Generative Adversarial Networks

Mohammad H. Nazeri

January 28, 2019

outline

Why do we need generative adversarial networks?

The most interesting idea in the last 10 years in Machine Learning

Yann Lecun

Supervised Learning

Data: (x, y)
x is data, y is label
Goal: Learn a function to map $x \rightarrow y$
Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

---> cat

Unsupervised Learning

Data: x
x just data, no labels!!
Goal: Learn some underlying *hidden structure* of the data
Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

We show images in a vector manner

We show images in a vector manner

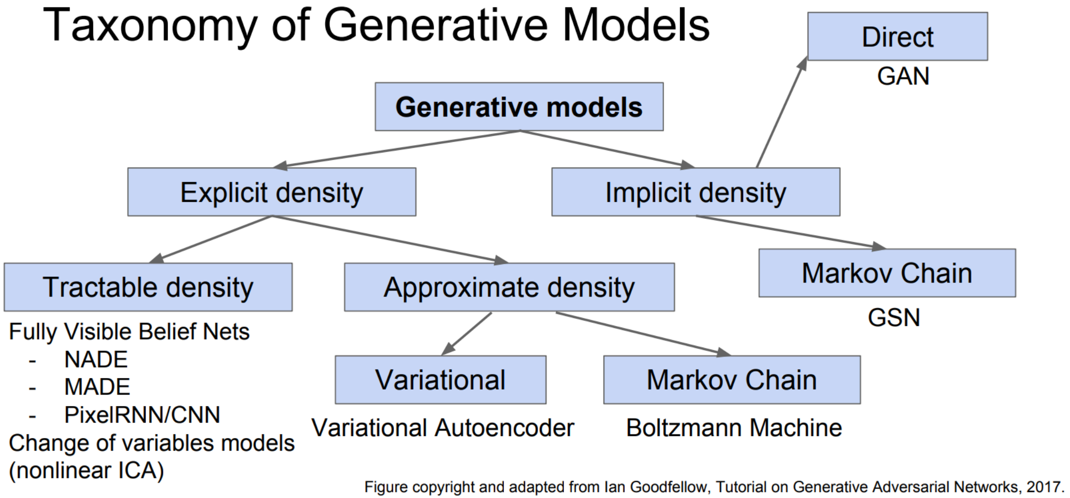
- This cat probability distribution is a very complex distribution over a very large space
- even though we assume the existence of this distribution, we obviously don't know how to express this distribution explicitly

The Goal in generative models, is to learn the distribution of data

In simple form, the problem that **Generative Models** try to tackle can be expressed as random variable generation

Types of generative models:

- Gaussian Mixture Model
- Hidden Markov Model
- Latent Dirichlet allocation
- Boltzmann Machine (e.g. Restricted Boltzmann Machine, Deep belief Network)
- Variational Autoencoder


Taxonomy of Generative Models
Generative models
Explicit density
Implicit density
Direct
GAN
Markov Chain
GSN
Tractable density
Approximate density
Fully Visible Belief Nets
- NADE
- MADE
- PixelRNN/CNN
Change of variables models (nonlinear ICA)
Variational
Variational Autoencoder
Markov Chain
Boltzmann Machine
Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Why do we need generative adversarial networks?

- Addresses density estimation, a core problem in unsupervised learning
- Simulate possible futures for planning or simulated Reinforcement Learning
- Missing data
- Multi-model outputs
- Realistic generation tasks
- Super resolution
- Training generative models can also enable inference of latent representations that can be useful as general features

Advantages of GANs:

- Use latent code that describes everything that generates later
- Asymptotically consistant: if you can find the *Nash Equilibrium* point of the game, defining GAN, you guaranteed that you actually recover the true distribution
- No Markov Chain needed: neither to train GAN, nor to drawn samples
- Often regarded as producing the best samples

How do GANs work?

Two methods for training GANs:

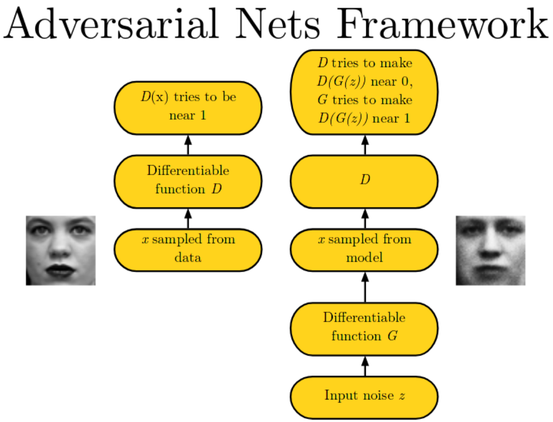
- Directed one
- Undirected one

How do GANs work?

In simple words:

- GANs use **two** models at a same time, one generates (G) data, the other discriminates (D) data as true or fake
- G tries to fool D to accepts its outputs as authentic
- D tries to minimize it's error
- G tries to maximize D's error
- It's a minimax game

How do GANs work?


Adversarial Nets Framework
D(x) tries to be near 1
Differentiable function D
x sampled from data
D tries to make $D(G(z))$ near 0, G tries to make $D(G(z))$ near 1
D
x sampled from model
Differentiable function G
Input noise z

[0]Ian Goodfellow: Generative Adversarial Networks (NIPS 2016 tutorial)

How do GANs work?

We can show the generator as a simple graphical model:

$$x = G(z; \theta^{(G)})$$

- Must be differentiable
- No invertability required
- Trainable for any size of z
- Some guarantees require z to have higher dimension than x

Training Procedure

- Use SGD-like algorithm of choice (like Adam) on two mini batches:
    - a mini batch of training examples
    - a minibatch of generated samples
- **Optional:** Run k steps of one player (D) for every step of the other player (G)

Run GD on both sides simultaneously

Loss function

Each player has it's own cost, there are many cost functions
We can use a **Minimax Game:**
$J^{(D)} = -\frac{1}{2} E_{x \sim P_{data}} \log D(x) - \frac{1}{2} E_z \log(1 - D(G(z)))$
$J^{(G)} = -J^{(D)}$

- Equilibrium is a saddle point of the discriminator loss
- Resembles Jensen-Shannon divergence
- Generator minimizes the log-probability of the discriminator being correct

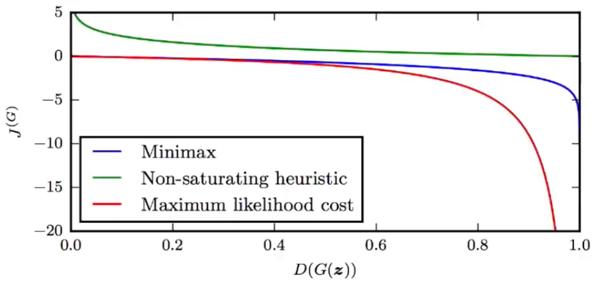
Loss function

Non-Saturating Game:

$$J^{(D)} = -\frac{1}{2} E_{x \sim P_{data}} \log D(x) - \frac{1}{2} E_z \log(1 - D(G(z)))$$
$$J^{(G)} = -\frac{1}{2} E_z \log D(G(z))$$

- Equilibrium is no longer describable with a single loss (it's heuristic, can't find out if it is *Nash*)
- Generator maximizes the log-probability of the discriminator being mistaken
- Heuristically motivated, generator can still learn even when discriminator successfully rejects all generator samples

Comparison of Generator Losses



(Goodfellow 2014)

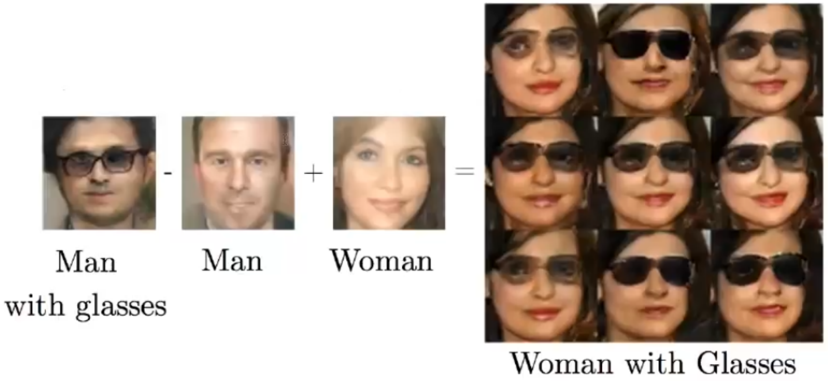
Balancing G and D

- Usually the discriminator wins
- This is a good thing - the theoretical justifications are based on assuming D is perfect
- Usually D is bigger and deeper than G
- Sometimes run D more often than G, mixed results
- Do not try to limit D to avoid making it ”too smart”
- use non-saturating cost

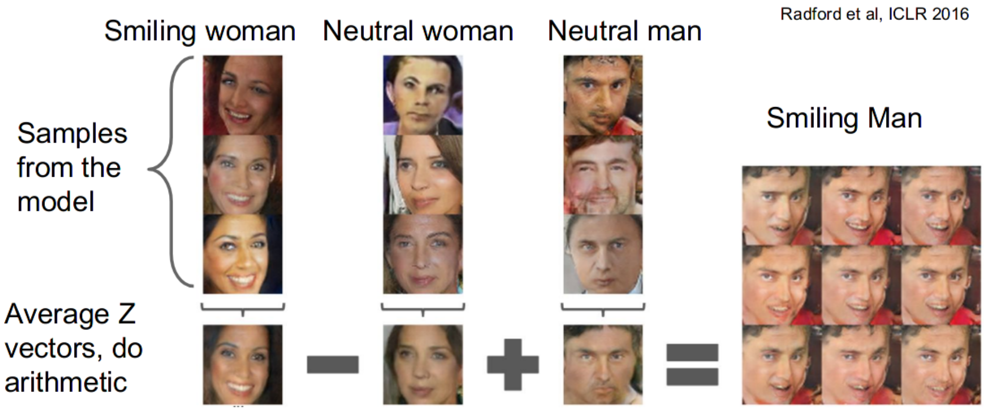
Reducing GANs to RL

- Generator makes a sample
- Discriminator evaluates a sample
- Generator's cost (negative reward) is a function of $D(G(z))$
- Note that generator's cost does not include the data x
- Generator's cost is always monotically decreasing in $D(G(z))$
- Different divergences change the location of the cost's fastest decrease

Vector Space Arithmetic



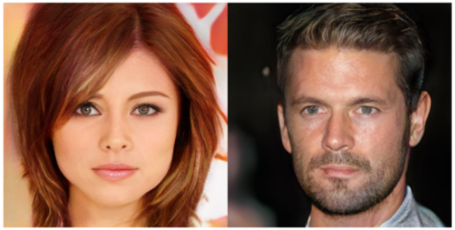
(Radford et al, 2015)


Radford et al, ICLR 2016
Smiling woman
Neutral woman
Neutral man
Smiling Man
Samples from the model
Average Z vectors, do arithmetic

Generative Modeling:
Sample Generation

Training Data
(CelebA)

$\longrightarrow$



Sample Generator
(Karras et al, 2017)

On backpropagation:
"My view is throw it all away and start again."

"The future depends on some graduate student who is deeply suspicious of everything I have said."

Geoffrey Hinton
"Godfather of Deep Learning"

Any Questions?

Thank You!

References

- Generative Adversarial Nets by Ian Goodfellow et al click here
- Ian Goodfellow: Generative Adversarial Networks (NIPS 2016 tutorial) click here
- Understanding Generative Adversarial Networks (GANs) click here
- CS231n: Convolutional Neural Networks for Visual Recognition click here
- The GAN Zoo click here
- Tips and tricks to make GANs work by Soumith Chintala click here