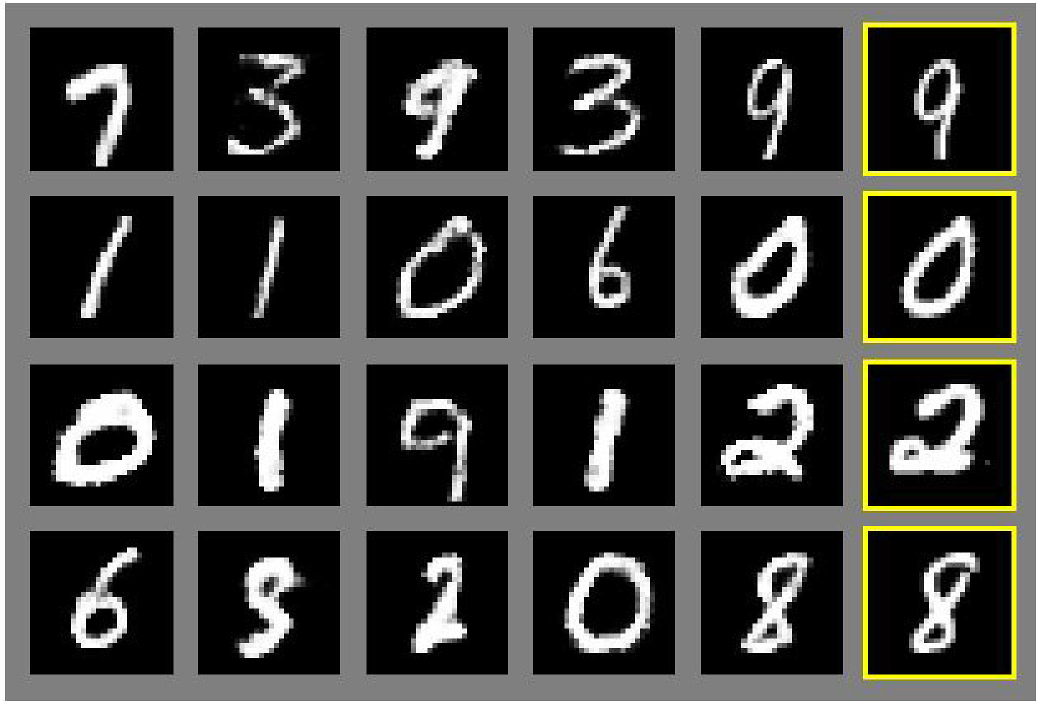
Generative Adversarial Networks (GANs)

- More Results

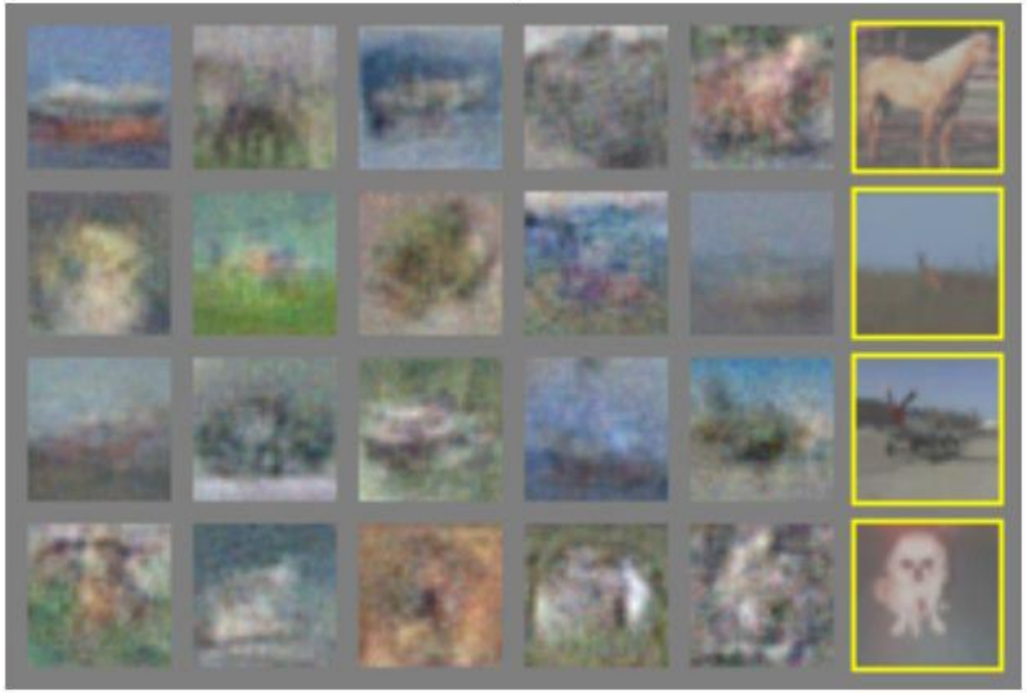


Generative Adversarial Networks (GANs)

- More Results

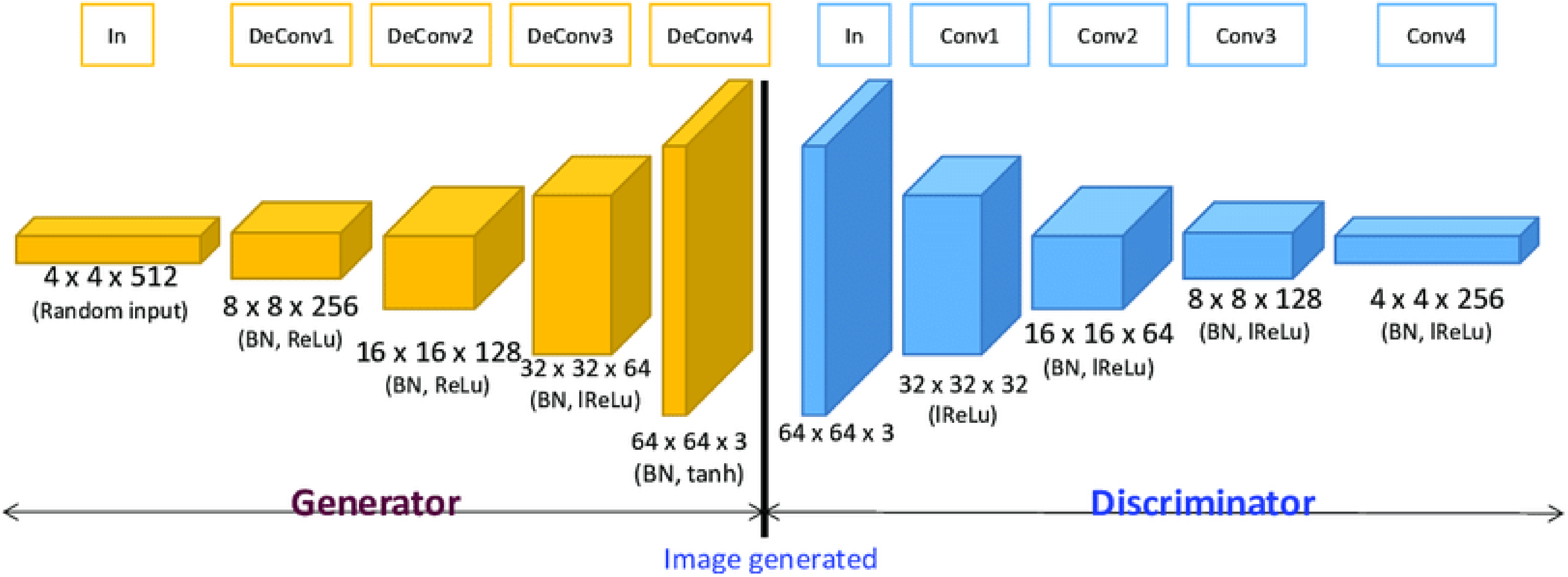


MNIST Digits



CIFAR-10

Example GAN – DCGAN

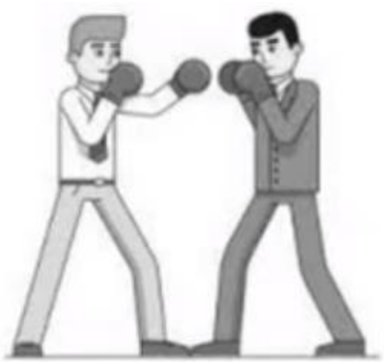


Generative Adversarial Networks

Generative

Generate data
(Creates fake data)

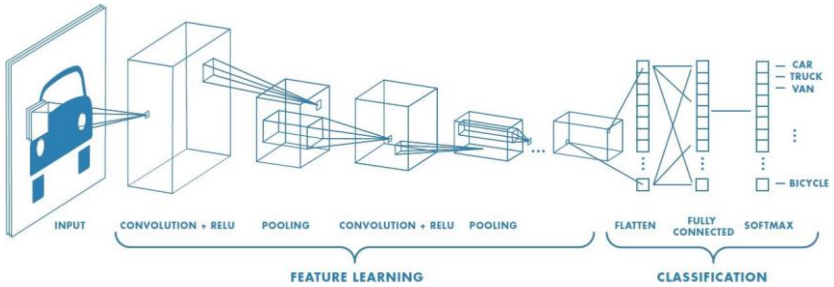
Adversarial

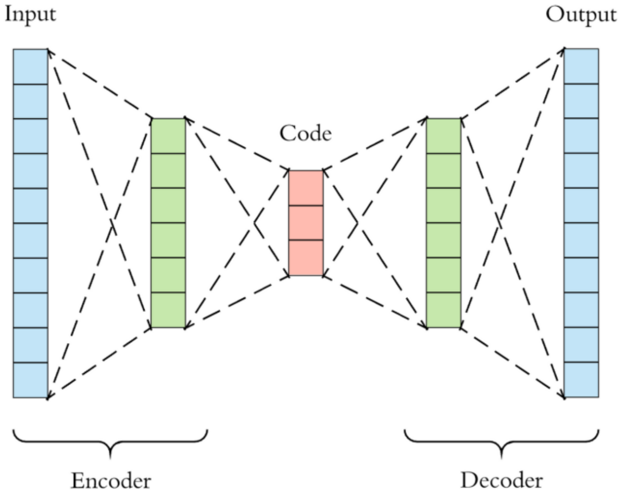


Generator and discriminator, each competing to win

Generator trying to fake and discriminator, trying not to be fooled

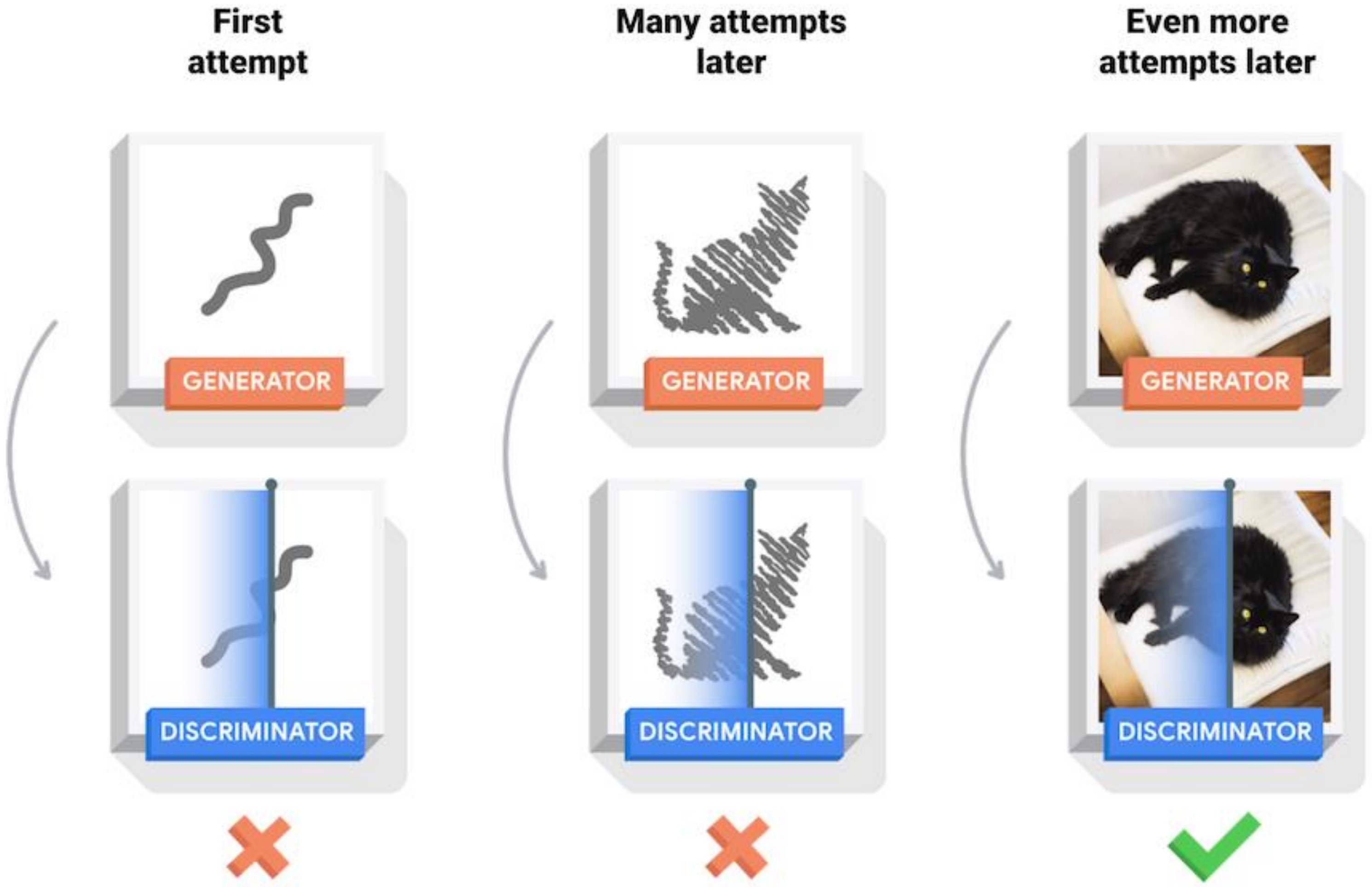
Networks



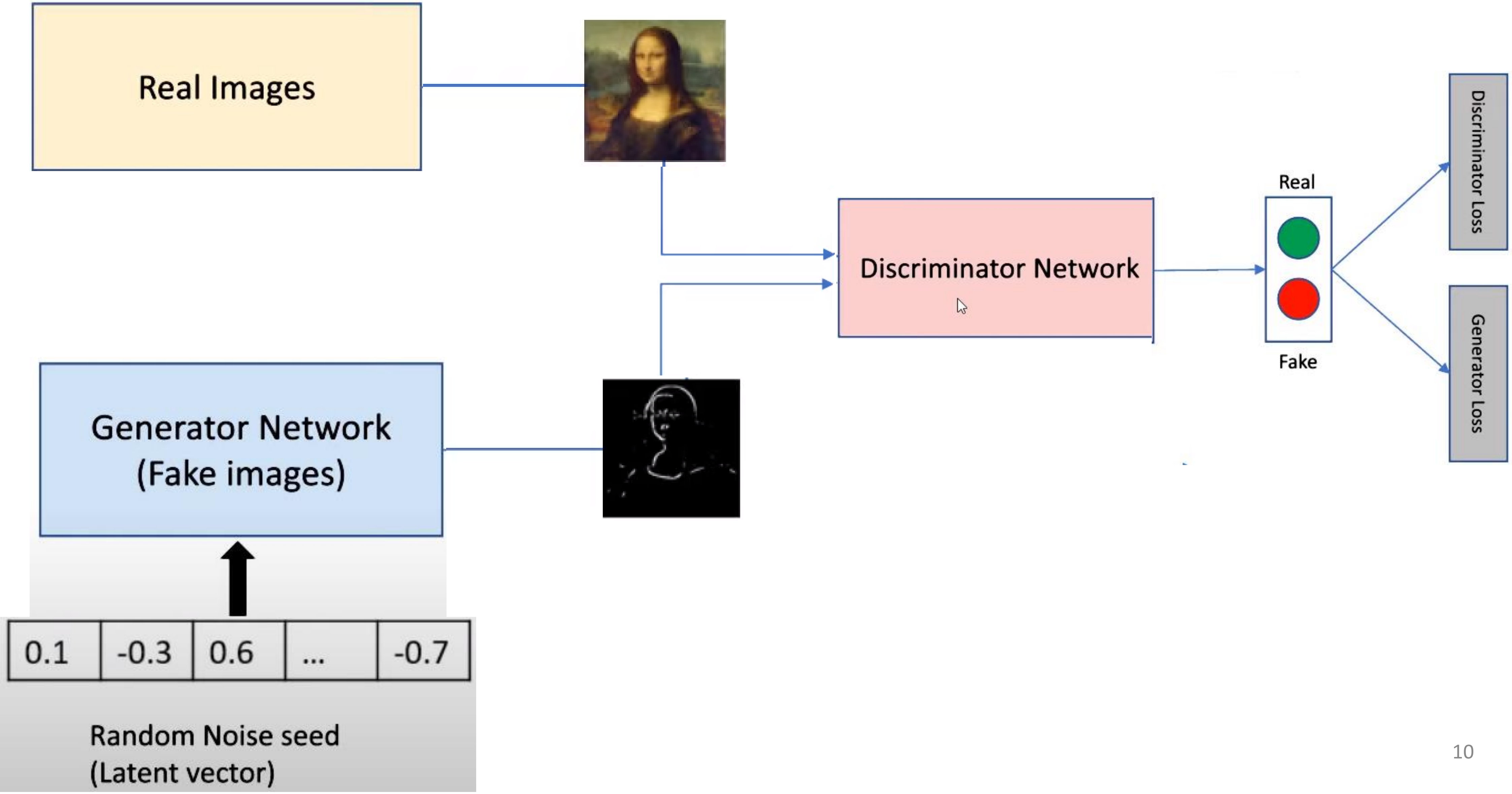


GANs take game-theoretic approach: learn to generate from training distribution through 2-player game

GAN Architecture



GAN Architecture



Real Images

Discriminator Network

Real

Fake

Discriminator Loss

Generator Loss

Generator Network
(Fake images)

0.1	-0.3	0.6	...	-0.7

Random Noise seed
(Latent vector)

Generator and Discriminator

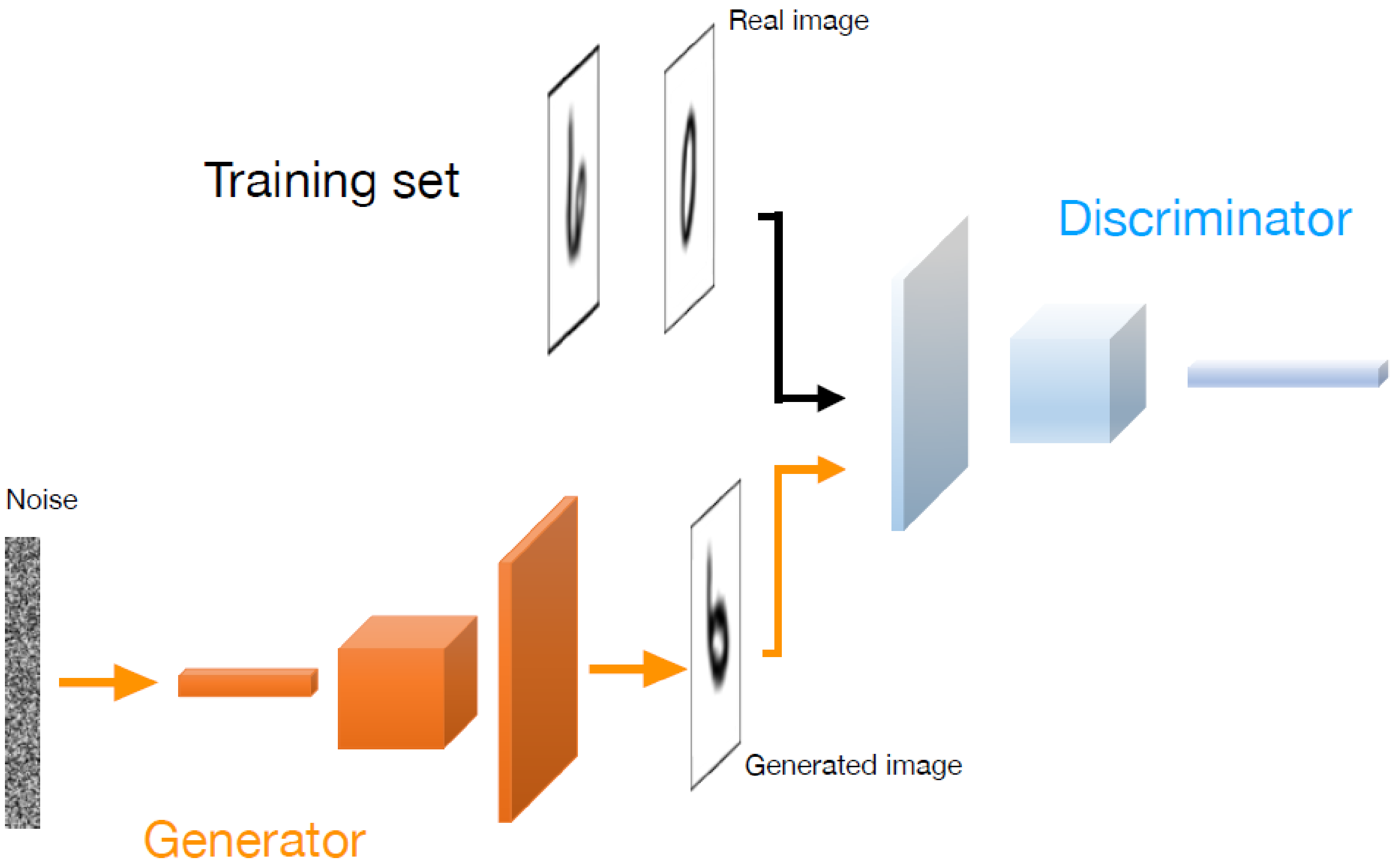
- Generator network: tries to fool the discriminator by generating real-looking images

- Discriminator network: tries to distinguish between real and fake images

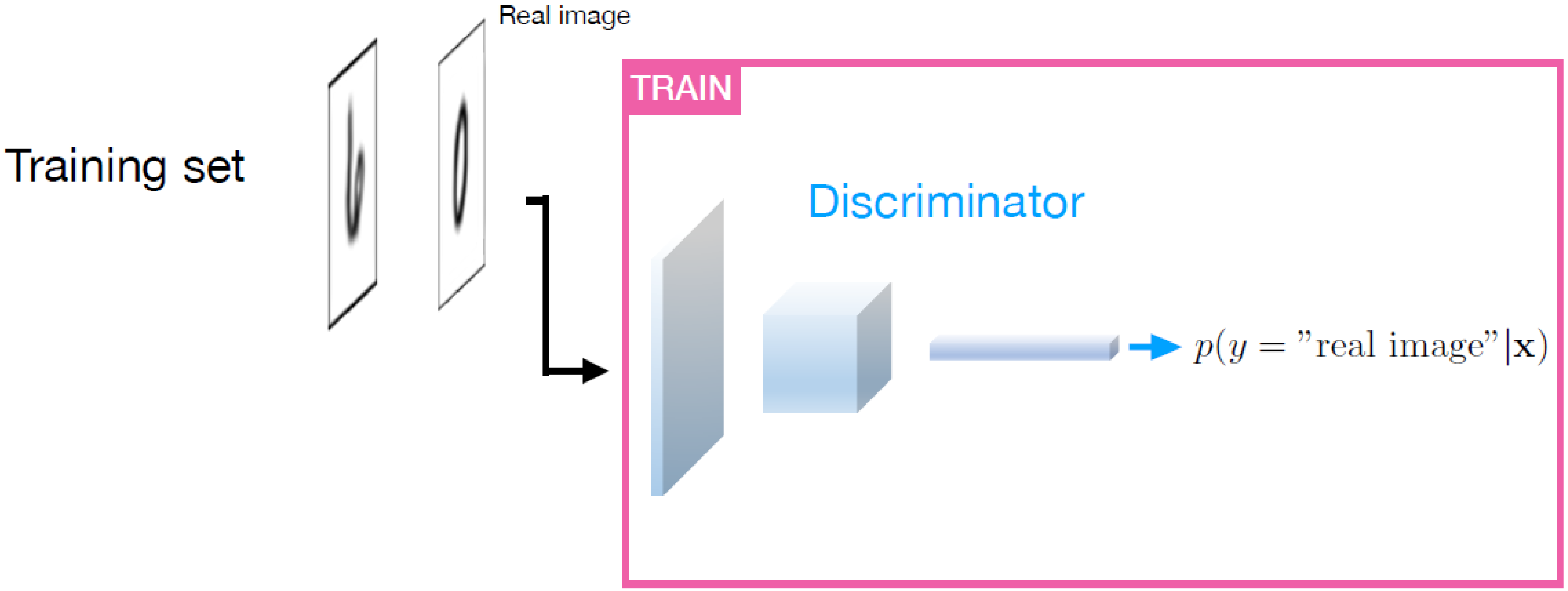
Generator and Discriminator

- The generator and discriminator are adversaries in a game
- The generator controls only its parameters
- The discriminator controls only its parameters
- Each seeks to maximize its own success and minimize the success of the other: related to minimax theory

Training GANs



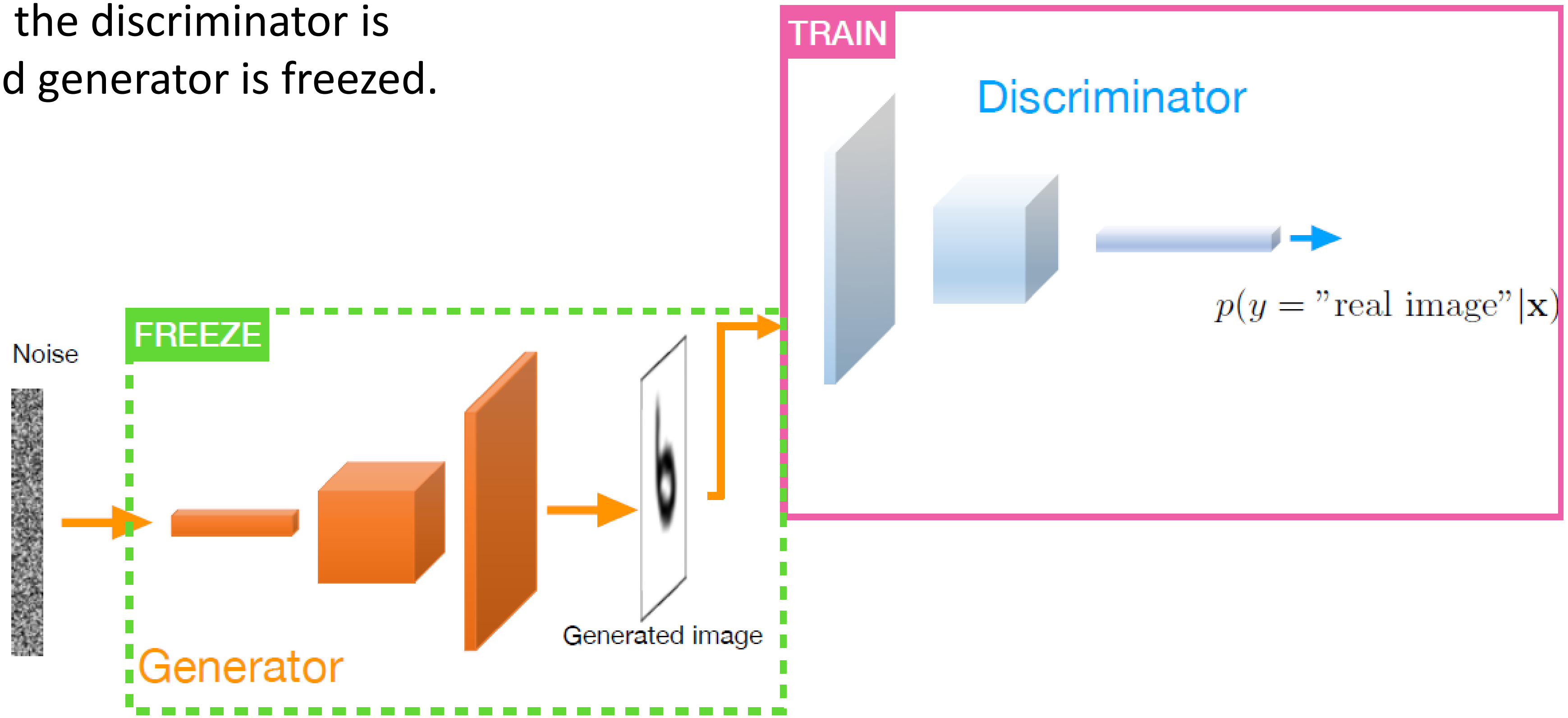
Training GANs



Train to predict that real image is real.

Training GANs

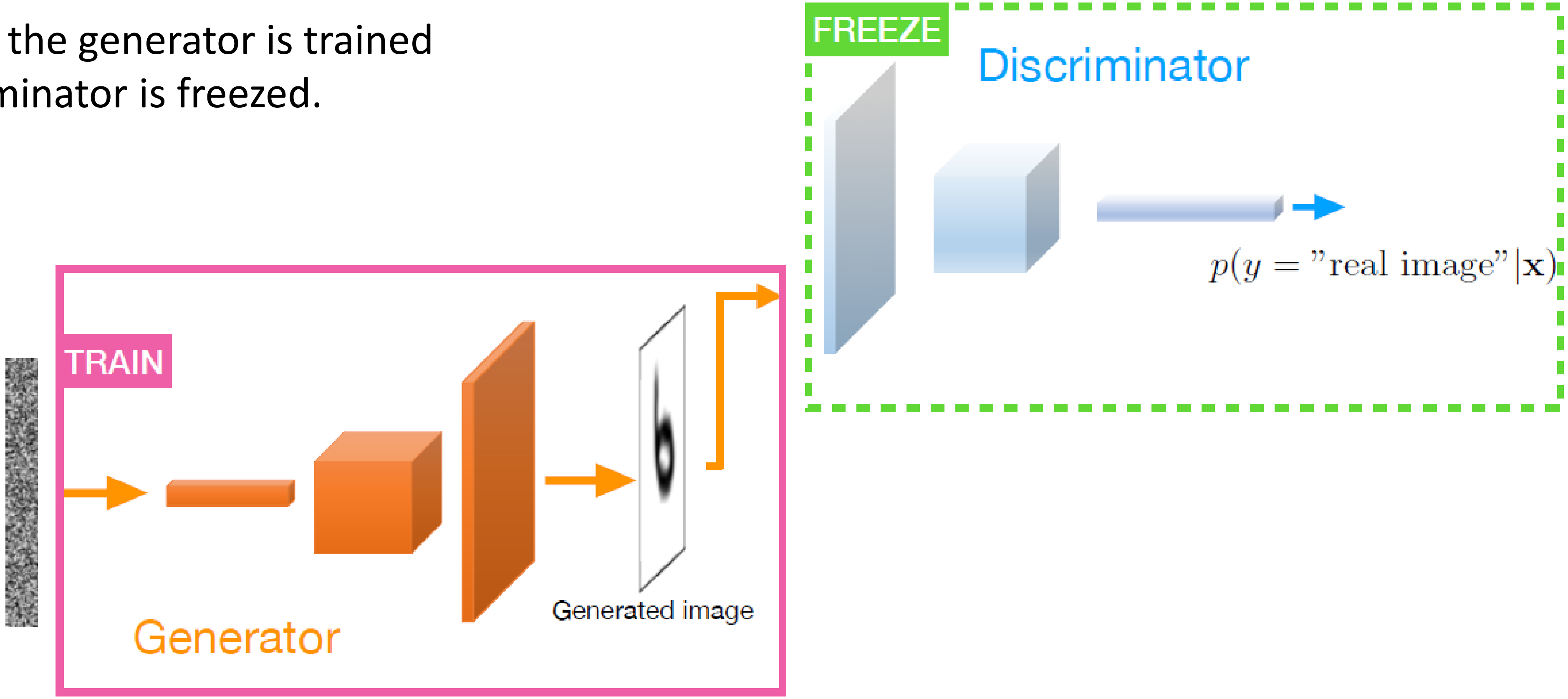
When the discriminator is trained generator is freezed.



Train to predict that fake image is fake.

Training GANs

When the generator is trained
discriminator is freezed.



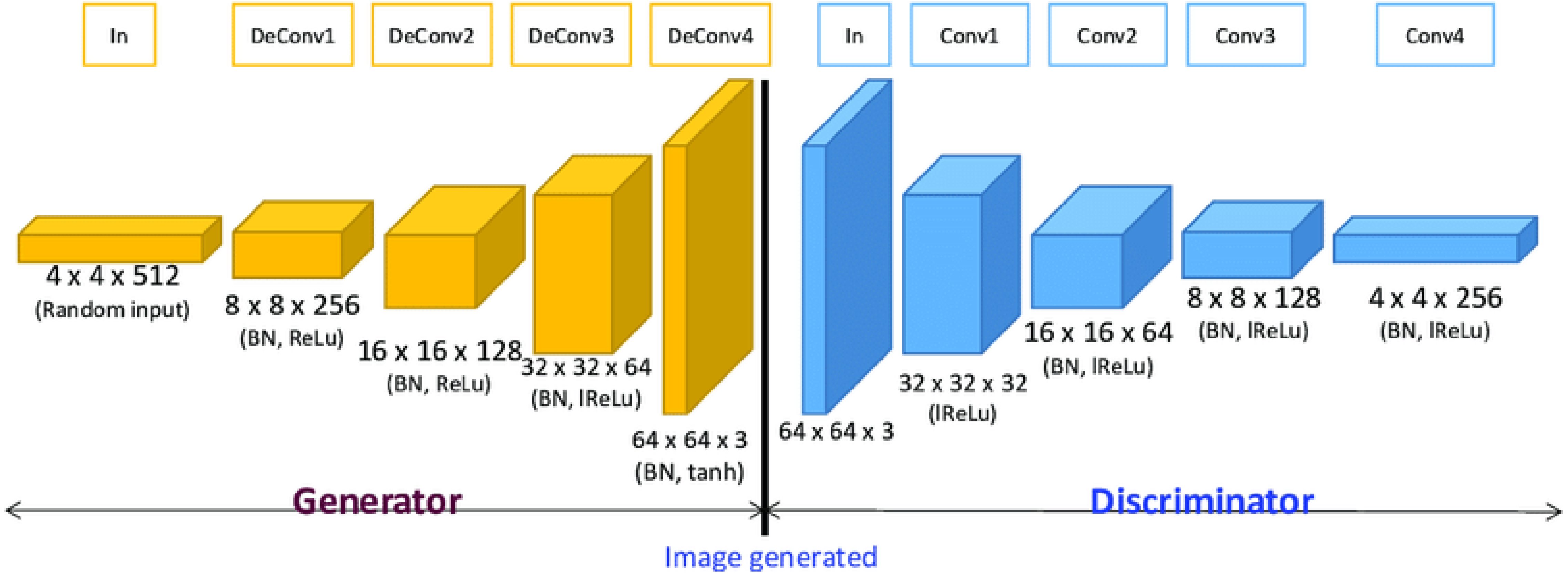
Train to predict that fake image is real.

Deep Convolutional GANs (DCGANs)

- DCGAN is one of the popular and successful network design for GAN. It mainly composes of convolution layers without max pooling or fully connected layers.

- It uses convolutional stride and transposed convolution for the downsampling and the upsampling.

DCGANs



In
DeConv1
DeConv2
DeConv3
DeConv4
In
Conv1
Conv2
Conv3
Conv4
4 x 4 x 512
(Random input)
8 x 8 x 256
(BN, ReLu)
16 x 16 x 128
(BN, ReLu)
32 x 32 x 64
(BN, lReLu)
64 x 64 x 3
(BN, tanh)
64 x 64 x 3
32 x 32 x 32
(lReLu)
16 x 16 x 64
(BN, lReLu)
8 x 8 x 128
(BN, lReLu)
4 x 4 x 256
(BN, lReLu)
Generator
Discriminator
Image generated

DCGAN - Generator and Discriminator

- Generator is an upsampling network with fractionally-strided convolutions

- Discriminator is a convolutional network

Deep Convolutional GANs (DCGANs)

- DCGAN, uses a couple of guidelines, in particular:
    - Replacing any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
    - Using batchnorm in both the generator and the discriminator.
    - Removing fully connected hidden layers for deeper architectures.
    - Using ReLU activation in generator for all layers except for the output, which uses tanh.
    - Using LeakyReLU activation in the discriminator for all layer.

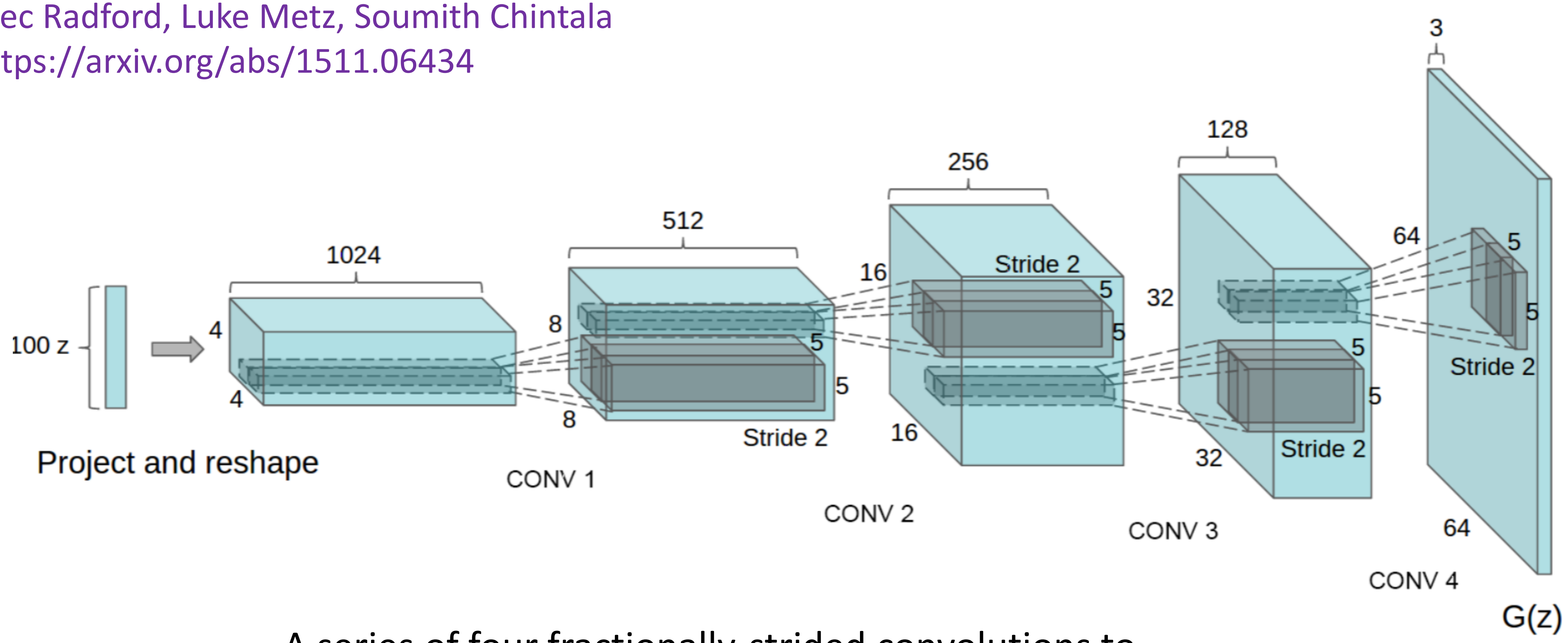
DCGAN - Example

Generator proposed in the original DCGAN paper for LSUN dataset

Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks

Alec Radford, Luke Metz, Soumith Chintala

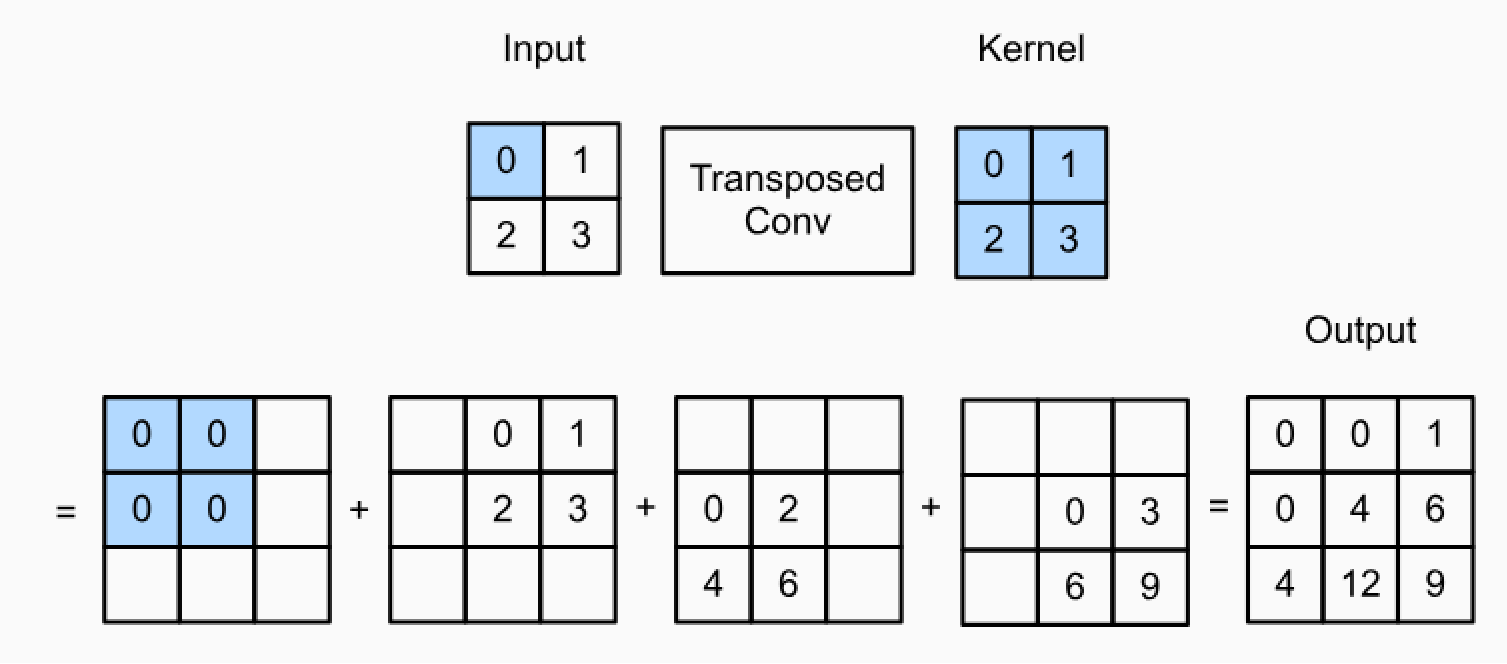
https://arxiv.org/abs/1511.06434



A series of four fractionally-strided convolutions to convert the input vector to 64x64 pixel image.

Fractionally Strided (Transposed) Convolutions

- A transposed convolutional layer is an upsampling layer that generates the output feature map greater than the input feature map.
- **Example1** - Transposed convolution with a 2x2 kernel is computed for a 2x2 input.

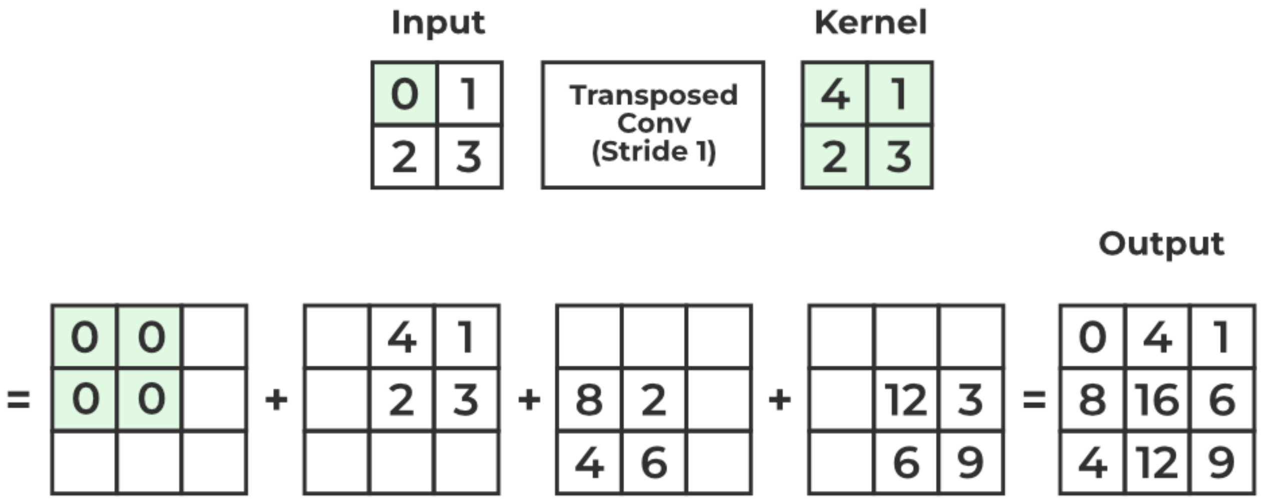


Fractionally Strided (Transposed) Convolutions

- Instead of sliding the kernel over the input and performing element-wise multiplication and summation, a transposed convolutional layer slides the input over the kernel and performs element-wise multiplication and summation.

- This results in an output that is larger than the input, and the size of the output can be controlled by the stride and padding parameters of the layer.

Example 2



Types of GANs

- Since GANs were introduced in 2014, there have been hundreds of papers introducing various architectures and training methods.
- Most modern architectures are based on the Deep Convolutional GAN(DC-GAN), where the generator and discriminator are both conv nets.
- GAN Zoo: https://github.com/hindupuravinash/the-gan-zoo

Types of GANs

- Vanilla GAN
- Deep convolutional GAN (DCGAN)
- Conditional GAN (cGAN)
- CycleGAN
- StyleGAN
- Super resolution GAN (SRGAN)
- DiscoGAN
- PixelRNN
- Text-to-image
- Pix2Pix