ADEL: Adaptive Distribution Effective-matching Method for Guiding Generators of GANs

Jungeun Kim[1,a][0000−0002−5341−726X], Jeongeun Park[2,a][0000−0001−9431−952X], and Ha Young Kim[2,b][0000−0001−5853−7589]

[1] Yonsei University, Department of AI, South Korea
jekim5418@yonsei.ac.kr
[2] Yonsei University, Graduate School of Information, South Korea
{park.je,hayoung.kim}@yonsei.ac.kr

Abstract. Research on creating high-quality, realistic fake images has engendered immense improvement in GANs. However, GAN training is still subject to mode collapse or vanishing gradient problems. To address these issues, we propose an adaptive distribution effective-matching method (ADEL) that sustains the stability of training and enables high performance by ensuring that the training abilities of the generator and discriminator are maintained in balance without bias in either direction. ADEL can help the generator's training by matching the difference between the distribution of real and fake images. As training is ideal when the discriminator and generator are in a balanced state, ADEL works when it is out of a certain optimal range based on the loss value. Through this, ADEL plays an important role in guiding the generator to create images similar to real images in the early stage when training is difficult. As training progresses, it naturally decays and gives model more freedom to generate a variety of images. ADEL can be applied to a variety of loss functions such as Kullback-Liebler divergence loss, Wasserstein loss, and Least-squares loss. Through extensive experiments, we show that ADEL improves the performance of diverse models such as DCGAN, WGAN, WGAN-GP, LSGAN, and StyleGANv2 upon five datasets, including low-resolution (CIFAR-10 and STL-10) as well as high-resolution (LSUN-Bedroom, Church, and ImageNet) datasets. Our proposed method is very simple and has a low computational burden, so it is expandable and can be used for diverse models.

Keywords: Deep learning · Generative adversarial networks · Training stabilization · Probability distribution matching

[a] These authors contributed equally.
[b] Corresponding author.
This work was supported by the Korea Agency for Infrastructure Technology Advancement(KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (22CTAP-C163908-02). This work was also supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-01361, Artificial Intelligence Graduate School Program (Yonsei University)).

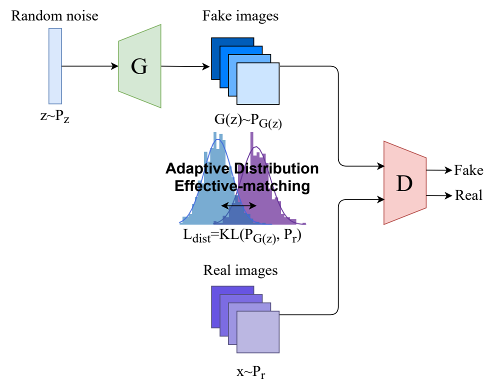


Fig. 1: Overall framework of ADEL. G denotes a generator and D is a discriminator. During training, when the discriminator loss is out of a certain optimal range, the distribution matching loss (L_{dist}) is applied, which encourages generator to match the actual data distribution. The generator trains based on this L_{dist} as well as L_G.

1 Introduction

Generative Adversarial Network (GAN), which is composed of a generator and discriminator, has been the focus of considerable research since it was first proposed by Ian Goodfellow [1]. GANs have demonstrated remarkable achievements in the field of computer vision [2–6] and continue to develop rapidly by being applied in the fields of natural language processing [7,8] and audio generation [9,10]. However, they still face many challenges. In particular, it is difficult to train a GAN stably, and mode collapse or vanishing gradient can often occur. To address these challenges, several methods have been devised to allow the model to learn the boundaries of the entire data distribution evenly and to keep it memorized [11–13].

Furthermore, helping the discriminator and the generator to learn evenly when training the GAN can considerably improve its performance. Therefore, various methods have tried to balance the training between the discriminator and generator [14–17]. For example, Wasserstein-GAN [18] applied Earth-Mover (EM) distance to a GAN and tried to stabilize it by clipping weights during back-propagation. BEGAN [19] used the auto-encoder instead of the discriminator to match the distribution of loss to allow the two networks to be in equilibrium. For EBGAN [19] and χ^2GAN [20], the distribution of data was matched to achieve Nash equilibrium during training. However, despite these efforts, maintaining training balance between a generator and discriminator is still challenging. In addition, most of these studies focus on adjusting the discriminator, or controlling the data distribution.

We propose a new training method, adaptive distribution effective-matching method (ADEL), to stabilize GAN training and improve its performance. Fig-

ure 1 shows our overall framework. Through the newly introduced distribution matching loss (L_{dist}), ADEL encourages the training of the generator by matching the distribution between the real images and the fake images created by the generator. By providing hints to the generator about the real image distribution so that the performance of the two networks is not skewed to one side, ADEL guides the GAN to create images that are more "realistic". However, this distribution matching method may interfere with the generator's degree of freedom. That is, the model to be trained may not cover the entire actual data distribution and may experience a loss of diversity, and vanishing gradient or a mode collapse may occur. Therefore, in this study, the level of distribution matching loss was adjusted as the training progressed, eventually allowing the generator to freely train without any constraints to cover the actual data distributions.
We demonstrate through comprehensive experiments that GAN training can be stabilized and better performance can be achieved by reducing the difference between the distribution of the fake images produced by the generator and the real images. In the Results section, we report the Inception Score (IS) [11], Fréchet Inception Score (FID) [21], and Kernel Inception Distance (KID) [22] scores to show quantitative results, while generated images demonstrating qualitative results are given in Experiments section.
Our main contributions are as follows:

- We propose a novel distribution matching method to guide generator's training of GANs. This method has a regularization effect, which helps the GAN to be trained stably.
- ADEL is a straightforward and effective method that can be easily applied to existing GAN models with a low computational burden.
- We show that ADEL can provide good performance in GANs with various losses and in small- as well as large-scale datasets.

2 Related Work

Various attempts have been made in many studies to address the challenges faced by GAN, such as that experiences mode collapse or the gradient of the generator is not updated because the loss of the discriminator rapidly converges to 0. Among them, some methods that change the model's structure or modify the loss function are stated below.

2.1 GAN architectures

DCGAN [15] dramatically improved a GAN's image generation ability by applying the structure of convolutional neural networks (CNNs) to the GAN. Owing to the advantages of CNN, DCGAN produces images with stable yet high resolution. Progressive GAN (PGGAN) [2] creates a high-resolution image by starting from a low-resolution image and gradually increasing the image resolution by adding network layers. Through the method of gradually stacking layers, learning stabilization and time reduction were achieved. However, PGGAN has limited ability to control specific detailed features, such as the color of the hair or eyes of the generated image.

This limitation is overcome by StyleGAN [4]. StyleGAN adds a mapping network and Adaptive Instance Normalization to the generator to create more realistic fake images. After that, StyleGANv2 [23] was recently introduced, which resolved the artifacts problem of StyleGAN, and contributed astonishing improvement of the generated images quality. SAGAN [24] applied a self-attention module in a GAN, which improved its performance in natural language processing considerably. This proves that it is possible to effectively model long-range dependencies within the internal representations of images, and thus this approach can outperform existing models. Zhang et al. also showed that spectral normalization stabilizes the training of GANs, and that the two time-scale update rule improves the training speed of the discriminator. BigGAN [5] also improved SAGAN structurally; they showed that a higher performance could be achieved only by increasing the number of channels and batch size of SAGAN. Furthermore, shared embedding, hierarchical latent space, and orthogonal regularization methods were added to provide even better performance. However, there is still room for improvement in stabilizing training. In addition to these studies, bi-directional GANs have been proposed, which introduce a bidirectional mechanism to generate latent space z from images [25]. There have been studies in which the number of generators or discriminators used is different [26] [27]. The adversarial auto-encoder takes GAN a step further by combining a variational auto-encoder with the GAN [28]. A recent study attempted to improve GAN's performance by applying a U-Net-based [29] architecture to the discriminator [30]. Another study achieved SOTA performance by applying the structure of a transformer to a GAN [31].

2.2 GAN loss functions

In the conventional GAN, the sigmoid cross entropy loss function was used for the discriminator. However, this loss function causes a vanishing gradient problem when updating the generator, which cannot be resolved by a structural change to the model. However, approaches that modified the loss function of the GAN appeared to alleviate this problem. Least squares GAN (LSGAN) [32] allows the generator to create images similar to real images by assigning a penalty to samples far from the decision boundary using least squares loss. By simply changing the loss function of GAN, LSGAN not only improves the quality of the generated image, but also enables stable training. While LSGAN uses the least squares loss, Banach Wasserstein GAN [33] simply replaces this least square norm with a dual norm, providing a new degree of freedom to select the appropriate Sobolev space, allowing WGAN to be placed in the Banach space. Relativistic GAN [34] modifies the loss as a relative evaluation by applying a probability indicating how realistic the real data is to the generated image. Another approach improves the performance of the model by applying the Hinge loss to the GAN [4, 35, 36]. As such, various studies have investigated the loss function of the GAN as a way to solve the vanishing gradient problem and eliminate mode collapse.

Table 1: Performance comparison of baseline models with and without ADEL method applied. Large and small values are preferred for ↑ and ↓ respectively

Model	Dataset	Method	IS (↑)	FID (↓)	KID (↓)	Memory (MB)	Training Time (sec/iteration)
DCGAN	CIFAR-10	Baseline	**5.47±0.09**	20.91	0.0100±0.0005	1604	0.07
		ADEL	5.32±0.23	**20.74**	**0.0085±0.0005**	2113	0.08
	STL-10	Baseline	4.70±0.24	63.86	0.0385±0.0009	2012	0.13
		ADEL	**5.03±0.19**	**60.37**	**0.0361±0.0009**	3945	0.16
WGAN	CIFAR-10	Baseline	5.22±0.07	20.23	**0.0076±0.0005**	2025	0.12
		ADEL	**5.23±0.17**	**19.26**	0.0081±0.0005	2031	0.13
	STL-10	Baseline	5.34±0.13	38.67	0.0194±0.0006	3795	0.23
		ADEL	**5.46±0.16**	**37.91**	**0.0190±0.0005**	3949	0.25
WGAN-GP	CIFAR-10	Baseline	5.24±0.08	23.75	0.0129±0.0006	2333	0.13
		ADEL	**5.46±0.09**	**23.08**	**0.0126±0.0006**	2462	0.13
	STL-10	Baseline	4.97±0.11	55.36	0.0360±0.0006	3807	0.23
		ADEL	**4.97±0.14**	**54.91**	**0.0336±0.0006**	3936	0.24
LSGAN	LSUN-B	Baseline	-	63.27	0.0663±0.0007	3864	0.74
		ADEL	-	**40.44**	**0.0359±0.0005**	4064	0.76
	ImageNet	Baseline	8.05±0.22	**46.30**	0.0354±0.0006	3864	1.01
		ADEL	**8.21±0.33**	46.32	**0.0343±0.0006**	4054	1.06

3 Methodology

The purpose of the generator and discriminator in GAN is to generate fake images that seem real and to distinguish between real and fake images, respectively. In general, while training a GAN, the discriminator converges faster than the generator, and this causes the vanishing gradient problem [37, 38]. Based on this observation, we propose an adaptive distribution matching method. Our method is not only easy to apply by adding few lines of code but also simple to understand.

3.1 ADEL

When training a GAN, the generator creates fake images using the latent vector as input while the discriminator D distinguishes between the real images and

Algorithm 1: Overall ADEL algorithm

Input: max iteration T, batch size m, dimension of latent vector d, learning rate η, weight of generator loss α, weight of discriminator loss β, global optimal solution δ, optimal range ϵ
Initialize θ_G and θ_D, i.e., the parameters of the generator and discriminator, respectively
for $c = 1 \cdots T$ **do**
  Sample random latent vector $z \sim N(0, 1)$, $z \in \mathbb{R}^d$
  Sample a batch of real images $r \sim P_r$, $r \in \mathbb{R}^m$
  Update $\theta_D \leftarrow \theta_D - \eta * \nabla_{\theta_D}(\beta * L_D)$
  $\gamma, L_{dist} \leftarrow 0$
  if $|L_D - \delta| > \epsilon$ **then**
    $P_{G(z)}, P_r \leftarrow Softmax(G(z)), Softmax(r)$
    $\gamma \leftarrow \exp^{-(c/T)}$
    $L_{dist} \leftarrow D_{KL}(P_{G(z)}, P_r)$
  else
    pass
  end if
  Update $\theta_G \leftarrow \theta_G - \eta * \nabla_{\theta_G}(\alpha * L_G + \gamma * L_{dist})$
end for

the fake images generated by the generator G, i.e., it identifies which images are real and which are fake. The loss function of a general GAN is as Eq. 1.

$$\begin{aligned} L_G &= \mathbb{E}_{z \sim P_z(z)}[log(1 - D(G(z)))] \\ L_D &= \mathbb{E}_{r \sim P_r(r)}[logD(r)] + \mathbb{E}_{z \sim P_z(z)}[log(1 - D(G(z)))] \end{aligned} \tag{1}$$

Here, r is a real data sample, P_r is a real data distribution, z is a random latent vector, P_z is a standard normal distribution, L_G is a generator loss, and L_D is a discriminator loss.

The discriminator only needs to determine whether a given image is real or fake (1 or 0), but the generator needs to generate entire images (e.g. 28×28×3 [H×W×C]), so there exists a difference in the relative difficulties of the two tasks. That is, the task of the generator that needs to generate a complex image is more laborious than that of the discriminator in deciding whether an input image is real. Furthermore, given that the generator's training depends only on the decision of the discriminator, the information that the generator can obtain is limited. In the view of the aforementioned two reasons, it becomes difficult for the two networks to maintain a balance during training. Therefore, we provide additional information about the distribution of real images to the generators performing tasks that are more difficult. To this end, the proposed method introduces a distribution matching loss (L_{dist}) described in Eq. 2, based on matching the distribution between real images and fake images to balance the evenness of the training of the generator and discriminator.

$$L_{dist} = \begin{cases} D_{KL}(P_{G(z)}||P_r) = \mathbb{E}_{x \sim P_{G(z)}}[log\frac{P_{G(z)}(x)}{P_r(x)}] & , \text{if } |L_D - \delta| > \epsilon \\ 0 & , \text{otherwise} \end{cases} \tag{2}$$

L_{dist} is computed using the Kullback-Liebler (KL) divergence (D_{KL}) to match between real and generated data distribution, $P_{G(z)}$. ϵ is a predefined constant and δ is the global optimal solution of each GAN model, i.e. the discriminator's value at $P_{G(z)} = P_r$. Those will be described in detail below.

In this study, we define the objective functions of the generator ($J(\theta_G)$) and the discriminator ($J(\theta_D)$) as given below in Eq. 3.

$$\begin{gathered} J(\theta_G) = \alpha * L_G(\theta_G) + \gamma * L_{dist}(\theta_G) \\ J(\theta_D) = \beta * L_D(\theta_D) \end{gathered} \tag{3}$$

α, β, and γ are weights to the generator, discriminator, and distribution matching loss. Also, θ_G and θ_D are the parameters of the generator and discriminator, respectively. During distribution matching, the ratio of L_G, L_D, and L_{dist} and whether to perform distribution matching is an essential issue. The value used as standard for distribution matching is also an important concern. To address these concerns, we propose a distribution matching method that considers not only the generator's information, using generated data distribution, but also the discriminator's information. Therefore, we introduce ϵ as the criterion for the difference between L_{dist} and δ. This serves as a threshold to measure how poorly the generator has trained. In Eq. 2, when the discriminator loss exceeds the range of $[\delta - \epsilon, \delta + \epsilon]$, L_{dist} exists. The general GAN loss consists of L_G and L_D, and the training of the discriminator dominates, so that L_D quickly converges to zero. However, in ADEL, when L_D is within the above-specified range, there exists an L_{dist} value. Therefore, in this scenario, the generator learns based on L_{dist} as well as L_G.

In Eq. 3, if the value of γ is increasing, then the generator will focus more on matching the distribution between real and fake data and interfere the degree of freedom available to the generator. Therefore, we design γ as an adaptive function based on the training iteration. The function γ is defined as follows.

$$\gamma := \gamma(c) = \exp^{-(c/T)} \tag{4}$$

where c is a current iteration and T is a max iteration. In Eq. 4, γ exponentially decreases as training progresses. As it is a large value at the beginning of the training, γ gives the generator a lot of information to guide it to generate similar images. When an iteration proceeds by decreasing the ratio of distribution matching loss, it gives the generator the degree of freedom to create various images, which is one of the goals of GAN models. To see the effect of L_{dist}, we fixed α and β as 1. In addition, the distribution of real and generated data (P_r and $P_{G(z)}$ in Eq. 2) are computed using a softmax function, as per Eq. 5. We applied the softmax function pixel by pixel to obtain the distribution of each image in every batch. That is, x_i is an image pixel value, N is the total number of pixels in an image and τ is set to 1.

$$P_d(x_i) = Softmax(x_i) = \frac{\exp^{\frac{x_i}{\tau}}}{\sum_{i=1}^{N} \exp^{\frac{x_i}{\tau}}} \text{ ,where } d \in \{r, G(z)\} \tag{5}$$

Algorithm 1 is the procedure for our proposed method, ADEL.

4 Experiments

In this section, we introduce our experimental environments and evaluations using various datasets and models. All experiments were conducted in a single GeForce RTX 3090 GPU.

4.1 Environments

Dataset We apply DCGAN, WGAN, and WGAN-GP on small-scale images (CIFAR-10 and STL-10) and LSGAN, and StyleGANv2 with large-scale images (LSUN-Bedroom, Church, and ImageNet). CIFAR-10 contains 32×32 images belonging to 10 classes, with a total of 60k images divided into 50k training data and 10k test data. STL-10 also has 10 classes, with 96×96 images, and consists of 500 training images and 800 test images for each class. ImageNet is a representative large-scale dataset. It has over 1M images belonging to 1000 classes. LSUN-Bedroom and LSUN-Church is also a large-scale dataset of 128×128 and 256×256 resolution, respectively.
Baselines We applied our proposed method to the following GAN models:

- DCGAN: GAN architecture with CNN, which replaced the fully-connected neural network.
- WGAN: WGAN explains the limitation of KL divergence loss and introduces Wasserstein loss which is a new type of loss.
- WGAN-GP: WGAN does not work well with momentum-type optimizers such as Adam. WGAN-GP address this issue by applying the gradient penalty.
- LSGAN: LSGAN proposed least square loss to solve the vanishing gradients problem in the traditional GAN's discriminator. This helps to generate a higher quality of image and stability during training.
- StyleGANv2: StyleGAN is a variation of GAN model, which is possible to being able to adjust the style when generating images. StyleGAN2 is a version that removed unnatural parts when generating images in StyleGAN and create higher quality images.

Training Details We trained WGAN using an RMSProp [?] and other models with an Adam optimizer ($\beta_1 = 0.5$, $\beta_2 = 0.999$) [40]. DCGAN and WGAN-GP are trained for 200k iterations. WGAN and LSGAN are trained for 1M iterations. The generator of DCGAN, WGAN and WGAN-GP produces 32×32 and 48×48 images for CIFAR-10 and STL-10, respectively. Like the LSGAN study, we generated 112×112 resolution images for LSNU-Bedroom and ImageNet. We set the learning rate as 0.0002 for DCGAN and LSGAN, 0.00005 for WGAN, and 0.0001 for WGAN-GP. All models are trained with a batch size of 64 using a single GPU. When applying our proposed method, we fixed all the hyper-parameters the same as baseline models, and we experimented by changing ϵ. The optimal ϵ value is obtained by grid search. For each model, the value showing the best performance was selected. In the case of StyleGANv2, the experimental environment is the same as all settings of StyleGANv2.

Evaluations We used three metrics, IS, FID, and KID as indicators to evaluate the performance of the GAN at generating various images and how realistic they were. IS is applicable only to class-conditional generation, so we do not calculate the LSUN-Bedroom score. To compute each metric score, 10k real images and 10k fake images, produced by the generator, were used. Metrics were calculated at a constant interval. Reported scores, when two or more metric scores were superior, were selected and compared. Since, StyleGAN [4] reflects quantitative as well as qualitative quality in the high-resolution dataset of the GAN model by suggesting a new metric called Perceptual Path Length (PPL). Therefore, we reported the PPL metric in this study.

5 Results

5.1 Base model performance

To examine the effectiveness of the ADEL method, we observed how the performance changes by applying ADEL to each model. As presented in the results in Table 1, the performance of each model is further improved when ADEL is applied regardless of the dataset. Except in three cases, i.e., the IS for DC-GAN's CIFAR-10, the KID for WGAN's CIFAR-10, and the FID for LSGAN's ImageNet, all three computational indicators consistently indicated improved performance in all models. In addition, DCGAN (ADEL) shows a noticeable performance improvement in the STL-10 dataset experiment. These results imply that when the generator is trained, ADEL guides the generator to produce images similar to real images by providing instructive information about the distribution of real images. In particular, in the LSUN-Bedroom dataset experiment of the LSGAN model, the FID and KID indicators showed performance improvements of 22.84 and 0.03, respectively. We also compared the difference between the computational costs and training time when ADEL was applied to baseline models and when it was not. The GPU usage memory increased from a minimum of 1.003 to a maximum of 1.961 times, and the training time took at least 0.003 seconds and a maximum of 0.05 seconds for one iteration. Thus, given that the additional workload or training time caused by applying ADEL is very small, ADEL can be applied to various models without any noticeable additional burden. Memory and training time is computed on a single RTX 3090 GPU.

In this study, only the FID metric showing the most change was presented. Figures 2, and 3 are graphs comparing the performance of each model and the

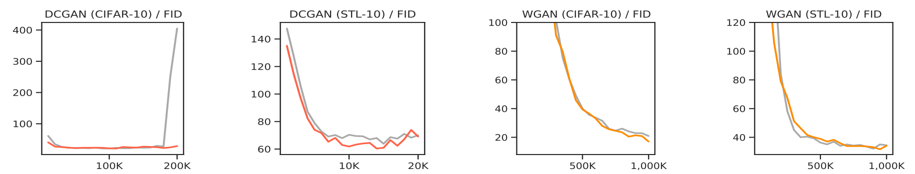


Fig. 2: Performance comparison between DCGAN and DCGAN (ADEL), WGAN and WGAN (ADEL) for all iterations.

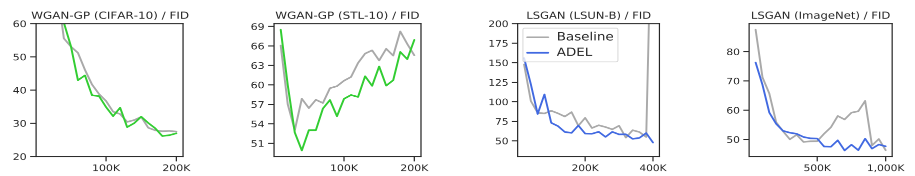


Fig. 3: Performance comparison between WGAN-GP and WGAN-GP (ADEL), LSGAN and LSGAN (ADEL) for all iterations.

model to which ADEL is applied at each evaluation iteration. In the graph, the colored line indicates the performance when ADEL is applied. It can be seen that each model has a section where FID is kept lower a specific point. It should be noted that mode collapse occurs after 400k iterations in the LSGAN (LSUN-Bedroom) experiment and 190k iterations in the DCGAN (CIFAR-10) experiment. However, the training progresses stably without mode collapse, when ADEL is applied during the same evaluation iteration. Furthermore, as can be observed in the LSGAN ImageNet graph, training is unstable at 600k in baseline models. By contrast, when ADEL is applied, it tends to be relatively stable.

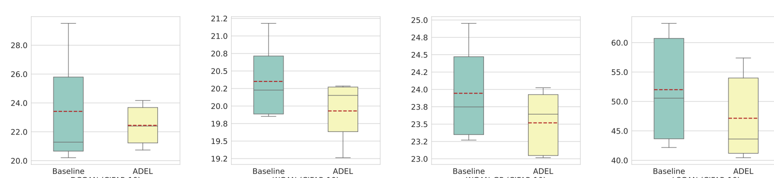


Fig. 4: Boxplots of FID scores of each model for five replicates. The red dotted line represents the average scores.

To show that ADEL's performance is consistent, we replicated experiments five times for all models and datasets. In Figure 4, we provide boxplots to compare the deviations of FID scores, which is a representative metric for GANs. It can be seen that ADEL demonstrates less deviation than the baselines. This was repeated for all metrics and models, but we could not include all the figures owing to page limitations. In the case of LSGAN ImageNet, where the difference between baseline and ADEL was the largest in all metrics, IS, FID, and KID values are 0.38, 3.82, and 0.0032, respectively. As shown in the figure, ADEL serves to make the model more robust because of reduced oscillation. Furthermore, although not presented in this paper, we find that ADEL has a low incidence of mode collapse. In particular, in the DCGAN CIFAR-10 experiment, mode collapse occurred with a 40% probability in the baseline model. By contrast, in ADE, training progressed stably without mode collapse. Thus, our proposed ADEL has the advantages of performance improvement and training stabilization. Besides these advantages, our proposed ADEL is a very simple approach with low computational burden. Hence, it is easily extendable to various GANs

Table 2: Performance comparison of StyleGANv2 with and without ADEL.

Model	Dataset	Method	PPL (↓)	FID (↓)
StyleGANv2	LSUN-Church	Baseline	342	3.86
	256×256	ADEL	**303**	**3.82**

5.2 Performance in the State-of-the-art model

We selected baseline models that serve as milestones with various loss functions. As these models are the basis of the latest GAN studies, we determined that if ADEL is applied in these models, even the latest derived studies would be covered. Additional experiments were conducted to prove that ADEL is also effectiveness in the SOTA model. Specifically, ADEL is applied to StyleGANv2 [23], a SOTA model in LSUN-Church, a single class and higher resolution dataset that is suitable for our experimental purpose. As presented in Table 2, StyleGANv2 improved FID and PPL. However, the KID score has not been officially reported from StyleGANv2, and it is difficult to make a fair comparison, so the score is not presented. According to Table 2, FID had a relatively marginal performance improvement. However, there was a very large performance improvement in PPL, the main metric of StyleGANv2.

5.3 Analysis of adjusting functions

ADEL adjusts the γ adaptively as per the discriminator's loss as in Eq. 4 when matching the distribution. Thus, it plays the role of guiding the generator by increasing the matching in the early stage of training and increasing the degree of freedom of the generator by ensuring a lower amount of matching towards the latter part of training to help generate various images. Before ADEL was chosen as the final matching method, we experimented with three different methods to compare the performance. ADEL's performance was the best of all. Table 3

Fig. 5: (Top) examples of StyleGANv2 without ADEL applied, and (bottom) ADEL is applied.

compares the performance of the proposed method ADEL with three other distribution matching methods using a different adjusting function and a condition. Detailed explanations are as follows:

- The linearly decaying distribution matching method (LDDM) for every iteration without considering the condition given in Eq. 2 is as follows:

$$\gamma(c) = 1 - c/T \tag{6}$$

- The exponentially decaying distribution matching method (EDDM) for every iteration without considering the condition of Eq. 2 and γ is the same as Eq. 4.
- A method to adaptively adjust the degree of distribution matching in the form of a penalty based on the discriminator loss value (DLDM) is given as follows:

$$\gamma(L_D) = \exp^{|L_D - \delta|} \tag{7}$$

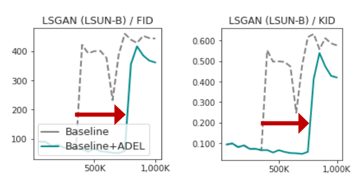
LDDM and EDDM are matching methods based on the current iteration versus max iteration, and DLDM is a method based on the current iteration's discriminator loss.

Table 3: Performance comparison between baseline, LDDM, EDDM, DLDM, and our proposed ADEL method.

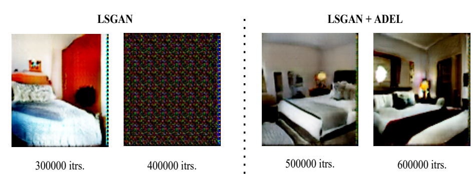
	CIFAR-10			STL-10		
Method	IS	FID	KID	IS	FID	KID
Baseline	5.47	20.91	0.0100	4.70	63.86	0.0385
LDDM	5.20	21.19	0.0087	4.82	65.89	0.0374
EDDM	**5.54**	20.82	0.0097	5.02	61.73	**0.0341**
DLDM	5.36	23.72	0.0108	4.82	62.81	0.0368
ADEL (ours)	5.32	**20.74**	**0.0085**	**5.03**	**60.37**	0.0361

ADEL uses a function where γ decays exponentially. That is, in the early stage of training, the large γ value makes matching large to provide additional information to the generator. As iteration goes, ADEL has a small γ value and reduces the matching so that the generator can create more diverse images. In order to check the effect of the exponential decaying function, we compared the LDDM method, which manually changes the max iteration linearly, and the EDDM method, which changes it exponentially without any restrictions on the condition. As observed from the results summarized in Table 3, the result of EDDM exceeds that of LDDM, and the result of LDDM does not even reach the baseline. This implies that adjusting the decaying function exponentially is more effective.

In addition, the performance of ADEL exhibits to EDDM in terms of applying conditions is compared. Applying a condition as in Eq. 2 has a more positive effect on performance improvement than when no such condition is applied. Finally, we compared ADEL and DLDM. DLDM is based on the discriminator loss; the discriminator loss tends to diverge toward the latter half of training, so the γ value is applied in the direction of increasing rather than continuously attenuating. Based on the results, ADEL's performance is much better than DLDM, which indicates that adjusting the decaying function based on the iteration is more effective than based on the loss value. These results indicates that it is helpful to give the model a degree of freedom to generate more diverse images at the end of training.



(a) Comparison between LSGAN and ADEL.



(b) Left of the dashed line: LSGAN with mode collapse. Right of the dashed line: ADEL is applied.

Fig. 6: Result when ADEL is applied to LSGAN where mode collapse has occurred.

5.4 Stabilizing and robustness effect of ADEL

In order to determine the stabilization effect of ADEL, we examined the kind of change that appeared by applying ADEL to LSGAN (LSUN-Bedroom) where mode collapse occurred. In Fig. 6 a), the gray dotted line represents the performance of the existing LSGAN baseline model, while the green line represents the changed performance after applying ADEL to the LSGAN baseline model before mode collapse occurs. Previously, mode collapse occurs at 400k evaluation iterations, but when ADEL is applied at the 350kth iteration, it can be confirmed that training proceeds stably without mode collapse until about 750k evaluation iterations. Thus, ADEL has the effect of making the model more robust to mode collapse to find the optimal performance. In Fig. 6 b), the images on the left side of the dashed lines are example images generated by baseline LSGAN, while those on the right are example images generated when LSGAN is continuously trained by applying ADEL before mode collapse occurs.

Figure 7 shows the training loss graph of DCGAN and DCGAN (ADEL). In the graph of the DCGAN baseline model, L_G diverges and L_D is unstable, so that the training of the discriminator does not proceed smoothly. By contrast, the training of the discriminator in DCGAN (ADEL) dominates, resulting in L_D almost converging to 0. This can make it difficult to train the generator. However, when L_D converges to 0, $|L_D - \delta| > \epsilon$ is satisfied, because we set ϵ to a very small value. Therefore, L_{dist} still exists. We demonstrate this experimentally in Fig. 7. According to Algorithm 1, the generator is trained based on L_G and L_{dist} so that the gradient is continuously updated. Therefore, when ADEL is applied, it is robust to the vanishing gradient problem.

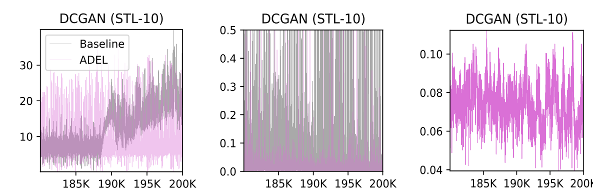


Fig. 7: Training loss graphs of L_G, L_D, and L_{dist} in order from left. The pink line is DCGAN (ADEL) and the gray line is DCGAN.

6 Conclusion

The proposed method ADEL is inspired by the fact that GAN's discriminator and generator demonstrate improved performance when they are trained in a balanced manner. ADEL works when the discriminator's loss exceeds a certain optimal range. At this time, the γ value, which controls the degree of ADEL, decreases exponentially. Therefore, it is applied strongly at the beginning of training to provide more guidance for the actual images to the generator, and it is applied weakly at the end of training to give the generator a degree of freedom to create more diverse images. To examine the effect of ADEL, five GAN models using different objective functions were selected as baselines, and the performances with and without ADEL applied to these models were compared. It was observed that the overall model performance improved when ADEL was applied. Furthermore, ADEL has the advantage that training is stable and robust to vanishing gradient problems. ADEL is not very burdensome because of its low computational cost and short training time.

References

1. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. arXiv (2014). https://doi.org/10.48550/ARXIV.1406.2661
2. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv (2018). https://doi.org/10.48550/ARXIV.1710.10196
3. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8798–8807, IEEE, Salt Lake City (2018)
4. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 4401–4410, IEEE, Long Beach (2019)
5. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. arXiv (2019). https://doi.org/10.48550/ARXIV.1809.11096
6. Lee, K.S., Tran, N.T., Cheung, N.M.:Infomax-gan: Improved adversarial image generation via information maximization and contrastive learning. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp. 3942–3952, Hawaii (2021)
7. Guo, J., Lu, S., Cai, H., Zhang, W., Yu, Y., Wang, J.: Long text generation via adversarial training with leaked information. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32, AAAI Press, New Orleans (2018)
8. Dash, A., Gamboa, J.C.B., Ahmed, S., Liwicki, M., Afzal, M.Z.:Tac-gan - text conditioned auxiliary classifier generative adversarial network. arXiv (2017). https://doi.org/10.48550/ARXIV.1703.06412
9. Gao, Y., Singh, R., Raj, B.: Voice impersonation using generative adversarial networks. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2506–2510, IEEE, Calgary (2018)
10. Kumar, R., Kumar, K., Anand, V., Bengio, Y., Courville, A.: Nu-gan: High resolution neural upsampling with gan. arXiv (2020). https://doi.org/10.48550/ARXIV.2010.11362
11. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. Advances in neural information processing systems 29, Morgan Kaufmann Publishers Inc, Barcelona (2016)
12. Metz, L., Poole, B., Pfau, D., Sohl-Dickstein, J.: Unrolled generative adversarial networks. arXiv (2017). https://doi.org/10.48550/ARXIV.1611.02163
13. Goodfellow, I.: Nips 2016 tutorial: Generative adversarial networks. arXiv (2017). https://doi.org/10.48550/ARXIV.1701.00160
14. Nowozin, S., Cseke, B., Tomioka, R.: f-gan: Training generative neural samplers using variational divergence minimization. Advances in neural information processing systems, vol. 29, pp. 271–279, Morgan Kaufmann Publishers Inc. (2016)
15. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv (2016). https://doi.org/10.48550/ARXIV.1511.06434
16. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. Advances in neural information processing systems, vol.30, Morgan Kaufmann Publishers Inc. (2017)

17. Wei, X., Gong, B., Liu, Z., Lu, W., Wang, L.: Improving the improved training of wasserstein gans: A consistency term and its dual effect, arXiv (2018). https://doi.org/10.48550/ARXIV.1803.01541
18. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv (2017). https://doi.org/10.48550/ARXIV.1701.07875
19. Berthelot, D., Schumm, T., Metz, L.: Began: Boundary equilibrium generative adversarial networks. arXiv (2017). https://doi.org/10.48550/ARXIV.1703.10717
20. . Tao, C., Chen, L., Henao, R., Feng, J., Duke, L.C.: Chi-square generative adversarial network. In: International conference on machine learning, PMLR (2018), pp. 4887–4896, Curran Associates, Inc. (2018)
21. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Proceedings of the Advances in neural information processing systems vol. 30, Morgan Kaufmann Publishers Inc., Long Beach (2017)
22. Bi nkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying mmd gans. arXiv (2018). https://doi.org/10.48550/ARXIV.1801.01401
23. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 8110–8119, IEEE, Virtual (2020)
24. Zhang, H., Goodfellow, I., Metaxas, D., Odena, A.: Self-attention generative adversarial networks. In: Proceedings of the International conference on machine learning (PMLR), pp. 7354–7363, (2019)
25. Donahue, J., Kr ahenb uhl, P., Darrell, T.: Adversarial feature learning. arXiv (2017). https://doi.org/10.48550/ARXIV.1605.09782
26. Hoang, Q., Nguyen, T.D., Le, T., Phung, D.: Multi-generator generative adversarial nets. arXiv (2017). https://doi.org/10.48550/ARXIV.1708.02556
27. Nguyen, T.D., Le, T., Vu, H., Phung, D.: Dual discriminator generative adversarial nets. arXiv (2017). https://doi.org/10.48550/ARXIV.1709.03831
28. Yu, X., Zhang, X., Cao, Y., Xia, M.: Vaegan: A collaborative filtering framework based on adversarial variational autoencoders. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 4206–4212, Morgan Kaufmann Publishers Inc., Macau (2019)
29. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Proceedings of the International Conference on Medical image computing and computer-assisted intervention, pp. 234–241, Springer (2015)
30. Schonfeld, E., Schiele, B., Khoreva, A.: A u-net based discriminator for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 8207–8216, IEEE, Virtual (2020)
31. Jiang, Y., Chang, S., Wang, Z.: Transgan: Two pure transformers can make one strong gan, and that can scale up. Advances in Neural Information Processing Systems, vol. 34, pp. 14745–14758, Morgan Kaufmann Publishers Inc., Virtual (2021)
32. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least squares generative adversarial networks. In: Proceedings of the IEEE international conference on computer vision, pp. 2794–2802, IEEE, Venice (2017)
33. Zhang, H., Goodfellow, I., Metaxas, D., Odena, A.: Self-attention generative ad- versarial networks. In: International conference on machine learning, PMLR, pp.7354–7363,, Curran Associates Inc. (2019)
34. Jolicoeur-Martineau, A.: The relativistic discriminator: a key element missing from standard gan. arXiv (2018) https://doi.org/10.48550/ARXIV.1807.00734

35. Lim, J.H., Ye, J.C.: Geometric gan, arXiv (2017) https://doi.org/10.48550/ARXIV.1705.02894
36. Kavalerov, I., Czaja, W., Chellappa, R.: A multi-class hinge loss for conditional gans. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp.1290–1299, Virtual (2021)
37. Arjovsky, M., Bottou, L.: Towards principled methods for training generative adversarial networks. arXiv (2017). https://doi.org/10.48550/ARXIV.1701.04862
38. Wiatrak, M., Albrecht, S.V., Nystrom, A.: Stabilizing generative adversarial networks: A survey. arXiv (2019). https://doi.org/10.48550/ARXIV.1910.00927
39. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, vol.12, pp.2121–2159, Microtome Publishing (2011)
40. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv (2017). https://doi.org/10.48550/ARXIV.1412.6980