Wasserstein GAN

Dingquan Wang
wdd@jhu.edu
What’s wrong with GANs?

• A careful balance discriminator and the generator
• Mode collapse: Low output diversity
• A careful design of the network
• No loss metric that correlates with the generator’s convergence and sample quality
• Unstable optimization process
Wasserstein GAN (WGAN)

- A careful balance discriminator and the generator
- Mode collapse: Low output diversity
- A careful design of the network architecture
- No loss metric that correlates with the generator’s convergence and sample quality
- Unstable of the optimization process
Takeaways of WGAN

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

\textbf{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.

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

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

When the discriminator successfully rejects generator samples with high confidence, the generator’s gradient vanishes. (Goodfellow 2016)

\[ L(D, g_{\theta}) = \mathbb{E}_{x \sim P_r}[\log D(x)] + \mathbb{E}_{x \sim P_g}[\log(1 - D(x))] \]

\[ D^*(x) = \frac{P_r(x)}{P_r(x) + P_g(x)} \]

\[ L(D^*, g_{\theta}) = 2JS(D(P_r || P_g) - 2 \log 2 \]
Jensen–Shannon divergence

\[ JS(D(P_r||P_g)) = \frac{1}{2} KL(P_r||P_A) + \frac{1}{2} KL(P_g||P_A) \]

\[ P_A = \frac{P_r + P_g}{2} \]

**Theorem 2.3.** Let \( P_r \) and \( P_g \) be two distributions whose support lies in two manifolds \( M \) and \( P \) that don’t have full dimension and don’t perfectly align. We further assume that \( P_r \) and \( P_g \) are continuous in their respective manifolds. Then,

\[ JS(D(P_r||P_g)) = \log 2 \]

\[ KL(P_r||P_g) = +\infty \]

\[ KL(P_g||P_r) = +\infty \]

**Lemma 2.** Let \( M \) and \( P \) be two regular submanifolds of \( \mathbb{R}^d \) that don’t have full dimension. Let \( \eta, \eta' \) be arbitrary independent continuous random variables. We therefore define the perturbed manifolds as \( \tilde{M} = M + \eta \) and \( \tilde{P} = P + \eta' \). Then

\[ P_{\eta, \eta'}(\tilde{M} \text{ does not perfectly align with } \tilde{P}) = 1 \]

\[ L(D^*, g_\theta) \] is almost a constant!
Learning parallel lines

Example 1 (Learning parallel lines). Let $Z \sim U[0, 1]$ the uniform distribution on the unit interval. Let $\mathbb{P}_0$ be the distribution of $(0, Z) \in \mathbb{R}^2$ (a 0 on the x-axis and the random variable $Z$ on the y-axis), uniform on a straight vertical line passing through the origin. Now let $g_\theta(z) = (\theta, z)$ with $\theta$ a single real parameter. It is easy to see that in this case,
Learning parallel lines

\[ JSD(P_0 || P_\theta) = \begin{cases} 
  \log 2 & \text{if } \theta \neq 0, \\
  0 & \text{if } \theta = 0,
\end{cases} \]

\[ KL(P_\theta || P_0) = KL(P_0 || P_\theta) = \begin{cases} 
  +\infty & \text{if } \theta \neq 0, \\
  0 & \text{if } \theta = 0,
\end{cases} \]
Wasserstein Distance

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

(1)

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 \( \mathbb{P}_g \). Intuitively, \( \gamma(x, y) \) indicates how much “mass” must be transported from \( x \) to \( y \) in order to transform the distributions \( \mathbb{P}_r \) into the distribution \( \mathbb{P}_g \). The EM distance then is the “cost” of the optimal transport plan.

\[ W(\mathbb{P}_0, \mathbb{P}_\theta) = |\theta| \]
Kantorovich-Rubinstein duality

\[ W(P_r, P_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_\theta}[f(x)] \]  

where the supremum is over all the 1-Lipschitz functions \( f : \mathcal{X} \to \mathbb{R} \). Note that if we replace \( \|f\|_L \leq 1 \) for \( \|f\|_L \leq K \) (consider \( K \)-Lipschitz for some constant \( K \)), then we end up with \( K \cdot W(P_r, P_\theta) \).

- \( f : R \to R \) is called \( K \)-Lipschitz continuous if there exists a positive real constant \( K \) such that, for all real \( x_1 \) and \( x_2 \),

\[ |f(x_1) - f(x_2)| \leq K|x_1 - x_2|. \]
Parameterize $f$

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim p(z)}[f_w(g_\theta(z))]$$

- K-Lipschitz?
  - Clip the weights to a fixed box (say $[-0.01, 0.01]$)

---

**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 $t = 0, ..., n_{\text{critic}}$ 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
Meaningful loss metric
Compare with DCGAN

*Figure 5: Algorithms trained with a DCGAN generator. Left: WGAN algorithm. Right: standard GAN formulation. Both algorithms produce high quality samples.*
Without Batch Normalization

Figure 6: Algorithms trained with a generator without batch normalization and constant number of filters at every layer (as opposed to duplicating them every time as in [17]). Aside from taking out batch normalization, the number of parameters is therefore reduced by a bit more than an order of magnitude. Left: WGAN algorithm. Right: standard GAN formulation. As we can see the standard GAN failed to learn while the WGAN still was able to produce samples.
With simpler architecture

Figure 7: Algorithms trained with an MLP generator with 4 layers and 512 units with ReLU nonlinearities. The number of parameters is similar to that of a DCGAN, but it lacks a strong inductive bias for image generation. Left: WGAN algorithm. Right: standard GAN formulation. The WGAN method still was able to produce samples, lower quality than the DCGAN, and of higher quality than the MLP of the standard GAN. Note the significant degree of mode collapse in the GAN MLP.
References

• Martin Arjovsky, Léon Bottou; Towards Principled Methods for Training Generative Adversarial Networks, ICLR2017

• Martin Arjovsky, Soumith Chintala, Léon Bottou; Wasserstein GAN, arXiv