IMPROVING THE IMPROVED TRAINING OF WASSERSTEIN GANS: A CONSISTENCY TERM AND ITS DUAL EFFECT

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite being impactful on a variety of problems and applications, the generative adversarial nets (GANs) are remarkably difficult to train. This issue is formally analyzed by Arjovsky & Bottou (2017), who also propose an alternative direction to avoid the caveats in the minmax two-player training of GANs. The corresponding algorithm, called Wasserstein GAN (WGAN), hinges on the 1-Lipschitz continuity of the discriminator. In this paper, we propose a novel approach to enforcing the Lipschitz continuity in the training procedure of WGANs. Our approach seamlessly connects WGAN with one of the recent semi-supervised learning methods. As a result, it gives rise to not only better photo-realistic samples than the previous methods but also state-of-the-art semi-supervised learning results. In particular, our approach gives rise to the inception score of more than 5.0 with only 1,000 CIFAR-10 images and is the first that exceeds the accuracy of 90% on the CIFAR-10 dataset using only 4,000 labeled images, to the best of our knowledge.

1 INTRODUCTION

We have witnessed a great surge of interests in deep generative networks in recent years (Kingma & Welling, 2013; Goodfellow et al., 2014; Li et al., 2015). The central idea therein is to feed a random vector to a (*e.g.*, feedforward) neural network and then take the output as the desired sample. This sampling procedure is very efficient without the need of any Markov chains.

In order to train such a deep generative network, two broad categories of methods are proposed. The first is to use stochastic variational inference (Kingma & Welling, 2013; Rezende et al., 2014; Kingma et al., 2014) to optimize the lower bound of the data likelihood. The other is to use the samples as a proxy to minimize the distribution divergence between the model and the real through a two-player game (Goodfellow et al., 2014; Salimans et al., 2016), maximum mean discrepancy (Li et al., 2015; Dziugaite et al., 2015; Li et al., 2017b), f-divergence (Nowozin et al., 2016; Nock et al., 2017), and the most recent Wasserstein distance (Arjovsky et al., 2017; Gulrajani et al., 2017).

With no doubt, the generative adversarial networks (GANs) among them (Goodfellow et al., 2014) have the biggest impact thus far on a variety of problems and applications (Radford et al., 2015; Denton et al., 2015; Im et al., 2016; Isola et al., 2016; Springenberg, 2015; Sutskever et al., 2015; Odena, 2016; Zhu et al., 2017). GANs learn the generative network (generator) by playing a two-player game between the generator and an auxiliary discriminator network. While the generator has no difference from other deep generative models in the sense that it translates a random vector into a desired sample, it is impossible to calculate the sample likelihood from it. Instead, the discriminator serves to evaluate the quality of the generated samples by checking how difficult it is to differentiate them from real data points.

However, it is remarkably difficult to train GANs without good heuristics (Goodfellow et al., 2014; Salimans et al., 2016; Radford et al., 2015) which may not generalize across different network architectures or application domains. The training dynamics are often unstable and the generated samples could collapse to limited modes. These issues are formally analyzed by Arjovsky & Bottou (2017), who also propose an alternative direction (Arjovsky et al., 2017) to avoid the caveats in the minmax two-player training of GANs. The corresponding algorithm, namely, Wasserstein GAN

(WGAN), shows not only superior performance over GANs but also a nice correlation between the sample quality and the value function that GANs lack.

1.1 BACKGROUND: WGAN AND THE IMPROVED TRAINING OF WGAN

WGAN (Arjovsky et al., 2017) aims to learn the generator network $G(\boldsymbol{z})$, for any random vector $\boldsymbol{z} \sim \mathbb{P}_z$, such that the Wasserstein distance is minimized between the resulting distribution $\mathbb{P}_G$ of the generated samples $\{G(\boldsymbol{z})\}$ and the real distribution $\mathbb{P}_r$ underlying the observed data points $\{\boldsymbol{x}\}$; in other words, $\min_G W(\mathbb{P}_r, \mathbb{P}_G)$. The Wasserstein distance $W(\mathbb{P}_r, \mathbb{P}_G)$ is shown a more sensible cost function for learning the distributions supported by low-dimensional manifolds than the other popular distribution divergences and distances — for example, the Jensen-Shannon (JS) divergence implicitly employed in GANs (Goodfellow et al., 2014).

Due to the Kantorovich-Rubinstein duality (Villani, 2008) for calculating the Wasserstein distance, the value function of WGAN is then written as

$$\min_G \max_{D \in \mathcal{D}} \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_r}\big[D(\boldsymbol{x})\big] - \mathbb{E}_{\boldsymbol{z} \sim \mathbb{P}_z}\big[D(G(\boldsymbol{z}))\big], \tag{1}$$

where $\mathcal{D}$ the set of 1-Lipschitz functions. Analogous to GANs, we still call D the "discriminator" although it is actually a real-valued function and not a classifier at all. Arjovsky et al. (2017) specify this family of functions $\mathcal{D}$ by neural networks and then use weight clipping to enforce the Lipschitz continuity. However, as the authors note, the networks' capacities become limited due to the weight clipping and there could be gradient vanishing problems in the training.

Improved training of WGAN. Gulrajani et al. (2017) give more concrete examples to illustrate the perils of the weight clipping and propose an alternative way of imposing the Lipschitz continuity. In particular, they introduce a gradient penalty term by noting that the differentiable discriminator $D(\cdot)$ is 1-Lipschitz if and only if the norm of its gradients is at most 1 *everywhere*,

$$GP|_{\widehat{\boldsymbol{x}}} := \mathbb{E}_{\widehat{\boldsymbol{x}}}\Big[\big(\|\nabla_{\widehat{\boldsymbol{x}}} D(\widehat{\boldsymbol{x}})\|_2 - 1\big)^2\Big] \tag{2}$$

where $\widehat{\boldsymbol{x}}$ is uniformly sampled from the straight line between a pair of data points sampled from the model $\mathbb{P}_G$ and the real $\mathbb{P}_r$, respectively. A similar regularization is used by Kodali et al. (2017).

Potential caveats. Unlike the weight clipping, however, by no means one can penalize *everywhere* using this term through a finite number of training iterations. As a result, the gradient penalty term GP takes effect only upon the sampled points $\widehat{\boldsymbol{x}}$, leaving significant parts of the support domain not examined at all. In particular, consider the observed data points and their underlying manifold that supports the real distribution $\mathbb{P}_r$. At the beginning of the training stage, the generated sample $G(\boldsymbol{z})$ and hence $\widehat{\boldsymbol{x}}$ could be distant from the manifold. The Lipschitz continuity over the manifold is not enforced until the generative model $\mathbb{P}_G$ becomes close enough to the real one $\mathbb{P}_r$, if it can.

1.2 OUR APPROACH AND CONTRIBUTIONS

In light of the above pros and cons, we propose to improve the improved training of WGAN by additionally laying the Lipschitz continuity condition over the manifold of the real data $\boldsymbol{x} \sim \mathbb{P}_r$. Moreover, instead of focusing on one particular data point at a time, we devise a regularization over a pair of data points drawn near the manifold following the most basic definition of the 1-Lipschitz continuity. In particular, we perturb each real data point $\boldsymbol{x}$ twice and use a Lipschitz constant to bound the difference between the discriminator's responses to the perturbed data points $\boldsymbol{x}', \boldsymbol{x}''$.

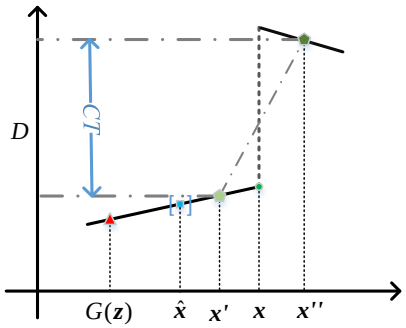


Figure 1: Illustration of our main idea.

Figure 1 illustrates our main idea. The gradient penalty $GP|_{\widehat{\boldsymbol{x}}}$ often fails to check the continuity of region near the real data $\boldsymbol{x}$, around which the discriminator function can freely violate the 1-Lipschitz continuity. We

alleviate this issue by explicitly checking the continuity
condition using the two perturbed version $\boldsymbol{x}', \boldsymbol{x}''$ near any observed real data point $\boldsymbol{x}$.

In this paper, we make the following contributions. (1) We propose an alternative mechanism for enforcing the Lipschitz continuity over the family of discriminators by resorting to the basic definition of the Lipschitz continuity. It effectively improves the gradient penalty method (Gulrajani et al., 2017) and gives rise to generators with more photo-realistic samples and higher inception scores (Salimans et al., 2016). (2) Our approach is very data efficient in terms of being less prone to overfitting even for very small training sets. We do not observe obvious overfitting phenomena even when the model is trained on only 1000 images of CIFAR-10 (Krizhevsky & Hinton, 2009). (3) Our approach can be seamlessly integrated with GANs to be a competitive semi-supervised training technique (Chapelle et al., 2009) thanks to that both inject noise to the real data points.

As results, we are able to report the state-of-the-art results on the generative model with a inception score of 8.81 ± 0.13 on CIFAR-10, and the semi-supervised learning results of 9.98 ± 0.21 on CIFAR-10 using only 4,000 labeled images — especially, they are significantly better than the existing GAN-based semi-supervised learning results.

2 Approach

We firstly review the definition of the Lipschitz continuity and then discuss how to use it to regularize the training of WGAN. We then arrive at an approach that can be seamlessly integrated with the semi-supervised learning method (Chapelle et al., 2009). By bringing the best of the two worlds, we report better semi-supervised learning results than both (Laine & Aila, 2016) and previously GAN-based methods.

2.1 Improving the improved training of WGAN

Let d denote the ℓ_2 metric on an input space used in this paper. A discriminator $D : \mathcal{X} \mapsto \mathcal{Y}$ is Lipschitz continuous if there exists a real constant $M \geqslant 0$ such that, for all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}$,

$$d(D(\boldsymbol{x}_1), D(\boldsymbol{x}_2)) \leqslant M \cdot d(\boldsymbol{x}_1, \boldsymbol{x}_2). \tag{3}$$

Immediately, we can add the following soft consistency term (CT) to the value function of WGAN in order to penalize the violations to the inequality in eq. (3),

$$CT|_{\boldsymbol{x}_1, \boldsymbol{x}_2} = \mathbb{E}_{\boldsymbol{x}_1, \boldsymbol{x}_2} \left[\max \left(0, \frac{d(D(\boldsymbol{x}_1), D(\boldsymbol{x}_2))}{d(\boldsymbol{x}_1, \boldsymbol{x}_2)} - M' \right) \right] \tag{4}$$

Remarks. Here we face the same snag as in (Gulrajani et al., 2017), *i.e.*, it is impractical to substitute all the possibilities of $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ pairs into eq. (4). What pairs and which regions of the input set $\mathcal{X}$ should we check for eq. (4)? Arguably, it is fairly safe to limit our scope to the manifold that supports the real data distribution $\mathbb{P}_r$ and its surrounding regions mainly for two reasons. First, we keep the gradient penalty term and improve it by the proposed consistency term in our overall approach. While the former enforces the continuity over the points sampled between the real and generated points, the latter complement the former by focusing on the region around the real data manifold instead. Second, the distribution of the generative model $\mathbb{P}_G$ is virtually desired to be as close as possible to $\mathbb{P}_r$. We use the notation M in eq. (3) and a different M' in eq. (4) to reflect the fact that the continuity will be checked only sparsely at some data points in practice.

Perturbing the real data. To this end, the very first version we tried was to directly add Gaussian noise $\boldsymbol{\delta}$ to each real data point, resulting in a pair of $\boldsymbol{x} + \boldsymbol{\delta}_1, \boldsymbol{x} + \boldsymbol{\delta}_2$, where $\boldsymbol{x} \sim \mathbb{P}_r$. However, as noted by Arjovsky et al. (2017) and Wu et al. (2016), we found that the samples from the generator become blurry due to the Gaussian noise used in the training. We have also tested the dropout noise that gives rise to generated MNIST samples, which are cut off here and there.

The success comes after we perturb the hidden layers of the discriminator using dropout, as opposed to the input $\boldsymbol{x}$. When the dropout rate is small, the perturbed discriminator's output can be considered as the output of the clean discriminator in response to a "virtual" data point $\boldsymbol{x}'$ that is not far from $\boldsymbol{x}$. Thus, we denote by $D(\boldsymbol{x}')$ the discriminator output after applying dropout to its hidden layers. In

Algorithm 1 our proposed **CT-GAN** for training a perfect generator. Most experiments in the paper used the default values $\lambda_1 = 10$,$\lambda_2 = 2$, $N = 5$, $\gamma = 0.0002$, $b = 64$, $iter = 10^6$

Require: b, the batch size. λ_1,λ_2, weights. γ, the Learning rate. $iter$, number of iterations.
for $iter$ of training iterations **do**
  for N of iterations **do**
    for $i = 1, ..., b$ **do**
      Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$,random vector $\boldsymbol{z} \sim \mathbb{P}_z$, a random number $\epsilon \sim U[0, 1]$.
      $\hat{x} \leftarrow \epsilon \boldsymbol{x} + (1 - \epsilon) G(\boldsymbol{z})$
      $L^{(i)} \leftarrow D(G(\boldsymbol{z})) - D(\boldsymbol{x}) + \lambda_1 GP|_{\widehat{\boldsymbol{x}}} + \lambda_2 CT|_{\boldsymbol{x}', \boldsymbol{x}''}$
    end for
    $\theta_d \leftarrow Adam(\nabla_{\theta_d} \frac{1}{b} \sum_{i=1}^{b} L^{(i)}, \theta_d, \gamma)$
  end for
  Sample b random vectors $\left\{\boldsymbol{z}^{(i)}\right\}$
  $\theta_g \leftarrow Adam(\nabla_{\theta_g} \frac{1}{b} \sum_{i=1}^{b} -D(G(\boldsymbol{z}^{(i)})), \theta_g, \gamma)$
end for

the same manner, we find the second virtual point $\boldsymbol{x}''$ around $\boldsymbol{x}$ by applying the (stochastic) dropout again to the hidden layers of the discriminator, and denote by $D(\boldsymbol{x}'')$ the corresponding output.

Note that, however, it becomes impossible to compute the distance $d(\boldsymbol{x}', \boldsymbol{x}'')$ between the two virtual data points. In this work, we assume it is bounded by a constant and absorb the constant to M'. Accordingly, we tune M' in our experiments to take account of this unknown constant; the best results are obtained between $M' = 0$ and $M' = 0.2$. For consistency, we use $M' = 0$ to report all the results in this paper.

A consistency regularization. Our final consistency regularization takes the following form,

$$CT|_{\boldsymbol{x}', \boldsymbol{x}''} = \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_r} \left[\max \left(0, d\left(D(\boldsymbol{x}'), D(\boldsymbol{x}'')\right) + 0.1 \cdot d\left(D_-(\boldsymbol{x}'), D_-(\boldsymbol{x}'')\right) - M' \right) \right] \tag{5}$$

where $D(\boldsymbol{x}')$ is the output of the discriminator given the input $\boldsymbol{x}$ and after we apply dropout to the hidden layers of the discriminator. We envision this is equivalent to passing a "virtual" data point $\boldsymbol{x}'$ through the clean discriminator. We find that it slightly improves the performance by further controlling the second-to-last layer $D_-(\cdot)$ of the discriminator, *i.e.*, the $d\left(D_-(\boldsymbol{x}'), D_-(\boldsymbol{x}'')\right)$ term above.

This new consistent regularization $CT|_{\boldsymbol{x}', \boldsymbol{x}''}$ enforces the Lipschitz continuity over the data manifold and its surrounding regions, effectively complementing and improving the gradient penalty $GP|_{\widehat{\boldsymbol{x}}}$ used in the improved training of WGAN. Putting them together, our new objective function for updating the weigts of the discriminator is

$$L = \mathbb{E}_{\boldsymbol{z} \sim \mathbb{P}_z} \left[D(G(\boldsymbol{z})) \right] - \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_r} \left[D(\boldsymbol{x}) \right] + \lambda_1 GP|_{\widehat{\boldsymbol{x}}} + \lambda_2 CT|_{\boldsymbol{x}', \boldsymbol{x}''}. \tag{6}$$

Algorithm 1 shows the complete algorithm for learning a WGAN in this paper. For the hyper-parameters, we borrow $\lambda_1 = 10$ from Gulrajani et al. (2017) and use $\lambda_2 = 2$ for all our experiments no matter on which dataset. Another hyper-parameter is M' in eq. (5). As stated previously, $M' = 0$ in our experiments.

2.2 A SEAMLESS CONNECTION WITH A SEMI-SUPERVISED LEARNING METHOD

In this section, we extend the WGAN to a semi-supervised learning approach by drawing insights from two related works.

- Following (Salimans et al., 2016), we modify the output layer of the discriminator such that it has $K + 1$ output neurons, where K is the number of classes of interest and the additional neuron is reserved for contrasting the generated samples with the real data using the Wasserstein distance in the WGAN. We use a $(K + 1)$-way softmax as the activation function of the last layer.
- Following (Laine & Aila, 2016), we add our consistency regularization $CT|_{\boldsymbol{x}', \boldsymbol{x}''}$ to the objective function of the semi-supervised training, in order to take advantage of the effect of temporal ensembling.

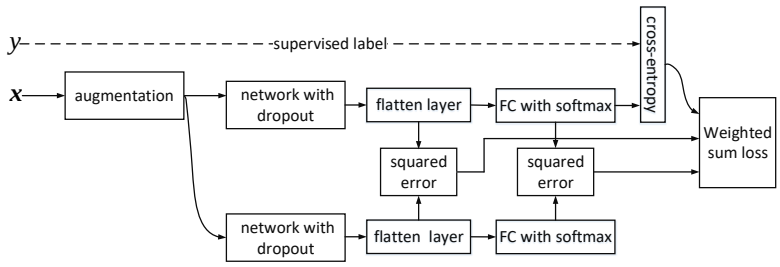


Figure 2: Framework for the semi-supervised training. For clarity, we have omitted the generator.

Figure 2 shows the framework for training the discriminator, with the objective function detailed below,

$$\begin{aligned} L_{\text{semi_dis}} = &- \mathbb{E}_{\boldsymbol{x},y\sim\mathbb{P}_{\text{x,y}}}\left[\log D(y|x)\right] - \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z}\left[\log D(K+1|G(z))\right] \\ &- \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}\left[\log(1-D(K+1|x))\right] + \lambda CT|_{\boldsymbol{x}',\boldsymbol{x}''}, \end{aligned} \tag{7}$$

where the first three terms are the same as in (Salimans et al., 2016), while the last consistency regularization is calculated after we apply dropout to the discriminator. The last term essentially leads to a temporal self-ensembling scheme to benefit the semi-supervised learning. Please see (Laine & Aila, 2016) for more insightful discussions about it.

For the Generator loss, we follow (Salimans et al., 2016) and the objective function is detailed below,

$$L_{\text{semi_gen}} = mean(|\mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_{\text{z}}}(D_(G(\boldsymbol{z})) - \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{r}}}(D_(\boldsymbol{x})|) \tag{8}$$

3 Experimental results

We conduct experiments on the prevalent MNIST (LeCun et al., 1998), and CIFAR-10 (Krizhevsky & Hinton, 2009) datasets. The code will be released for reproducibility of our results.

3.1 MNIST

The MNIST dataset provides 70,000 handwritten digits in total, and 10,000 of them are often left out for the testing purpose. Following (Gulrajani et al., 2017), we use only 1,000 of them to train the WGAN for a fair comparison with it. We use all the 60,000 training examples in the semi-supervised learning experiments and reveal the labels of 100 of them (10 per class). This setup is the same as in (Rasmus et al., 2015), and no data augmentation is used. Please see Appendix A for the network architectures of the generator and discriminator, respectively.

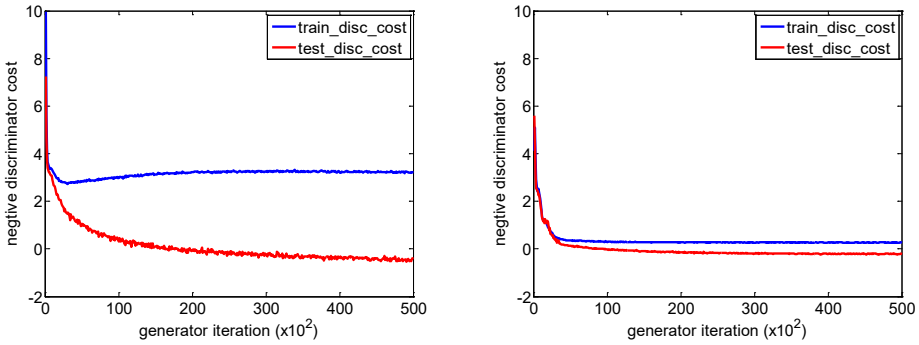
Qualitative results. Figure 3 shows the generated samples with improved training of WGAN by the gradient penalty (GP-WGAN), and ours with the consistency regularization (CT-GAN), respectively, after 50,000 generator iterations. It is clear that our approach gives rise to more realistic samples than GP-WGAN. The contrasts of our samples between the foreground and the background are in general sharper than those of GP-WGAN.

Overfitting. We find that our approach is less prone to overfitting. To demonstrate this point, we show the convergence curves of the discriminator's value functions by GP-WGAN and our CT-GAN in Figure 4. The red curves are evaluated on the training set and the blue ones are on the test set. We can see that the results on the test set become saturated pretty early in GP-WGAN, while ours can consistently decrease the costs on both the training and the test sets. This observation also holds for the CIFAR-10 dataset (cf. Appendix E).

Semi-supervised learning results. We compare our semi-supervised learning results with those of several competitive methods in Table 1. Our approach is among the best on this MNIST dataset.

(a) GP-WGAN (b) Ours

Figure 3: Images generated by (a) GP-WGAN (Gulrajani et al., 2017) and (b) Our CT-GAN, respectively.



(a) (b)

Figure 4: Convergence curves of the discriminator cost: (a) GP-WGAN and (b) Our CT-GAN.

3.2 CIFAR-10

CIFAR-10 (Krizhevsky & Hinton, 2009) contains 50,000 natural images of size 32×32. We use it to test two networks for the generative model: a small CNN and a ResNet (cf. Appendix A for the network structures). For the former, we use only 1,000 images to train the model, and we use the whole training set to learn the ResNet.

Qualitative results. Figure 5 contrasts the samples we generated to those by GP-WGAN when the generator is the small CNN. Figure 7 shows the results by a larger-scale ResNet. Our results are more photo-realistic.

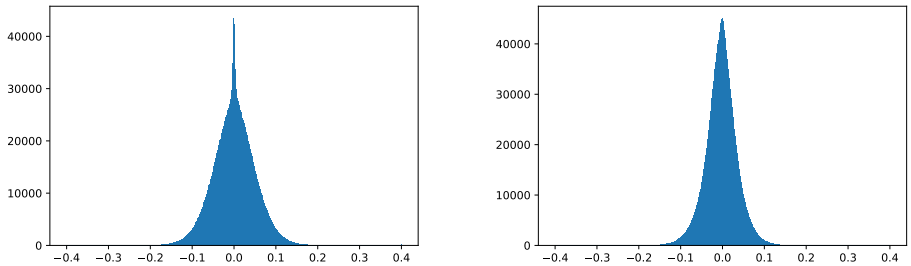
Additionally, we also draw the histograms of the discriminator's weights in Figure 6 after we train it using GP-WGAN and CT-GAN, respectively. It is interesting to see that ours controls the weights within a smaller and more symmetric range $[-0.67, 0.96]$ than the $[-2.00, 10.12]$ by GP-WGAN, partially explaining why our approach is less prone to overfitting.

Table 1: Comparing our semi-supervised learning approach with state-of-the-art ones on MNIST.

Method	**Test error (%)**
Ladder (Rasmus et al., 2015)	1.06±0.37
VAT (Miyato et al., 2017)	1.36
CatGAN (Springenberg, 2015)	1.39±0.28
Improved GAN (Salimans et al., 2016)	0.93 ± 0.065
Triple GAN (Li et al., 2017a)	0.91 ± 0.58
Our CT-GAN	$\mathbf{0.89 \pm 0.13}$

(a) GP-WGAN (inception score: 2.98 ± 0.11) (b) Our CT-GAN (inception score: 5.13 ± 0.12)

Figure 5: Generated samples by (a) GP-WGAN and (b) our CT-GAN. Here the generator is a small CNN. See Figure 7 for the samples by a ResNet.



(a) (b)

Figure 6: Histograms of the weights of the discriminators trained by (a) GP-WGAN and (b) CT-GAN, respectively.

Comparison of the inception scores. Finally, we compare our approach with GP-WGAN on the whole training set for both unsupervised and supervised generative-purpose task using ResNet. For model selection, we use the first 50,000 samples to compute the inception scores (Salimans et al., 2016), then choose the best model, and finally report the "test" score on another 50,000 samples. The experiment follows the previous setup in (Odena et al., 2016). From the comparison results in Tables 2, we conclude that our proposed model achieves the highest inception score in the c dataset, to the best of our knowledge. Correspondingly, the generated samples are shown in Figure 7.

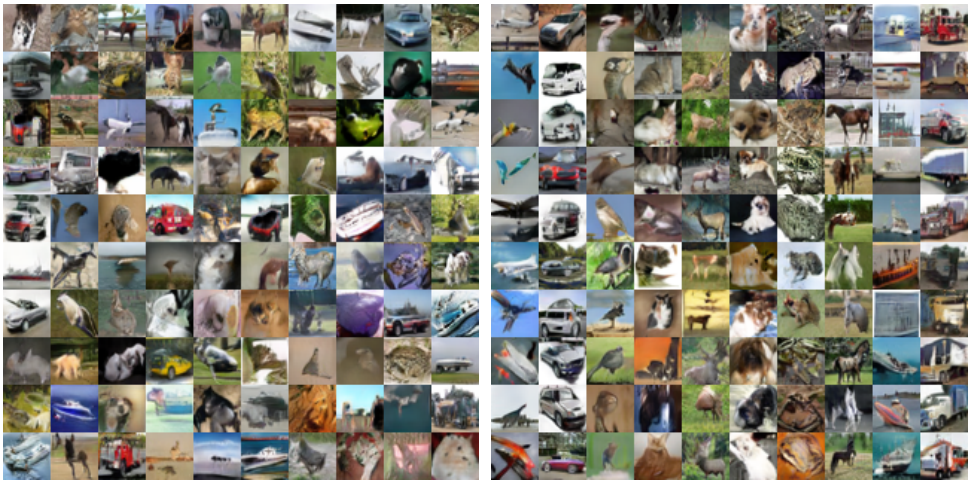
For the small CNN based generator, the inception scores of GP-WGAN and our CT-GAN are 2.98 ± 0.11 and 5.13 ± 0.12, respectively.

Table 2: Inception score and accuracy of different models on CIFAR-10

Method	**Supervised IS**	**Unsupervised IS**	**Accuracy(%)**
SteinGANs (Wang & Liu, 2016)	6.35	–	–
DCGANs (Radford et al., 2015)	6.58	6.16 ± 0.07	–
Improved GANs (Salimans et al., 2016)	8.09 ± 0.07	–	–
AC-GANs (Odena et al., 2016)	8.25 ± 0.07	–	–
GP-WGAN (Gulrajani et al., 2017)	8.42 ± 0.10	7.86 ± 0.07	91.85
SGANs (Huang et al., 2016)	8.59 ± 0.12	–	–
ALI (Warde-Farley & Bengio, 2016)	–	5.34 ± 0.05	–
BEGAN (Berthelot et al., 2017)	–	5.62	–
EGAN-Ent-VI (Dai et al., 2017)	–	7.07 ± 0.10	–
DFM (Warde-Farley & Bengio, 2016)	–	7.72 ± 0.07	–
Our CT-GAN	**8.81±0.13**	**8.12±0.12**	**95.91**

Table 3: Comparing our semi-supervised learning approach with state-of-the-art ones on CIFAR-10

Method	**Test error (%)**
Ladder (Rasmus et al., 2015)	20.40 ± 0.47
VAT (Miyato et al., 2017)	10.55
TE (Laine & Aila, 2016)	12.16 ± 0.24
Teacher-Student (Tarvainen & Valpola, 2017)	12.31 ± 0.28
CatGANs (Springenberg, 2015)	19.58 ± 0.58
Improved GANs (Salimans et al., 2016)	18.63 ± 2.32
ALI (Dumoulin et al., 2016)	17.99 ± 1.62
Triple GAN (Li et al., 2017a)	16.99 ± 0.36
Improved semi-GAN (Kumar et al., 2017)	16.78 ± 1.80
Our CT-GAN	$\mathbf{9.98 \pm 0.21}$



(a) (b)

Figure 7: Generated samples by our ResNet model. (a). Generated samples by unsupervised model. (b). Generated samples by supervised model, each column corresponds to one class which is consistent with the CIFAR-10 dataset.

Semi-supervised learning. For the semi-supervised learning approach, we follow the standard training/test split of the dataset but use only 4,000 labels in the training. A regular data augmentation with flipping the images horizontally and randomly translating the images within [-2,2] pixels is used in our paper (No ZCA whitening). We report the semi-supervised learning results in Table 3. The mean and standard errors are are obtained by running the experiments 5 rounds. Comparing to several very competitive methods, ours is able to achieve state-of-the-art results. Notably, our CT-GAN outperfroms all the GAN based methods by a large margin.

Please see Appendix A for the network architectures and Appendix C for the ablation study of our algorithm.

4 Conclusion

In this paper, we present a consistency term derived from Lipschitz inequality, which boosts the performance of GANs model. The proposed term has been demonstrated to be an efficient manner to ease the over-fitting problem when data amount is limited. Experiments show that our model obtains the state-of-the-art accuracy and Inception score on CIFAR-10 dataset for both semi-supervised learning task and generative model.

REFERENCES

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

Zihang Dai, Amjad Almahairi, Philip Bachman, Eduard Hovy, and Aaron Courville. Calibrating energy-based generative adversarial networks. *arXiv preprint arXiv:1702.01691*, 2017.

Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pp. 1486–1494, 2015.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft, and Serge Belongie. Stacked generative adversarial networks. *arXiv preprint arXiv:1612.04357*, 2016.

Daniel Jiwoong Im, Chris Dongjoo Kim, Hui Jiang, and Roland Memisevic. Generating images with recurrent adversarial networks. *arXiv preprint arXiv:1602.05110*, 2016.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pp. 3581–3589, 2014.

Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

Abhishek Kumar, Prasanna Sattigeri, and P Thomas Fletcher. Improved semi-supervised learning with gans using manifold invariances. *arXiv preprint arXiv:1705.08850*, 2017.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Chongxuan Li, Kun Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. *arXiv preprint arXiv:1703.02291*, 2017a.

Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. *arXiv preprint arXiv:1705.08584*, 2017b.

Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1718–1727, 2015.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *arXiv preprint arXiv:1704.03976*, 2017.

Richard Nock, Zac Cranko, Aditya Krishna Menon, Lizhen Qu, and Robert C Williamson. f-gans in an information geometric nutshell. *arXiv preprint arXiv:1707.04385*, 2017.

Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pp. 271–279, 2016.

Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pp. 3546–3554, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

Ilya Sutskever, Rafal Jozefowicz, Karol Gregor, Danilo Rezende, Tim Lillicrap, and Oriol Vinyals. Towards principled unsupervised learning. *arXiv preprint arXiv:1511.06440*, 2015.

Antti Tarvainen and Harri Valpola. Weight-averaged consistency targets improve semi-supervised deep learning results. *arXiv preprint arXiv:1703.01780*, 2017.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Dilin Wang and Qiang Liu. Learning to draw samples: With application to amortized mle for generative adversarial learning. *arXiv preprint arXiv:1611.01722*, 2016.

David Warde-Farley and Yoshua Bengio. Improving generative adversarial networks with denoising feature matching. 2016.

Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger Grosse. On the quantitative analysis of decoder-based generative models. *arXiv preprint arXiv:1611.04273*, 2016.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

Appendices

APPENDIX A NETWORK ARCHITECTURES

Table 4 (MNIST) and Table 5 (CIFAR-10) detail the network architectures used in our classification-purpose CT-GAN, where the classifiers are the same as the widely used ones in most semi-supervised networks (Laine & Aila, 2016) except that we apply weight-norm rather than batch-norm. For the generators, we follow the network structures in GP-WGAN (Salimans et al., 2016), but we use lower dimensional noise (50D) as the input to the generator for CIFAR-10 in order not to reproduce the complicated images and instead shift the focus of the training to the classifier.

Table 4: Networks for semi-supervised learning on MNIST

Classifier C	**Generator G**
Input: Labels y, 28*28 Images $\boldsymbol{x}$,	Input: Noise 100 $\boldsymbol{z}$
Gaussian noise 0.3, MLP 1000, ReLU	MLP 500, Softplus, Batch norm
Gaussian noise 0.5, MLP 500, ReLU	MLP 500, Softplus, Batch norm
Gaussian noise 0.5, MLP 250, ReLU	MLP 784, Sigmoid, Weight norm
Gaussian noise 0.5, MLP 250, ReLU	
Gaussian noise 0.5, MLP 250, ReLU	
Gaussian noise 0.5, MLP 10, Softmax	

Table 5: Networks for semi-supervised learning on CIFAR-10

Classifier C	**Generator G**
Input: Labels y, 32*32*3 Colored Image $\boldsymbol{x}$,	Input: Noise 50 $\boldsymbol{z}$
0.2 Dropout	MLP 8192, ReLU, Batch norm
3*3 conv. 128, Pad =1, Stride =1, lReLU, Weight norm	Reshape 512*4*4
3*3 conv. 128, Pad =1, Stride =1, lReLU, Weight norm	5*5 deconv. 256*8*8,
3*3 conv. 128, Pad =1, Stride =2, lReLU, Weight norm	ReLU, Batch norm
0.5 Dropout	
3*3 conv. 256, Pad =1, Stride =1, lReLU, Weight norm	
3*3 conv. 256, Pad =1, Stride =1, lReLU, Weight norm	5*5 deconv. 128*16*16,
3*3 conv. 256, Pad =1, Stride =2, lReLU, Weight norm	ReLU, Batch norm
0.5 Dropout	
3*3 conv. 512, Pad =0, Stride =1, lReLU, Weight norm	
3*3 conv. 256, Pad =0, Stride =1, lReLU, Weight norm	5*5 deconv. 3*32*32,
3*3 conv. 128, Pad =0, Stride =1, lReLU, Weight norm	Tanh, Weight norm
Global pool	
MLP 10, Weight norm, Softmax	

APPENDIX B HYPER-PARAMETERS AND OTHER TRAINING DETAILS

For the semi-supervised learning experiments, we set $\lambda = 1.0$ in Eq.(7) in all our experiments. For CIFAR-10, the number of training epochs is set to 1,000 with a constant learning rate of 0.0003. For MNIST, the number of training epochs is set to 300 with a constant learning rate of 0.003. The other hyper-parameters are exactly the same as in the improved GAN (Salimans et al., 2016).

For the experiments on the generative models, to have a fair comparison for our results with the existing ones, we keep the network structure and hyper-parameters the same as in the improved training of WGAN (Gulrajani et al., 2017) except that we add three *dropout* layers to some of the hidden layers as shown in tables 6 to 8.

Table 6: Generative model for MNIST

Discriminator	**Generator**
Input: 1*28*28 Image $\boldsymbol{x}$	Input: Noise $\boldsymbol{z}$ 128
5*5 conv. 64, Pad = same, Stride = 2, lReLU 0.5 Dropout	MLP 4096, ReLU Reshape 256*4*4
5*5 conv. 128, Pad = same, Stride = 2, lReLU 0.5 Dropout	5*5 deconv. 128*8*8 ReLU, Cut 128*7*7
5*5 conv. 256, Pad = same, Stride = 2, lReLU 0.5 Dropout	5*5 deconv. 64*14*14 ReLU
Reshape 256*4*4 ($D_{_}$)	5*5 deconv. 1*28*28
MLP 1 (D)	Sigmoid

Table 7: Generative model for CIFAR-10

Discriminator	**Generator**
Input: 3*32*32 Image $\boldsymbol{x}$,	Input: Noise $\boldsymbol{z}$ 128
5*5 conv. 128, Pad = same, Stride = 2, lReLU 0.5 Dropout	MLP 8192, ReLU, Batch norm Reshape 512*4*4
5*5 conv. 256, Pad = same, Stride = 2, lReLU 0.5 Dropout	5*5 deconv. 256*8*8 ReLU, Bach norm
5*5 conv. 512, Pad = same Stride = 2, lReLU 0.5 Dropout	5*5 deconv. 128*16*16 ReLU, Batch norm
Reshape 512*4*4 ($D_{_}$)	5*5 deconv. 3*32*32
MLP 1 (D)	Tanh

Table 8: ResNet for CIFAR-10

Discriminator	**Generator**
Input: 3*32*32 Image $\boldsymbol{x}$	Input: Noise bmz 128
[3*3]*2 Residual Block, Resample = DOWN 128*16*16	MLP 2048 Reshape 128*4*4
[3*3]*2 Residual Block, Resample = DOWN 128*8*8 0.2 Dropout [3*3]*2 Residual Block, Resample = None 128*8*8 0.5 Dropout [3*3]*2 Residual Block, Resample = None 128*8*8 0.5 Dropout	[3*3]*2 Residual Block, Resample = UP 128*8*8 [3*3]*2 Residual Block, Resample = UP 128*16*16 [3*3]*2 Residual Block, Resample = UP 128*32*32
ReLU, Global mean pool ($D_{_}$)	3*3 conv. 3*32*32
MLP 1 (D)	Tanh

APPENDIX C ABLATION STUDY OF OUR APPROACH TO SSL

We run an ablation study of our approach to the semi-supervised learning (SSL). Thanks to the dual effect of the proposed consistency (CT) term, we are able to connect GAN with the temporal ensembling (TE) Laine & Aila (2016) method for SSL. Our superior results thus benefit from both of them, verified by the ablation study detailed in Table 9.

If we remove the CT term, the test error goes up to 14.98, signifying the effectiveness of the CT regularization.

If we remove GAN from our approach, it almost reduces to TE; in fact, all the settings here are the same as TE except that we use an extra regularization ($D_{\text{-}}(.,.)$ in CT) over the second-to-last layer. We can see that the error is still significantly larger than our overall method.

We use the weight normalization as in Salimans et al. (2016), which becomes a core constituent of our approach. The batch normalization would actually invalidate the feature matching in Salimans et al. (2016).

Finally, we also test the version without the regularization to the second-to-last layer and observe a little drop in the performance.

Table 9: Ablation study of our semi-supervised learning method

Method	**Test Error**
w/o CT	14.98±0.43
w/o GAN	11.98±0.32
w batch norm	–
w/o $D_{\text{-}}(.,.)$ over the second-to-last layer	10.70±0.24
Ours	9.98±0.21

APPENDIX D EXAMINING THE 1-LIPSCHITZ CONTINUITY

Norm of gradient. In our experiments, we find that although the GP-WGAN (Gulrajani et al., 2017) has applied a Lipschitz constraint in the form of the gradient penalty over the input sampled between a real data point and a generated one, the actual effect on the ℓ_2 norm of the gradient is not as good as our CT-GAN model in the real data points, as illustrated in Figure 1. We empirically verify this fact by Figure 8, which shows the ℓ_2 norms of the gradients of the discriminator with respect to the real data points. The closer to 1 the norms are, the better the 1-Lipschitz continuity is preserved. Figure 8 further demonstrates that our consistency (CT) regularization is able to improve GP-WGAN (Gulrajani et al., 2017).

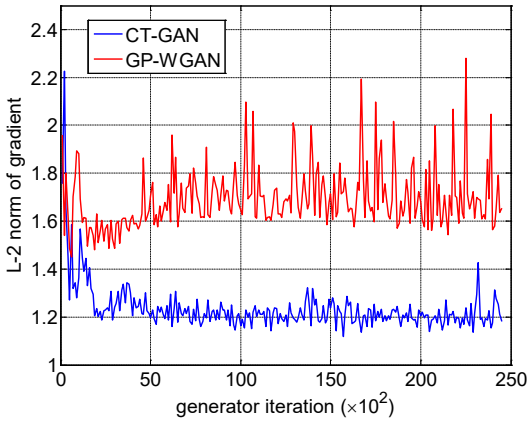


Figure 8: The maximum ℓ_2 norm of the gradients of the discriminator with respect to the input on CIFAR-10 testing set in each iteration of the training using 1000 CIFAR-10 training images.

Definition of the Lipschitz continuity. Additionally, we check how much the 1-Lipschitz continuity is satisfied according to its vanilla definition (cf. Eq. (3)). Figures 9 and 10 plot the CTs of Eq. (4) and Eq. (5), respectively, over different iterations of the training using 1000 CIFAR-10 images. Figure 10 is the actual CTs used to train the generative model. Figure 9 is drawn as follows. For every 100 iterations, we randomly pick up 64 real examples and split them into two subsets of the same size. We compute d(D(x1)-D(x2))/d(x1-x2) for all the (x1,x2) pairs, where x1 is from the first subset and x2 is from the second. The maximum of d(D(x1)-D(x2))/d(x1-x2) is plotted in Figure 9. We can see that the CT-GAN curve converges under a certain value much faster than GP-WGAN. The 1-Lipschitz continuity is better maintained by CT-GAN than GP-WGAN over the whole course of the training procedure on the manifold of the real data.

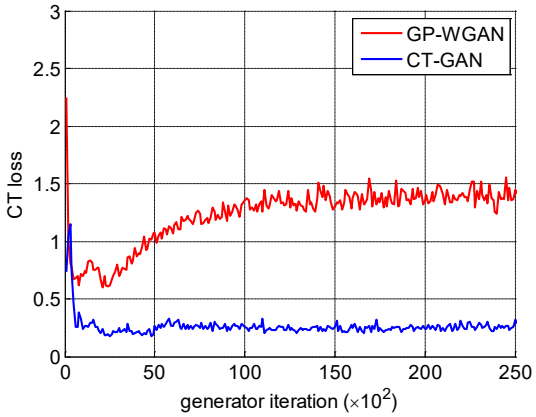


Figure 9: CT (cf. Eq. (4)) over different iterations of the training using 1000 CIFAR-10 images. In each iteration, we randomly pick up two real data points to compute the CT.

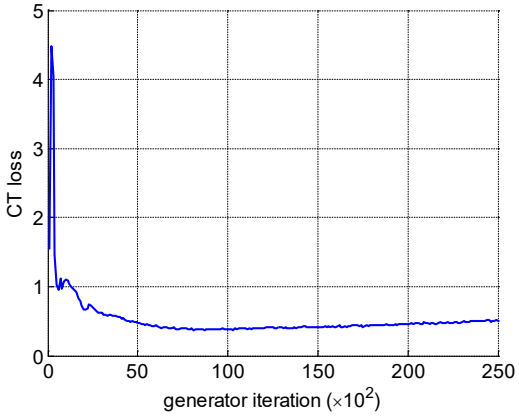
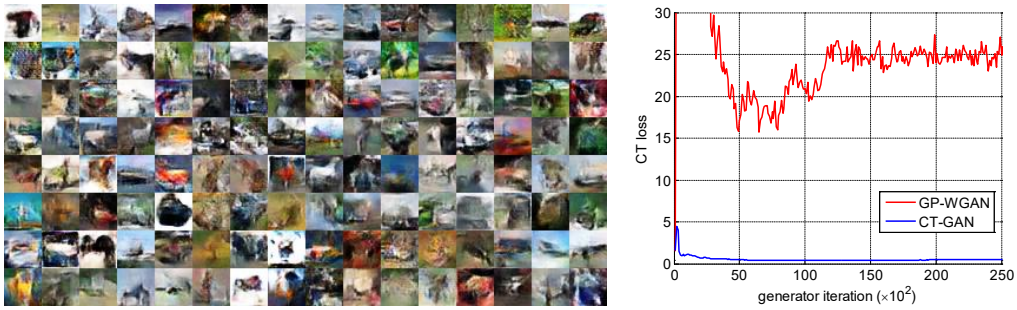


Figure 10: CT (cf. Eq. (5)) over different iterations of the training using 1000 CIFAR-10 images.

APPENDIX E GP-WGAN WITH DROPOUT

We run another experiment of GP-WGAN by adding the same dropout layers used in our method to the networks of GP-WGAN, denoted by GP-WGAN+dropout. This may help understand the contribution of our approach in preventing the overfitting problem — is it due to the CT regularization, or merely due to the dropout? This experiment is done by removing the CT term out of our value function for training WGAN using 1,000 CIFAR-10 images and yet keep the dropout layers.

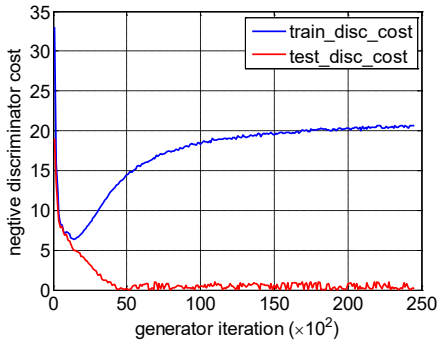
The Inception score of GP-WGAN+dropout is 4.29 ± 0.12, and the generated samples are shown in Figure 11(a). We also plot the curve of CT in the training procedure of GP-WGAN+dropout and contrast it to the curve of our CT-GAN in Figure 11(b). It is clear that GP-WGAN+dropout outperforms GP-WGAN, and our method outperforms GP-WGAN with a large margin. In other words, the dropout itself actually weakens GP-WGAN, while the CT regularization we proposed improves GP-WGAN. The improvement is because CT enforces the 1-Lipschitz continuity of the discriminator, and not because of the dropout used by CT.



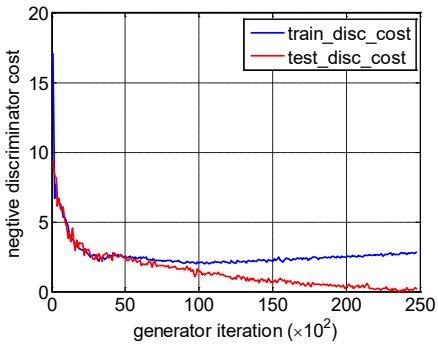
(a) Generated samples (inception score: 4.09 ± 0.12) (b) The curves of CT over different iterations

Figure 11: The results of GP-WGAN+dropout.

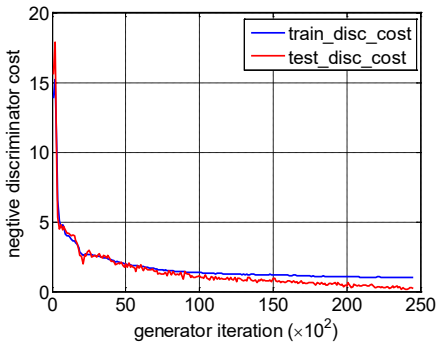
Finally, we plot the convergence curves of the discriminators' negative cost function learned by GP-WGAN, GP-WGAN+Dropout, and our CT-GAN in Figure 12. We can see that dropout is able to reduce the overfitting of GP-WGAN, but it is not as effective as our CT-GAN.



(a) GP-WGAN



(b) GP-WGAN+Dropout



(c) CT-GAN

Figure 12: Convergence curves of the discriminator cost: (a) GP-WGAN, (b) GP-WGAN+Dropout, and (c) CT-GAN (ours).