Training Generative Adversarial Networks Via Turing Test

Jianlin Su
School of Mathematics
Sun Yat-sen University
Guangdong, China
bojone@spaces.ac.cn

Abstract

In this article, we introduce a new mode for training Generative Adversarial Networks (GANs). Rather than minimizing the distance of evidence distribution $\tilde{p}(x)$ and the generative distribution $q(x)$, we minimize the distance of $\tilde{p}(x_r)q(x_f)$ and $\tilde{p}(x_f)q(x_r)$. This adversarial pattern can be interpreted as a Turing test in GANs. It allows us to use information of real samples during training generator and accelerates the whole training procedure. We even find that just proportionally increasing the size of discriminator and generator, it succeeds on 256x256 resolution without adjusting hyperparameters carefully.

1 Reviews of GANs

GANs has been developed a lot since Goodfellow's fist work (Goodfellow et al., 2014). The main idea of GANs is to train a generator $G(z)$ such that the generative distribution

$$q(x) = \int \delta(x - G(z))q(z)dz \tag{1}$$

will be a good approximation of the evidence distribution $\tilde{p}(x)$, while $q(z)$ is a prior distribution which will be standard normal distribution usually. Generally, the current GANs aim to minimize the distribution distance of $\tilde{p}(x)$ and $q(x)$.

1.1 Standard GANs

Here a series of GANs which are based on the Goodfellow's fist work are called Standard GANs (SGANs). Firstly, we fix generator and train a discriminator $T(x)$ by the following goal

$$\arg\max_T \mathbb{E}_{x\sim\tilde{p}(x)}[\log\sigma(T(x))] + \mathbb{E}_{x\sim q(x)}[\log(1-\sigma(T(x)))] \tag{2}$$

whose $\sigma(x) = 1/(1+e^{-x})$ means sigmoid activation. Then we fix discriminator and train a generator $G(z)$ by minimizing

$$\arg\min_G \mathbb{E}_{x\sim q(x)}[h(T(x))] = \arg\min_G \mathbb{E}_{z\sim q(z)}[h(T(G(z)))] \tag{3}$$

whose h can be any scalar function to make $h(log(t))$ be a convex function of variable t. Run two steps alternately and we may get a good generator finally.

Using variational method, we can show that the optimum solution of (2) is

$$\frac{\tilde{p}(x)}{q(x)} = \frac{\sigma(T(x))}{1-\sigma(T(x))} = e^{T(x)} \tag{4}$$

Preprint. Work in progress.

replace $T(x)$ in (3) with this result, we get

$$\begin{aligned} &\arg\min_G \mathbb{E}_{x\sim q(x)}\left[h\left(\log\frac{\tilde{p}(x)}{q(x)}\right)\right] \\ =&\arg\min_G \int q(x)\left[h\left(\log\frac{\tilde{p}(x)}{q(x)}\right)\right]dx \end{aligned} \tag{5}$$

Let $f(t) = h(\log(t))$, we can see the essential goal of SGANs is to minimize the f-divergence (Nowozin et al., 2016) between $\tilde{p}(x)$ and $q(x)$. Function f is constrained in convex function. Therefore, any function h making $h(\log(t))$ be a convex function is allowed to use, such as $h(t) = -t, h(t) = -\log\sigma(t), h(t) = \log(1-\sigma(t))$, which lead to the following loss of generator:

$$-T(x), \quad -\log\sigma(T(x)), \quad \log(1-\sigma(T(x))) \tag{6}$$

1.2 Wasserstein GANs

An important breakthrough in GANs is Wasserstein GANs (WGANs, Arjovsky et al. (2017)). Compared with SGANs, WGANs can improve the stability of learning and get rid of problems like mode collapse. The main idea of WGANs is to minimize the Wasserstein distance of $\tilde{p}(x)$ and $q(x)$, rather than f-divergence in SGANs. The Wasserstein distance

$$W(\tilde{p}(x), q(x)) = \inf_{\gamma\in\Pi(\tilde{p}(x),q(x))} \mathbb{E}_{(x,y)\sim\gamma}\|x-y\| \tag{7}$$

is an excellent metric of two distribution. $\gamma \in \Pi(\tilde{p}(x), q(x))$ means γ is any joint distribution of variable x and y whose marginal distributions are $\tilde{p}(x)$ and $q(y)$. With a dual transformation, Wasserstein distance can be rewritten as

$$W(\tilde{p}(x), q(x)) = \sup_{\|T\|_L\le 1} \mathbb{E}_{x\sim\tilde{p}(x)}[T(x)] - \mathbb{E}_{x\sim q(x)}[T(x)] \tag{8}$$

whose $T(x)$ is a scalar function and $\|T\|_L$ is Lipschitz norm of function T:

$$\|T\|_L = \max_{x\neq y}\frac{|T(x)-T(y)|}{\|x-y\|} \tag{9}$$

With these foundations, we can train the generator as a min-max game under the Wasserstein distance:

$$\arg\min_G \arg\max_{T,\|T\|_L\le 1} \mathbb{E}_{x\sim\tilde{p}(x)}[T(x)] - \mathbb{E}_{x\sim q(x)}[T(x)] \tag{10}$$

The first $\arg\max$ attempts to acquire a approximate function of Wasserstein distance and the second $\arg\min$ attempts to minimize the Wasserstein distance of $\tilde{p}(x)$ and $q(x)$.

One difficulty of WGANs is how to impose Lipschitz constraint $\|T\|_L \le 1$ on T, which currently has serveral solutions: weight clipping (Arjovsky et al., 2017), gradient penalty (Gulrajani et al., 2017) and spectral normalization (Miyato et al., 2018).

1.3 Problems

GANs has achieved a great success but there are still some problems waiting to be solved.

The distinct one is that training of GANs will be very unstable on large-scale datasets, such as 256x256 images and higher. Simply increasing the size of discriminator and generator can always not achieve this goal. It always needs certain tricks and well-designed hyperparameters for discriminator and generator, and even needs a large amount of computing resources (Karras et al., 2017; Brock et al., 2018; Peng et al., 2018).

2 A New GANs' Mode

There are two things in common between SGANs and WGANs: 1. They both attempts to minimize one kind of distribution distance between $\tilde{p}(x)$ and $q(x)$; 2. While updating generator, only fake samples from generative distribution is available.

So the updating of generator depends on whether discriminator can remember characteristics of real samples or not. In other words, generator just improve its production by the memory of discriminator, using no signal of real samples directly. It may be too hard to discriminator and lower the convergence rate of generator.

Here we demonstrate a new mode of GANs: to minimize distance of $\tilde{p}(x_r)q(x_f)$ and $\tilde{p}(x_f)q(x_r)$. This idea can make real images available while updating generator and can be integrated into all the current GANs. It is a new thought to train all generative models rather than one specific GANs.

2.1 Under SGANs

Define two joint distributions

$$P(x_r, x_f) = \tilde{p}(x_r)q(x_f), \quad Q(x_r, x_f) = \tilde{p}(x_f)q(x_r) \tag{11}$$

now we want to minimize the distance of $P(x_r, x_f)$ and $Q(x_r, x_f)$. Regard (x_r, x_f) as one whole random variable, and from (2) we get

$$\begin{aligned} &\arg\max_T \mathbb{E}_{(x_r,x_f)\sim P(x_r,x_f)}[\log\sigma(T(x_r,x_f))] + \mathbb{E}_{(x_r,x_f)\sim Q(x_r,x_f)}[\log(1-\sigma(T(x_r,x_f)))] \\ =&\arg\max_T \mathbb{E}_{(x_r,x_f)\sim \tilde{p}(x_r)q(x_f)}[\log\sigma(T(x_r,x_f)) + \log(1-\sigma(T(x_f,x_r)))] \end{aligned} \tag{12}$$

Then from (3) we have

$$\begin{aligned} &\arg\min_G \mathbb{E}_{(x_r,x_f)\sim Q(x_r,x_f)}[h(T(x_r,x_f))] \\ =&\arg\min_G \mathbb{E}_{x_r\sim\tilde{p}(x_r),x_f\sim q(x_f)}[h(T(x_f,x_r))] \\ =&\arg\min_G \mathbb{E}_{x_r\sim\tilde{p}(x_r),z\sim q(z)}[h(T(G(z),x_r))] \end{aligned} \tag{13}$$

Therefore, we can train a generative model by alternately running the following two steps:

$$\begin{aligned} &\arg\max_T \mathbb{E}_{(x_r,x_f)\sim\tilde{p}(x_r)q(x_f)}[\log\sigma(T(x_r,x_f)) + \log(1-\sigma(T(x_f,x_r)))] \\ &\arg\min_G \mathbb{E}_{x_r\sim\tilde{p}(x_r),x_f\sim q(x_f)}[h(T(x_f,x_r))] \end{aligned} \tag{14}$$

A natural choice of h leads to

$$\begin{aligned} &\arg\max_T \mathbb{E}_{(x_r,x_f)\sim\tilde{p}(x_r)q(x_f)}[\log\sigma(T(x_r,x_f)) + \log(1-\sigma(T(x_f,x_r)))] \\ &\arg\max_G \mathbb{E}_{(x_r,x_f)\sim\tilde{p}(x_r)q(x_f)}[\log(1-\sigma(T(x_r,x_f))) + \log\sigma(T(x_f,x_r))] \end{aligned} \tag{15}$$

2.2 Under WGANs

Corresponding to (8), we can estimate Wasserstein distance between $P(x_r, x_f)$ and $Q(x_r, x_f)$ by

$$\begin{aligned} &W(P(x_r,x_f), Q(x_r,x_f)) \\ =&\sup_{\|T\|_L\le 1} \mathbb{E}_{(x_r,x_f)\sim P(x_r,x_f)}[T(x_r,x_f)] - \mathbb{E}_{(x_r,x_f)\sim Q(x_r,x_f)}[T(x_r,x_f)] \\ =&\sup_{\|T\|_L\le 1} \mathbb{E}_{(x_r,x_f)\sim\tilde{p}(x_r)q(x_f)}[T(x_r,x_f) - T(x_f,x_r)] \end{aligned} \tag{16}$$

Hence we can train a generative model by a new min-max game:

$$\arg\min_G \arg\max_{T,\|T\|_L\le 1} \mathbb{E}_{(x_r,x_f)\sim\tilde{p}(x_r)q(x_f)}[T(x_r,x_f) - T(x_f,x_r)] \tag{17}$$

It is a really pretty result, which allows us to use an exactly symmetrical target to train discriminator and generator.

2.3 Relate to Turing Test

There is a very intuitive interpretation for minimizing the distance of $\tilde{p}(x_r)q(x_f)$ and $\tilde{p}(x_f)q(x_r)$: Turing test (Turing, 1995).

As we known, Turing test is a test of a machine's ability to exhibit intelligent behavior equivalent to, or indistinguishable from, that of a human. The tester communicates with both the robot and the human in unpredictable situations. If the tester fails to distinguish the human from the robot, we can say the robot has (in some aspects) human intelligence.

How about it in GANs? If we sample x_r from real distribution $\tilde{p}(x_r)$ and x_f from fake distribution $q(x_f)$, then mix them. Can we identify where they come from? That is, how much difference between $\tilde{p}(x_r)q(x_f)$ and $\tilde{p}(x_f)q(x_r)$? A good generator means we have $\tilde{p}(x) \approx q(x)$ everywhere, so we can not distinguish $\tilde{p}(x_r)q(x_f)$ and $\tilde{p}(x_f)q(x_r)$, so does (x_r, x_f) and (x_f, x_r).

Therefore, to minimize the distance of $\tilde{p}(x_r)q(x_f)$ and $\tilde{p}(x_f)q(x_r)$ is like a Turing test in GANs. We mix real samples and fake samples such that discriminator has to distinguish them by pairwise comparison and generator has to improve itself by pairwise comparison.

We call GANs in this mode as Turing GANs (T-GANs), correspondingly, (14) as T-SGANs and (17) as T-WGANs.

3 Related Works

Both (14) and (17) allow optimizer to obtain the signal of real samples directly to update generator. Formally, compared with SGANs and WGANs, the discriminator of T-GANs is two-variables function which needs both real and fake sample as inputs. It means that discriminator needs a pairwise comparison to make a reasonable judgement.

This idea fistly occurs in RSGANs (Jolicoeurmartineau, 2018). Our result can be regarded as an expansion of RSGANs. Just define $T(x_r, x_f) \triangleq T(x_r) - T(x_f)$ in (15), with $1 - \sigma(x) = \sigma(-x)$ we can obtain RSGANs:

$$\begin{aligned} &\arg\max_T \mathbb{E}_{(x_r,x_f)\sim\tilde{p}(x_r)q(x_f)}[\log\sigma(T(x_r) - T(x_f))] \\ &\arg\max_G \mathbb{E}_{(x_r,x_f)\sim\tilde{p}(x_r)q(x_f)}[\log\sigma(T(x_f) - T(x_r))] \end{aligned} \tag{18}$$

RSGANs have demonstrate some potential to improve GANs and we will demonstrate more efficient and sustainable progress of T-GANs at the section 4.

However, RSGANs is not the first GANs which make real samples available during training generator. As far as I know, the first one is Cramer GANs (Bellemare et al., 2017), which is based on energy distance:

$$\arg\min_G \arg\max_{E,\|E\|_L\leq 1} \mathbb{E}_{x_{r,1},x_{r,2}\sim\tilde{p}(x_r),\, x_{f,1},x_{f,2}\sim q(x_f)}[f(E(x_{r,1}), E(x_{r,2}), E(x_{f,1}), E(x_{f,2}))] \tag{19}$$

whose E is an encoder network and

$$f(x_1, x_2, y_1, y_2) = \|x_1 - y_2\| + \|y_1 - x_2\| - \|x_1 - x_2\| - \|y_1 - y_2\| \tag{20}$$

and

$$\iiiint p(x_1)p(x_2)q(y_1)q(y_2)f(x_1, x_2, y_1, y_2)dx_1dx_2dy_1dy_2 \tag{21}$$

is called energy distance of $p(x)$ and $q(x)$. Cramer GAN is not a perfect and complete inference framework of generative models. In fact it seems like a empirical model and it does not work well on large-scale datasets. It need more samples for echo updating iteration which is computation intensive.

4 Experiments

Our experiments are conducted on CelebA HQ dataset (Liu et al., 2015) and cifar10 dataset (Krizhevsky & Hinton, 2009). We test both (14) and (17) on CelebA HQ of 64x64, 128x128 and 256x256 resolution. cifar10 is an additional auxiliary experiment to demonstrate T-GANs work better than current GANs.

Code was written in Keras (Chollet et al., 2015) and available in my repository[1]. The architectures of models were modified from DCGANs (Radford et al., 2015). And models were trained using Adam optimizer (Kingma & Ba, 2014) with learning rate 0.0002 and momentum 0.5.

Experiments on 64x64 and 128x128 resolution were run on a GTX 1060 and experiments on 256x256 resolution were run on a GTX 1080Ti.

4.1 Design of Discriminator

In theory, any neural network with double inputs x_r, x_f can be used as $T(x_r, x_f)$. But for simplicity, inspired by RSGANs, we design $T(x_r, x_f)$ into the following form:

$$T(x_r, x_f) \triangleq D(E(x_r) - E(x_f)) \tag{22}$$

whose $E(\cdot)$ is an encoder for input image and $D(\cdot)$ is a multilayer perception with hidden difference vector of $E(x_r), E(x_f)$ as input and a scalar as output. It can also be regarded as a relativistic discriminator comparing the hidden features of x_r and x_f, rather than comparing the final scalar output in RSGANs.

If we use T-SGANs (14), no constraints for T theoretically. But as we known, gradient vanishing usually occurs in SGANs and spectral normalization is an effective strategy to prevent it. Therefore, spectral normalization has been a popular trick to be added into discriminator, no matter SGANs or WGANs, so do T-SGANs and T-WGANs.

Our experiment demonstrates (14) and (17) have similar performance while spectral normalization is applied on their discriminator $T(x_r, x_f)$.

4.2 Result Analysis

On 64x64 resolution's experiments, we find T-SGANs and T-WGANs has a faster convergence rate than popular GANs, such as DCGANs, DCGANs-SN, WGANs-GP, WGANs-SN, RSGANs.

On 128x128 resolution, we find most popular GANs does not work or only work under very particular hyperparameters and convergences unsteadily, but T-GANs still work well and have the same convergence rate as on 64x64 resolution.

We even find that just proportionally increasing the size of discriminator and generator, it succeeds on 256x256 resolution. It is very incredible that training a generative adversarial model on high resolution does not need to adjust hyperparameters carefully under T-GANs framework.

4.2.1 Faster Convergence Rate

Figure 1 shows comparison of convergence rate of different GANs. All of these GANs has same architecture of discriminator and generator. And they actually have same final performance but different convergence rate. We found that T-SGANs and T-WGANs converges almost twice as fast as other GANs. other GANs need about 20k iterations to achieve the same performance as T-GANs do in 10k iterations.

It needs to be pointed out that WGAN-GP seems to have a similar performance like T-GANs but actually it needs more time for echo iteration. During echo iteration, we update discriminator 5 times and update generator 1 times while training WGAN-GP and WGAN-SN, but update discriminator 1 times and update generator 2 times while training other GANs (including T-SGANs and T-WGANs).

Experiments on cifar10 also demonstrates this conclusion further (Figure 2).

4.2.2 High Quality Generation

Now we will focus attention on high quality generation. On 128x128 resolution, we compare serveral GANs but few of them work well. Table 1 demonstrates T-GANs still work well while increasing resolution, only need to expand the size of models.

We also test T-GANs on 256x256 resolution and T-GANs also work well but all of others fail to do that (Figure 3). It is worth mentioning that no matter 64x64, 128x128 or 256x256 resolution, T-GANs

[1] `https://github.com/bojone/T-GANs`

Table 1: Final results of serveral GANs on 128x128 resolution.

SGAN-SN	WGAN-SN	RSGAN-SN	TSGAN-SN	TWGAN-SN
				

would achieve a good performance after 12000 iterations. That is to say, large-scale does not affect the convergence of the T-GANs.

5 Conclusion

In this paper, we propose a new adversarial mode for training generative models called T-GANs. This adversarial pattern can be interpreted as a Turing test in GANs. It is a guiding ideology for training GANs rather than a specific GANs model. It can be integrated with current popular GANs such SGANs and WGANs, leading to T-SGANs and T-WGANs.

Our experiments demonstrate that T-GANs have good and stable performance on dataset varying from small scale to large scale. It suggests the signal of real samples is really important during updating generator in GANs. However, the mechanism of T-GANs to improve stability and convergence rate remains to be explored further.

References

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein gan.

Bellemare, M. G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., & Munos, R. (2017). The cramer distance as a solution to biased wasserstein gradients.

Brock, A., Donahue, J., & Simonyan, K. (2018). Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*.

Chollet, F., et al. (2015). *Keras.* `https://keras.io`.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial networks. *Advances in Neural Information Processing Systems*, *3*, 2672-2680.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of wasserstein gans.

Jolicoeurmartineau, A. (2018). The relativistic discriminator: a key element missing from standard gan.

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *Computer Science*.

Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images* (Tech. Rep.). Citeseer.

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of international conference on computer vision (iccv).*

Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral normalization for generative adversarial networks.

Nowozin, S., Cseke, B., & Tomioka, R. (2016). f-gan: Training generative neural samplers using variational divergence minimization.

Peng, X. B., Kanazawa, A., Toyer, S., Abbeel, P., & Levine, S. (2018). Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow.

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *Computer Science*.

Turing, A. M. (1995). *Computing machinery and intelligence*. MIT Press.

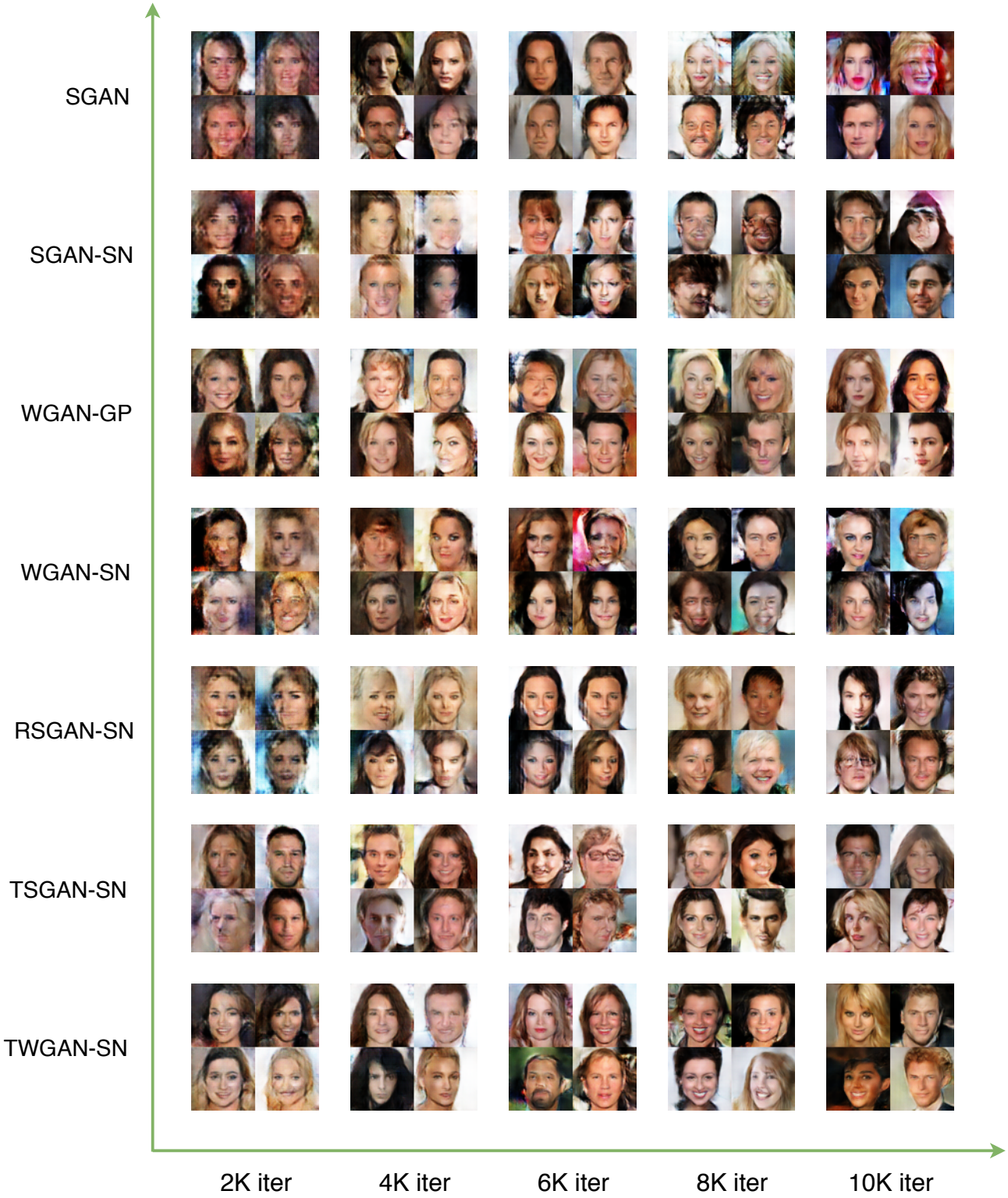


Figure 1: Comparison of convergence rate of different GANs on 64x64 CelebA. T-GANs converges almost twice as fast as other GANs. "-SN" means spectral normalization is added into discriminator.

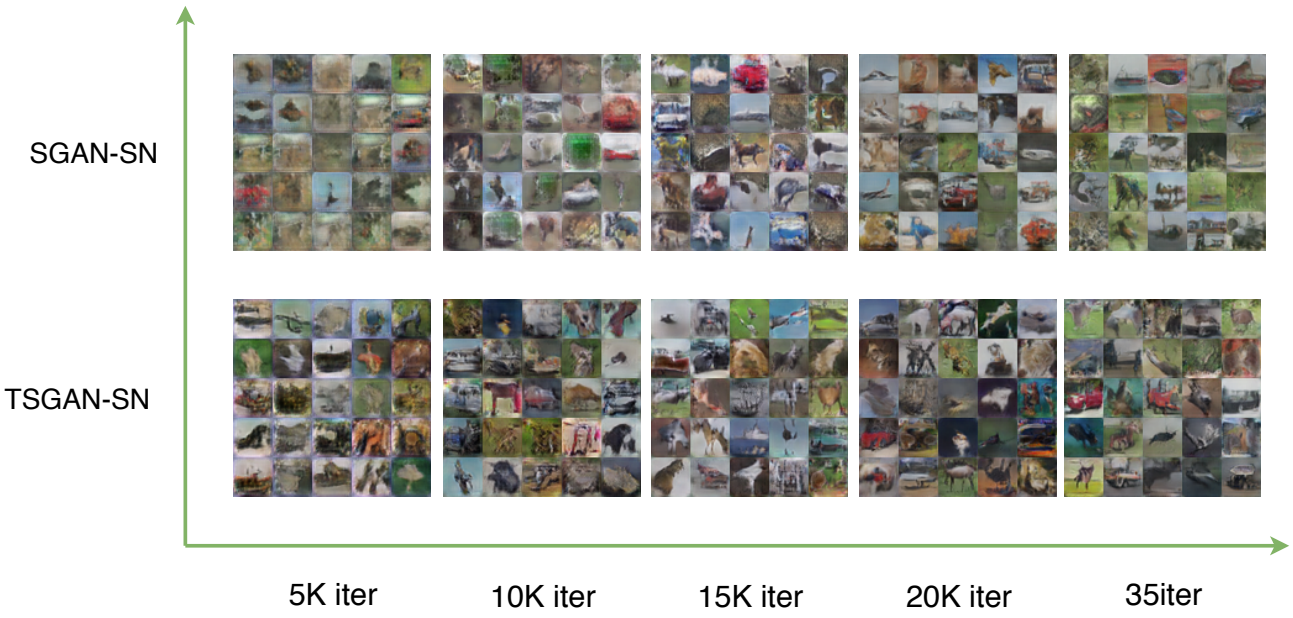


Figure 2: Comparison of convergence rate of different GANs on cifar10. It suggests that GANs under mode of Turing test has a better convergence than conventional. WGAN-SN performs like SGAN-SN and TWGAN-SN performs like TSGAN-SN, so we just show the result of SGAN-SN and TSGAN-SN.

Figure 3: Random samples of T-SGANs on 256x256 resolution.

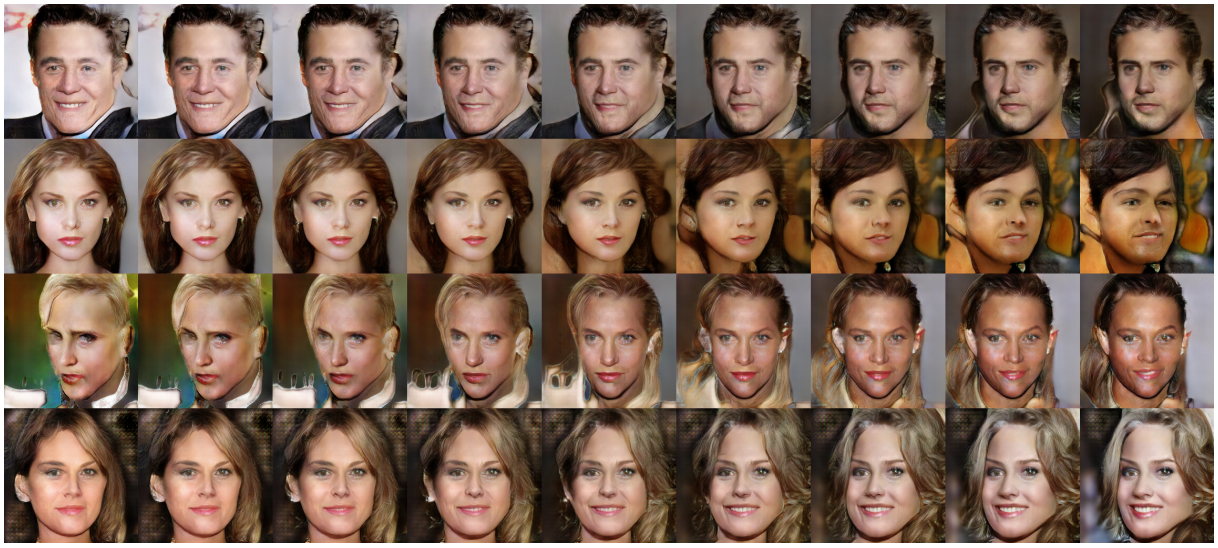
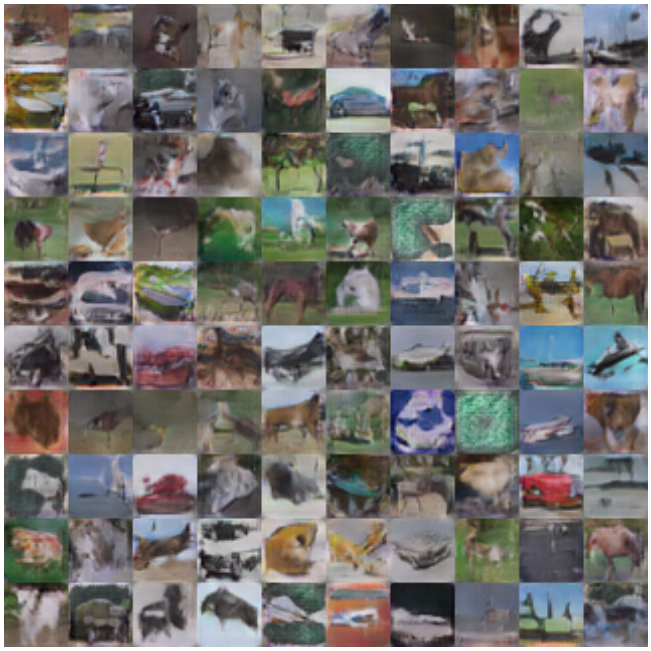
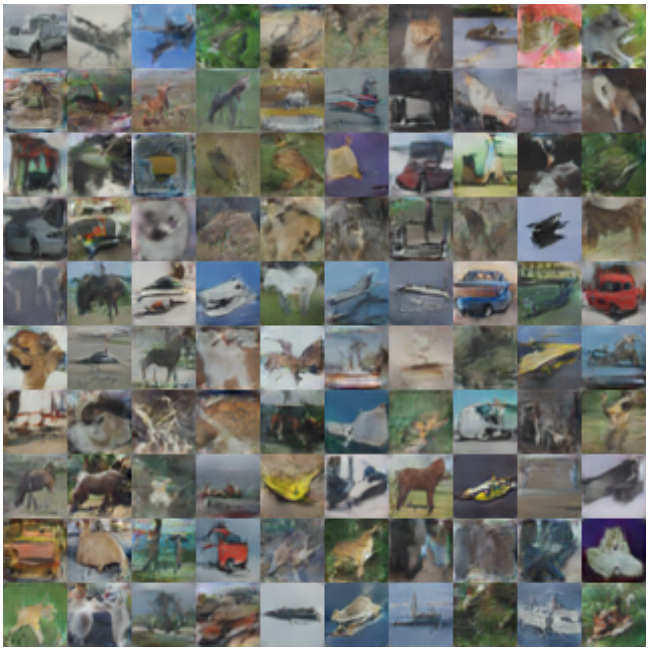


Figure 4: Random interpolation of T-SGANs on 256x256 resolution.



(a) Random samples from SGAN-SN



(b) Random samples from TSGAN-SN

Figure 5: Random samples from cifar10 (60K iteratons).