Generative Adversarial Networks (GANs)

Semester 2, 2022

Kris Ehinger and Tom Drummond

Demo

- https://thispersondoesnotexist.com/

Outline

- GAN architecture
- Evaluating GANs
- Conditional GANs

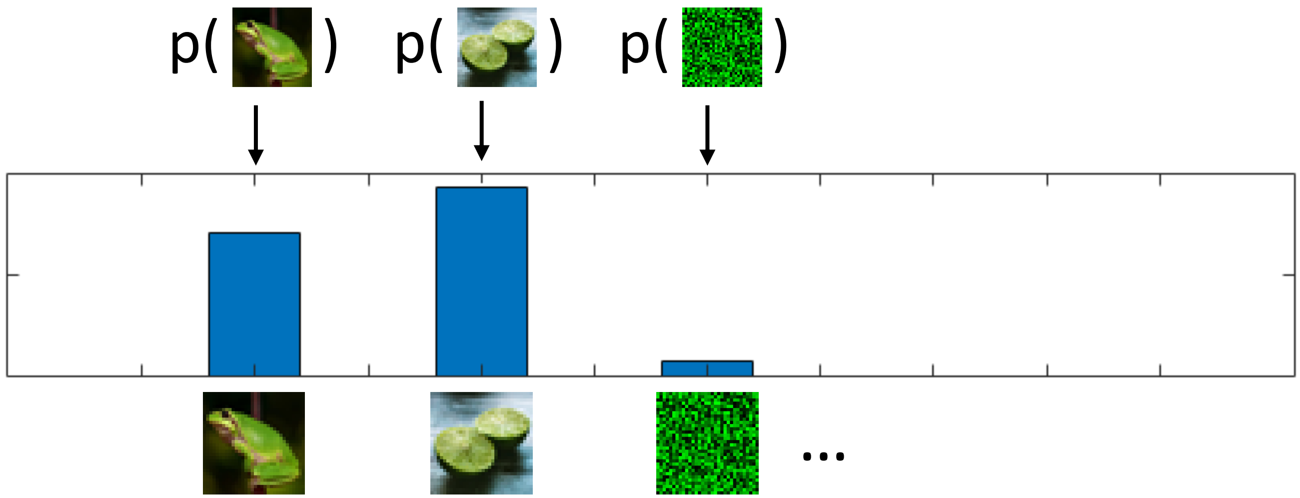
Learning Outcomes

- Explain the architecture and objective function of a standard GAN
- Explain common pitfalls in GANs, and how GANs can be evaluated
- Explain common architectures for conditional GANs

GAN architecture

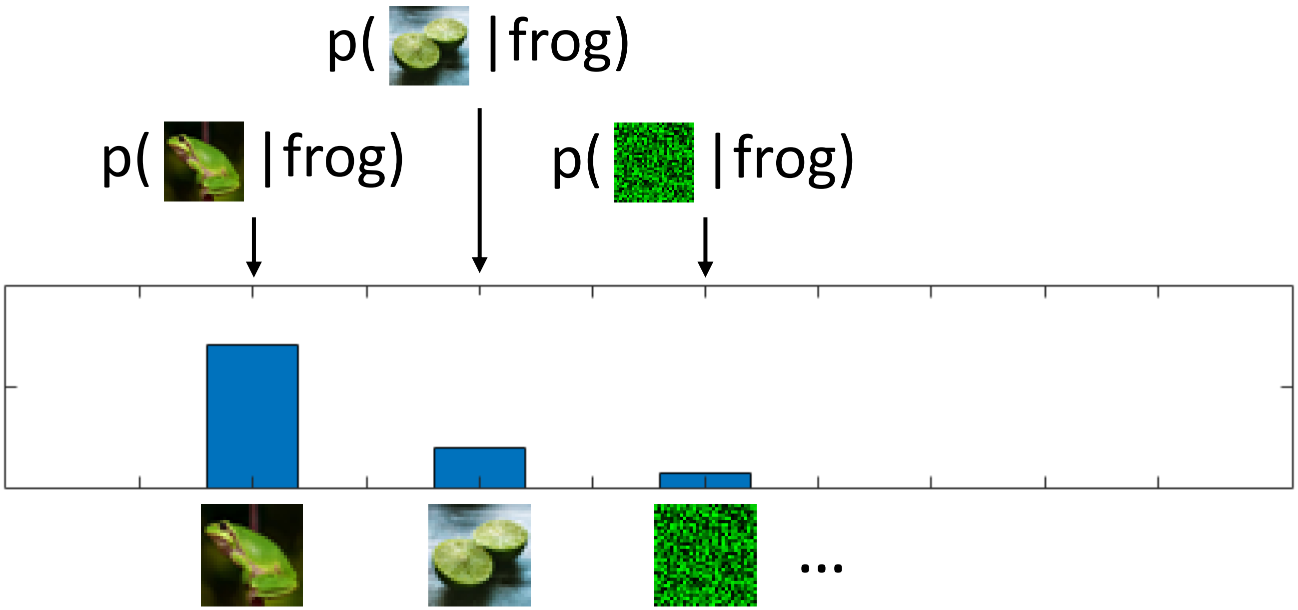
(Unconditional) generative model

- Output is a probability distribution p(x)
- What is the probability that this is an image?



(Conditional) generative model

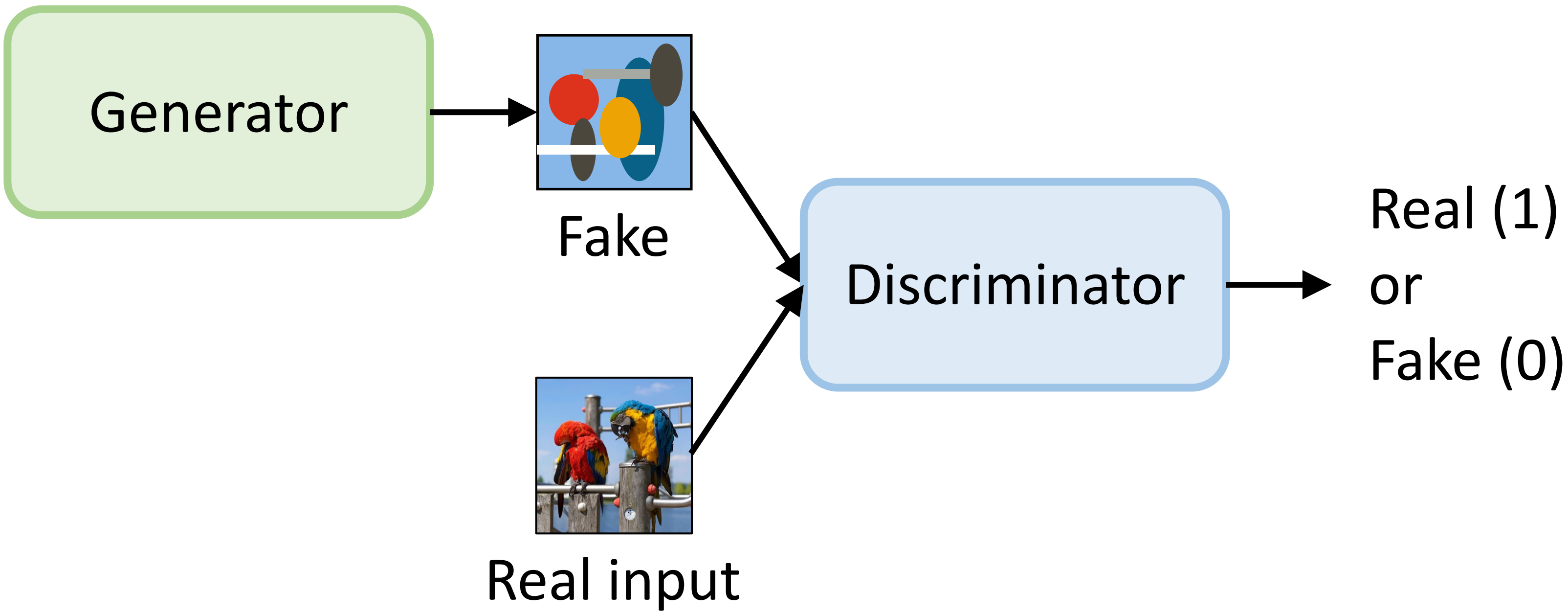
- Input is a label
- Output is a probability density function over images p(x|y)



GANs

- **Generative Adversarial Networks** (GANs) are neural networks that learn to generate instances from a particular distribution (e.g., images of faces)
- Actually consist of two neural networks: a **generator** and a **discriminator**
- Training involves a sort of competition between the two networks

GAN architecture

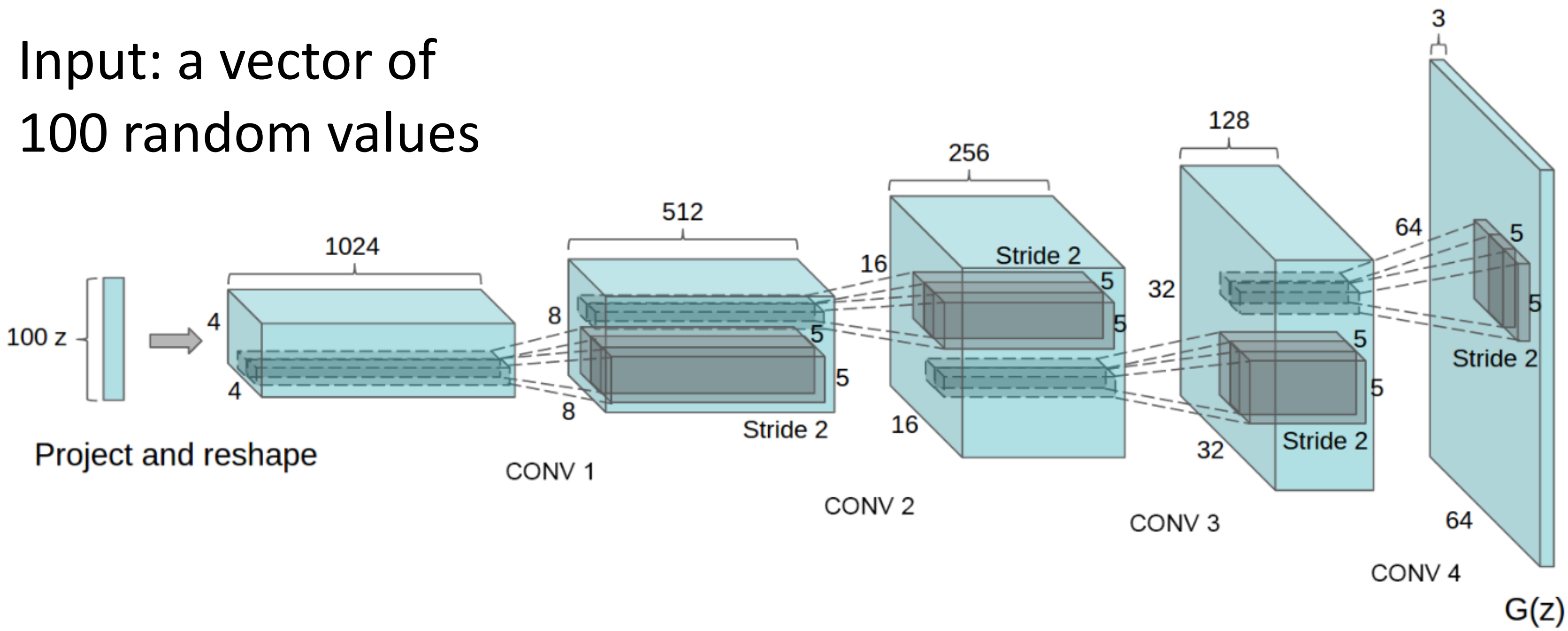


Generator

- GAN Generator doesn't actually learn the probability distribution p(x), but learns to sample from it
- Generator input is a latent variable z with a simple prior (e.g., uniform random or standard normal)
- Generator output is an image
- Generator network learns a function to map p(z) to a distribution that approximates p(x)

Generator architecture example

Input: a vector of 100 random values



Output: 64 x 64 pixel colour image

Image: Radford, Metz, & Chintala (2016)

Discriminator

- Discriminator learns to identify real inputs vs. fake inputs created by generator
- Neural network classifier with two output classes (real, fake)
- Architecture depends on task: e.g., for images the discriminator might be a CNN with several convolutional layers, followed by softmax

Training

- The networks are trained together on a combination of real data $\mathbf{x}$ and generator input $\mathbf{z}$
- Given a generator G and discriminator D:
  - Discriminator's goal is to correctly classify real vs. fake
  - Discriminator wants to maximize $D(\mathbf{x})$ and minimize $D\big(G(\mathbf{z})\big)$
  - Generator's goal is to fool the Discriminator
  - Generator wants to maximize $D\big(G(\mathbf{z})\big)$
- Can treat this as a zero-sum game with the goal of finding equilibrium between G and D

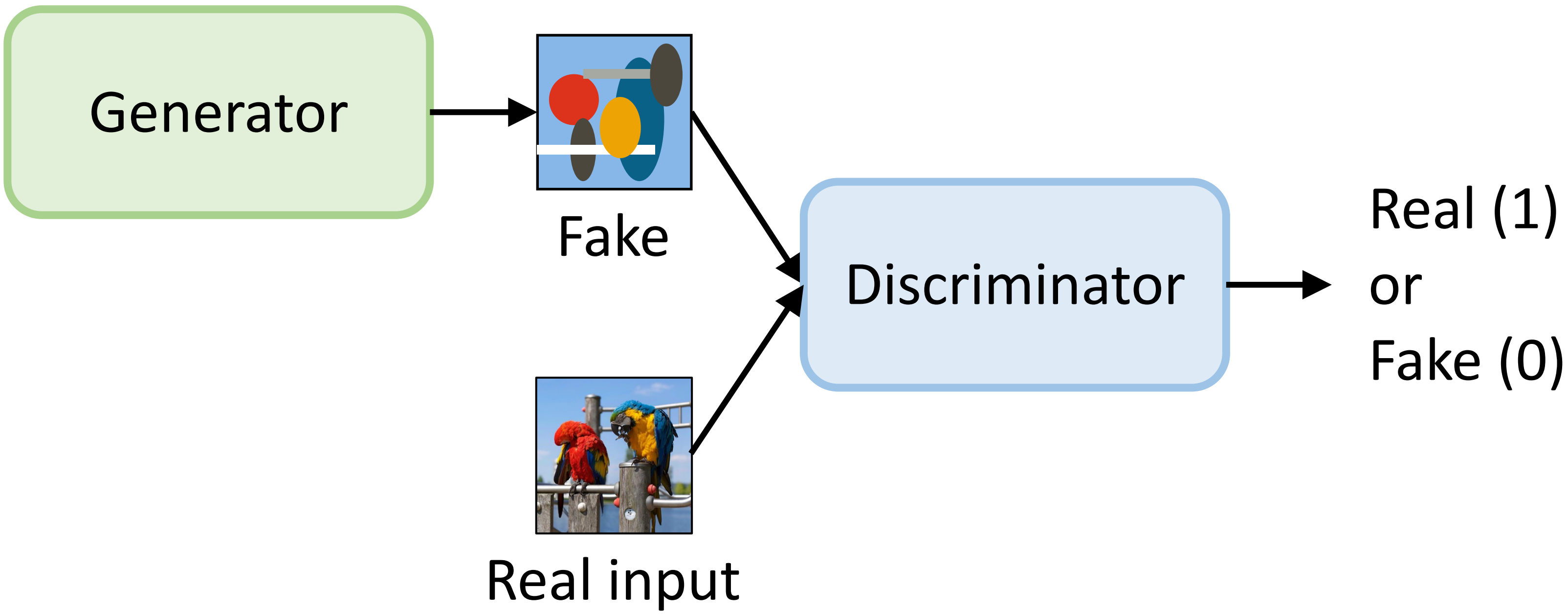
Training objective

- GAN training objective is a minimax game:

$$\min_{G} \max_{D} \Big(\underbrace{\underbrace{E_{x \sim p_{data}} [\log(D(x))]}_{\text{Discriminator's response to real images}} + \underbrace{E_{z \sim p(z)} [\log(1 - D(G(z)))]}_{\text{1 - Discriminator's response to fake images produced by Generator}}}_{\text{The Discriminator tries to maximize this by learning weights for D that will give D(x)=1 for real images and D(G(z))=0 for fake images}} \Big)$$

The Generator tries to minimize this by learning weights that will give D(G(z)) = 1

Training



Training

- If the discriminator is too good:
  - Easily rejects all fake inputs
  - Not much information to train the generator
- If the discriminator is too bad:
  - Easily confused by fake inputs that don’t look real
  - Generator will learn a poor solution
- Training can be difficult – hard to find a balance between discriminator and generator

Demo

- https://poloclub.github.io/ganlab/

Summary

- GAN is a pair of networks trained together: generator creates images based on latent input z, discriminator judges whether images are real vs. fake
- Objective function is a competition in which generator tries to fool discriminator
- Generator doesn't learn p(x) (distribution of real images) but does learn to sample this distribution

Evaluating GANs

GAN evaluation

- GAN equilibrium does not necessarily mean the GAN has found a good solution
- How to tell if a GAN has learned? Ideally:
  - Outputs should not be identical to inputs (memorised training data)
  - Outputs should look like inputs (look “real” and not “fake”)
  - Outputs should be as diverse as real data (avoid **mode collapse** = the generator only creates one or a few outputs)

Progressive Growing of GANs for Improved Quality, Stability, and Variation

Tero Karras
NVIDIA

Timo Aila
NVIDIA

Samuli Laine
NVIDIA

Jaakko Lehtinen
NVIDIA
Aalto University

NVIDIA

https://www.youtube.com/watch?v=G06dEcZ-QTg

Memorisation?

GAN output:

3 nearest neighbours in training dataset

Karras, Aila, Laine, & Lehtinen (2018)

Realism?

Karras, Aila, Laine, & Lehtinen (2018)

Realism?

Karras, Aila, Laine, & Lehtinen (2018)

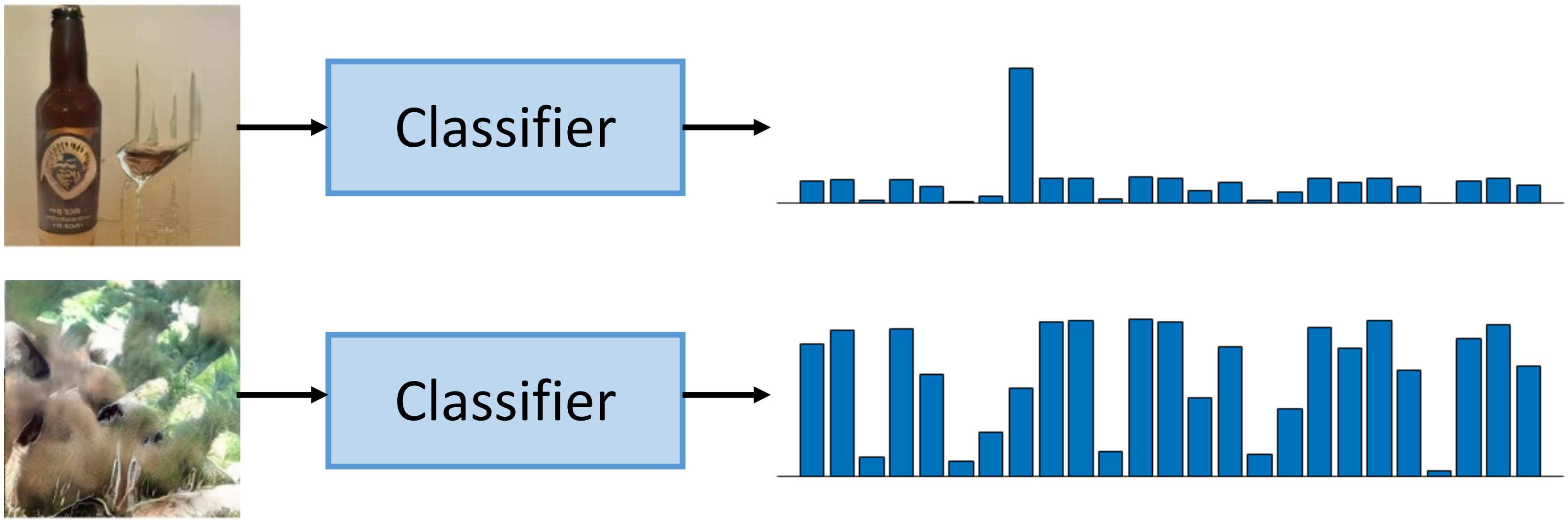
Realism?

Karras, Aila, Laine, & Lehtinen (2018)

Evaluating realism

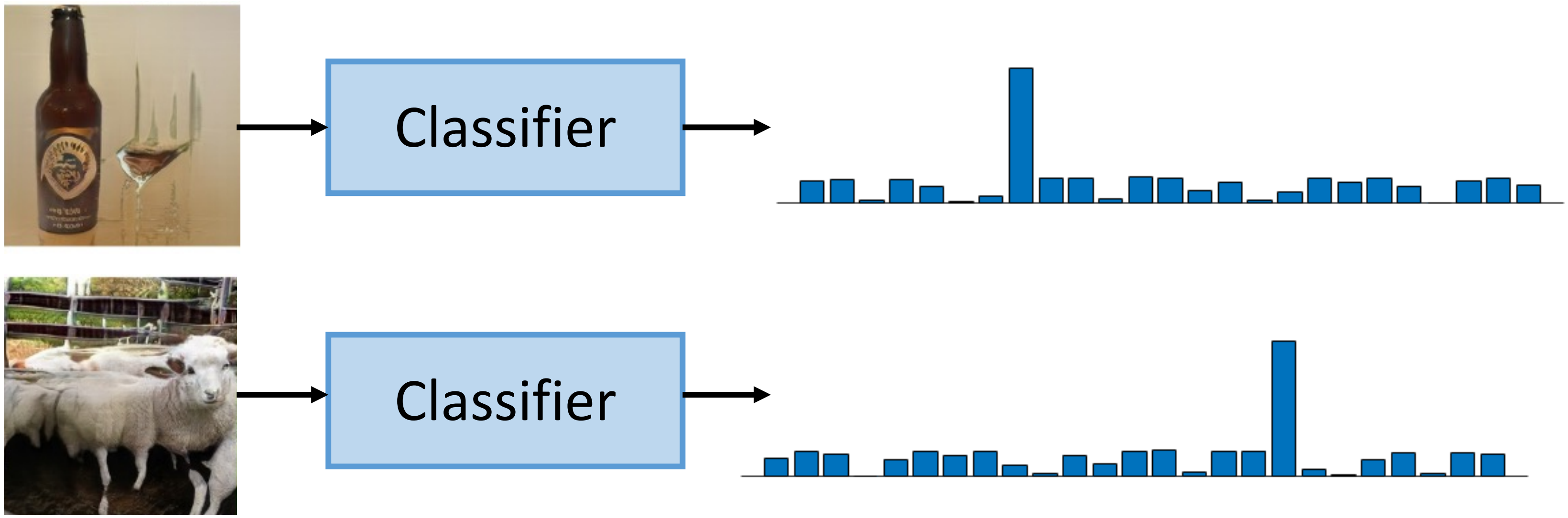
- How to evaluate realism?
- Gold standard: human evaluation (but this is slow and expensive)
- Automatic methods compare responses of an image classifier (e.g., a CNN trained on ImageNet) to real vs. GAN-generated images

Inception score



- Within a class, all images should be confidently classified with the correct label

Inception score



- Within a class, all images should be confidently classified with the correct label
- Across classes, the GAN should produce a wide variety of confidently-classified images

Inception score

- Advantages
  - Automatic, efficient
  - Neural network responses correlate with human judgements of image quality
- Disadvantages
  - Doesn’t require high diversity within categories
  - Sensitive to noise, adversarial images

Diversity?

- The GAN isn't just memorizing training examples
- But does it capture *all* of the diversity in the training set?
  - How would you measure this?

Birthday paradox

- What are the odds that someone else in this subject has the same birthday you do?
  - About 59% ($= 1 - (364/365)^{324}$)
- What are the odds that any two people in this subject share a birthday?
  - Close to 100%
- What's the smallest class size that has at least a 50/50 chance of two people sharing a birthday?
  - 23

Birthday paradox for GANs

- Arora, Risteski, & Zhang (2018)
- Suppose a generator that can produce N discrete outputs, all equally likely
- Experiment: take a small sample of s outputs and count duplicates
  - The odds of observing duplicates in a sample of size s can be used to compute N
  - A sample of about $\sqrt{N}$ outputs is likely to contain at least one pair of duplicates

Duplicates and diversity

- Example duplicates (and 1-NN in training dataset):

- Most GANs tested produced about the same diversity (number of different images) as was in their training set

Arora, Risteski, & Zhang (2018)

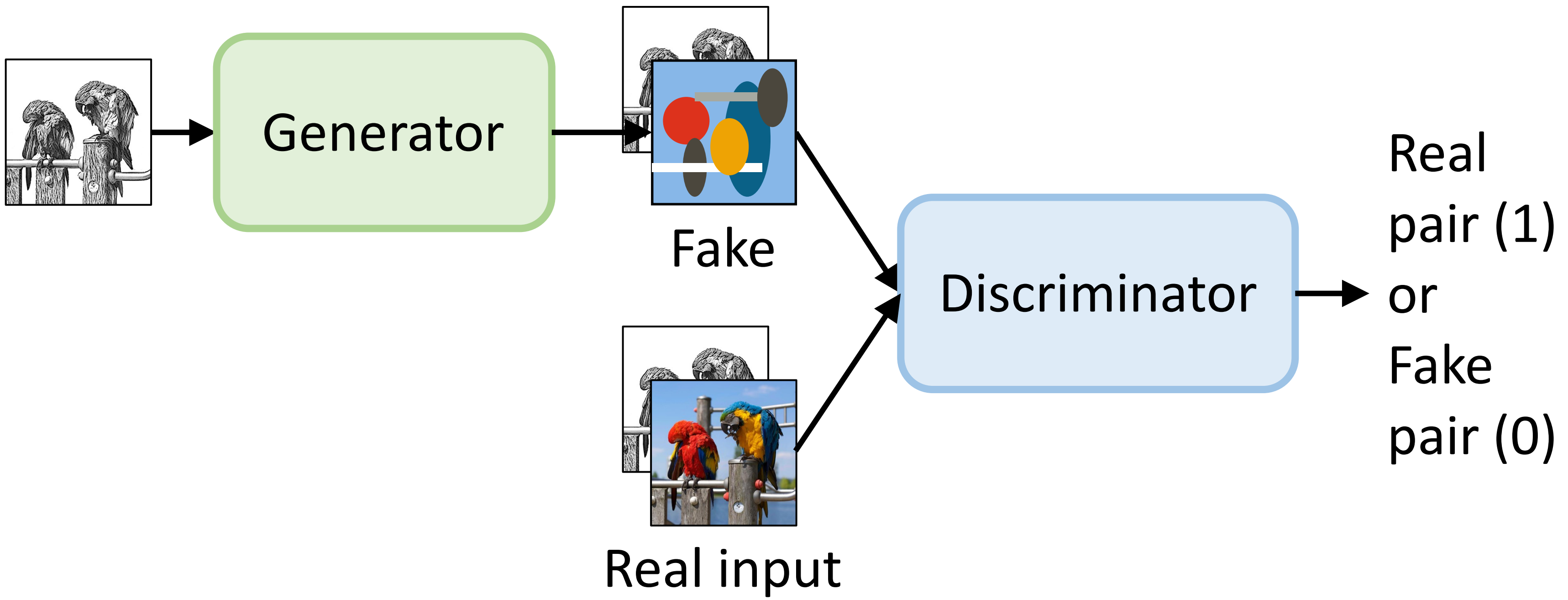
Summary

- GAN evaluation is important, because successful training does not necessarily mean the generator's output is similar to p(x)
- Generally check for:
  - Memorisation
  - Realism
  - Diversity

Conditional GANs

Conditional GANs

- Conditional model: learn p(x|y) rather than p(x)
- Both Discriminator and Generator take y as additional input

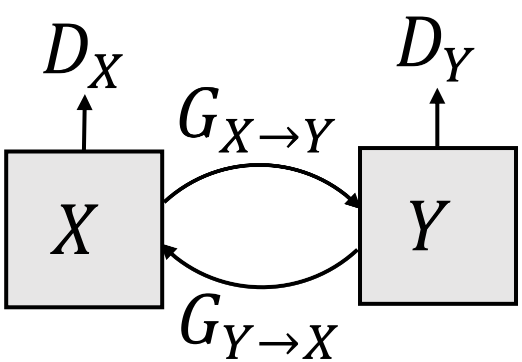


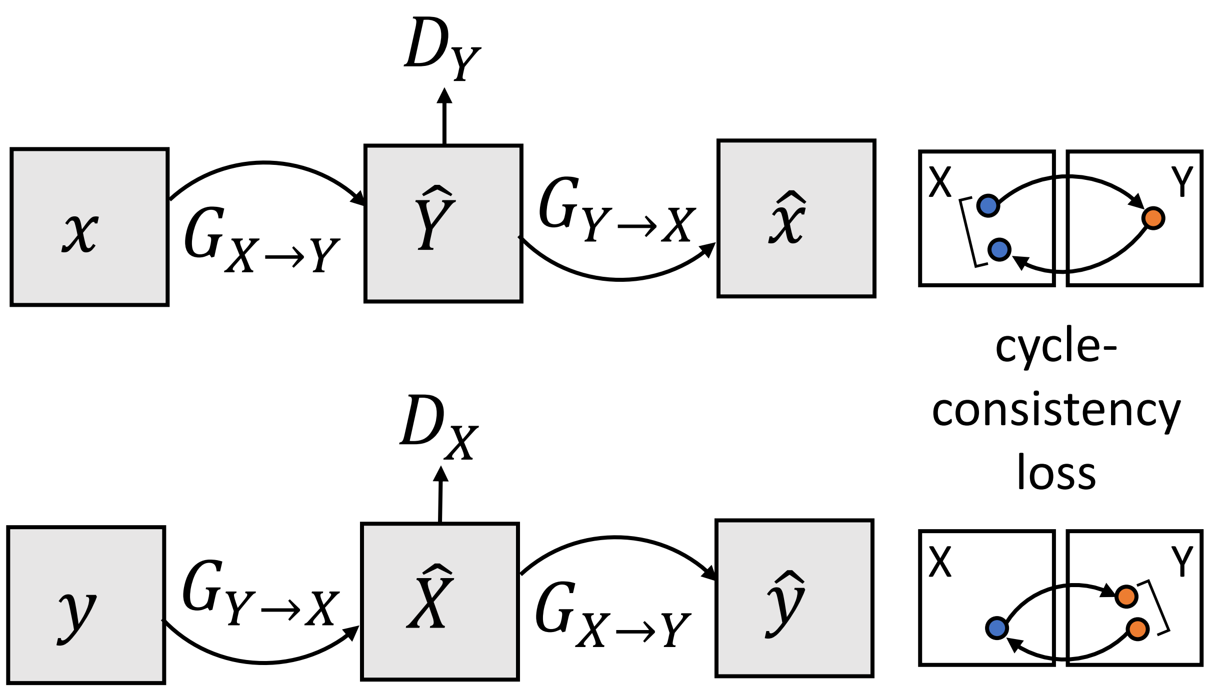
Demo

- https://affinelayer.com/pixsrv/

Conditional GANs

- What if you don't have a dataset of real x-y pairs?
- CycleGAN: train a pair of Generators to map x->y and y->x

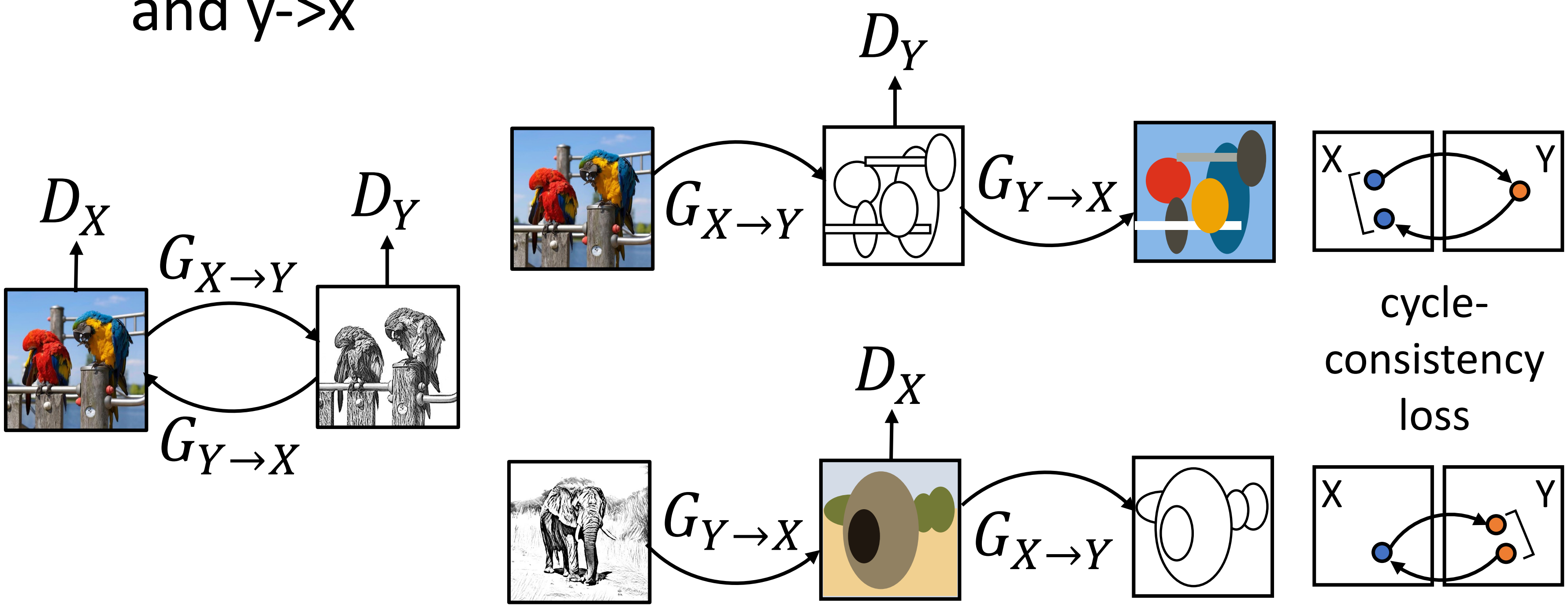




Zhu, Park, Isola, & Efros (2017)

Conditional GANs

- What if you don't have a dataset of real x-y pairs?
- CycleGAN: train a pair of Generators to map x->y and y->x



Zhu, Park, Isola, & Efros (2017)

Example: CycleGAN

apple → orange

orange → apple

Image: https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

Summary

- Conditional GAN learns to sample p(x|y) (images conditional on y)
- If x-y pairs are available, can use the standard GAN architecture with additional input y
- If x-y pairs are not available, one option is CycleGAN – learns to transform samples from x->y and from y->x

Summary

- Advantages
  - GANs can generate samples from complex probability distributions without actually representing the distribution
- Disadvantages
  - Can be unstable / hard to train
  - Difficult to evaluate
  - Even models that don't show complete mode collapse tend to have lower-than-desired diversity