Image Generation using Wasserstein Generative Adversarial Network

Zhankun Luo
Dept. of ECE
Purdue University Northwest
Hammond, IN, USA
luo333@pnw.edu

Andres Jara
Dept. of ECE
Purdue University Northwest
Hammond, IN, USA
ajaralom@pnw.edu

Wen Ou
Dept. of ECE
Purdue University Northwest
Hammond, IN, USA
ou21@pnw.edu

***Abstract*—GAN shows the capability to generate fake authentic images by evaluating and learning from real and fake samples. This paper introduces an alternative algorithm to the traditional DCGAN, named Wasserstein GAN (WGAN). It introduces the Wasserstein distance for the first time, and the improved model WGAN-GP added the gradient penalty in the loss function. The experimental results demonstrate an improvement in the learning stability comparing to DCGAN whereas WGAN-GP convergence speed outcompetes WGAN.**

***Index Terms*—GAN, Wasserstein Distance, Gradient Penalty**

I. Introduction

Generative adversarial network was firstly introduced in 2014 by Goodfellow *et al.* [1]. The model consists of one pair of neural networks, generator G and discriminator D, both trying to learn from each other simultaneously. While the generator is trained to produce fake samples as similar as possible to the real data, the discriminator is trained to be able to classify fake samples from real samples. Nevertheless, there are difficulties in the training that make the model fail. G tends to stuck when the same output is being produced with extremely low variety. The lack of a proper evaluation metric results in possible convergence failure. Such difficulties are improved with an alternative model named Wasserstein GAN (WGAN).

WGAN was proposed in 2017 by Arjovsky *et al.* [2]. This algorithm enhances one important aspect of GAN: training stability. WGAN uses a critic model that scores the "realness" or "fakeness" instead of a discriminator model (DCGAN) that predicts the probability of a sample as "real" or "fake". This conceptual shift comes from the motivation of using the Wasserstein distance (Earth-mover distance). It helps to overcome the gradient vanishing problem, thus strengthens the stability of the GAN model. Hence, the improved model WGAN-GP is introduced [3] where the gradient penalty is added to accelerate the convergence speed.

In this paper, an overview of the Wasserstein GAN model is discussed to provide a comprehensive analysis of Wasserstein distance in comparison to other probability distances and divergences. Afterwards, the WGAN algorithm is demonstrated along with the improved WGAN-GP algorithm. The experimental results are compared making use of WGAN and WGAN-GP on MNIST dataset and CIFAR-10 dataset.

II. Problem Description

There are a real image x and a fake image $G(z)$ generated by neural network G whose input is random noise z. The real image x and fake image $G(z)$ can be rearranged to be a vector. It is assumed that if the x and $G(z)$ are close enough, people will not able to identify the differences between the real image x and the generated fake image $G(z)$. Here the neural network is used D as the discriminator metric to separate real images x from fake images $G(z)$, that is, $D(x)$ and $D(G(z))$ respectively.

A. DCGAN

The discriminator is expected to identify between the real image x and the fake image $G(z)$, that is $D(x) \approx 1$ and $D(G(z)) \approx 0$ for the real image x and the fake image $G(z)$, and the range of $D(\cdot) \in (0, 1)$. If the cross entropy (1) is used to measure the similarity between the predicted value given by D and the true values, then it results in the following:

$$H(y, p) = \sum -y \log(p) - (1 - y) \log(1 - p) \tag{1}$$

Here, if x is a real image, the label is $y = 1$, and the predicted probability of real image for x is $p = D(x)$. The cross entropy becomes:

$$H(y, p) = \sum -\log(D(x)) \tag{2}$$

$G(z)$ is a fake image, the label is $y = 0$, and the predicted probability of fake image for $DG(z)$ is $1 - p = 1 - D(G(z))$, the cross entropy results in:

$$H(y, p) = \sum -\log(1 - D(G(z))) \tag{3}$$

Equation (4) is the sum of (2) and (3), $x^{(i)}$ and $G(z^{(i)})$ denote the i-th real image and the i-th fake image in batch size m samples respectively.

$$\text{loss}_D = \frac{1}{m} \sum_{i=1}^{m} -\log(D(x^{(i)})) - \log(1 - D(G(z^{(i)}))) \tag{4}$$

loss_D is known as the loos function and reaches the minimal point when $D(x_i) = 1$ and $D(G(z_i)) = 0$. If the generator G is fixed, WGAN is used to update the parameters of the discriminator, θ_d. Typically, the gradient-based optimization method is applied to minimize loss_D.

Since the discriminator cannot separate real x from fake images $G(z)$ by itself, that is $D(G(z)) \approx 1$, the goal is to maximize $D(G(z))$ to be equivalent to minimizing (5) or (6)

$$\text{loss}_G = \frac{1}{m}\sum_{i}^{m} -\log(D(G(z^{(i)}))) \tag{5}$$

$$\text{loss}_G = \frac{1}{m}\sum_{i}^{m} \log(1 - D(G(z^{(i)}))) \tag{6}$$

where $G(z^{(i)})$ denotes the i-th fake image in batch size m samples and loss_G reaches the minimal point when $D(G(z^{(i)})) = 1$. If the discriminator D is fixed, the algorithm can be used to update the parameters of generator θ_g. Once again, the gradient-based optimization method is used to minimize the loss function of the generator loss_G.

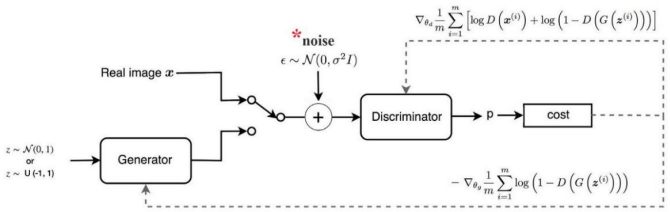

Fig. 1. DCGAN Training Process.

B. Problem of DCGAN

The expectation of the discriminator loss $\mathbb{E}[\text{loss}_D]$ is minimized by the following equation (7):

$$\mathbb{E}[\text{loss}_D] = \int [-p(x)\log(D(x)) - q(x)\log(1 - D(x))]\, dx \tag{7}$$

where $p(x)$ is the distribution of the real data x and $q(x')$ is the estimated distribution of the fake data $x' = G(z)$ from the generator. Notice that $p(x), q(x)$ are fixed values when the generator G is fixed. By setting the functional $\mathcal{J}(D) \equiv \mathbb{E}[\text{loss}_D]$, the variation of $\mathcal{J}(D)$ becomes:

$$\delta\mathcal{J}(D) = \int \left[-\frac{p(x)}{D(x)} + \frac{q(x)}{1 - D(x)}\right] \delta D dx = 0 \tag{8}$$

$$\left[-\frac{p(x)}{D(x)} + \frac{q(x)}{1 - D(x)}\right] = 0 \Rightarrow D(x) = \frac{p(x)}{p(x) + q(x)} \tag{9}$$

From equation (9), $\mathbb{E}[\log(D(x))]$ is a constant value. The $\mathbb{E}[\text{loss}_G]$ (6) is minimized. The sum of the expectation of the discriminator loss and $\mathbb{E}[\log(D(x))]$ is derived as follows:

$$\begin{aligned}
&\mathbb{E}[\text{loss}_G] + \mathbb{E}[\log(D(x))] \\
&= \int p(x)\log(D(x))dx + \int q(x')\log(1 - D(x'))dx' \\
&= \int \left[p\log\left(\frac{p}{\frac{1}{2}[p+q]}\right) + q\log\left(\frac{q}{\frac{1}{2}[p+q]}\right)\right] dx - 2\log(2) \\
&= 2D_{JS}(p||q) - 2\log(2)
\end{aligned} \tag{10}$$

When there is a huge gap increasing between p and q, the Jensen-Shannon divergence is $D_{JS}(p||q) = \log(2)$ presented in Fig. 3. Consequently, the gradient for $D_{JS}(p||q)$ will vanish, that is, the generator G learns nothing from the gradient-based optimization. (Fig.1-Fig.3 come from Jonathan Hui's blog [4])

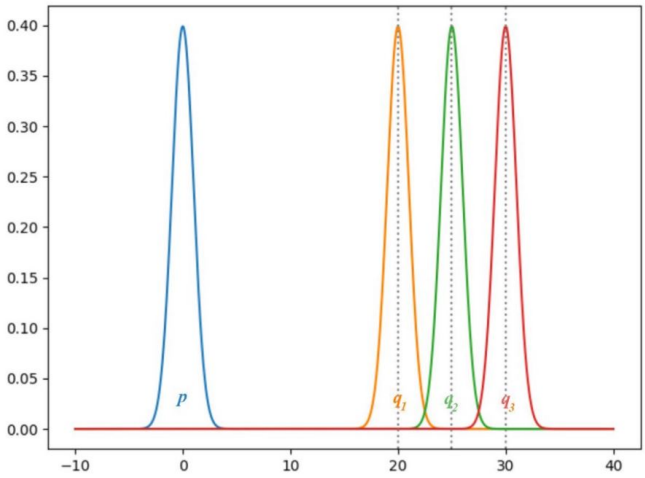

Fig. 2. Probability distribution of p and q.

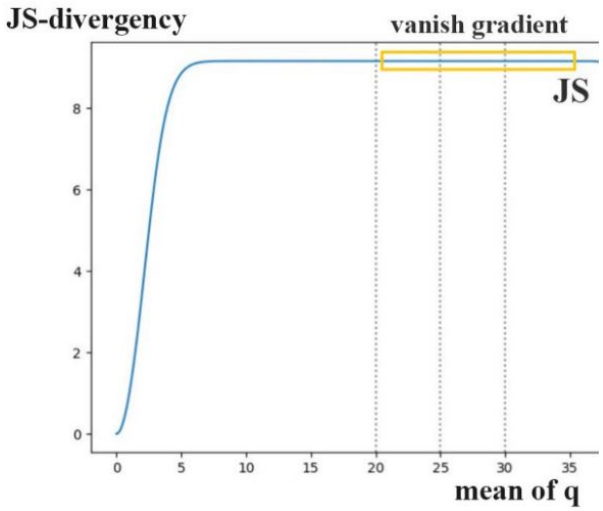

Fig. 3. The JS divergence for different q.

III. Algorithm of Wasserstein GAN

A. Wasserstein Distance

To solve the problem that G may learn nothing when D is well-trained, Arjovsky introduced the Wasserstein distance, whose gradient does not vanish when the distribution of real image $P_r(x)$ and the distribution of the fake image $P_g(y)$ are far away from each other. Arjovsky proved that Wasserstain

distance could be written as: [2].

$$
\begin{aligned}
W\left(P_r, P_g\right) &= \inf_{\gamma \sim \Pi\left(P_r, P_g\right)} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|] \\
&= \frac{1}{K} \sup_{\|f\|_L \leq K} \mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_g}[f(x)] \\
&\approx \frac{1}{K} \max_{w:|f_w|_L<K} \mathbb{E}_{x\sim P_r}\left[f_w(x)\right] - \mathbb{E}_{x\sim P_g}\left[f_w(x)\right]
\end{aligned} \tag{11}
$$

where γ is the joint distribution of P_r, P_g, that is, $\int \gamma(x,y)dy = P_r(x), \int \gamma(x,y)dx = P_g(y)$ holds for the joint probability distribution function $\gamma(x,y)$. $\|f\|_L \leq K$ means all the functions satisfy the Lipschitz constraint

$$
\left|f\left(x_1\right) - f\left(x_2\right)\right| \leq K\left|x_1 - x_2\right| \tag{12}
$$

In simple and informal words, the Wasserstein distance (also known as Earth Mover's distance) can be interpreted as finding the shortest path of transporting the distribution $P_r(x)$ to the distribution $P_g(y)$.

A subset of f that satisfies the Lipschitz constraint is chosen to find the maximal value and finally estimate the Wasserstein distance. Typically, neural networks are used to represent a family of functions f_w, where w represents the weights of neural networks.

$$
\begin{aligned}
|w| < c, |x| < M \Rightarrow \left|\frac{\partial f_w}{\partial x}\right| = |g(w,x)| < K \\
\Rightarrow \left|f\left(x_1\right) - f\left(x_2\right)\right| \leq K\left|x_1 - x_2\right|
\end{aligned} \tag{13}
$$

The exact value of K is not used. However, it ensures that f_w satisfies the Lipschitz constraint. Then, the best w^* is obtained to reach the Wasserstein distance $KW(P_r, P_g) = KW(p(x), q(x'))$ between the distributions $p(x)$ and $q(x')$.

$$
\begin{aligned}
w^* &= \operatorname*{argmax}_{|w|<c} \mathbb{E}_{x\sim P_r}\left[f_w(x)\right] - \mathbb{E}_{x\sim P_g}\left[f_w(x)\right] \\
&\approx \operatorname*{argmin}_{|w|<c} \frac{1}{m}\sum_{i=1}^{m} -f_w(x^{(i)}) + f_w(x'^{(i)})
\end{aligned} \tag{14}
$$

The estimation formula of the Wasserstein distance used for fixed w^* and $f_{w^*}(x)$

$$
\begin{aligned}
&\mathbb{E}_{x\sim P_r}\left[f_{w^*}(x)\right] - \mathbb{E}_{x\sim P_g}\left[f_{w^*}(x)\right] \\
&\approx \frac{1}{m}\sum_{i=1}^{m} f_{w^*}(x^{(i)}) - f_{w^*}(x'^{(i)})
\end{aligned} \tag{15}
$$

B. Wasserstein GAN

The discriminator D expects to maximize the "distance" between the real image x and the fake image $x' = g(z)$ when the generator G is fixed, that is, θ, g_θ. From (11), we know the Wasserstein distance is the super limit of $\mathbb{E}_{x\sim p(x)}[f(x)] - \mathbb{E}_{x'\sim q(x')}[f(x')]$. Thus, the loss function of D for minimization is determined.

$$
\begin{aligned}
\text{loss}_D &= \frac{1}{m}\sum_{i=1}^{m} -f_w(x^{(i)}) + f_w(g_\theta(z^{(i)})) \quad (|w| < c) \\
&\approx -\left(\mathbb{E}_{x\sim p(x)}[f_w(x)] - \mathbb{E}_{x'\sim q(x')}[f_w(x')]\right)
\end{aligned} \tag{16}
$$

$$
\begin{aligned}
w &\leftarrow -\nabla_w \text{loss}_D = \nabla_w \frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - f_w(g_\theta(z^{(i)})) \\
w &\leftarrow \text{clip}(w, -c, c)
\end{aligned} \tag{17}
$$

The generator G expects to minimize the distance between the distribution of real image x and the distribution of fake image x' when the discriminator D is fixed, that is, w, f_w are fixed. From (15), the Wasserstein distance can be estimated by $KW(p(x), q(x')) \approx \frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - f_w(x'^{(i)})$, where $f_w(x^{(i)})$ is a constant value when w, f_w are fixed. Thus, the loss function of G for minimization is found to be:

$$
\text{loss}_G = -\frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) \tag{18}
$$

$$
\theta \leftarrow -\nabla_\theta \text{loss}_G = \nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) \tag{19}
$$

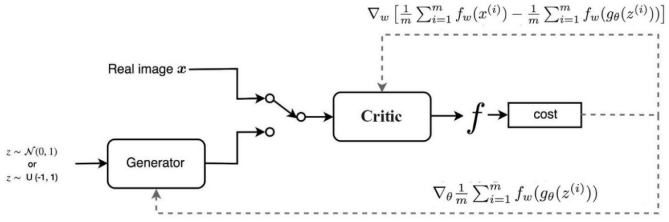

Fig. 4. WGAN training process.

C. Wasserstein GAN with Gradient Penalty

This model of WGAN is very sensitive to the hyper parameter c that ensures f_w satisfies the Lipschitz condition. When c is big, the convergence speed can be very slow, but when c is small, the gradient of w may vanish if the batch normalization is not used or the number of layers is huge.

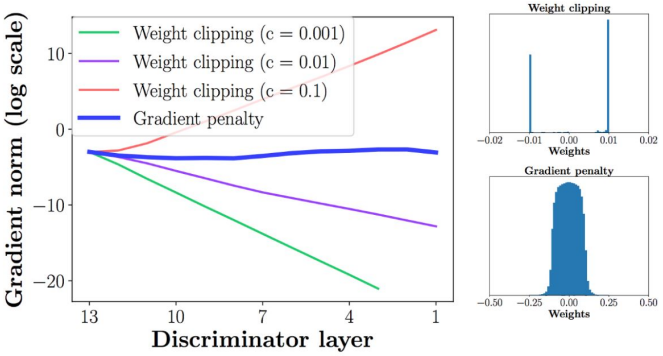

Fig. 5. Gradient norm of 0.8loss_D with values of c and numbers of layers, and weight distribution for WGAN and WGAN-GP.

Fig. 5 [3] demonstrates the weakness of WGAN with weight clipping. If the batch normalization is removed, the gradient of loss_D can be converted from vanishing gradients to exploding ones with small changes of c. Since the weights w of f_w

concentrate within $[-c, c]$, WGAN cannot make full use of the fitting capabilities of deep neural networks.

Lemma 1: A differentiable function f is 1-Lipschitz $\Leftrightarrow$ f has gradients norm 1 at most everywhere.

From lemma 1, if $\left|\left|\frac{\partial f_w}{\partial x}\right|\right|_2$ can be close to 1, the model can satisfy Lipschitz condition as well. By creating $\hat{x} = \epsilon x + (1 - \epsilon)x'$, $\epsilon \sim U[0, 1]$ to represent both real and fake images and added to the L2 gradient penalty, the loss function D of is obtained as:

$$\text{loss}_D = \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)})) - f_w(x^{(i)}) + \lambda\left(\left|\left|\frac{\partial f_w(\hat{x})}{\partial \hat{x}}\right|\right|_2 - 1\right)^2 \tag{20}$$

Algorithm 1 WGAN with gradient penalty. We use default values of λ = 10, n_{critic} = 5, α = 0.0001, β_1 = 0, β_2 = 0.9

Require: The gradient penalty coefficient λ, the number of critic iterations per generator iteration n_{critic}, the batch size m, Adam hyperparameters α, β_1, β_2.

Require: initial critic parameters w_0, generator parameters θ_0.

while θ has not converge **do**
  for $t = 1, \cdots, n_{critic}$ **do**
    for $i = 1, \cdots, m$ **do**
      Sample real data $x \sim P_r$, latent variable $z \sim p(z)$, a random number $\epsilon \sim U[0, 1]$.
      $x' \leftarrow g_\theta(z)$
      $\hat{x} = \epsilon x + (1 - \epsilon)x'$
      $L^{(i)} \leftarrow -f_w(x) + f_w(x') + \lambda\left(\left|\left|\frac{\partial f_w(\hat{x})}{\partial \hat{x}}\right|\right|_2 - 1\right)^2$
    end for
    $w \leftarrow \text{Adam}\left(\nabla_w \frac{1}{m}\sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2\right)$
  end for
  Sample a batch of latent variables $\left\{z^{(i)}\right\}_{i=1}^{m} \sim p(z)$.
  $\theta \leftarrow \text{Adam}\left(\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} -f_w\left(g_\theta(z)\right), \theta, \alpha, \beta_1, \beta_2\right)$
end while

IV. Experimental Results

The methods WGAN and WGAN-GP are applied to the MNIST handwritten digits data set that are trained for 200 epochs. The hyper parameter of weight clipping $c = 0.01$ to be used in WGAN. In terms of step size and optimizer, WGAN used α = 0.00005 and RMSprop optimizer while Wasserstein GAN gradient penalty (WGAN-GP) used α = 0.0002 and Adams optimizer (β_1=0.5 and β_2=0.999).

Fig. 6 presents the fake handwritten digits generated by WGAN, while Fig. 7 shows the digits generated by WGAN-GP. The results obtained by WGAN-GP show better performance than WGAN after training the MNIST data for the same amount of epochs.

V. Conclusion

GAN was firstly introduced to provide a general idea of Wasserstein generative adversarial network (WGAN) and its improvement using gradient penalty (WGAN-GP). The gradient of the discriminator (GAN) was converted from vanishing to exploding gradients and small changes were noticed in the weight clipping using the hyper parameter c. The gradient penalty creates a gradient norm of 1 so that it makes complete use of the fitting capabilities of deep neural networks. The experimental results demonstrated that WGAN improves the learning stability, one of the problems of DCGAN; whereas WGAN-GP convergence speed is faster and perform better results than WGAN.

Fig. 6. Fake images generated by WGAN after training for 200 epochs.

Fig. 7. Fake images generated by WGAN-GP after training for 200 epochs.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[2] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[3] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, pp. 5767–5777, 2017.

[4] J. Hui, "Gan - wasserstein gan & wgan-gp." , Medium, 2018. [Online]. Available: https://jonathan-hui.medium.com/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490. [Accessed: Nov-16-2020].