Generative AI

Some slides were adapted/taken from various sources, including Prof. Andrew Ng's Coursera Lectures, Stanford University, Prof. Kilian Q. Weinberger's lectures on Machine Learning, Cornell University, Prof. Sudeshna Sarkar's Lecture on Machine Learning, IIT Kharagpur, Prof. Bing Liu's lecture, University of Illinois at Chicago (UIC), CS231n: Convolutional Neural Networks for Visual Recognition lectures, Stanford University and many more. We thankfully acknowledge them. Students are requested to use this material for their study only and **NOT** to distribute it.

Today

- What and why?
- Learning models
- Generative Learning
- Applications

Generative AI

- Deals with AI-based algorithms to generate synthetic content like images, text, videos, etc.

- It's an unsupervised approach.

- It learns from input data of a given distribution to generate new examples that belong to a distribution close to the previously given distribution.

Why they are important?

It can be observed in the recent literature that supervised learning-based approaches have achieved phenomenal performance, especially for deep learning-based prediction models.

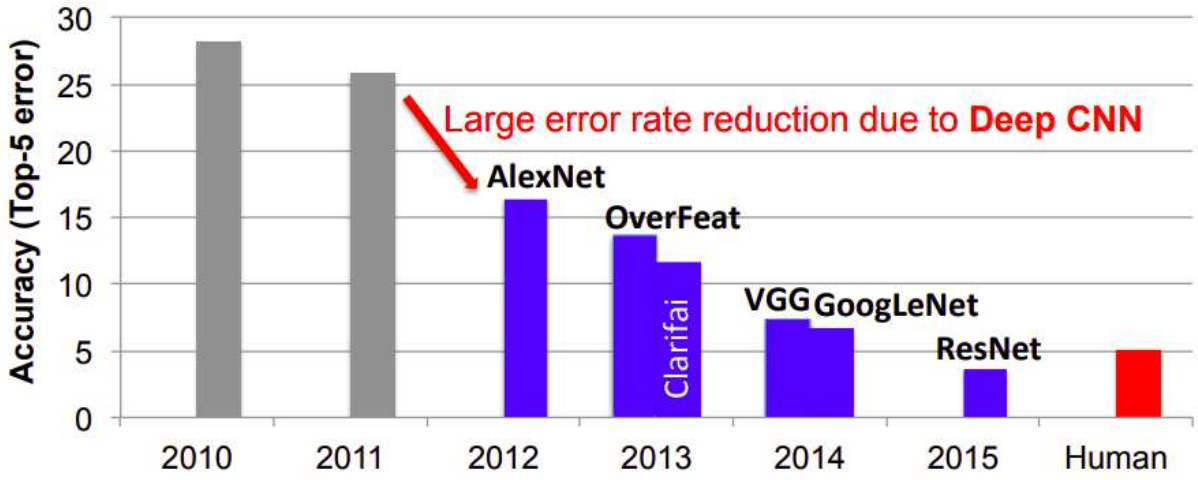


Figure: Top-5 classification error in test set - results from ImageNet challenge [1]

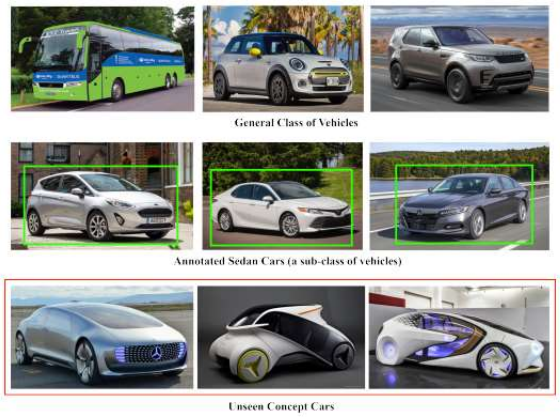
[1] Sze, Vivienne, et al. "Efficient processing of deep neural networks: A tutorial and survey." Proceedings of the IEEE 105.12 (2017): 2295-2329.

Problems with Supervised Learning

- Collecting sufficient annotated data for a large number of objects is expensive and requires expert supervision
- Difficult to collect data for rare objects in Nature
- Objects may undergo physical changes over time

Different tree types

Rare object



Intra-class variability

We can generate labeled examples synthetically using Generative AI-based algorithms

Two classes of models in machine learning

- Discriminative Models: It is learning by discriminating between two classes of data.
  - Classification (e.g. Face is fake or real)

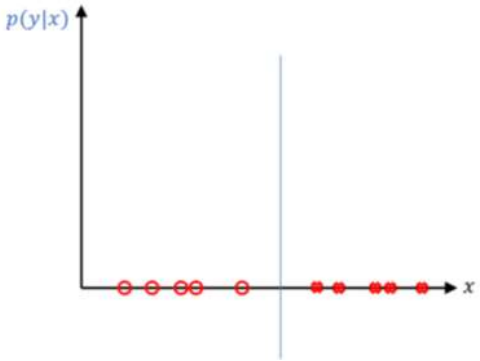
- Generative Model: A generative model G to be trained on training data X sampled from some true distribution D is the one which given some standard random distribution Z produces a distribution D' which is close to D according to some closeness metric, mathematically

  $z \sim Z$ maps to a sample $G(z) \sim D'$

Discriminative Model

Discriminative models: classify data

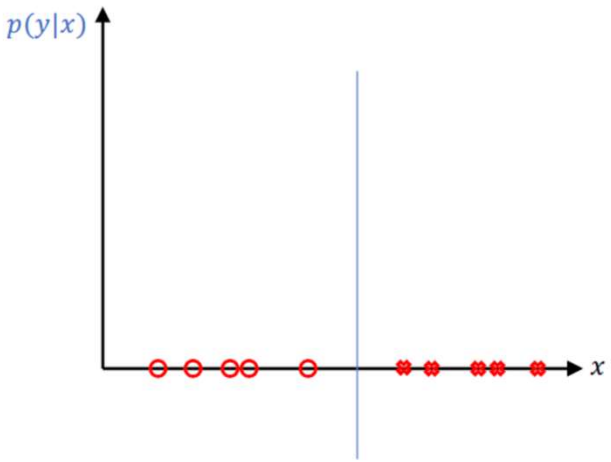
finding the **decision boundary** $P(Y|X)$



Discriminative vs. Generative

Discriminative models: classify data

finding the **decision boundary** $P(Y|X)$

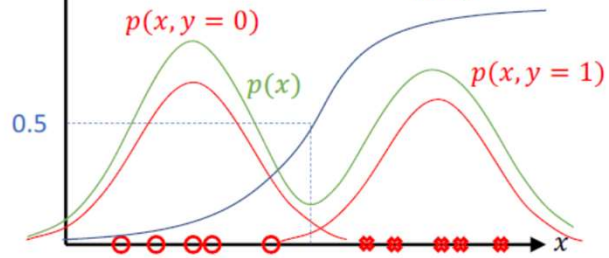


Generative models: generate data

finding **joint distribution** $P(Y,X)$

$$Conditional = \frac{Joint}{Marginal}$$

$$p(y=1|x) = \frac{p(x,y=1)}{p(x)}$$



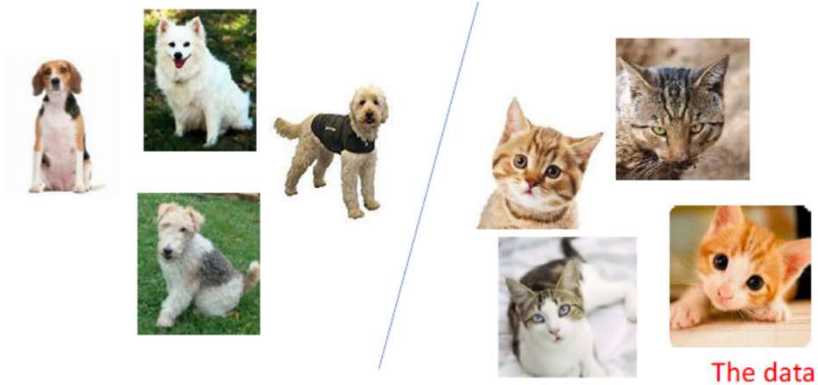
Note: Generative models can perform both generative and discriminative tasks

Discriminative vs. Generative

Discriminative models: classify data

finding **conditional distribution** $P(Y|X)$

$P(Y = Cat|X =$  $) = 0.99$

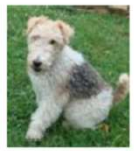


Decision boundary

Generative models: generate data

finding **joint distribution** $P(Y, X)$

$Y = Cat, X =$ 

$Y = Dog, X =$ 

The data distribution can be high-dimensional, like images

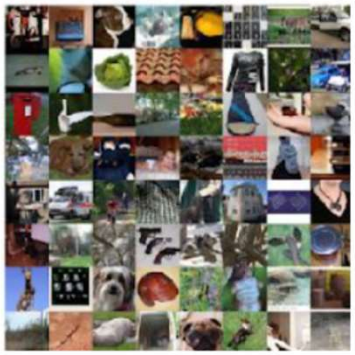
Discriminative vs. Generative

- Discriminative models do not model/learn the probability distribution of data $p(x)$
  and find the decision boundary directly to form $p(y|x)$

- Generative models need to
  first model/learn the probability distribution of data $p(x)$
  and the joint probability distribution $p(x, y)$
  and the estimated the conditional probability $p(y|x) = \frac{p(x,y)}{p(x)}$

Generative Model

Given training data, generate new samples from same distribution



Training data ~ $p_{data}(x)$ Generated samples ~ $p_{model}(x)$

Want to: learn $p_{model}(x)$ similar to $p_{data}(x)$

Addresses density estimation which is a core problem in unsupervised learning

Generative Model

Given training data, generate new samples from same distribution

Training data ~ $p_{data}(x)$

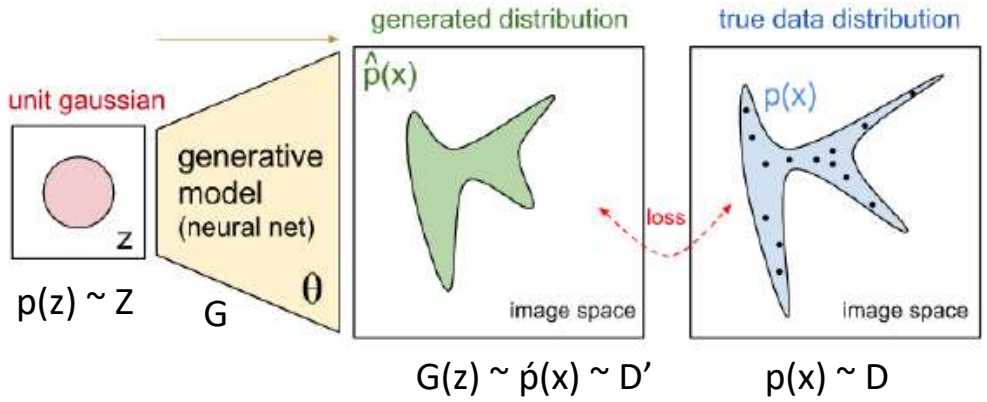
Generated samples ~ $p_{model}(x)$

Want to: learn $p_{model}(x)$ similar to $p_{data}(x)$

Addresses density estimation which is a core problem in unsupervised learning

- Explicit density estimation: explicitly define and solve for $p_{model}(x)$
- Implicit density estimation: learn model that can sample from $p_{model}(x)$ without explicitly defining it

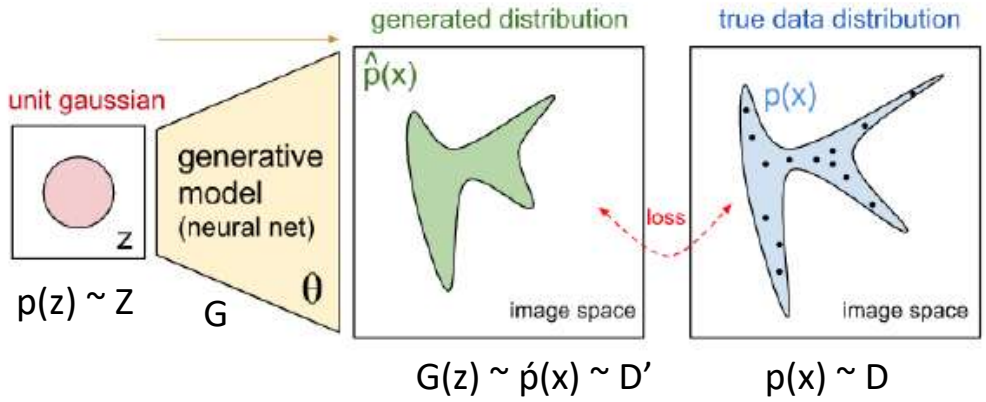
Generative Model



Let we have a set of samples (let say MNIST dataset) which form a distribution D which is not known to us. So, we have known samples but we don't know the distribution of those samples.

Goal: Approximate the distribution D and generate a data distribution (D') as close as possible to D.

How to achieve this goal: Given some standard distribution Z, use a generator which can be used to transfer the distribution Z to D',



Let Z is a random distribution with samples set z ($z \epsilon Z$), we use a generator to produce some samples (called generative sample G(z)) having distribution D' which is as close as possible to distribution D with respect to some statistical distance (let say L1 norm, L2 norm etc.) . So, mathematically, $G(z) \sim D'$

What is G: Feed forward neural network (may be DNN). It produces the generative samples G(z) having distribution D'.

Loosely speaking, G(z) are also MNIST like samples but they are not present within D. They are synthetically generated following distribution D.

Why Generative Model

- **We've only seen discriminative models so far**
  - Given an image **X**, predict a label **Y**
  - Estimates **P(Y|X)**

- **Discriminative models have several key limitations**
  - Can't model **P(X)**, i.e. the probability of seeing a certain image
  - Thus, can't sample from **P(X)**, i.e. **can't generate new images**

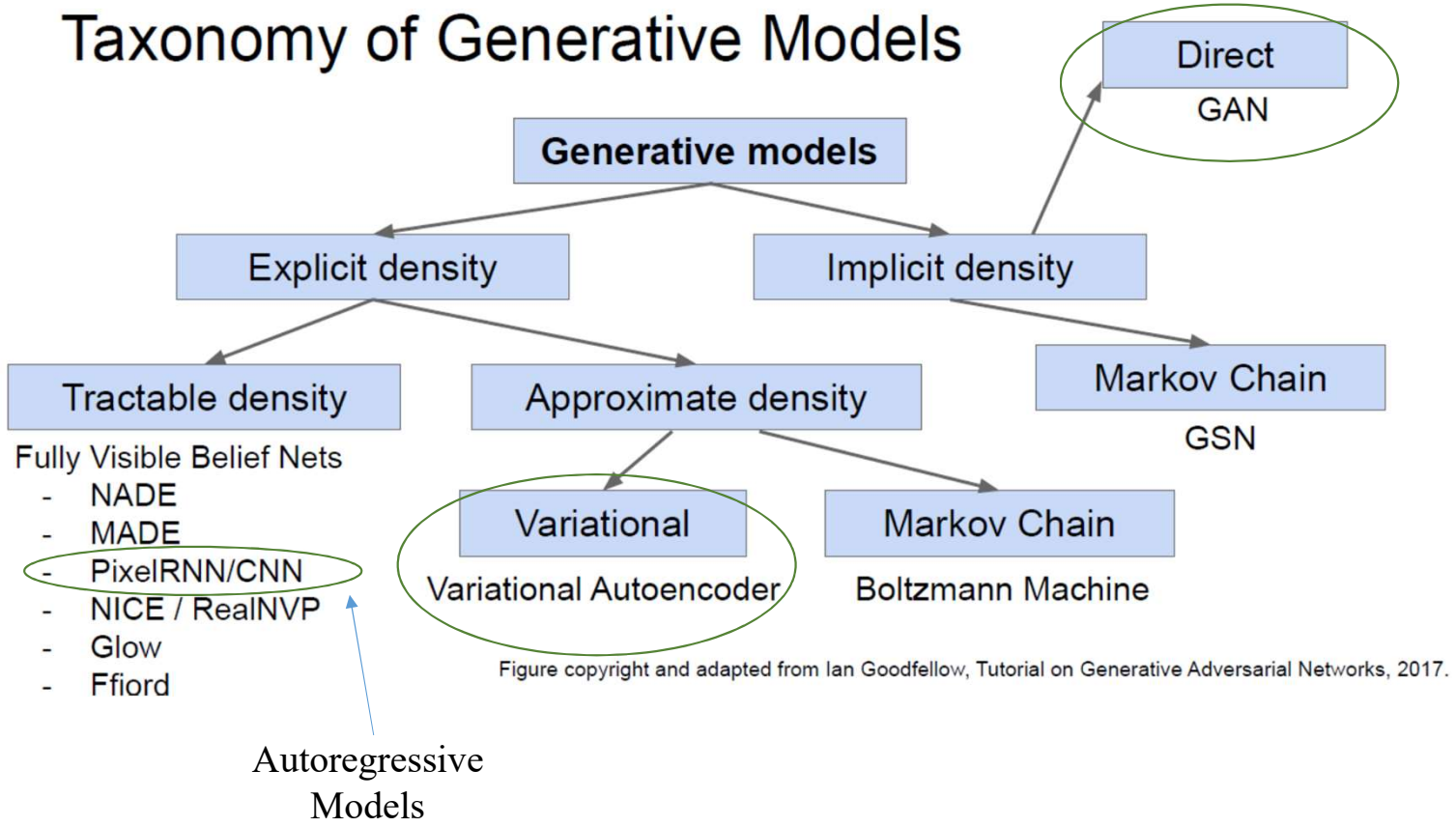
- **Generative models (in general) cope with all of above**
  - Can model **P(X)**
  - Can generate new images

Why Generative Model

- Realistic samples for artwork, super-resolution, colorization, etc.



- Generative models of time-series data can be used for simulation and planning (reinforcement learning applications!)
- Training generative models can also enable inference of latent representations that can be useful as general features

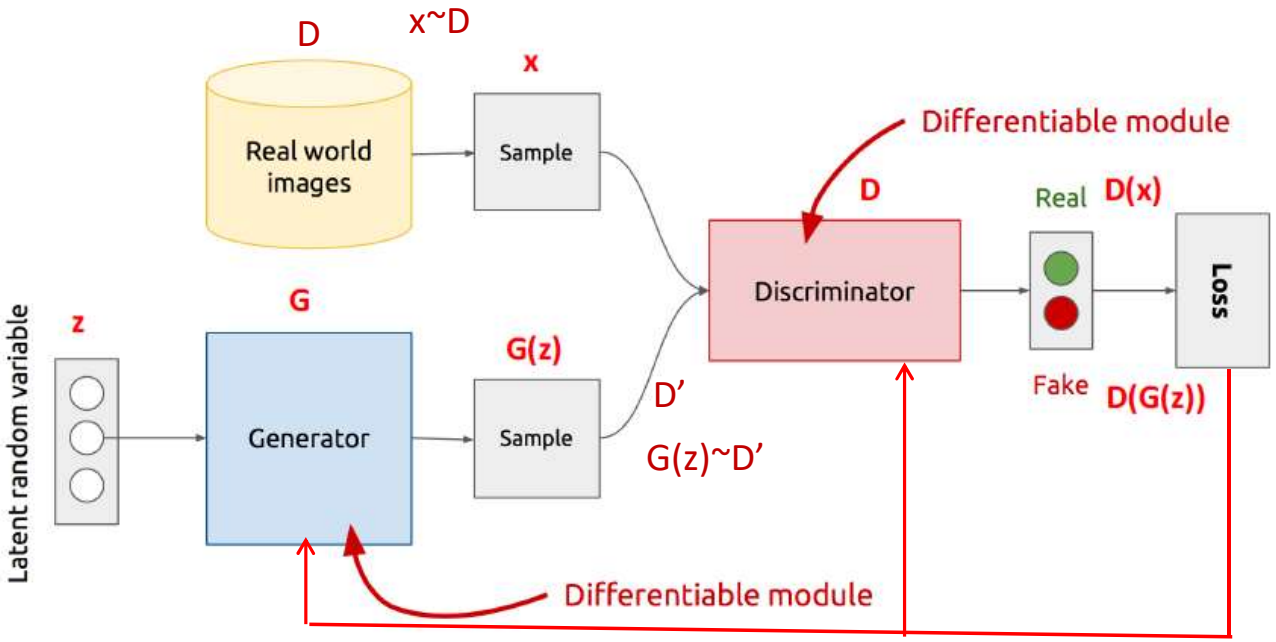

Taxonomy of Generative Models
Generative models
Explicit density
Implicit density
Direct
GAN
Tractable density
Approximate density
Markov Chain
GSN
Fully Visible Belief Nets
- NADE
- MADE
- PixelRNN/CNN
- NICE / RealNVP
- Glow
- Ffiord
Variational
Variational Autoencoder
Markov Chain
Boltzmann Machine
Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.
Autoregressive Models

Generative Adversarial Network

- Definition: GAN are deep neural net architectures comprised of two neural networks, competing one against the other (thus adversarial).

- GAN are neural networks that are trained in an adversarial manner to generate data mimicking some distribution.

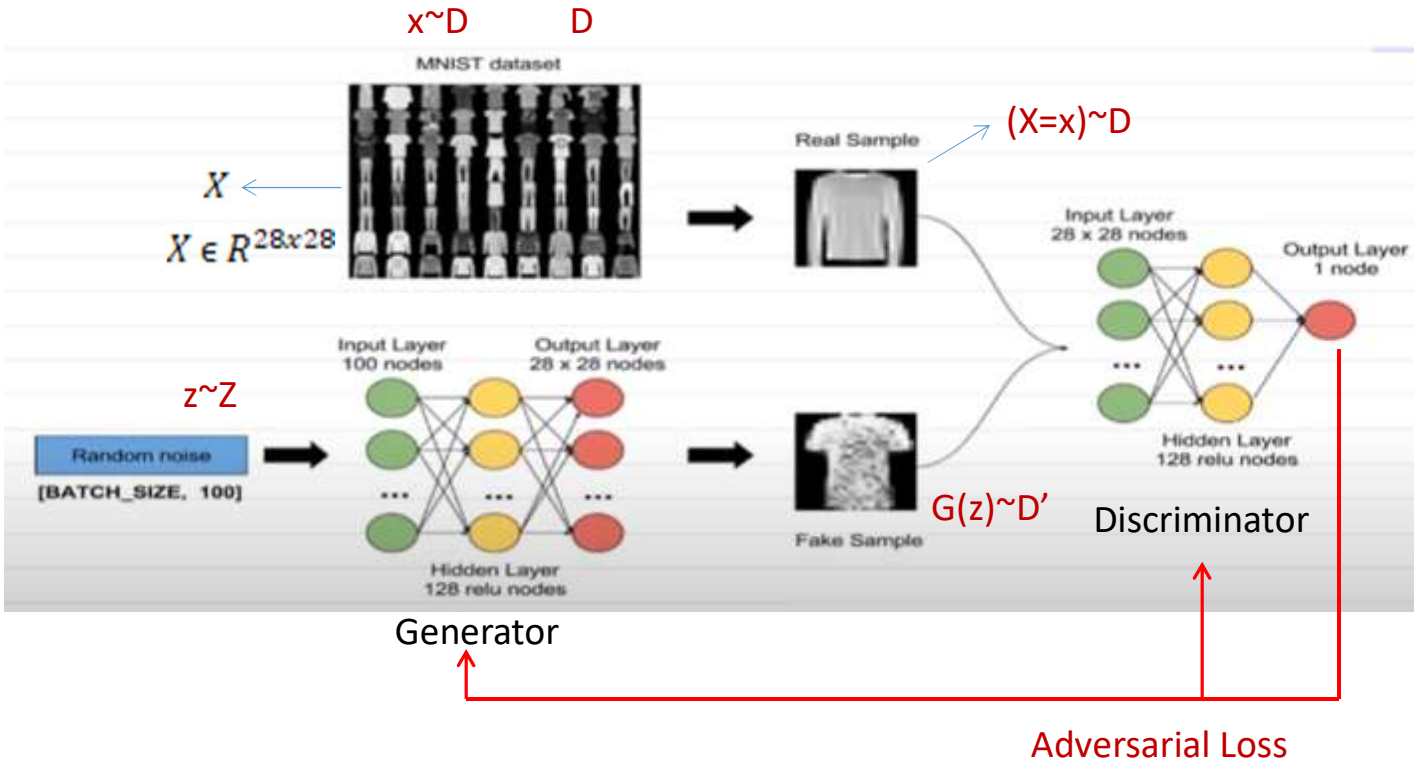
GAN Architecture



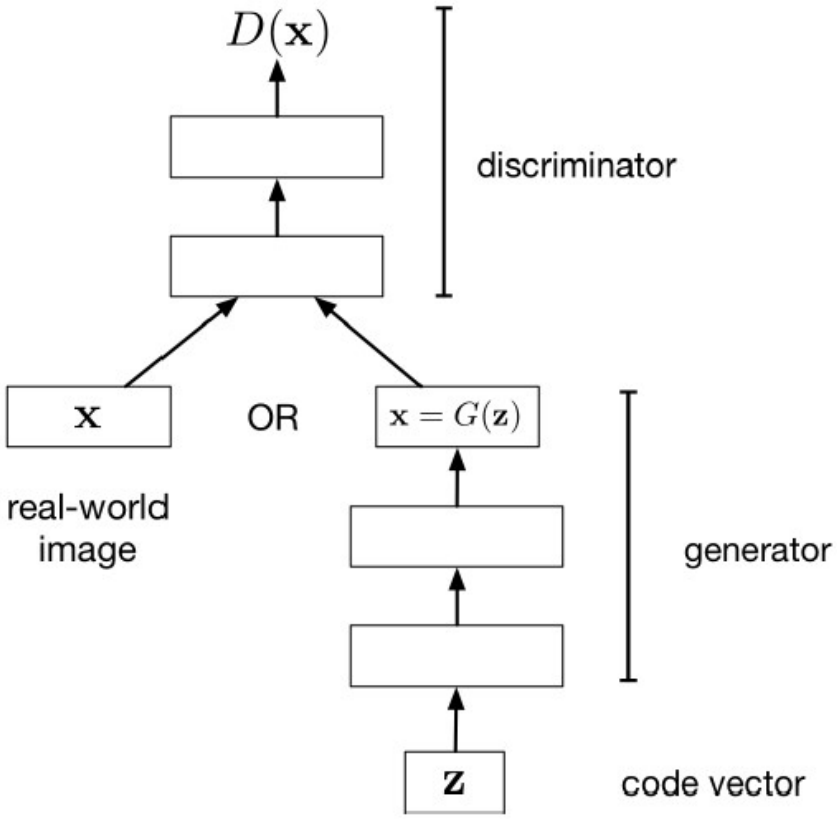
Z is some random noise (Gaussian/Uniform).

Z can be thought as the latent representation of the image.

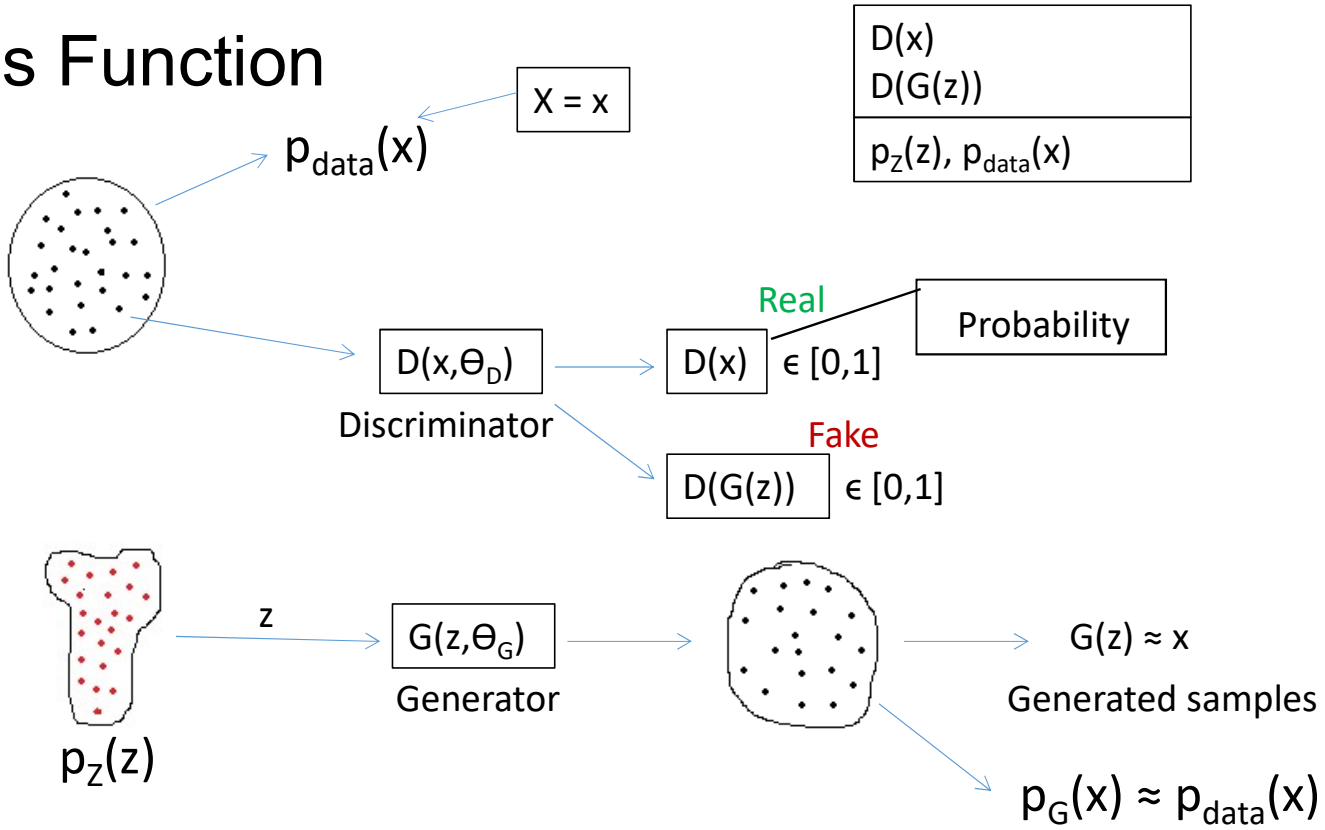
Example: MNIST Fashion



GAN Architecture



GAN: Loss Function



GAN: Loss Function

Binary Cross Entropy: $L(\hat{y}, y) = y log\, \hat{y} + (1 - y) \log (1 - \hat{y})$

The label for the data coming from $p_{data}(x)$ is y=1 & $\hat{y}$ = D(x), so putting them we obtain

L(D(x),1) = log(D(x)) ---- (A)

And the data coming from generator, the label is y=0 & $\hat{y}$ = D(G(z))

So in that case,

L(D(G(z)),0) = (1-0)log(1-D(G(z))) ---- (B)

GAN: Loss Function (Discriminator)

Now, the objective of the discriminator is to correctly classify fake vs the real dataset. For this (A) and (B) should be maximal.

$L(D(x),1) = \log(D(x))$ ---- (A) ✓

$L(D(G(z)),0) = (1-0)\log(1-D(G(z)))$ ---- (B) ✓

Now, how can we maximize both A and B?

By making D(x) = 1 we can maximize A and

By making D(G(z)) = 0, we can maximize B

} how

$$\max \{\log(D(x)) + \log(1-D(G(z)))\}$$

Thanks