Introduction to the Generative Adversarial Networks (GAN)

Blaise Hanczar

ibiSc
Generative Adversarial Network (GAN)

• Neural network that generates data from noise that looks real
• Does not look for an explicit distribution of the real data
• One of the most popular methods in deep learning (2014)
Data Generation

Original Brain MR Images

Realistic Tumors in Random Locations

(GAN) Generate

Realistic Tumors with Desired Size/Location by Adding Conditioning

Generate (Conditional GAN)

Synthetic Images for Data Augmentation

T1  T1c
T2  FLAIR

Synthetic Images for Physician Training
Image colorization
Image Editing
3D object generation
Generative Adversarial Networks
Generative Models

- Does not look for an explicit distribution of the real data
- Generate data from noise (random distribution)
- Learn a neural network that projects noise into the real data distribution
Generative Models

Objective: $P_{\text{data}} = P_{\text{model}}$

Learning: Minimizing the distance between $P_{\text{data}}$ and $P_{\text{model}}$
Generative Adversarial Networks

$P_z \xrightarrow{} z \xrightarrow{} G \xrightarrow{} G(z) \xrightarrow{} P_{\text{model}}$ 

$P_{\text{data}} \xrightarrow{} \text{real images} \xrightarrow{} X$ 

$D \xrightarrow{} \text{Real} \xrightarrow{} \text{Fake} \xrightarrow{} \text{loss}$
Generative Adversarial Networks

\[
\max_D \min_G \left[ \mathbb{E}_{x \sim p_{data}} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right]
\]

Output of the discriminator for real data
Output of the discriminator for artificial data
Training Discriminator

\[ \max_D \left[ \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \]

Output of the discriminator for real data

Output of the discriminator for artificial data
Training Generator

\[
min_G \left[ \mathbb{E}_{x \sim p_{data}} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right]
\]

Output of the discriminator for real data

Output of the discriminator for artificial data
GAN’s formulation

A 2-players game, where:

- The discriminator is trying to maximize $L(D, G)$
- The generator is trying to minimize $L(D, G)$

$$(D^*, G^*) = \max_D \min_G L(D, G)$$

$$L(D, G) = \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right]$$

The Nash equilibrium of this particular game is achieved at:

- $P_{data} = P_{model}$
- $D(x) = \frac{1}{2} \ \forall x$
GAN’s loss function

\[
L(D, G) = \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \\
= \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{x \sim p_m} \log(1 - D(x))
\]

• For a given G, what is the optimal D?

\[
D^* = \text{argmax}_D (L(D, G))
\]

\[
\frac{\partial L(D, G)}{\partial D(x)} = 0 \implies D^*(x) = \frac{p_d(x)}{p_d(x) + p_m(x)}
\]

Bayes classifier
GAN’s loss function

• Given $D^*$, what is the optimal $G$?
  
  $G^* = \arg\min_D (L(D^*, G))$

\[
L = \mathbb{E}_{x \sim p_d} \log \frac{p_m(x)}{\frac{1}{2} (p_d(x) + p_m(x))} + \mathbb{E}_{x \sim p_m} \log \frac{p_d(x)}{\frac{1}{2} (p_d(x) + p_m(x))} - 2 \log 2
\]

\[
L = \frac{1}{2} KL(p_m \| \frac{p_d + p_m}{2}) + \frac{1}{2} KL(p_d \| \frac{p_d + p_m}{2}) - 2 \log 2
\]

\[
L = 2 JS(p_m, p_d) - 2 \log 2
\]

Kullback-Leibler divergence:

\[
KL(p_1, p_2) = \mathbb{E}_{x \sim p_1} \log \frac{p_1(x)}{p_2(x)}
\]

Jensen-Shannon divergence:

\[
JS(p_1, p_2) = \frac{1}{2} KL(p_1, \frac{p_1 + p_2}{2}) + \frac{1}{2} KL(p_2, \frac{p_1 + p_2}{2})
\]
GAN Training

Objective: \( \max_D \min_G \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \)

Alternate the training of the two networks:

1. Discriminator: Gradient ascent
   \[ \max_D \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \]

2. Generator: Gradient descent
   \[ \min_G \left[ \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \]
Apprentissage GAN : un jeu à deux joueurs

Objective :  \( \max_D \min_G \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \)

Alternate the training of the two networks :
1. Discriminator : Gradient ascent
   \[
   \max_D \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right]
   \]
   Generator is good
   High gradient

2. Generator : Gradient descent
   \[
   \min_G \left[ \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right]
   \]
   Generator is bad
   Low gradient
Apprentissage GAN : un jeu à deux joueurs

Objective :  \( \max_D \min_G \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \)

Alternate the training of the two networks :

1. Discriminator : Gradient ascent
   \[
   \max_D \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right]
   \]

2. Generator : Gradient ascent
   \[
   \min_G \left[ \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \\
   \max_G \left[ \mathbb{E}_{z \sim p_z} \log(D(G(z))) \right]
   \]
Apprentissage GAN : un jeu à deux joueurs

Objective :  \( \max_D \min_G \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \)

Alternate the training of the two networks :
1. Discriminator : Gradient ascent
   \[ \max_D \left[ \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \]
2. Generator : Gradient ascent
   \[ \min_G \left[ \mathbb{E}_{z \sim p_z} \log(1 - D(G(z))) \right] \]
   \[ \max_G \left[ \mathbb{E}_{z \sim p_z} \log(D(G(z))) \right] \]
Unstable training

Unstable training caused by JS divergence

- With an optimal discriminator
- If no overlap between $P_m$ and $P_d$
  - $JS(p_m, p_d) = \log 2$
  - $L = 2JS(p_m, p_d) - 2\log 2 = 0$
  - Gradient is null
  - No training on $G$

Kullback-Leiber divergence:

$$KL(p_1, p_2) = \mathbb{E}_{x \sim p_1} \log \frac{p_1(x)}{p_2(x)}$$

Jensen-Shannon divergence:

$$JS(p_m, p_d) = \frac{1}{2}KL\left(\frac{p_m + p_d}{2}\right) + \frac{1}{2}KL\left(\frac{p_m + p_d}{2}\right)$$

$JS(p_m, p_d)^* = 0.693$

$JS(p_m, p_d)^* = 0.336$
Dropping mode

• With $D^*$, $KL(p_m||p_d)$ can be reformulated as:
  $$KL(p_m||p_d) = \mathbb{E}_{x \sim p_m} \log(1 - D^*(x)) - \mathbb{E}_{x \sim p_m} \log D^*(x)$$

• $G$ tries to minimize
  $$-\mathbb{E}_{x \sim p_m} \log D^*(x)$$
  $$-\mathbb{E}_{x \sim p_m} \log D^*(x) = KL(p_m||p_d) - \mathbb{E}_{x \sim p_m} \log(1 - D^*(x))$$

• In introducing the $L = 2JS(p_m||p_d) - 2\log 2$
  $$-\mathbb{E}_{x \sim p_m} \log D^*(x) = KL(p_m||p_d) - 2JS(p_m||p_d) - 2\log 2 + \mathbb{E}_{x \sim p_d} \log D^*(x)$$

• $G$ tries both to minimize $KL(p_m||p_d)$ and to maximize $JS(p_m||p_d)$
Mode collapse

• KL divergence is an unsymmetrical distribution measure

\[ \text{if } p_m(x) \to 0, p_d(x) \to 1 \quad \text{then} \quad KL(p_m||p_d) \to 0 \]
  • G does not produce very plausible data, tiny penalization
    ➢ Generated data lack the diversity

\[ \text{if } p_m(x) \to 1, p_d(x) \to 0 \quad \text{then} \quad KL(p_m||p_d) \to +\infty \]
  • G produces implausible data, large penalization
    ➢ Generated data are not accurate

• G will prefer the first case
Mode Collapse: Many $z$ to $\sim$one $x$

- Latent $z$
- Generated $G(z)$
- Real data $x$

Mode Dropping: Real Data not in $G(z)$

- Mode collapse and Mode Dropping can co-occur
- Complete training collapse is a separate phenomenon, for which extreme Mode Dropping and/or Mode Collapse are often symptoms

Mode Collapse on CelebA (Source: Geometric GAN)

Mode Dropping on MNIST (Source: TripletGAN)
Nombreuses variantes
Loss Variant GAN
Wasserstein GAN (WGAN)

- Wasserstein distance computes the distance between 2 distributions based on the optimal transport theory

\[ W(p_d, p_m) = \inf_{\gamma \in \Pi(p_d, p_m)} \mathbb{E}_{(x,y) \sim \gamma} \|x - y\| \]
Wasserstein GAN (WGAN)

• Wasserstein distance computes the distance between 2 distributions based on the optimal transport theory
  \[ W(p_d, p_m) = \inf_{\gamma \in \Pi(p_d, p_m)} \mathbb{E}_{x,y \sim \gamma} ||x - y|| \]

• W distance is intractable! But can be estimated
• The discriminator is used to estimate the W distance
  - \( D_w \) : regression problem, sigmoid output layer
  \[ L_G = -\mathbb{E}_{z \sim p_z} \log(D_w(G(z))) \]
Spectral Normalization GAN (SN-GAN)

• Weight normalization to stabilize the training of D
• D should be a K-Lipshitz continuous function

\[ \tilde{W}_{SN}(W) = \frac{W}{\sigma(W)} \]

With \( W \) the weights matrix of D
\( \sigma(W) \) the \( L_2 \) matrix norm of \( W \)

• Computationally light and easily applied to other GAN
Architecture variant GAN
Progressive GAN (PROGAN)

Training time: 0 days
4x4 resolution

- $z = \text{random code}$
- $x = \text{real image}$
- $x' = \text{generated image}$
Self Attention GAN (SAGAN)
Conditional GAN (cGAN)

\[
L(D, G) = \left[ \mathbb{E}_{x \sim p_{data}} \log D(x|y) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z|y))) \right]
\]
cGAN for image translation (Pix2Pix)

\[ L(D, G) = \left[ \mathbb{E}_{x \sim p_{data}} \log D(x, y) + \mathbb{E}_{z \sim p_z} \log (1 - D(G(z|y), y)) \right] \]
cGAN for image translation (Pix2Pix)
The small bird has a red head with feathers that fade from red to gray from head to tail.

This bird is black with green and has a very short beak.
Cycle GAN

\[ \min_{G_{A \to B}, G_{B \to A}} \max_{D_{A}, D_{B}} \left( L_{GAN}(G_{A \to B}) + L_{GAN}(G_{B \to A}) \right) + \mathbb{E}_{x \sim p_{d_{A}}} \left\| G_{A \to B}(G_{B \to A}(x)) - x \right\| + \mathbb{E}_{x \sim p_{d_{B}}} \left\| G_{B \to A}(G_{A \to B}(x)) - x \right\| \]

Cycle consistency on domain A

Cycle consistency on domain B
Other applications

• Semi-supervised learning
• Anomaly detection
• Representation learning
• Model interpretation
• Transfer learning
• Security of predictive models