Generative Adversarial Nets

framework for estimating gen models via an adversarial process

Simel train two models
G • A gen model - capture data dist
D • A discrim model - est prob that sample came from train data

training for G is to ~~mak~~ max prob of D making a mistake

framework corresponds to a minimax two player game

A unique solution exists where G recovers the training data distribution

& D equal to $\frac{1}{2}$ everywhere

If G & D = MLPs then can be trained w/ back-prop

---

introduction

historically, the best DL successes involved Discriminative models

↳ usually mapping high Dim inputs to a class label

Mostly based on Backprop, dropout, & pecewise linear units

Deep generative models less succes
~~to~~ Due to difficult of Approx Many intractable probabilistic computations from max like Estimators

Adversial nets framework

gen model is pitted against an adversary

Adverse = Discrim that learns to Determin whether a sample is from model dist or data dist

both models improve in the training

gen & Disc training together

in feedforward Disc = Discarded & only generative is used

no approximate or Markov chains are nees

2 Related works

↳

3 Adversarial Nets

Most simple to do when D & G are MLP's

to learn the generators dist $p_g(\cdot)$ over x

we define prior input noise variables $p_z(z)$
(just provides a random input)

then map to data space $G(z, \theta_g)$ θ_g

G = Differentiable func, rep by MLP w/ Params
(Params = all weights & biases)

then Define a second MLP: $D(x, \theta_d)$
→ this outputs a scalar

$D(x)$ = Prob that x came From data
rather than p_g

train D to maximise prob of assigning
the correct label to both train examples
& samples From the generator

Simeltan train G to minimize: $\log(1 - D(g(z)$

D & G Play two player Mini-Max game
w/ value func $V(G,D)$ $\min_G \max_D V(D,G)$

Sometimes, early in train when G is poor there is not sufficient gradient to train G

in this case $\log(1 - D(g(z)))$ saturates

rather than min $\log(1 - D(g(z)))$

can instead max $\log D(G(z))$

provides stronger gradients in early learning

4 theorehcal Results

the Generator implicitly Defines a prob distribution P_g as the dist of the samples $G(z)$ when $z \sim P_z$

5 Experiments

generators uses a Mix of rectifier & sigmoid activation functions

Disc used Maxout activations

Dropout used for Disc

has a challenge table for Autoencode vs Adversial Models

Autoencoders = recon vs gen
Adverse = Disc vs gen

Inference for Autoencoders = MCMC

6 Advantages & Disadvantages

(1) Disadvantage = no explicit representation of $P_g(x)$

Provides no formula or func that tells you the prob dens of any given point x

Gans learn through implicit process

learn mapping $g(z, \theta_g)$ from noise $P_z(z)$

gen is trained to fool disc, not explain the prob Density functions of outputs

to gen must sample $P_z(z)$ and pass to get

no pass on arbitary x

Makes it v hard to compare GANs

tends to rely on the qualatis evals

Other ~~VAEs~~ other gen models like VAEs use log-likihood metrics

GANs dont have an Encoder—
no way to get back to input

GANs are good @ sample from complex dist

Poor @ density estimation of dist