Generative Adversarial Network (GAN)

Outlook:

Restricted Boltzmann Machine:

Gibbs Sampling:
NIPS 2016 Tutorial: Generative Adversarial Networks

• Author: Ian Goodfellow
• Paper: https://arxiv.org/abs/1701.00160

You can find tips for training GAN here: https://github.com/soumith/ganheks
Review
Generation

http://www.rb139.com/index.php?s=/Lot/44547

Writing Poems?

Drawing?
Review: Auto-encoder

As close as possible

Randomly generate a vector as code

Image ?
Review: Auto-encoder

NN Decoder → code → 2D → NN Decoder

$\begin{bmatrix} -1.5 \\ 0 \end{bmatrix}$ → [1.5]

Input $\rightarrow$ Output $\rightarrow$ Reconstructed Image
Review: Auto-encoder
Auto-encoder

From a normal distribution

Problems of VAE

• It does not really try to simulate real images

```
code
```

![Diagram showing problem areas of VAE](image)

- One pixel difference from the target (Realistic)
- One pixel difference from the target (Fake)
The evolution of generation

- **NN Generator v1**
- **Disriminator v1**

- **NN Generator v2**
- **Disriminator v2**

- **NN Generator v3**
- **Disriminator v3**

- **Binary Classifier**

Real images: 5 0 4 1
The evolution of generation

NN Generator v1 → NN Generator v2 → NN Generator v3

Discriminator v1 → Discriminator v2 → Discriminator v3

Real images: 5041
GAN - Discriminator

Randomly sample a vector

Something like Decoder in VAE

Generator v1

Real images:

1/0 (real or fake)
GAN - Generator

Updating the parameters of generator

The output be classified as “real” (as close to 1 as possible)

Generator + Discriminator = a network

Using gradient descent to update the parameters in the generator, but fix the discriminator
GAN – 二次元人物頭像鍊成

Source of images: https://zhuanlan.zhihu.com/p/24767059
DCGAN: https://github.com/carpedm20/DCGAN-tensorflow
GAN－二次元人物頭像鍊成

100 rounds
GAN — 二次元人物頭像鍊成

1000 rounds
GAN – 二次元人物頭像鍊成

2000 rounds
GAN – 二次元人物頭像鍊成

5000 rounds
GAN - 二次元人物頭像鍊成

10,000 rounds
GAN – 二次元人物頭像鍊成

20,000 rounds
GAN – 二次元人物頭像鍊成

50,000 rounds
Basic Idea of GAN
Maximum Likelihood Estimation

• Given a data distribution $P_{data}(x)$
• We have a distribution $P_G(x; \theta)$ parameterized by $\theta$
  • E.g. $P_G(x; \theta)$ is a Gaussian Mixture Model, $\theta$ are means and variances of the Gaussians
  • We want to find $\theta$ such that $P_G(x; \theta)$ close to $P_{data}(x)$

Sample $\{x^1, x^2, \ldots, x^m\}$ from $P_{data}(x)$

We can compute $P_G(x^i; \theta)$

Likelihood of generating the samples

$$L = \prod_{i=1}^{m} P_G(x^i; \theta)$$

Find $\theta^*$ maximizing the likelihood
Maximum Likelihood Estimation

\[ \theta^* = \arg \max_{\theta} \prod_{i=1}^{m} P_G(x^i; \theta) = \arg \max_{\theta} \log \prod_{i=1}^{m} P_G(x^i; \theta) \]

\[ = \arg \max_{\theta} \sum_{i=1}^{m} \log P_G(x^i; \theta) \]

\[ \approx \arg \max_{\theta} E_{x \sim P_{data}}[\log P_G(x; \theta)] \]

\[ = \arg \max_{\theta} \int_{x} P_{data}(x) \log P_G(x; \theta)dx - \int_{x} P_{data}(x) \log P_{data}(x)dx \]

\[ = \arg \min_{\theta} KL(P_{data}(x) || P_G(x; \theta)) \]

How to have a very general \( P_G(x; \theta) \)?
Now $P_G(x; \theta)$ is a NN

$$P_G(x; \theta) = \int_{z} P_{prior}(z) I_{[G(z)=x]} \, dz$$

It is difficult to compute the likelihood.

https://blog.openai.com/generative-models/
Basic Idea of GAN

• Generator G
  - G is a function, input z, output x
  - Given a prior distribution $P_{\text{prior}}(z)$, a probability distribution $P_G(x)$ is defined by function G

• Discriminator D
  - D is a function, input x, output scalar
  - Evaluate the “difference” between $P_G(x)$ and $P_{\text{data}}(x)$

• There is a function $V(G,D)$.

$$G^* = \arg\min_G \max_D V(G, D)$$
Basic Idea

\[
G^* = \arg \min_G \max_D V(G, D)
\]

\[
V = E_{x \sim P_{data}}[\log D(x)] + E_{x \sim P_G}[\log (1 - D(x))]
\]

Given a generator \( G \), \( \max_D V(G, D) \) evaluate the "difference" between \( P_G \) and \( P_{data} \)

Pick the \( G \) defining \( P_G \) most similar to \( P_{data} \)
max \( V(G, D) \) \quad \quad G^* = \arg \min_G \max_D V(G, D)

- Given \( G \), what is the optimal \( D^* \) maximizing

\[
V = E_{x \sim P_{data}}[\log D(x)] + E_{x \sim P_G}[\log(1 - D(x))]
\]

\[
= \int_{x} P_{data}(x) \log D(x) \, dx + \int_{x} P_{G}(x) \log(1 - D(x)) \, dx
\]

\[
= \int_{x} \left[ P_{data}(x) \log D(x) + P_{G}(x) \log(1 - D(x)) \right] \, dx
\]

Assume that \( D(x) \) can have any value here

- Given \( x \), the optimal \( D^* \) maximizing

\[
P_{data}(x) \log D(x) + P_{G}(x) \log(1 - D(x))
\]
\[\max_D V(G, D) \quad G^* = \arg\min_G \max_D V(G, D)\]

- Given \(x\), the optimal \(D^*\) maximizing

\[P_{data}(x)\log D(x) + P_G(x)\log(1 - D(x))\]

\[a \quad D^* \quad b \quad D\]

- Find \(D^*\) maximizing: \(f(D) = a \log(D) + b \log(1 - D)\)

\[\frac{df(D)}{dD} = a × \frac{1}{D} + b × \frac{1}{1 - D} × (-1) = 0\]

\[a × \frac{1}{D^*} = b × \frac{1}{1 - D^*} \quad a × (1 - D^*) = b × D^* \quad a - aD^* = bD^*\]

\[D^* = \frac{a}{a + b}\]

\[D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} < 1\]
\[
\max_D V(G, D) \quad G^* = \arg \min_G \max_D V(G, D)
\]

\[
D_1^*(x) = \frac{P_{\text{data}}(x)}{P_{\text{data}}(x) + P_{G_1}(x)}
\]

\[
D_2^*(x) = \frac{P_{\text{data}}(x)}{P_{\text{data}}(x) + P_{G_2}(x)}
\]

“difference” between \(P_{G_1}\) and \(P_{\text{data}}\)
\[
\max_D V(G, D)
\]

\[
\max_D V(G, D) = V(G, D^*)
\]

\[
D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)}
\]

\[
V = E_{x \sim P_{data}}[\log D(x)] + E_{x \sim P_G}[\log (1 - D(x))]
\]

\[
= E_{x \sim P_{data}} \left[ \log \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} \right] + E_{x \sim P_G} \left[ \log \frac{P_G(x)}{P_{data}(x) + P_G(x)} \right]
\]

\[
= \int_{x} P_{data}(x) \log \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} dx + \int_{x} P_G(x) \log \frac{P_G(x)}{P_{data}(x) + P_G(x)} dx
\]

\[
+ 2 \log \frac{1}{2} - 2 \log 2
\]
\[ \max_D V(G, D) \]

\[ \max_D V(G, D) = V(G, D^*) \]

\[ = -2\log 2 + \int x P_{\text{data}}(x) \log \left( \frac{P_{\text{data}}(x)}{P_{\text{data}}(x) + P_G(x)} \right) dx \]

\[ + \int x P_G(x) \log \left( \frac{P_G(x)}{P_{\text{data}}(x) + P_G(x)} \right) dx \]

\[ = -2\log 2 + \text{KL} \left( P_{\text{data}}(x) \| \frac{P_{\text{data}}(x) + P_G(x)}{2} \right) \]

\[ + \text{KL} \left( P_G(x) \| \frac{P_{\text{data}}(x) + P_G(x)}{2} \right) \]

\[ = -2\log 2 + 2 \text{JSD} \left( P_{\text{data}}(x) \| P_G(x) \right) \]

Jensen-Shannon divergence
In the end ......

- Generator G, Discriminator D
- Looking for G* such that
  
  \[ G^* = \arg \min_G \max_D V(G, D) \]

- Given G, \( \max_D V(G, D) \)
  
  \[ = -2 \log 2 + 2 \text{JSD}(P_{data}(x) \| P_G(x)) \]

- What is the optimal G?
  
  \[ P_G(x) = P_{data}(x) \]
Algorithm

\[ G^* = \arg \min_G \max_D V(G, D) \]

- To find the best \( G \) minimizing the loss function \( L(G) \),

\[ \theta_G \leftarrow \theta_G - \eta \frac{\partial L(G)}{\partial \theta_G} \]

\[ f(x) = \max\{D_1(x), D_2(x), D_3(x)\} \]

\[ \frac{df(x)}{dx} = ? \quad \frac{dD_i(x)}{dx} \]

If \( D_i(x) \) is the max one
Algorithm

- Given $G_0$
- Find $D_0^*$ maximizing $V(G_0, D)$

$V(G_0, D_0^*)$ is the JS divergence between $P_{data}(x)$ and $P_{G_0}(x)$

- $\theta_G \leftarrow \theta_G - \eta \frac{\partial V(G, D_0^*)}{\partial \theta_G}$ Obtain $G_1$ Decrease JS divergence(?)
- Find $D_1^*$ maximizing $V(G_1, D)$

$V(G_1, D_1^*)$ is the JS divergence between $P_{data}(x)$ and $P_{G_1}(x)$

- $\theta_G \leftarrow \theta_G - \eta \frac{\partial V(G, D_1^*)}{\partial \theta_G}$ Obtain $G_2$ Decrease JS divergence(?)
- ......
Algorithm

• Given $G_0$

• Find $D_0^*$ maximizing $V(G_0, D)$

$V(G_0, D_0^*)$ is the JS divergence between $P_{data}(x)$ and $P_{G_0}(x)$

$\theta_G \leftarrow \theta_G - \eta \frac{\partial V(G, D_0^*)}{\partial \theta_G} \quad \text{Obtain } G_1$

$V(G_0, D_0^*) \quad \text{smaller}$

$V(G_1, D_0^*)$

Assume $D_0^* \approx D_1^*$

Don’t update G too much
In practice ...

\[
V = E_{x \sim P_{data}}[\log D(x)] + E_{x \sim P_G}[\log(1 - D(x))]
\]

- Given G, how to compute \(\max_D V(G, D)\)
  - Sample \(\{x^1, x^2, \ldots, x^m\}\) from \(P_{data}(x)\), sample \(\{\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m\}\) from generator \(P_G(x)\)

\[
\text{Maximize } \tilde{V} = \frac{1}{m} \sum_{i=1}^{m} \log D(x^i) + \frac{1}{m} \sum_{i=1}^{m} \log (1 - D(\tilde{x}^i))
\]

Binary Classifier

Output is \(D(x)\)  Minimize Cross-entropy
If x is a positive example  Minimize \(-\log D(x)\)
If x is a negative example  Minimize \(-\log(1-D(x))\)
D is a binary classifier (can be deep) with parameters $\theta_d$

$$
\{x^1, x^2, \ldots, x^m\} \text{ from } P_{data}(x) \quad \Rightarrow \quad \text{Positive examples}
$$

$$
\{\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^m\} \text{ from } P_G(x) \quad \Rightarrow \quad \text{Negative examples}
$$

Minimize

$$
L = \frac{1}{m} \sum_{i=1}^{m} \log D(x^i) + \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D(\tilde{x}^i)\right)
$$

Maximize

$$
\tilde{V} = \frac{1}{m} \sum_{i=1}^{m} \log D(x^i) + \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D(\tilde{x}^i)\right)
$$
Algorithm

Initialize $\theta_d$ for D and $\theta_g$ for G

Repeat $k$ times

Learning D

• In each training iteration:
  • Sample m examples $\{x^1, x^2, ..., x^m\}$ from data distribution $P_{data}(x)$
  • Sample m noise samples $\{z^1, z^2, ..., z^m\}$ from the prior $P_{prior}(z)$
  • Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, ..., \tilde{x}^m\}$, $\tilde{x}^i = G(z^i)$
  • Update discriminator parameters $\theta_d$ to maximize
    
    \[
    \tilde{V} = \frac{1}{m} \sum_{i=1}^{m} \log D(x^i) + \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D(\tilde{x}^i)\right)
    \]
    
    \[
    \theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)
    \]

• Sample another m noise samples $\{z^1, z^2, ..., z^m\}$ from the prior $P_{prior}(z)$

Learning G

• Update generator parameters $\theta_g$ to minimize
  
  \[
  \tilde{V} = \frac{1}{m} \sum_{i=1}^{m} \log D(x^i) + \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D(G(z^i))\right)
  \]

  \[
  \theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)
  \]

Can only find lower bound of $\max_D V(G, D)$
Objective Function for Generator in Real Implementation

\[ V = \mathbb{E}_{x \sim P_{\text{data}}} \left[ \log D(x) \right] \]

\[ + \mathbb{E}_{x \sim P_{G}} \left[ \log (1 - D(x)) \right] \]

Slow at the beginning

\[ V = \mathbb{E}_{x \sim P_{G}} \left[ -\log (D(x)) \right] \]

Real implementation:
label x from \( P_G \) as positive

\[ -\log(D(x)) \]

\[ \log(1 - D(x)) \]
Demo

• The code used in demo from:
  • https://github.com/osh/KerasGAN/blob/master/MNIST_CNN_GAN_v2.ipynb
Issue about Evaluating the Divergence
Evaluating JS divergence

Evaluating JS divergence

- JS divergence estimated by discriminator telling little information

https://arxiv.org/abs/1701.07875

Weak Generator

Strong Generator
Discriminator

\[ V = E_{x \sim P_{data}} [\log D(x)] + E_{x \sim P_G} [\log (1 - D(x))] \]
\[ \approx \frac{1}{m} \sum_{i=1}^{m} \log D(x^i) + \frac{1}{m} \sum_{i=1}^{m} \log (1 - D(\tilde{x}^i)) \]
\[ \max_D V(G, D) = -2\log 2 + 2 \text{JS}D(P_{data}(x) || P_G(x)) = 0 \]

Reason 1. Approximate by sampling

Weaken your discriminator?
Can weak discriminator compute JS divergence?
Discriminator

\[ V = E_{x \sim P_{data}}[logD(x)] + E_{x \sim P_G}[log(1 - D(x))]. \]

\[ \approx \frac{1}{m} \sum_{i=1}^{m} logD(x^i) + \frac{1}{m} \sum_{i=1}^{m} log \left( 1 - D(\tilde{x}^i) \right) \]

max \_ V(G, D) = -2\log2 + 2\text{JS}D(P_{data}(x) || P_G(x)) = 0

Reason 2. the nature of data

Both \( P_{data}(x) \) and \( P_G(x) \) are low-dim manifold in high-dim space

Usually they do not have any overlap
Evaluation

http://www.guokr.com/post/773890/

Better
Evaluation

\[ J_S(P_{G_1} \mid \mid P_{data}) = \log 2 \]

\[ J_S(P_{G_2} \mid \mid P_{data}) = \log 2 \]

Not really better ......
Add Noise

• Add some artificial noise to the inputs of discriminator
• Make the labels noisy for the discriminator

Discriminator cannot perfectly separate real and generated data

\[ P_{data}(x) \] and \[ P_G(x) \] have some overlap

Noises decay over time
Mode Collapse
Mode Collapse

Generated Distribution

Data Distribution
Mode Collapse

What we want ...

In reality ...

$P_{data}$
Flaw in Optimization?

\[ KL = \int P_{data} \log \frac{P_{data}}{P_G} \, dx \]

Reverse \( KL = \int P_G \log \frac{P_G}{P_{data}} \, dx \)

Maximum likelihood (minimize \( KL(P_{data} \| P_G) \))

Minimize \( KL(P_G \| P_{data}) \)

This may not be the reason (based on Ian Goodfellow’s tutorial)
So many GANs ……

<table>
<thead>
<tr>
<th>Modifying the Optimization of GAN</th>
<th>Different Structure from the Original GAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>fGAN</td>
<td>Conditional GAN</td>
</tr>
<tr>
<td>WGAN</td>
<td>Semi-supervised GAN</td>
</tr>
<tr>
<td>Least-square GAN</td>
<td>InfoGAN</td>
</tr>
<tr>
<td>Loss Sensitive GAN</td>
<td>BiGAN</td>
</tr>
<tr>
<td>Energy-based GAN</td>
<td>Cycle GAN</td>
</tr>
<tr>
<td>Boundary-seeking GAN</td>
<td>Disco GAN</td>
</tr>
<tr>
<td>Unroll GAN</td>
<td>VAE-GAN</td>
</tr>
<tr>
<td>……</td>
<td>……</td>
</tr>
</tbody>
</table>
Conditional GAN
Motivation


Motivation

• Challenge

Text $c$  $\rightarrow$ NN  $\rightarrow$ Image $x$

(a point, not a distribution)

Text: “train”

NN output
Conditional GAN

Training data: \((\hat{c}, \hat{x})\)

Prior distribution \(z\)

Learn to approximate \(P(x|c)\)

\((\hat{c}, x = G(\hat{c}))\) classified as positive

Learn to ignore this term ...

Can generated \(x\) not related to \(c\)

Positive example: \((\hat{c}, \hat{x})\)

Negative example: \((\hat{c}, G(\hat{c})), (\hat{c}', \hat{x})\)
<table>
<thead>
<tr>
<th>Caption</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>a pitcher is about to throw the ball to the batter</td>
<td>![Images of a pitcher throwing a baseball]</td>
</tr>
<tr>
<td>a group of people on skis stand in the snow</td>
<td>![Images of a group of people on skis in snow]</td>
</tr>
<tr>
<td>a man in a wet suit riding a surfboard on a wave</td>
<td>![Images of a man surfing]</td>
</tr>
</tbody>
</table>
Text to Image - Results

"red flower with black center"

<table>
<thead>
<tr>
<th>Caption</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>this flower has white petals and a yellow stamen</td>
<td><img src="image1.png" alt="Images of white flowers with a yellow stamen" /></td>
</tr>
<tr>
<td>the center is yellow surrounded by wavy dark purple petals</td>
<td><img src="image2.png" alt="Images of purple flowers with wavy petals" /></td>
</tr>
<tr>
<td>this flower has lots of small round pink petals</td>
<td><img src="image3.png" alt="Images of pink flowers with small round petals" /></td>
</tr>
</tbody>
</table>
Image-to-image Translation

Positive examples

Real or fake pair?

G tries to synthesize fake images that fool D

D tries to identify the fakes

Negative examples
Image-to-image Translation - Results
Speech Enhancement GAN

https://arxiv.org/abs/1703.09452
Speech Enhancement GAN

Using Least-square GAN
Least-square GAN

• For discriminator

\[
\min_D \frac{1}{2} E_{x\sim P_{\text{data}}} [(D(x) - b)^2] + \frac{1}{2} E_{x\sim P_G} [(D(x) - a)^2]
\]

D has linear output

• For Generator

\[
\min_D \frac{1}{2} E_{z\sim P_{\text{data}}} [(D(G(z)) - c)^2]
\]
Least-square GAN

- The code used in demo from:
  - https://github.com/osh/KerasGAN/blob/master/MNIST_CNN_GAN_v2.ipynb