Generative Adversarial Networks (GANs)

Hsieh Yu-Guan

January 8, 2018

Theoretical Viewpoint

Traditional Approaches

Generative modeling based on maximum likelihood estimation: we want to minimize the Kullback-Leibler (KL) divergence between the unknown data distribution $\mathbb{P}_r$ and our generator's distribution $\mathbb{P}_g$,

$$KL(\mathbb{P}_r \| \mathbb{P}_g) = \int_{\mathcal{X}} P_r(x) \log \frac{P_r(x)}{P_g(x)} dx,$$

where P_r and P_g are respectively the densities of $\mathbb{P}_r$ and $\mathbb{P}_g$ supposing that they're continuous. This cost function has a unique minimum zero at $\mathbb{P}_r = \mathbb{P}_g$. However, this divergence is not symetrical between the two distributions.

Original Model

Min-max game with value function:

$$V(G, D) = \mathbb{E}_{x \sim \mathbb{P}_r}[\log D(x)] + \mathbb{E}_{z \sim \mathbb{P}_z}[\log(1 - D(G(x)))].$$

We want to find $\min_G \max_D V(G, D)$. If the discriminator is trained to its optimum, and we note

$$C(G) = \max_D V(G, D),$$

we have, with $\mathbb{P}_g$ the distribution of the samples $G(z)$ obtained when $z \sim \mathbb{P}_z$, and $\mathbb{P}_m$ the mixture with densities $(P_r + P_g)/2$,

$$C(G) = -\log(4) + KL(\mathbb{P}_r \| \mathbb{P}_m) + KL(\mathbb{P}_g \| \mathbb{P}_m).$$

We recognize in the previous expression the Jensen-Shannon (JS) divergence between the model's distribution and the data generating process:

$$C(G) = -\log(4) + 2 \cdot JS(\mathbb{P}_r \| \mathbb{P}_g).$$

The idea is thus to minimize the JS divergence between $\mathbb{P}_g$ and $\mathbb{P}_r$.

Flaws of this model: synchronization between D and G is insdispensable, the choice of architectures have great influence and rely on heuristics that are extremely sensitive to modifications (batch normalization, dropout, ...).

Least Squares GAN

Key idea: replace the sigmoid cross entropy loss function with the least squares loss function. We fix some $a - b$ coding scheme for the discriminator and a value c that G wants D to believe for fake data, and define:

$$\begin{aligned} V^1_{\text{LSGAN}}(G, D) &= \frac{1}{2}\mathbb{E}_{x\sim\mathbb{P}_r}[(D(x) - b)^2] + \frac{1}{2}\mathbb{E}_{z\sim\mathbb{P}_z}[(D(G(z)) - a)^2], \\ V^2_{\text{LSGAN}}(G, D) &= \frac{1}{2}\mathbb{E}_{z\sim\mathbb{P}_z}[(D(G(z)) - c)^2]. \end{aligned}$$

During training, we minimize V^1_{LSGAN} with respect to D and V^2_{LSGAN} with respect to G alternatively.

It can be easily proved that if $b - c = 1$ and $b - a = 2$, we are in fact minimizing the Pearson χ^2 divergence between $\mathbb{P}_d + \mathbb{P}_g$ and $2\mathbb{P}_g$:

$$\chi^2_{\text{Pearson}}(\mathbb{P}_d + \mathbb{P}_g \| 2\mathbb{P}_g) = \int_{\mathcal{X}} \frac{(2P_g(x) - (P_d(x) + P_g(x)))^2}{P_d(x) + P_g(x)} dx.$$

LSGAN penalizes samples lying a long way to the decision boundary. It can therefore 1. move the generated samples toward the decision boundary which should go across the manifold of real data and 2. generate more gradients when updating the generator.

Wasserstein GAN

Let us consider the Earth-Mover (EM) or Wasserstein distance between $\mathbb{P}_r$ and $\mathbb{P}_g$:

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma\in\Pi(\mathbb{P}_r,\mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|],$$

where $\Pi(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively $\mathbb{P}_r$ and $\mathbb{P}_g$.

When G is a feedforward neural network parameterized by some vector θ, we denote $\mathbb{P}_g$ by $\mathbb{P}_\theta$, then $W(\mathbb{P}_r, \mathbb{P}_\theta)$ is continuous everywhere, and differentiable almost everywhere. This is not the case for neither the JS divergence nor any KLs.

In fact, in general, the supports of $\mathbb{P}_r$ and $\mathbb{P}_g$ lie in two low dimensional manifolds that have a intersection of measure zero. In this case, the KL divergence is not defined and the JS divergence equals always $\log 2$ and doesn't provide any usable gradient information. This is why we shouldn't train the discriminator to opitmal for classic GANs.

We can further prove that the topology induced by Wasserstein distance is weaker than the one induced by the JS divergence. The EM distance is indeed a sensible cost function when learning distribution supported by low dimensional manifolds. We now want to design a procedure to minimize the EM distance between $\mathbb{P}_r$ and $\mathbb{P}_g$.

By the Kantorovich-Rubinstein duality, for any $K > 0$,

$$W(\mathbb{P}_r, \mathbb{P}_g) = \frac{1}{K}\left(\sup_{\|f\|_L \leq K} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_g}[f(x)]\right)$$

where the supremum is taken over all the K-Lipschitz functions $f : \mathcal{X} \to \mathbb{R}$.

For $\mathbb{P}_g = \mathbb{P}_\theta$ defined as above, the supremum is attained. Let f^* be a function that attains this supremum, we have

$$\nabla_\theta W(\mathbb{P}_r, \mathbb{P}_\theta) = -\mathbb{E}_{z\sim\mathbb{P}_z}[\nabla_\theta f^*(G(z))]$$

when both terms are well-defined.

In practice, f will be another neural network parameterized with weights w lying in a compact space $\mathcal{W}$. The fact that $\mathcal{W}$ is compact implies that all possible f will be K-Lipschitz for some K that only depends on $\mathcal{W}$ and not the individual weights. In order to have parameters w lie in a compact space, we can for example clamp the weights to a fixed box after each gradient update.

At each step, we train the 'critic' f to optimality and can thus get an estimate of the quantity $K \cdot W(\mathbb{P}_r, \mathbb{P}_g)$. This estimate correlates quite well with the quality of the generated samples. We then update the generator using the above formula.

In summary, the two main benefits of WGAN compared to normal GAN are: improved stability (no more mode collapse, can train the critic till optimality, can be used with more architectures) and meaningful loss metric.