Energy-based GAN

Hung-yi Lee
Original Idea

• Discriminator leads the generator

Is it the only explanation of GAN?
Original GAN

The discriminator is flat in the end.

Source: https://www.youtube.com/watch?v=ebMei6bYeWw (credit: Benjamin Striner)
Evaluation Function

• We want to find an evaluation function $F(x)$
  • Input: object $x$, output: scalar $F(x)$ (how “good” the object is)
  • E.g. $x$ are images
    • Real $x$ has high $F(x)$
    • $F(x)$ can be a network
  • We can generate good $x$ by $F(x)$:
    • Find $x$ with large $F(x)$
  • How to find $F(x)$?

In practice, you cannot decrease all the $x$ other than real data.
Evaluation Function
- Structured Perceptron

- **Input:** training data set \( \{(x^1, \hat{y}^1), (x^2, \hat{y}^2), \ldots, (x^r, \hat{y}^r), \ldots\} \)
- **Output:** weight vector \( w \)
- **Algorithm:** Initialize \( w = 0 \)
  
  \[
  F(x, y) = w \cdot \phi(x, y)
  \]
  
  - do
  - For each pair of training example \( (x^r, \hat{y}^r) \)
  - Find the label \( \tilde{y}^r \) maximizing \( F(x^r, y) \)
  
  \[
  \tilde{y}^r = \arg\max_{y \in Y} F(x^r, y)
  \]
  
  - If \( \tilde{y}^r \neq \hat{y}^r \), update \( w \)
  
  Increase \( F(x^r, \hat{y}^r) \), decrease \( F(x^r, \tilde{y}^r) \)

  \[
  w \rightarrow w + \phi(x^r, \hat{y}^r) - \phi(x^r, \tilde{y}^r)
  \]

  - until \( w \) is not updated  

  **We are done!**
How about GAN?

- Generator is an intelligent way to find the negative examples.

“Experience replay”, parameters from last iteration

In the end ......
$P_{\text{data}} = \text{“origin”}$

$G = \text{1 hidden layer (100)}$

$D = \text{1 hidden layer (100)}$
100 iterations on G

100 iterations on D
$P_{data} = \text{"line"}$

100 iterations on $D$

$G = 1$ hidden layer (100)
$D = 1$ hidden layer (100)

$G = 2$ hidden layer (100)
$D = 1$ hidden layer (100)

$G = 1$ hidden layer (100)
$D = 2$ hidden layer (100)
$P_{\text{data}} = 1$-D Gaussian

100 iterations on D

G = 2 hidden layer (100)
D = 1 hidden layer (100)
Energy-based GAN (EBGAN)

• Viewing the discriminator as an energy function (negative evaluation function)
• Auto-encoder as discriminator (energy function)
• Loss function with margin for discriminator training
• Generate reasonable-looking images from the ImageNet dataset at 256 x 256 pixel resolution
  • without a multiscale approach

Sample real example $x$
Sample code $z$ from prior distribution
Update discriminator $D$ to minimize

$$L_D(x, z) = D(x) + \max\left(0, m - D(G(z))\right)$$

Sample code $z$ from prior distribution
Update generator $G$ to minimize

$$L_G(z) = D(G(z))$$
EBGAN

Discriminator D:
\[
L_D(x, z) = D(x) + \max \left( 0, m - D(G(z)) \right)
\]

Generator G:
\[
L_G(z) = D(G(z))
\]

Hard to reconstruct, easy to destroy
EBGAN

Discriminator D:
\[ L_D(x, z) = D(x) + \max(0, m - D(G(z))) \]

Generator G:
\[ L_G(z) = D(G(z)) \]

For auto-encoder, the region for low value is limited.

What would happen if \( x \) and \( G(z) \) have the same distribution?

\( \gamma \) is a value between 0 and \( m \)

\( \gamma \) is a value between 0 and \( m \)
More about EBGAN

• Pulling-away term for training generator

Given a batch $S = \{\cdots x_i \cdots x_j \cdots \}$ from generator

$$f_{PT}(S) = \sum_{i,j,i \neq j} \cos(e_i, e_j)$$

To increase diversity

• Better way to learn auto-encoder?
  • If auto-encoder only learns to minimize the reconstruction error of real images
    • Can obtain nearly identity function (not properly designed structure)
  • Giving larger reconstruction error for fake images regularized auto-encoder
Margin Adaptation GAN (MAGAN)

\[ L_D(x, z) = D(x) + \max\left(0, m - D(G(z))\right) \]

- Dynamic margin \( m \)
  - As the generator generates better images
  - The margin becomes smaller and smaller
Loss-sensitive GAN (LSGAN)

- LSGAN allows the generator to focus on improving poor data points that are far apart from real examples.
- Connecting LSGAN with WGAN

**LSGAN**

Assuming $D(x)$ is the *energy function*

Discriminator minimizing:

$$D(x) + \max \left( 0, \Delta(x, G(z)) + D(x) - D(G(z)) \right)$$
**LSGAN** Assuming $D(x)$ is the *energy function*

Discriminator minimizing:

$$D(x) + \max \left(0, \Delta(x, G(z)) + D(x) - D(G(z))\right)$$
\[ F(x^n, \hat{y}^n) \geq F(x^n, y) \]

\[ F(x^n, \hat{y}^n) - F(x^n, y) \geq \Delta(\hat{y}^n, y) \]
Boundary Equilibrium Generative Adversarial Networks (BEGAN)


• Auto-encoder based GAN
For discriminator: \(L_D = D(x) - k_t D(G(z))\)

For generator: \(L_G = D(G(z))\)

For each training step \(t\):
\[k_{t+1} = k_t + \lambda \left(\gamma D(x) - D(G(z))\right)\]

\(k_t\) increase

If \(\gamma D(x) > D(G(z))\), then \(\frac{D(G(z))}{D(x)} < \gamma\)
BEGAN

\[ \frac{D(G(z))}{D(x)} < \gamma \]

For discriminator: \[ L_D = D(x) - k_t D(G(z)) \]

For generator: \[ L_G = D(G(z)) \]

For each training step \( t \):

\[ k_{t+1} = k_t + \lambda \left( \gamma D(x) - D(G(z)) \right) \]
陳柏文 (大四) 提供實驗結果 (using CelebA)