Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
3. GAN architectures

Drawing? => learning from examples
Review: Auto-encoder

As close as possible

Minimize reconstruction error

Randomly generate a vector as code

Image ?
Review: Auto-encoder

NN Decoder → code → NN Decoder → Image

-1.5 → 0

[−1.5, 0]

1.5 → 0

[1.5, 0]
Review: Auto-encoder
Auto-encoder

From a normal distribution $N(0,1)$

$c_i = \sigma_i e_i + m_i$

Problems of AE/VAE

- It does not really try to simulate real images.

```
code
```

![Diagram showing the process of generating images from a code through a neural network decoder and output. The diagram illustrates the difference between realistic and non-realistic images, with one pixel differences from the target shown.]

One pixel difference from the target

Realistic

Non Realistic
Problems of AE/VAE

GAN to tackle this pb:

Realistic

Non Realistic

GAN: generative adversarial networks

Game scenario:

Player1, Generator, produces samples
Player2, – Its adversary Discriminator, attempts to distinguish real samples from fake generated ones (produced by Player1)!

Player1 aims at producing Realistic images to fool the Player2
Generative models

Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
   • GAN Algorithm
Adversarial Nets Framework

Game scenario:

Player1, Generator $G$
Player2, Discriminator $D$

$$V(G, D) = \mathbb{E}_{x \sim P_{data}}[\log D(x)] + \mathbb{E}_{x \sim P_G}[\log (1 - D(x))]$$

$$G^* = \arg \min_G \max_D V(G, D)$$
GAN Learning – D and G updates

Game scenario:

**Player1, Generator G,** produces samples.

**Player2,** its adversary **Discriminator D,** attempts to distinguish **real** samples from **fake** generated ones (produced by Player1)!

Player1 aims at producing **Realistic** images to fool the Player2.
GAN - Discriminator

Randomly sample a vector

Generator v1

NN

Real images:

1/0 (real or fake)

Discriminator Optimization on a batch of images:
Using gradient descent to update the parameters in the discriminator, with a fixed generator
GAN - Generator

Updating the parameters of generator

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

Generator + Discriminator = a network

Optimization:

Using gradient descent to update the parameters in the generator, but fixing the discriminator
GAN Learning – D and G updates

Real images: 5 0 4 1

Game over: Winner==Player1
Generator G producing fully realistic images that fool the Player2
Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, \( k \), is a hyperparameter. We used \( k = 1 \), the least expensive option, in our experiments.

\begin{algorithm}
\begin{algorithmic}
\For {number of training iterations}
\For {\( k \) steps}
\State Sample minibatch of \( m \) noise samples \( \{ z^{(1)}, \ldots, z^{(m)} \} \) from noise prior \( p_g(z) \).
\State Sample minibatch of \( m \) examples \( \{ x^{(1)}, \ldots, x^{(m)} \} \) from data generating distribution \( p_{data}(x) \).
\State Update the discriminator by ascending its stochastic gradient:
\[
\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D\left(x^{(i)}\right) + \log \left( 1 - D\left(G\left(z^{(i)}\right)\right) \right) \right].
\]
\EndFor
\State Sample minibatch of \( m \) noise samples \( \{ z^{(1)}, \ldots, z^{(m)} \} \) from noise prior \( p_g(z) \).
\State Update the generator by descending its stochastic gradient:
\[
\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log \left( 1 - D\left(G\left(z^{(i)}\right)\right) \right).
\]
\EndFor
\end{algorithmic}
\end{algorithm}

**GAN algorithm**

\[
V = \mathbb{E}_{x \sim p_{data}} \left[ \log D(x) \right] + \mathbb{E}_{x \sim p_G} \left[ \log (1 - D(x)) \right]
\]

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

Source of images: https://zhuanlan.zhihu.com/p/24767059
DCGAN: https://github.com/carpedm20/DCGAN-tensorflow
GAN

100 rounds
GAN

1000 rounds
GAN

50,000 rounds
Generative models
Outline

1. Preview: Auto-Encoders, VAE
2. Generative models with GAN
   • GAN Algorithm
   • KL vs. Jensen Shannon Divergence

\[ V(G, D) = \mathbb{E}_{x \sim P_{data}}[\log D(x)] + \mathbb{E}_{x \sim P_{G}}[\log (1 - D(x))] \]
\[ G^* = \arg \min_{G} \max_{D} V(G, D) \]
Which measure to evaluate how $P_G(x; \theta)$ is close to $P_{data}(x)$ in Maximum Likelihood optimization?

• 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 $\{x^1, x^2, ..., x^m\}$ 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
Which measure to evaluate how \( P_G(x; \theta) \) is close to \( P_{\text{data}}(x) \) in Maximum Likelihood optimization?

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

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

\[
\approx \arg \max_\theta \mathbb{E}_{x \sim P_{\text{data}}} [\log P_G(x; \theta)]
\]

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

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

In Maximum Likelihood it is a KLD Kullback Leibler Divergence.
If $P_G(x; \theta)$ is a coming with a NN

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

It is difficult to compute the likelihood.

Credits: https://blog.openai.com/generative-models/
Basic Idea of GAN: the 2 players G-D game

- **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 (and $P_{\text{prior}}$)

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

- **Global objective function** $V(G, D)$

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

\[ G^* = \arg \min_G \max_D \mathbb{E} \]
\[
\max_D V(G, D) \quad G^* = \arg \min_G \max_D V(G, D)
\]

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

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

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

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

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

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

- Find \(D^*\) maximizing:

\[
f(D) = a \log(D) + b \log(1 - D)
\]

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

\[
a \times \frac{1}{D^*} = b \times \frac{1}{1 - D^*}
\]

\[
a \times (1 - D^*) = b \times D^*
\]

\[
a - aD^* = bD^*
\]

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

\[
D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} < 1
\]

\[0 < \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} < 1\]
\[
\max_D V(G, D) \quad G^* = \arg \min_G \max_D V(G, D)
\]

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

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

“difference” between \(P_{G_1}\) and \(P_{data}\)
\[
\max_D V(G, D) = V(G, D^*) = \mathbb{E}_{x \sim P_{data}} \left[ \log \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} \right] + \mathbb{E}_{x \sim P_G} \left[ \log \frac{P_G(x)}{P_{data}(x) + P_G(x)} \right]
\]

\[
\Rightarrow +2 \log \frac{1}{2} = -2 \log 2
\]
\[
\max_D V(G, D) = V(G, D^*) = \\
\quad -2\log 2 + \int_x P_{\text{data}}(x) \log \frac{P_{\text{data}}(x)}{(P_{\text{data}}(x) + P_G(x))/2} dx \\
\quad + \int_x P_G(x) \log \frac{P_G(x)}{(P_{\text{data}}(x) + P_G(x))/2} dx \\
= -2\log 2 + \text{KL} \left( P_{\text{data}}(x) \| \frac{P_{\text{data}}(x) + P_G(x)}{2} \right) \\
\quad + \text{KL} \left( P_G(x) \| \frac{P_{\text{data}}(x) + P_G(x)}{2} \right) \\
= -2\log 2 + 2\text{JSD} \left( P_{\text{data}}(x) \| P_G(x) \right) \quad \text{Jensen-Shannon divergence}
\]

\[
\text{JSD}(P||Q) = \frac{1}{2} \text{KL}(P||M) + \frac{1}{2} \text{KL}(Q||M) \\
M = \frac{1}{2} (P + Q)
\]
• Generator G, Discriminator D
• Looking for G* such that \( G^* = \arg \min_G \max_D V(G, D) \)
• Given G, \( \max_D V(G, D) = -2\log 2 + 2JS\text{D}(P_{data}(x)||P_G(x)) \)
• What is the optimal G?
  \[
P_G(x) = P_{data}(x)
\]
  with/using the JS\((P_G, P_{data})\) Divergence
  (In Maximum Likelihood it is a KL Divergence)

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