An Improvement based on Wasserstein GAN for Alleviating Mode Collapsing

Yingying Chen[1,2], Xinwen Hou[1*]
[1]*Institute of Automation, Chinese Academy of Sciences*, Beijing, China
[2]*School of Artificial Intelligence, University of Chinese Academy of Sciences*, Beijing, China
chenyingying2018@ia.ac.cn, xwhou@nlpr.ia.ac.cn

***Abstract*—In the past few years, Generative Adversarial Networks as a deep generative model has received more and more attention. Mode collapsing is one of the challenges in the study of Generative Adversarial Networks. In order to solve this problem, we deduce a new algorithm on the basis of Wasserstein GAN. We add a generated distribution entropy term to the objective function of generator net and maximize the entropy to increase the diversity of fake images. And then Stein Variational Gradient Descent algorithm is used for optimization. We named our method SW-GAN. In order to substantiate our theoretical analysis, we perform experiments on MNIST and CIFAR-10, and the results demonstrate superiority of our method.**

I. Introduction

In the past decade, the field of artificial intelligence has witnessed a frenzy of deep learning [1]. Compared with the great success of deep discriminative models in computer vision, natural language processing and other fields, the halo of deep generative models is slightly inferior due to the difficulty of approximating intractable probabilistic computations. Thanks to Goodfellow proposed Generative Adversarial Networks (GANs) [2] model in 2014, the deep generative model has gained increasing attention.

In the early research of deep generative model, researchers tried to model probability distribution functions to sample data that match some nature distributions, and optimize the probability model by Maximizing Likelihood Estimation (MLE). However, high dimensional MLE is usually computationally intractable. Instead of modeling the probabilistic distribution, GANs model takes a different approach. It generates the samples that match the data distribution directly through the deep network.

Like all other methods, GANs is not perfect. It avoids the difficulty of calculating the probability distributions, but there are still some inherent disadvantages. Training instability and mode collapsing are two typical problems. Training instability involves the generator and discriminator oscillating rather than converging to a fixed point during training. Mode collapsing means that the generator tends to produce only a single sample or a small family of very similar samples.

Within a few years after GANs being put forward, various improvements on it have emerged one after another. These methods can be roughly divided into two categories: one is to improve the model structure; the other proceed from the objective function. Deep Convolutional Generative Adversarial Networks (DC-GANs) [3] is a representative of the methods based on the model structure. The convolutional neural network is applied to GANs framework, and good results are achieved under the training in the subset of large scale scene dataset LSUN [4] . Wasserstein GAN (W-GAN) [5] improves original GANs objective function by introducing Wasserstein-1[1] distance. However W-GAN raises another issue: Lipschitz constraint on the discriminator network. It leads to several excellent improvement works on constraining the gradient or norm of discriminator, such as WGAN-GP [6], SN-GAN [7].

Based on the problem of mode collapsing, we introduce Stein Variational Gradient Descent (SVGD) [8] into W-GAN and theoretically deduce the feasibility of alleviating mode collapsing problem. Based on W-GAN, we add a generator distribution entropy term to the objective function. Maximizing the entropy will increase the diversity of generated data, and then alleviate the issue of mode collapsing. SVGD provides an excellent computing tool for optimizing the KL divergence between the generated distribution and the target distribution. It avoids the intractable computation of normalization constant of the target distribution. And its own property makes further efforts to stop mode collapsing. We use the widespread score criterion IS and FID to evaluate our model. The baseline model is original W-GAN. The experimental results on two commonly used datasets MNIST and CIFAR-10 show the superiority of our method.

II. Related Work

Until recently, a lot of works on GANs had been devoted to unstable training problem. Radford et al. [3] used convolutional neural network to replace the multilayer perceptrons of the original GANs. They explained that the convolutional neural net work can extract the hierarchical visual features of nature images so as to improve the stability of GAN's training. Arjvsky et al. [9] thought GAN's equivalent objective function – Jensen-Shannon (JS) divergence is not a good distance metric. Because JS divergence between two non overlapping distributions of high dimensional space is a constant, which

*The corresponding author is Xinwen Hou, xwhou@nlpr.ia.ac.cn

[1] The Wasserstein-1 is also called Earth-Mover (EM) distance, it measures the cost of optimal transport plan from distribution $\mathbb{P}_r$ to distribution $\mathbb{P}_g$: $W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}[||x - y||]$. All the Wasserstein distance in the following paper denotes Wasserstein-1 distance.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

will lead to generator gradient vanishing. So Arjvsky et al. recommend using Wasserstein-1 distance instead of JS divergence to maintain the stability of GAN's training process.

W-GAN has an irritating K-Lipschitz[2] constraint. Most of the improvements based on W-GAN have revolved around meeting this constraint. In W-GAN, Arjvsky et al. [5] clipped the weight of discriminator to satisfy the constraint. But this way is too simplistic to guarantee the constraint. Gulrajani et al. [6] found that the Lipschitz constraint is actually equivalent to identifying the magnitude of the discriminator's gradient less than a constant. Therefore, the Lipschitz constraint can be satisfied better only by adding the gradient penalty to the objective function. For the sake of creating a more explicit restriction on the gradient, Miyato et al. [7] used the spectral norm to normalized the parameter matrix of the discriminator. The above schemes seek better solutions to meet constraint instead of aiming at solving the two inherent problems of original GANs, so the effect of these methods on solving mode collapsing or training instability are within the scope of the original W-GAN framework.

Previous works have also taken approaches to alleviate mode collapsing problem. Gurumurthy et al. [10] thought the latent space of GANs model is too simple to tackling the extremely complex underlying distributions of real data. So they reparameterize the latent generative space as a mixture model and learn the mixture models parameters along with the GANs. This modification enables diversity in generated samples. Su et al. [11] introduced Turing test into the GANs model and lead to a new mode for training GANs. It allows GANs model to use information of real samples during training generator and accelerates the whole training procedure. Kumar et al. [12] proposed an energy-based model and train it by an amortized neural sampler. The resulting objective maximizes entropy of the generated samples. They use nonparametric mutual information maximization techniques to maximize entropy term. This is the main difference between us. We directly add an entropy term of the generated samples to the generator's objective function and optimize the objective function by Stein Variational Gradient Descent algorithm.

Some previous works have used Stein Variational method in GANs. The most relevant work is Stein-GAN [13]. Stein-GAN is a byproduct of Stein Contrastive Divergence [13] (Stein-CD) with a CD-type objective function:

$$\min_{w} \max_{\theta} \{KL(q_0||p_w) - KL(q_{[G_\theta]}||p_w)\} \tag{1}$$

where p_w is an unknown target distribution and $q_{[G_\theta]}$ denotes the distribution of the generator's output. $q_0(x) = 1/m \sum_{i=1}^{m} \delta(x - x_i)$ is the empirical distribution of observed data $\{x_i\}_{i=1}^{m}$. Then it uses Stein-CD algorithm to optimize w and minimize the Euclidean distance between images pixels to optimize θ. Our work differs from Stein-GAN. We base on W-GAN and add entropy term to the objective function for increasing diversity of generated samples. We maximize (4) to optimize discriminator's parameter w, and use Stein Variational Gradient Descent algorithm in the process of optimizing generator's parameter θ.

[2] The K-Lipschitz constraint denotes the L-norm of f smaller than some constant K: $||f||_L \leq K$

III. Background

In this section, we first introduce the background of GANs and W-GAN which forms the foundation of our work. Then a brief description of Stein Variational Gradient Descent algorithm will be introduced for the convenience of quoting conclusions directly in section IV.

A. Wasserstein GAN

GANs is a minimax two-player game between two competing networks. The generator G receives raw data to generate samples that match an observed distribution. A simple case is that a generator takes random noise as input and outputs face images. The discriminator D is trained to distinguish whether its input data comes from the true data. The generator and discriminator act adversarially in training process. The GANs objective function is:

$$\begin{aligned} \min_{G} \max_{D} V(G, D) = \mathbb{E}_{\boldsymbol{x} \sim p_r(\boldsymbol{x})}[\log D(\boldsymbol{x})] \\ + \mathbb{E}_{\boldsymbol{x} \sim p_g(\boldsymbol{x})}[\log(1 - D(\boldsymbol{x}))] \end{aligned} \tag{2}$$

where $p_r(\boldsymbol{x})$ is the real data distribution and $p_g(\boldsymbol{x})$ is the generated data distribution with $\boldsymbol{x} = G(\boldsymbol{z})$. $\boldsymbol{z}$ is the input to G obeying a given distribution $p_z(\boldsymbol{z})$. $D(\boldsymbol{x})$ is the output of discriminator and represents the probability that $\boldsymbol{x}$ comes from the real data rather than fake data.

GANs is trained by optimizing (2) which alternates the maximization and minimization. Specifically, maximize $V(D, G)$ by using gradient ascent algorithm to update discriminator (under the current generator) firstly, and then using gradient descent algorithm to adjust generator to minimize $V(G, D)$; repeating the above process until G produces realistic data. In each iteration, the optimal D measures the JS divergence between real data and generated data given G, and optimizing G will narrow the gap between two distributions. In the process of game, the capability of D and G is enhanced.

There are many ways to measure the distance between two distributions, which original GANs involved JS divergence. Arjvsky et al. [5] brilliantly observed that Wasserstein distance has nicer properties of optimization than JS divergence. So they introduced Wasserstein distance into GANs framework, then consider solving the problem:

$$\min_{G} \max_{D} V(G, D) = \mathbb{E}_{\boldsymbol{x} \sim p_r(\boldsymbol{x})}[f_w(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x} \sim p_g(\boldsymbol{x})}[f_w(\boldsymbol{x})] \tag{3}$$

where $f_w(\boldsymbol{x})$ is the discriminator with parameter w, and it is denoted by $D(\boldsymbol{x})$ in (2). $\max_D V(G, D)$ in (3) yields a calculation of Wasserstein (or Earth-Mover) distance between p_r and p_g.

W-GAN shows that Wasserstein distance is a sensible cost function when learning distributions supported by low dimensional manifolds. The discriminator is trained to maximize:

$$\max_{w \sim W} \mathbb{E}_{\boldsymbol{x} \sim p_r(\boldsymbol{x})}[f_w(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x} \sim p_g(\boldsymbol{x})}[f_w(\boldsymbol{x})] \tag{4}$$

where W is a parameter family of discriminator. The generator is trained to minimize:

$$\begin{aligned}\min_{\theta\sim\Theta}\mathbb{E}_{\boldsymbol{x}\sim p_r(\boldsymbol{x})}[f_w(\boldsymbol{x})]-\mathbb{E}_{\boldsymbol{x}\sim p_g(\boldsymbol{x})}[f_w(\boldsymbol{x})]\\ =\min_{\theta\sim\Theta}-\mathbb{E}_{\boldsymbol{x}\sim p_g(\boldsymbol{x})}[f_w(\boldsymbol{x})]\end{aligned} \tag{5}$$

where Θ is a parameter family of generator. Equation (5) holds for the term $\mathbb{E}_{\boldsymbol{x}\sim p_r(\boldsymbol{x})}[f_w(\boldsymbol{x})]$ is independent of the optimization.

B. Stein Variational Gradient Descent

The goal of variational inference is finding a simpler distribution $q(x)$ to approximate a target distribution $p(x)$. This problem can be solved by searching a $q^*(x)$ that satisfies certain accuracy requirements:

$$q^*(x)=\arg\min_q KL(q(x)||p(x)) \tag{6}$$

where gradient decent algorithm can solve $q^*(x)$ iteratively. While $p(x)$ always equips with an intractable normalization constant.

Stein Variational Gradient Descent (SVGD) is a general purpose variational inference algorithm introduced by Liu et al [8]. It works without knowing the normalization constant of $p(x)$. SVDG links the derivative of KL divergence and Stein Discrepancy [14]–[16]. It shows that:

$$\nabla_\epsilon \mathrm{KL}\left(q_{[\boldsymbol{M}]}\|p\right)\big|_{\epsilon=0}=-\mathbb{E}_{x\sim q}\left[\mathrm{trace}\left(\mathcal{A}_p\phi(x)\right)\right] \tag{7}$$

where $q_{[\boldsymbol{M}]}(z)$ is the density of $z=\boldsymbol{M}(x)=x+\epsilon\phi(x)$ when $x\sim q(x)$. $\phi(x)$ is a smooth vector function that characterizes the searching direction. ϵ represents the searching step. $\mathcal{A}_p$ is Stein Operator, which act on function $\phi(x)$:

$$\mathcal{A}_p\phi(x)=\nabla_x\log p(x)\phi(x)^T+\nabla_x\phi(x) \tag{8}$$

To the right side of (7), $\mathbb{E}_{x\sim q}[\mathrm{trace}(\mathcal{A}_p\phi(x)]$'s maximum value for $\phi(x)$ is Stein Discrepancy:

$$\mathbb{D}(q,p)=\max_\phi \mathbb{E}_{x\sim q}[\mathrm{trace}(\mathcal{A}_p\phi(x)] \tag{9}$$

The optimization has a closed form solution by maximizing $\phi(x)$ in the unit ball of a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$. In this case, Stein Discrepancy is called Kernelized Stein Discrepancy:

$$\begin{aligned}\mathbb{D}(q,p)=\max_{\phi\sim\mathcal{H}^d}\{&\mathbb{E}_{x\sim q}[\mathrm{trace}(\mathcal{A}_p\phi(x)],\\ &s.t.\qquad ||\phi||_{\mathcal{H}^d}\le 1\}\end{aligned} \tag{10}$$

where $\mathcal{H}^d$ is the space of vector functions $\boldsymbol{f}=[f_1,f_2,...,f_d]$ with $f_i\in\mathcal{H}$. The optimal solution of (10) can be shown as: $\phi(x)=\phi^*_{q,p}(x)/||\phi^*_{q,p}||_{\mathcal{H}^d}$, in which:

$$\begin{aligned}\phi^*_{q,p}(\cdot)&=\mathbb{E}_{x\sim q}[\mathcal{A}_p k(x,\cdot)]\\ &=\int_{x\sim q}[\nabla_x\log p(x)k(x,\cdot)+\nabla_x k(x,\cdot)]dx\end{aligned} \tag{11}$$

$k(x,\cdot)$ is the kernel of RKHS $\mathcal{H}$. RBF kernel $k(x,x')=\exp(-||x-x'||_2^2/h)$ is a viable option.

SVGD avoids the intractable derivation of the normalization constant to ensure the smooth operation of the optimization algorithm. At the same time, SVGD has its own characteristics for fostering particles diversity. By maximizing the Kernelized Stein Discrepancy, SVGD gets the fastest descent direction of KL divergence.

To sum up, SVGD suggests an iterative procedure that transforms an initial distribution $q_0(x)$ to the target distribution $p(x)$: starting with applying transform $\boldsymbol{M}^*_0(x)=x+\epsilon\phi^*_{q_0,p}(x)$ on q_0; this would give a new distribution $q_1(x)$ that closer to $p(x)$ than $q_0(x)$; repeating $\boldsymbol{M}$ operation until convergence. $\phi^*_{q_0,p}(x)$ can be view as a negative gradient of $KL(q_0,p)$ to x.

IV. Our Method:SW-GAN

W-GAN does not specifically consider the problem of generated data diversity. We solve this problem by adding the generated data distribution entropy term to (5) and rewrite it as:

$$G^*=\arg\min_{\theta\sim\Theta}\{-\gamma\mathbb{E}_{\boldsymbol{x}\sim p_g(\boldsymbol{x})}[f_w(\boldsymbol{x})]-H_{\boldsymbol{x}\sim p_g(\boldsymbol{x})}(\boldsymbol{x})\} \tag{12}$$

where $H_{\boldsymbol{x}\sim p_g(\boldsymbol{x})}(\boldsymbol{x})$ is the entropy of generated data, and γ is regularization coefficient. Minimizing (12) will maximize the fake data's entropy on the basis of Wasserstein distance. Add a constant C that is independent of optimization to (12):

$$\begin{aligned}G^*&=\arg\min_{\theta\sim\Theta}\{-\mathbb{E}_{\boldsymbol{x}\sim p_g(\boldsymbol{x})}[\gamma f_w(\boldsymbol{x})]-H_{\boldsymbol{x}\sim p_g(\boldsymbol{x})}(\boldsymbol{x})]+C(w)\}\\ &=\arg\min_{\theta\sim\Theta}\{-\mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(e^{\gamma f_w(x)}/Z(w))]+\mathbb{E}_{\boldsymbol{x}\sim p_g}[\log p_g(\boldsymbol{x})]\}\end{aligned} \tag{13}$$

where $e^{\gamma f_w(\boldsymbol{x})}/Z(w)$ is the Gibbs distribution of f_w that given w:

$$\begin{aligned}p(\boldsymbol{x}|w)&=e^{\gamma f_w(\boldsymbol{x})}/Z(w)\\ with\qquad Z(w)&=\int_{\boldsymbol{x}}\exp[\gamma f_w(\boldsymbol{x})]d\boldsymbol{x}\end{aligned} \tag{14}$$

That is to say, Gibbs distribution $p(\boldsymbol{x}|w)$ is used to model real data distribution p_r. Then (13) can be wrote as:

$$\begin{aligned}G^*&=\arg\min_{\theta\sim\Theta}\{-\mathbb{E}_{x\sim p_g}\log p_r+\mathbb{E}_{x\sim p_g}\log p_g\}\\ &=\arg\min_{\theta\sim\Theta}\{KL(p_g,p_r)\}\end{aligned} \tag{15}$$

Equation (15) shows that we can optimize generator G by minimizing $KL(p_g,p_r)$, and SVGD could work on this case. Firstly, G generates a batch of negative samples $\{\hat{\boldsymbol{x}}_i\}_{i=1}^m$ from a given $\boldsymbol{z}\sim p(\boldsymbol{z})$:

$$\hat{\boldsymbol{x}}_i=G(\boldsymbol{z}_i) \tag{16}$$

Then making operator $\boldsymbol{M}(x)$ acts on $\hat{\boldsymbol{x}}_i$:

$$\boldsymbol{M}(\hat{\boldsymbol{x}}_i)=\hat{\boldsymbol{x}}_i+\epsilon\phi^*(\hat{\boldsymbol{x}}_i) \tag{17}$$

where $\hat{\boldsymbol{x}}'_i=\boldsymbol{M}(\hat{\boldsymbol{x}}_i)$ forms increasingly better approximations for real data than $\hat{\boldsymbol{x}}_i$. Equation (17) mimics a gradient dynamics at the particle level for updating samples.

$$\phi^*_{p_g,p_r}(\cdot)=\frac{1}{m}\sum_{j=1}^m[\alpha k(x_j,\cdot)\nabla_{x_j}f_w(x_j)+\beta\nabla_{x_j}k(x_j,\cdot)] \tag{18}$$

where two terms play different roles: $k(x_j,\cdot)\nabla_{x_j} f_w(x_j)$ drives the particles $\hat{\boldsymbol{x}}_i$ towards the high probability areas of p_r, $\nabla_{x_j} k(x_j,\cdot)$ acts as a repulsive force that prevents all the $\hat{\boldsymbol{x}}_i$ to collapse together into local modes. α and β balance the effect of above two terms.

$-\phi^*(\hat{\boldsymbol{x}}_i)$ in (17) serves as the gradient of $KL(p_g, p_r)$ to $\hat{\boldsymbol{x}}$. According to the Chain Rule, the gradient of $KL(p_g, p_r)$ to G's parameters θ is $\frac{\partial \hat{\boldsymbol{x}}}{\partial \theta}\phi^*(\hat{\boldsymbol{x}})$. Then the rule for gradient descent to update θ is:

$$\theta^{l+1} = \theta^l + \delta \frac{\partial \hat{\boldsymbol{x}}}{\partial \theta}\phi^*(\hat{\boldsymbol{x}}) \tag{19}$$

where δ is the learning rate.

We train discriminator to maximize (4), it is the same as original W-GAN. The gradient ascent rule for updating parameter w is :

$$w^{l+1} = w^l + \delta \frac{\partial}{\partial w}\{\mathbb{E}_{\boldsymbol{x}\sim p_r}[f_w(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim p_g}[f_w(\boldsymbol{x})]\} \tag{20}$$

In a word, we add an entropy term to the objective function of generator G to expand the diversity of fake data. Then derive it can be optimized by SVGD in minimizing the KL divergence between p_g and p_r. The SVGD increases the diversity further due to its own speciality. And we verify the change on generator's objective function has no bad effect on the optimization of (20) by experiments. It even helps with stabilizing the training.

In practice, we train the whole nets iteratively. For each iteration, we update discriminator D n times firstly and then update the generator G one time. We use the update algorithm RMSProp [17] in W-GAN. The procedure of our method is summarized as Algorithm 1.

Algorithm 1 SW-GAN. All experiments in the paper used the default values $\delta = 0.00005$, $c = 0.01$, $m = 64$, $n_{critic} = 5$

Require:

δ, the learning rate; c, the clipping parameter;
m, the mini-batch size; n_{critic}, the number of iterations of the discriminator per generator iteration; w_0, initial discriminator's parameters; θ_0, initial generator's parameters.

while Θ has not converged **do**
  for $t = 0, ..., n_{critic}$ **do**
    Sample $\{\boldsymbol{x}_i\}_{i=1}^m \sim p_r(\boldsymbol{x})$ a batch of the real data.
    Sample $\{\boldsymbol{z}_i\}_{i=1}^m \sim p(\boldsymbol{z})$ a batch of prior samples.
    $g_w \leftarrow \nabla_w[\frac{1}{m}\sum_{i=1}^m f_w(\boldsymbol{x}_i) - \frac{1}{m}\sum_{i=1}^m f_w(G(\boldsymbol{z}_i))]$
    $w \leftarrow w + \delta \cdot RMSProp(w, g_w)$
    $w \leftarrow clip(w, -c, c)$
  end for
  Sample $\{\boldsymbol{z}_i\}_{i=1}^m \sim p(\boldsymbol{z})$ a batch of prior samples.
  $g_\theta \leftarrow -\frac{1}{m}\sum_{i=1}^m \frac{\partial G(\boldsymbol{z}_i)}{\partial \theta}\phi^*(G(\boldsymbol{z}_i))$
  $\theta \leftarrow \theta - \delta \cdot RMSProp(\theta, g_\theta)$
end while

V. Experiment

We do experiments on datasets MNIST and CIFAR-10. For MNIST, we design a toy experiment, and evaluate the quality of fake images by a pretrained LeNet [18]. For CIFAR-10, we run IS and FID experiments to evaluate the models. In two datasets, our algorithm shows the advantages over baseline W-GAN both in terms of the accuracy and speed.

We set the architecture and super parameters the same as those in benchmark model W-GAN[3]. We specify super parameters in Algorithm 1. The whole networks apply convolutional architecture: the discriminator employs convolutions layers and generator employs transposed convolution layers. The generator net uses ReLU [19] activation for all layers except for the output, which uses Tanh. The discriminator net uses LeakyReLU [20] activation for all layers except for the output, which does not has any activation. Batch Normalization [21] is applied in training both the G and D nets. For all our experiments, we use RBF kernel $k(x, x') = \exp(-||x - x'||_2^2/h)$, and take the bandwidth to be $h = rep^2/\log n$, where rep is the 0.6 times median of the pairwise distance between the current samples. We set the parameter $\alpha = 2$, $\beta = 0.5$ for all SW-GAN models training. In each comparison experiment, we input same random normal distribution noise into generator.

Model Evaluation Criteria There are many ways to evaluate the performance of GANs, but all of them come from very intuitive views. In the field of image generation, a simple idea to model automatic evaluation is classifying images by a classifiers with excellent performance. The better the image quality is, the easier it can be classified. This raises two common indicators: Inception Score (IS) [22] and Fréchet Inception Distance (FID) [23].

IS calculates the KL divergence between $p(y|\boldsymbol{x})$ and $p(y)$. Given the image $\boldsymbol{x}$, the conditional probability $p(y|\boldsymbol{x})$ indicates corresponding category. $p(y)$ shows the distribution of different categories . If $\boldsymbol{x}$ is a realistic image, its IS will be high. IS grows up as the quality of generated data becomes better. On the basis of IS, FID considers the effect of real data and a Fréchet distance between generated images and real images is calculated. FID as a distance metric, the smaller the better. Both of them used inception model [24] to extracting images feature. If the generated images do not fit to the world view of inception net, although the image quality is high, its score turns worse. Therefore, we need to choose appropriate scoring rules according to different situation.

A. Experiment on MNIST

We train a LeNet model on MNIST to classify the fake images. The generated images are 1-channel images of 32*32 pixels in size. The test accuracy of our LeNet model is 98%. Because the purpose of this paper is not to obtain an accurate LeNet classifier, the classification network does not need be finely tuned to the best accuracy. We put 128 images randomly selected from our model and baseline in Fig.1 which the models have the same random noise inputs.

Suppose in an ideal situation, a qualified model generates N images randomly. They should contain all the digits from 0 to

[3] We use the baseline model under https://github.com/martinarjovsky/WassersteinGAN

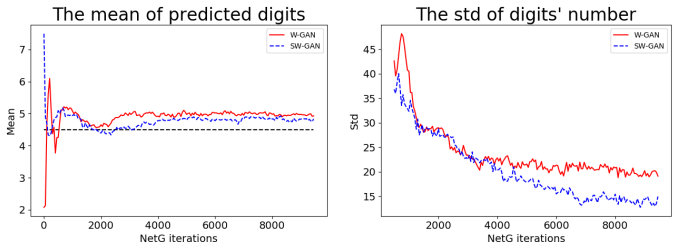
(a) Images from W-GAN (b) Images from SW-GAN

Fig. 1. Images randomly sampled from models with 9500 G's iterations. Note that two models can generate realistic figures. While the images from W-GAN appear mode collapsing in the red mark. While our method doesn't encounter this problem.

9 and each digit has $N/10$ images. The mean value of these digits is 4.5. From this point, we calculate the mean value of fake digits to compare two models. We also calculate the standard deviation (std) of digit's numbers to further reflect the balance of generated data:

$$std = \sqrt{\frac{1}{K-1}\sum_{i=1}^{K}(n_i - \overline{n})} \tag{21}$$

where K is the number of different digit categories, n_i is the number of digit category i and $\sum n_i = N$, $\overline{n}$ is the mean of n_i and $\overline{n} = N/K$. Fig.2 shows the mean and std of two models.



(a) Figures Mean (b) Figure Number's Std

Fig. 2. The mean value and digits number's std of 500 generated images. The solid red lines are from W-GAN model and the dash blue lines are from our method. Our method has lower mean and std than W-GAN. It means our model generates more abundant samples than W-GAN.

During the experiments, we find that our model is earlier than W-GAN in generating all the 10 digits. The number of generated digits with G's iterations is counted in Table I. '400' is the iteration that W-GAN generates all the 10 digits for the first time (sampling interval is 50 G's iterations). Our method generates 10 digits in 50 G's iterations. It shows that our method is faster than baseline.

B. Experiment on CIFAR-10

We train our model and baseline on CIFAR-10. The CIFAR-10 images is more vivid than MNIST. Generator is trained to generate 3-channel images of 32x32 pixels in size. Fig.3 shows the random samples of two models.

TABLE I
THE NUMBER OF CATEGORIES WITH G'S ITERATION

G iteration	Number of categories	
	W-GAN	SW-GAN
50	3	5
100	2	10
150	2	10
200	7	10
250	8	10
300	8	10
350	7	10
400	10	10

(a) Images from W-GAN (b) Images from SW-GAN

Fig. 3. Images randomly sampled from models with 11000 G's iterations. Two models have the same noise inputs.

For calculating IS on each model, we generate 1000 images 10 times and compute the averages inception score. The model performance goes up with oscillation, so we plot the IS curves with net G iterations in Fig.4. IS is related to image quality and diversity, the higher the better. The IS curve of our model is higher than baseline.

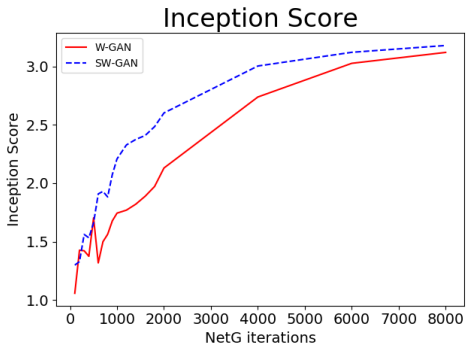


Fig. 4. The IS with net G iterations. The solid red lines are from W-GAN model and the dash blue lines are from our method. Our method has higher IS than baseline.

FID uses the real data statistics for calculating. We download the precalculated statistics from FID repository[4]. And each model generates 10000 images for computing the fake data statistics. Fig.5 shows the FID with net G iteration. FID measures the distance between real data and generated data,

[4] The precalculated statistics for FID: https://github.com/bioinf-jku/TTUR

the smaller the better. The FID curve of our method is below the baseline model.

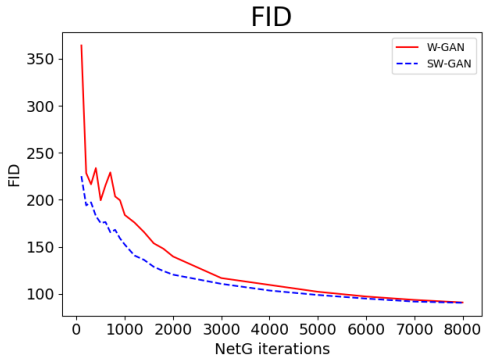
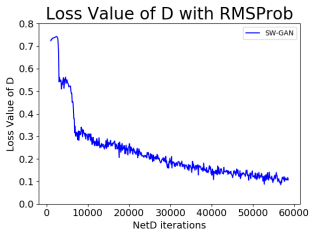
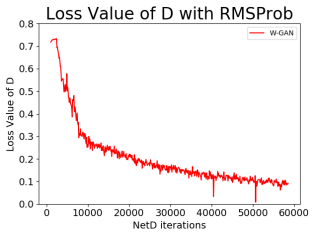


Fig. 5. The FID with net G iterations. The solid red lines are from W-GAN model and the dash blue lines are from our method. Our method has lower IS than baseline, and reaches minimum value faster than baseline.

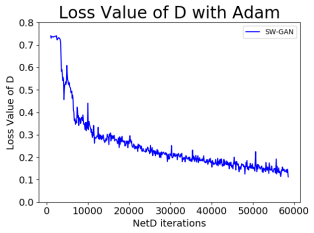
Since we change the objective function of the generator, we should make sure it doesn't affect the optimization of the discriminator. Actually, we find that our method helps to stabilize the training of discriminator. W-GAN states clearly that under the same discriminator's architecture, the objective function is a meaningful loss metric that correlates with the generators convergence and sample quality. As the training goes on, objective function becomes smaller and the quality of generated data gets better. We recorded the D loss curves during training in Fig.6. It shows that our method dose not have a bad effect on optimizing discriminator; it improves the stationary of optimization on the contrary.



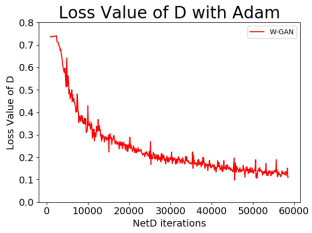
(a) SW-GAN with RMSprob updating



(b) W-GAN with RMSprob updating



(c) SW-GAN with Adam updating



(d) W-GAN with Adam updating

Fig. 6. The curves of D's training loss with D iterations. (a) and (b) are under the RMSProb updating rule. (a) and (b) have the same decline, it confirms that our method dose not have a bad effect on optimizing discriminator. (c) and (d) using the Adam [25] to update. (d) preforms worse than (c), it seems that our method improves the stationary of optimization.

VI. Conclusion

We introduce an algorithm that we named SW-GAN. In our method, we increase the diversity of generated data and alleviate mode collapsing by optimizing an additional entropy term. Then we use the Stein Variational Gradient Descent algorithm in optimization and expand the diversity further. Our method shows better experimental results relative to original W-GAN. In the future work, we plan to investigate more theoretical basis for our method and experiment our algorithm on larger and more complex datasets. We will also do some research on new way to realize Lipschitz constraint for SW-GAN.

VII. Acknowledgments

We would like to acknowledgment National Key R & D Program of China (2017YFC1200602), the National Natural Science Foundation of China (31672325). This paper is supported by them.

References

[1] I. G. Goodfellow, Y. Bengio, and A. C. Courville, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[2] 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 27 NIPS*, pp. 2672–2680, Curran Associates, Inc., 2014.

[3] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2015.

[4] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv:1506.03365*, 2015.

[5] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *ArXiv:1701.07875*, 2017.

[6] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," *arXiv:1704.00028*, 2017.

[7] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *ArXiv:1802.05957*, 2018.

[8] Q. Liu and D. Wang, "Stein variational gradient descent: A general purpose bayesian inference algorithm," in *Advances in Neural Information Processing Systems 29 NIPS*, pp. 2378–2386, Curran Associates, Inc., 2016.

[9] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *5th International Conference on Learning Representations ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[10] S. Gurumurthy, R. K. Sarvadevabhatla, and R. V. Babu, "Deligan: Generative adversarial networks for diverse and limited data," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4941–4949, 2017.

[11] J. Su, "Training generative adversarial networks via turing test," *ArXiv*, vol. abs/1810.10948, 2018.

[12] R. Kumar, A. Goyal, A. C. Courville, and Y. Bengio, "Maximum entropy generators for energy-based models," *CoRR*, vol. abs/1901.08508, 2019.

[13] Q. Liu and D. Wang, "Learning deep energy models: Contrastive divergence vs. amortized mle.," *arXiv:1707.00797*, 2017.

[14] Q. Liu, J. D. Lee, and M. I. Jordan, "A kernelized stein discrepancy for goodness-of-fit tests," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pp. 276–284, 2016.

[15] C. J. Oates, M. Girolami, and N. Chopin, "Control functionals for monte carlo integration," *Journal of The Royal Statistical Society Series B-statistical Methodology*, vol. 79, no. 3, pp. 695–718, 2017.

[16] K. Chwialkowski, H. Strathmann, and A. Gretton, "A kernel test of goodness of fit," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pp. 2606–2615, 2016.

[17] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning,*, pp. 26–30, 2012.
[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov 1998.
[19] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *AISTATS*, 2011.
[20] A. L. Maas, "Rectifier nonlinearities improve neural network acoustic models," in *International Conference on Machine Learning,ICML*, vol. 30, 2013.
[21] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *ArXiv:1502.03167*, 2015.
[22] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *ArXiv:1606.03498*, 2016.
[23] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems 30 NIPS*, 2017.
[24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition CVPR*, pp. 2818–2826, 2015.
[25] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations 2015 ICLR*, 2015.