[course site]

Day 4 Lecture 1

Generative models and adversarial training

Kevin McGuinness
kevin.mcguinness@dcu.ie

Research Fellow
Insight Centre for Data Analytics
Dublin City University

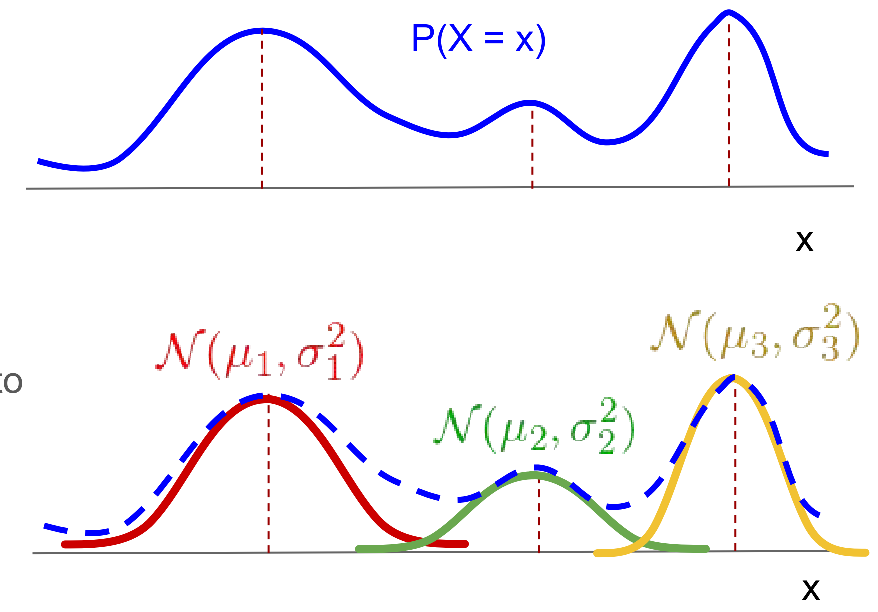
What is a generative model?

A model P(X; ϴ) that we can draw samples from.

E.g. **A Gaussian Mixture Model**

- Fitting: EM algorithm
- Drawing samples:
  - Draw sample from categorical distribution to select Gaussian
  - Draw sample from Gaussian

GMMs are not generally complex enough to draw samples of images from.



$$P(X) = \lambda_1 \mathcal{N}(\mu_1, \sigma_1^2) + \lambda_2 \mathcal{N}(\mu_2, \sigma_2^2) + \ldots$$

Why are generative models important?

- Model the probability density of images
- Understanding P(X) may help us understand P(Y | X)
- Generate novel content
- Generate training data for discriminative networks
- Artistic applications
- Image completion
- Monte-carlo estimators

Generative adversarial networks

New method of training deep generative models

Idea: pit a generator and a discriminator against each other

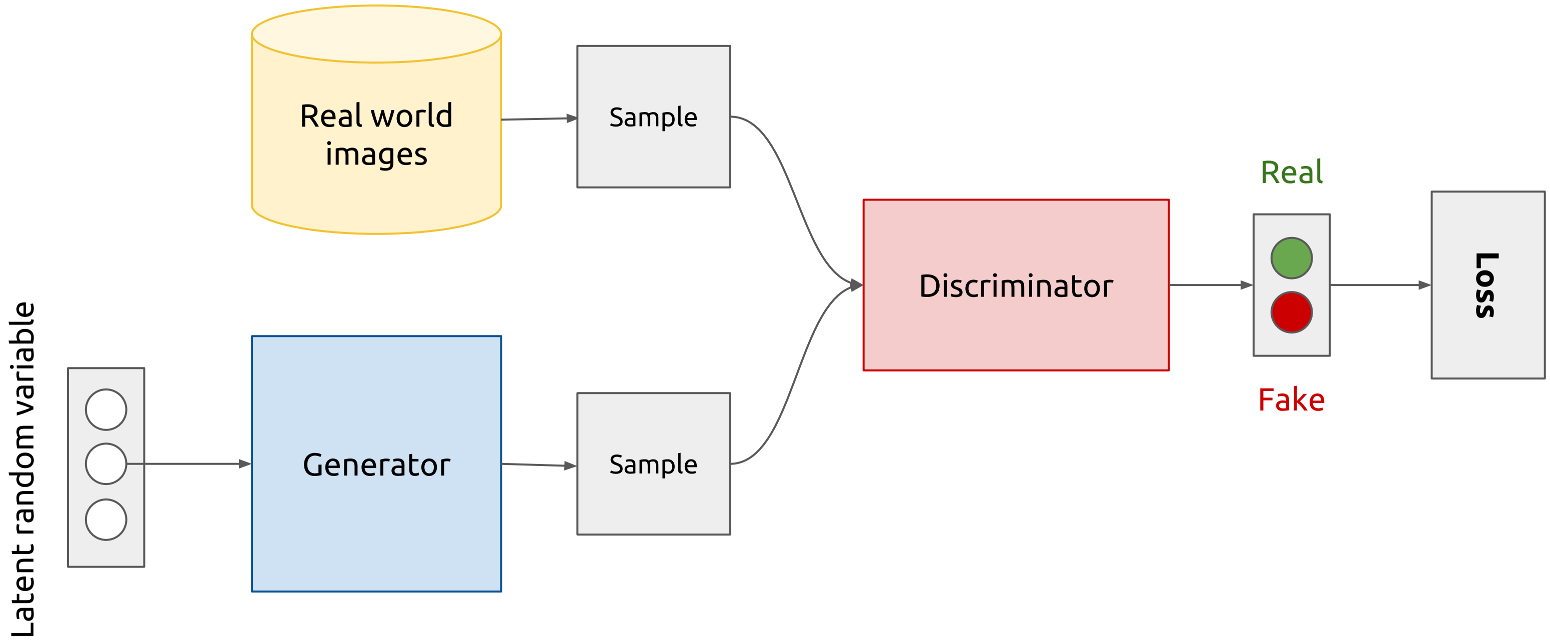
Generator tries to draw samples from P(X)

Discriminator tries to tell if sample came from the generator or the real world

Both discriminator and generator are deep networks (differentiable functions)

Can train with backprop: train discriminator for a while, then train generator, then discriminator, …

Generative adversarial networks (conceptual)

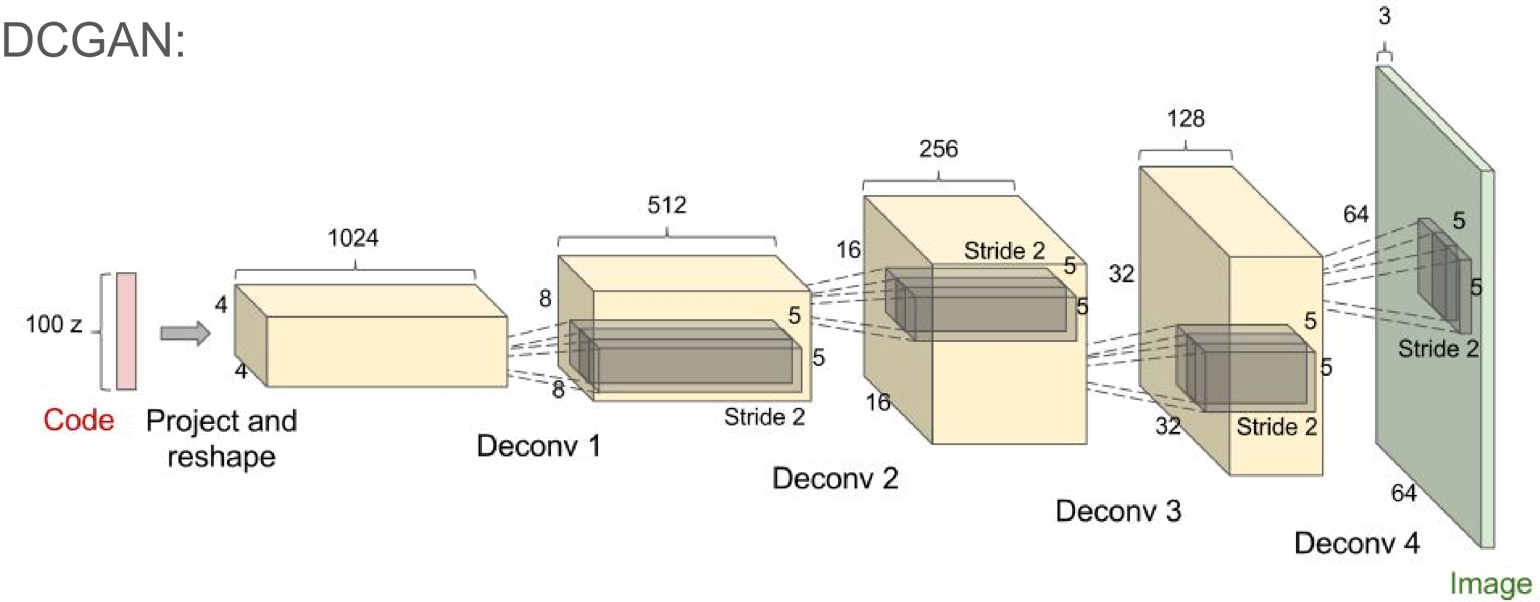


The generator

Deterministic mapping from a latent random vector to sample from q(x) ~ p(x)

Usually a deep neural network.

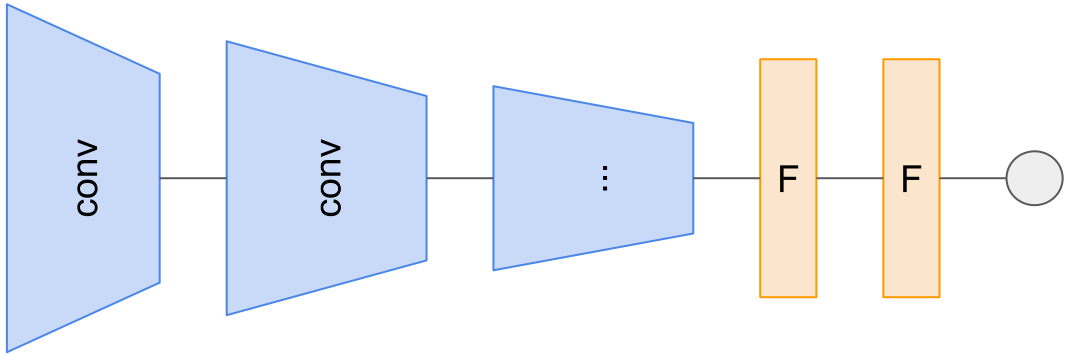
E.g. DCGAN:



The discriminator

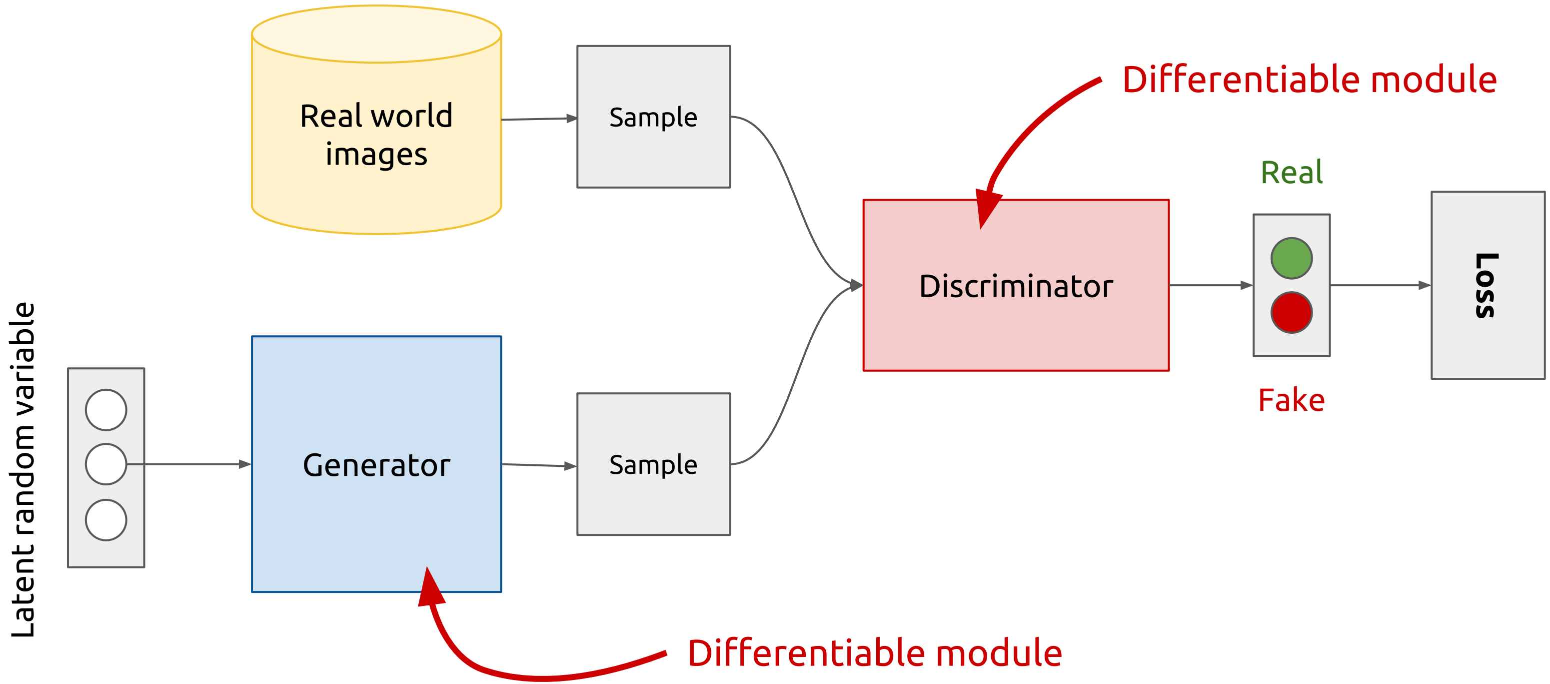
Parameterised function that tries to distinguish between samples from real images p(x) and generated ones q(x).

Usually a deep convolutional neural network.

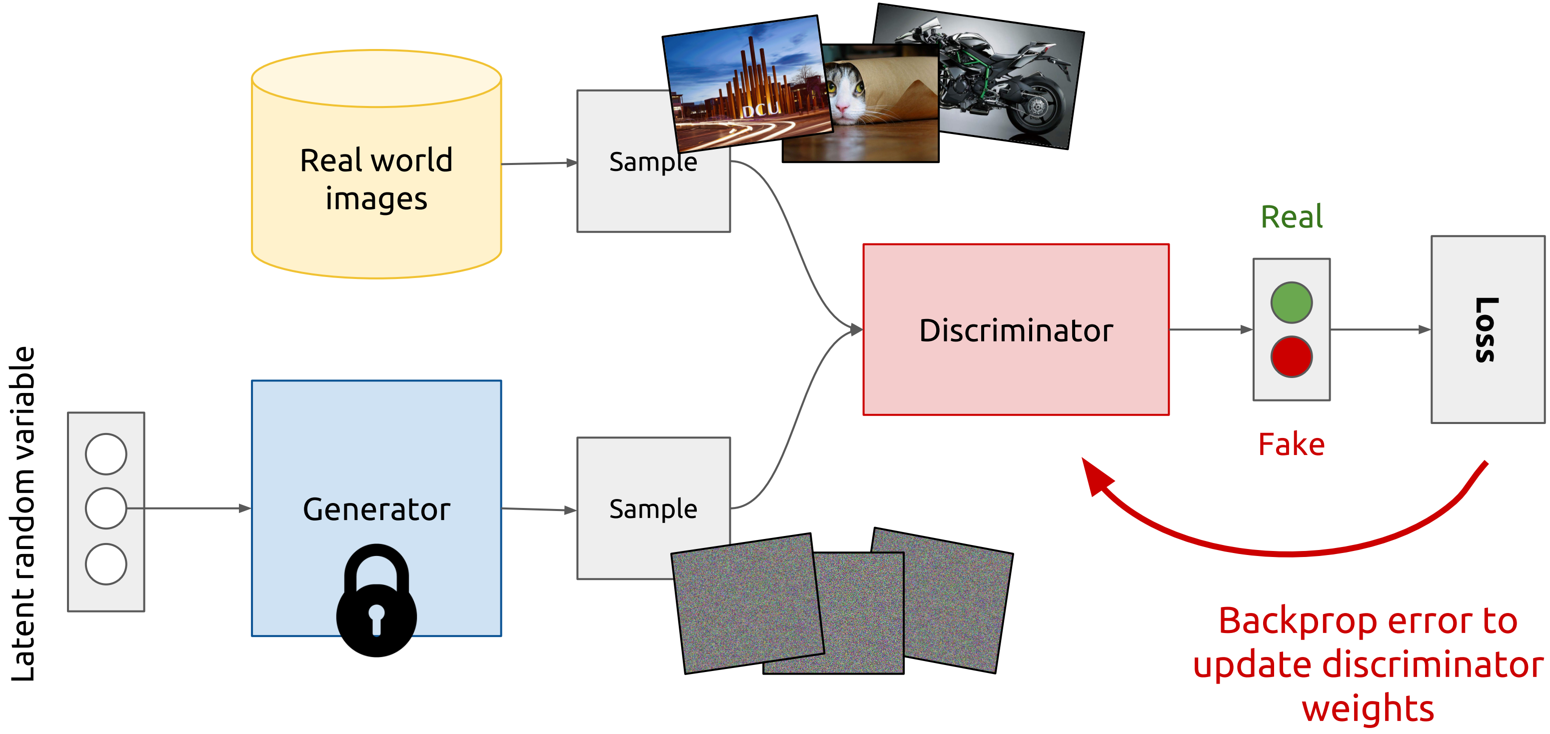


Training GANs

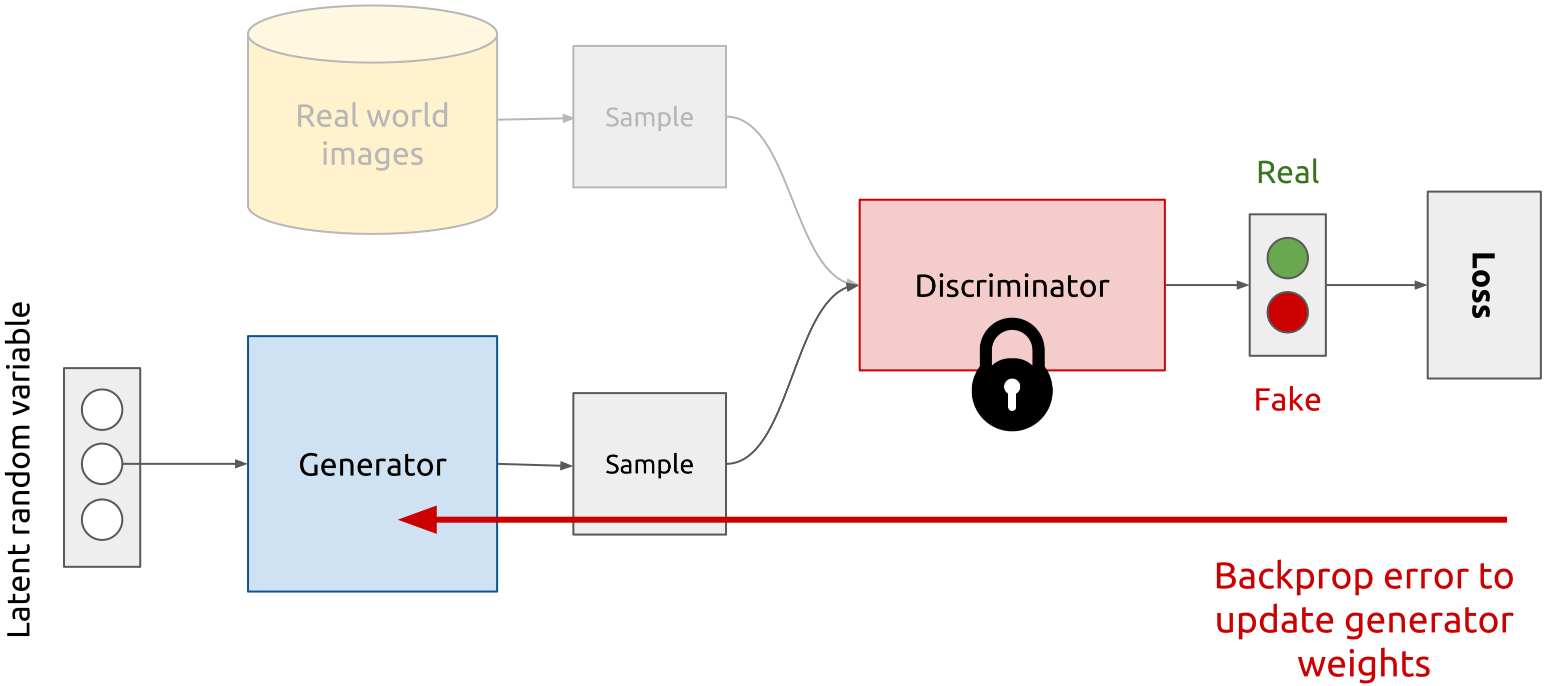
Alternate between training the discriminator and generator



1. Fix generator weights, draw samples from both real world and generated images
2. Train discriminator to distinguish between real world and generated images



1. Fix discriminator weights
2. Sample from generator
3. Backprop error through discriminator to update generator weights

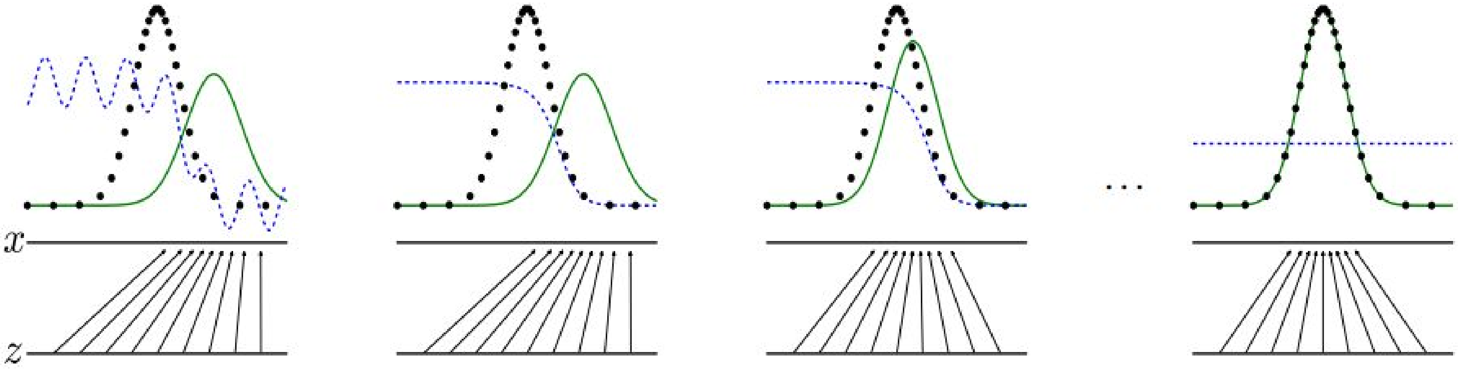


Training GANs

Iterate these two steps until convergence (which may not happen)

- Updating the discriminator should make it better at discriminating between real images and generated ones (**discriminator improves**)
- Updating the generator makes it better at fooling the current discriminator (**generator improves**)

Eventually (we hope) that the generator gets so good that it is impossible for the discriminator to tell the difference between real and generated images. Discriminator accuracy = 0.5



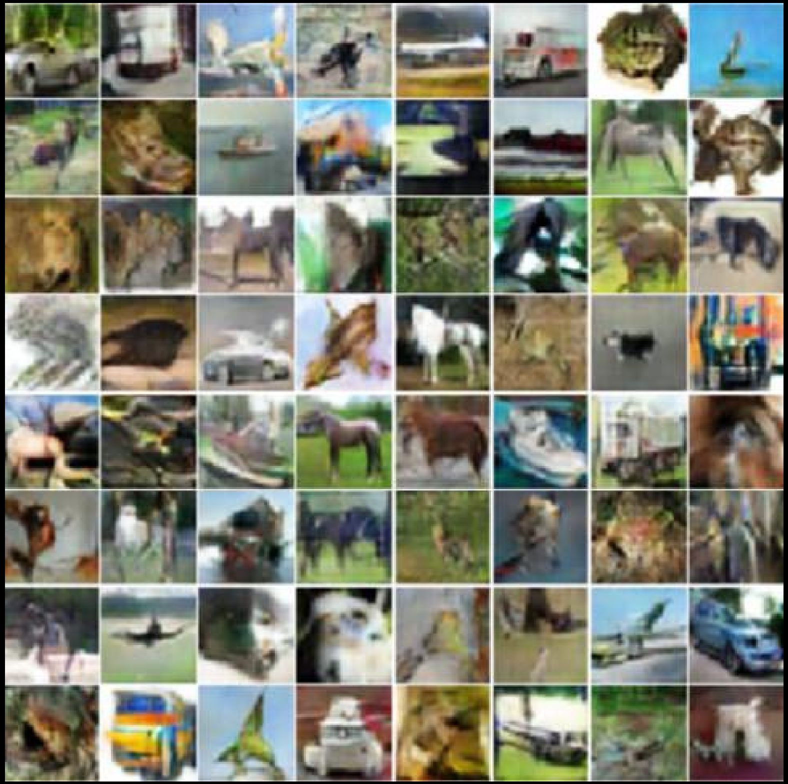
Some examples…

ImageNet

Source:
https://openai.com/blog/generative-models/

CIFAR-10

Source
https://openai.com/blog/generative-models/



Credit:
[Alec Radford]

Code on [GitHub]

Credit: Alec Radford   Code on GitHub

Issues

Known to be very difficult to train:

- Formulated as a “game” between two networks
- Unstable dynamics: hard to keep generator and discriminator in balance
- Optimization can **oscillate** between solutions
- Generator can collapse

Possible to use supervised labels to help prevent this: https://arxiv.org/abs/1606.03498

Predicting the future with adversarial training

Want to train a classifier to predict the pixels in frame (t+K) from pixels in frame t.

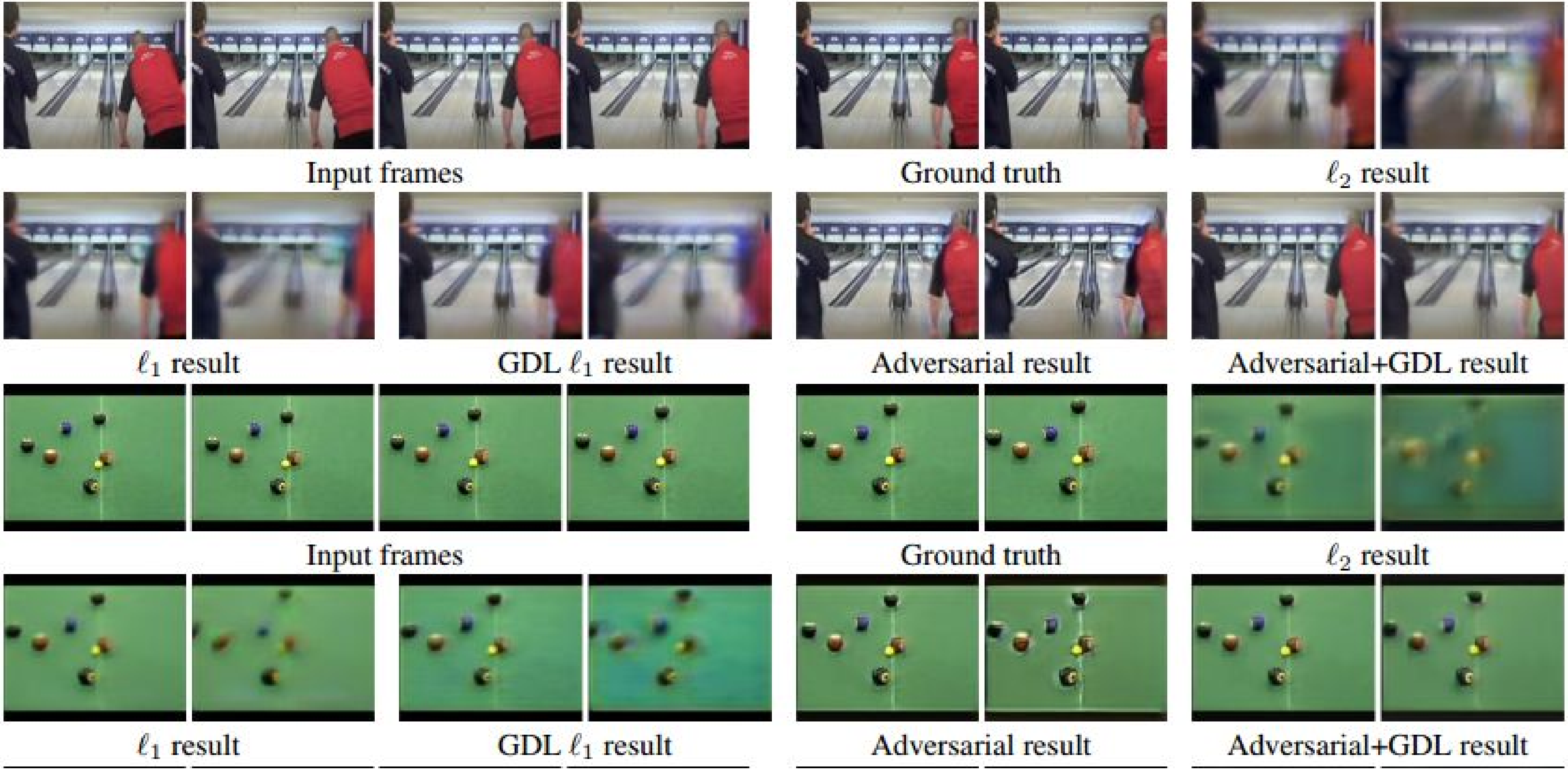
Many possible futures for same frame

Using supervised classification results in blurry solutions: loss if minimized if classifier averages over possibilities when predicting.

We really want a sample, not the mean

Adversarial training can solve this: easy for an adversary to detect blurry frames

Mathieu et al. **Deep multi-scale video prediction beyond mean square error**, ICLR 2016 (https://arxiv.org/abs/1511.05440)



Mathieu et al. **Deep multi-scale video prediction beyond mean square error**, ICLR 2016 (https://arxiv.org/abs/1511.05440)

Summary

Adversarial networks pit a generator network against a discriminator (adversary)

Can be trained to draw realistic sharp samples

Training can be difficult: can oscillate or generator can collapse