Generative Adversarial Networks (GAN)

Hanxiao Liu

April 17, 2017
Outline

- Implicit Generative Models
- GAN
- Variants
  - $f$-GAN
  - Wasserstein GAN
- Use cases
  - Conditional generation
  - SSL
Implicit Generative Models

Working with likelihood could be expensive

- Variational Autoencoder (variational inference)
- Boltzmann Machines (MCMC)
- PixelRNN (conditional prob.)

Representation learning may not require likelihood.
- Sometimes we are more interested in taking samples from \( p(x) \) instead of \( p \) itself.

More discussions: [Mohamed and Lakshminarayanan, 2016]
advocate/penalize samples within the blue/white region.
GAN

Unsupervised learning via supervised learning

Diagram:
- Data sample
- Discriminator
- Yes / No
- Noise
- Generator
- Generator sample

Diagram flow:
- Data sample to discriminator
- Discriminator to data sample?
- Yes / No
- Noise to generator
- Generator to generator sample
GAN

Proposed by [Goodfellow et al., 2014]

A minimax game

\[
J(D) = -\frac{1}{2} \mathbb{E}_{x \sim \text{data}} \log D(x) - \frac{1}{2} \mathbb{E}_z \log (1 - D(G(z))) \tag{1}
\]

\[
J(G) = -J(D) \tag{2}
\]

- The optimal discriminator \( D^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_{model}(x)} \).
- In this case, \( J(G) = 2D_{JS}(p_{data} \parallel p_{model}) + \text{const.} \).
- Jenson-Shannon divergence:
  \[
  D_{JS}(p \parallel q) = \frac{1}{2} D_{KL}(p \parallel \frac{p+q}{2}) + \frac{1}{2} D_{KL}(q \parallel \frac{p+q}{2}).
  \]
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.

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

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.
Visualizing the GAN training process

dots: data.
blue: discriminator.
green: generator.
Deep Convolutional GAN [Radford et al., 2015]
Walk around the data manifold

Vector Arithmetic

Results of doing the same arithmetic in pixel space
Can we simply optimize the divergence?

\[ D_f(p\|q) = \int q(x) f \left( \frac{p(x)}{q(x)} \right) dx \]  \hspace{1cm} (3)

- recovers KL divergence when \( f(u) = u \log u \).
- optimization \( \min_q D_f(p\|q) \) can hardly be carried out with \( D_f \) in its original form.
Fenchel conjugacy: $f(u) = \sup_t tu - f^*(t)$

$$\min_q D_f(P\|Q) = \min_q \int q(x) \left[ \sup_t \left( t \frac{p(x)}{q(x)} - f^*(t) \right) \right] dx$$  \hspace{1cm} (4)

$$\geq \min_q \sup_T \int p(x)T(x) - f^*(T(x))q(x)dx$$  \hspace{1cm} (5)

$$= \min_q \sup_T \mathbb{E}_pT(x) - \mathbb{E}_q f^*(T(x))$$ \hspace{1cm} (6)

$$\approx \min_w \max_\theta \mathbb{E}_pT_\theta(x) - \mathbb{E}_q \omega f^*(T_\theta(x))$$ \hspace{1cm} (7)

The above includes GAN as its special case.
Training GAN (finding the equilibrium) is hard.

- Gradient for $G$ will vanish when $D$ is very good.
- i.e. when $p_{data}$ and $p_{model}$ are very different.
  - usually true for high-dimensional data.
  - $f$-divergence may be ill-defined.
Wasserstein GAN

Replace the pointwise $f$-divergence with Wasserstein distance [Arjovsky et al., 2017].

\[
D_w(p\|q) = \inf_{\gamma \in \Pi(p,q)} \mathbb{E}_{(x,y) \sim \gamma} dist(x, y)
\]  

(9)

where $\Pi(p, q)$ is the set of all joint distributions $\gamma(x, y)$ which marginal distributions are respectively $p$ and $q$.

- well defined even if $p$ and $q$ have different support.
- leads to a very simple algorithm using a similar variational trick for the $f$-divergence.
Wasserstein GAN

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: $\alpha$, the learning rate. $c$, the clipping parameter. $m$, the batch size. $n_{\text{critic}}$, the number of iterations of the critic per generator iteration.

Require: $w_0$, initial critic parameters. $\theta_0$, initial generator’s parameters.

1: while $\theta$ has not converged do
2:    for $t = 0, \ldots, n_{\text{critic}}$ do
3:        Sample $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$ a batch from the real data.
4:        Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
5:        $g_w \leftarrow \nabla_w \left[ \frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)})) \right]$
6:        $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
7:        $w \leftarrow \text{clip}(w, -c, c)$
8:    end for
9:    Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
10:   $g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))$
11:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
12: end while
Conditional GAN

[Mirza and Osindero, 2014]
Conditional GAN

Image to image translation [Isola et al., 2016].
Semi-supervised Learning

- Data augmentation
- Regularization [Salimans et al., 2016]
- Categorical GAN [Springenberg, 2015]


