IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 15, 2019, accepted December 4, 2019, date of publication December 18, 2019,
date of current version December 30, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2960461

An Adaptive Control Algorithm for Stable Training of Generative Adversarial Networks

XIAOHAN MA[1], RIZE JIN[2], KYUNG-AH SOHN[1], JOON-YOUNG PAIK[2], AND TAE-SUN CHUNG[1]

[1]Department of Computer Engineering, Ajou University, Suwon 16499, South Korea
[2]School of Computer Science and Technology, Tianjin Polytechnic University, Tianjin 300160, China

Corresponding author: Tae-Sun Chung (tschung@ajou.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant 2019R1F1A1058548, in part by the Natural Science Foundation of Tianjin under Grant 18JCYBJC44000, and in part by the Tianjin Science and Technology Program under Grant 19PTZWHZ00020.

ABSTRACT Generative adversarial networks (GANs) have shown significant progress in generating high-quality visual samples, however they are still well known both for being unstable to train and for the problem of mode collapse, particularly when trained on data collections containing a diverse set of visual objects. In this paper, we propose an Adaptive k-step Generative Adversarial Network (Ak-GAN), which is designed to mitigate the impact of instability and saturation in the original by dynamically adjusting the ratio of the training steps of both the generator and discriminator. To accomplish this, we track and analyze stable training curves of relatively narrow datasets and use them as the target fitting lines when training more diverse data collections. Furthermore, we conduct experiments on the proposed procedure using several optimization techniques (e.g., supervised guiding from previous stable learning curves with and without momentum) and compare their performance with that of state-of-the-art models on the task of image synthesis from datasets consisting of diverse images. Empirical results demonstrate that Ak-GAN works well in practice and exhibits more stable behavior than regular GANs during training. A quantitative evaluation has been conducted on the *Inception Score* (*IS*) and the *relative inverse Inception Score* (*RIS*); compared with regular GANs, the former has been improved by 61% and 83%, and the latter by 21% and 60%, on the CelebA and the Anime datasets, respectively.

INDEX TERMS Generative adversarial networks, image generation, adaptive algorithm, mode collapse.

I. INTRODUCTION

Generative adversarial networks (GANs) [2], [4], [5], [9], [12] are well known for being effective at synthesizing samples for various applications such as image generation [5], [12], [26], [28], [29], industrial design [20], speech generation [11], and natural language processing [2], [9]. The objective of GANs is to train a *generator model G* and a *discriminator model D* in parallel. GANs are typically multilayer perceptrons (MLPs) or convolutional neural networks (CNNs). The generator synthesizes the data distribution from a noise prior, whereas the discriminator is trained to extract discriminative features of real data. Specifically, the discriminator distinguishes between real and synthesized data produced by the generator and outputs the posterior probabilities that data are from the real data distribution, which are used as a guiding signal to update the parameters of discriminator and generator networks. Ideally, the process continues until real and synthesized data are indistinguishable.

In practice, however, GANs are well known both for being unstable to train and for the problem of mode collapse [1], [19]. Mode collapse is a situation where G generates most of the samples sharing some common properties, especially when trained on data collections containing a discrete sets of visual objects. Two major issues that compound the problems are: 1) the constantly unbalanced model capacities of the generator and discriminator during the training process, which prevents the generator from learning effectively; and 2) the lack of a computable convergence criterion.

The associate editor coordinating the review of this manuscript and approving it for publication was Qi Zhou.

Recently, several studies have focused on improving the stability of GANs, mostly by adopting a heuristic approach [11], [12], which is extremely sensitive to the training data and hard to apply to new domains [19]. Unlike previous work, this study tries to approach the unstable training problem of GANs by investigating an adaptive hyper-parameter learning method. Specifically, we propose to dynamically adjust the number of training steps of the G and D for each set of training iterations based on the learning curves from relatively simple and narrow datasets such as MNIST [7], from which GANs have been shown to produce very imperative results. Extensive experimental results show that the proposed procedure can significantly improve the stability of GAN trainings and generate much more recognizable objects in Anime and CelebA datasets [8].

This study is an extension of our previous work [25], which makes further the following improvements.

- In this extended work, the proposed Ak-GAN model dynamically adjusts the ratio (denoted as k) of the training steps of D and G every c iterations intead of every one iteration as in our previous work. c is a newly added hyperparameter to improve the generalizaiton of the adaptive model.
- We introduced some variants in which the adjustment of k either depends on more sophisticated criterias, such as loss values, or in progressive or immediate manners to control the degree of stabilizing the GAN training. We also investigated blur strategies which are a set of filters applied to the input of D to examine whether the blurring affects the training balance of D and G.
- We conducted extensive experiments to compare Ak-GAN models with some state-of-the-art models, using *Inception Score* (*IS*) [13] [16] and *relative inverse Inception Score* (*RIS*) [24]. Evaluation results demonstrate the effectiveness of the proposed training procedure in terms of convergence rate and image quality.

II. BACKGROUND

A. PRELIMINARY

In order to avoid the confusion, the parameters used throughout in this paper are defined as follows.

Definition 1: k: the training ratio of D to G. When $k < 1$, G is trained ($\frac{1}{k}$) steps with D one step, else if $k > 1$, D is optimized k steps with G one step.

The conventional GAN [4] is set to alternate between k steps of optimizing D and one step of G, generally with k being one. In our algorithm, the value of k is adaptively controlled using deviation in probabilities or loss values between running dataset and criterion dataset.

Table 1 shows the notations and their descriptions.

TABLE 1. Notations used in this paper.

Notation	Description
k	the training ratio of D to G
m	the batch size
z	the random noise variables
$p_z(z)$	the noise distribution
p_g	the generated data distribution
$p_{data}(x)$	the real data distribution
P_r	the probability that D classifies real data into real data
P_g	the probability that D classifies generated data into real data
L_D	the loss value of D
L_G	the loss value of G
P_{mr}	the probability that D classifies real data into real data when training on MNIST
P_{mg}	the probability that D classifies generated data into real data when training on MNIST
L_{mD}	the loss value of D when training on MNIST
L_{mG}	the loss value of G when training on MNIST

B. GENERATIVE ADVERSARIAL NETWORKS

GANs estimate generative models by means of an adversarial process in which a generator G is pitted against a discriminator D. The inputs of D come from two data distributions: real and synthesized, where the latter is generated by G. D is trained to maximize its ability to categorize a sample exactly as real or fake, whereas G is trained to minimize the ability of D to categorize a fake sample from synthesized data. In the original framework, the training procedure is defined as a two-player minimax game:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (1)$$

where G is a function that maps input *noise variables* $p_z(z)$ to a *generated data distribution* p_g. Whereas D is a function that maps a data space to scalar values, where each value represents the probability that specifies a sample came from *real data distribution* $p_{data}(x)$ rather than p_g. In GANs, G and D are treated as neural networks and trained concurrently on their objective function.

In practice, we use the same loss function of D as in the conventional GAN [4], and to optimize G we adapt more effective objective function [19]. They are represented in formula as:

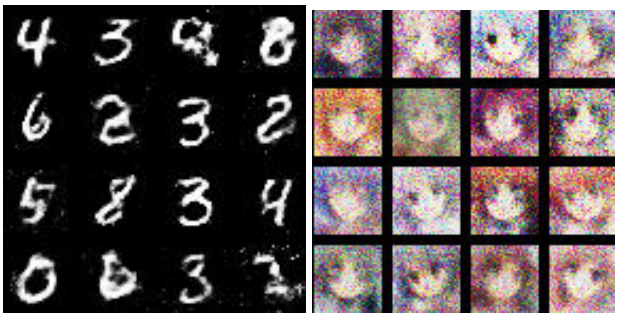
$$L_D = -\frac{1}{m}\sum_{i=1}^{m}[\log P(x^{(i)}; \theta_D) + \log(1 - P(G(z^{(i)}; \theta_G); \theta_D))] \quad (2)$$

$$L_G = -\frac{1}{m}\sum_{i=1}^{m}\log(P(G(z^{(i)}; \theta_G); \theta_D)) \quad (3)$$

where L_D and L_G are the loss functions, and θ_D and θ_G are the parameters of D and G, respectively. $x^{(i)}$ represents real data, whereas $G(z^{(i)})$ is the data generated by G with the arbitary noise $z^{(i)}$. Then $P(x^{(i)}; \theta_D)$ is the probability that D classifies real data $x^{(i)}$ as real label. $P(G(z^{(i)}; \theta_G); \theta_D)$ is the probability that D classifies generated data $G(z^{(i)})$ as real label. Hereinafter referred to as P_r and P_g.

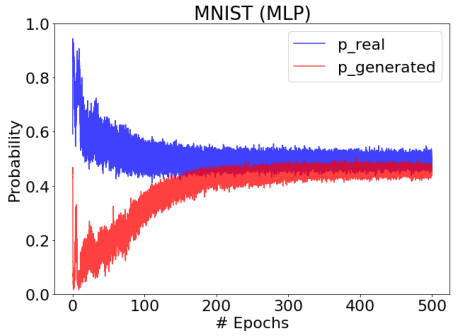
C. CHALLENGES AND LIMITATIONS OF GANS

Recently, GANs have gained tremendous attention because of their strength at producing compelling image synthesis,

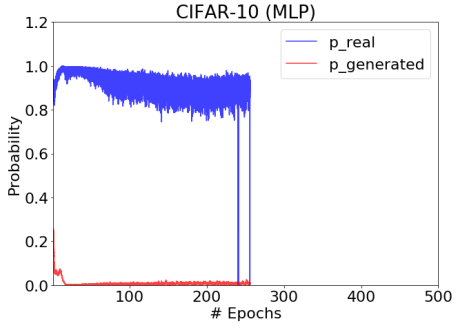


(a) the MNIST dataset (b) the Anime dataset

FIGURE 1. Effects of the original GAN from data collections with different levels of diversity.



(a) Learning curves on MNIST



(b) Learning curves on the diverse dataset

FIGURE 2. Learning curves for the conventional GAN model with multi-layer perceptrons (MLPs). The learning curves are measured by the probabilities of real and synthesized data (p_real and p_generated, respectively).

particularly when trained on image collections comprising relatively narrow domains, such as MNIST handwritten digits, as shown in Fig. 1(a). However, for diverse image collections, GANs generally yield less impressive results. For example, samples from models trained on the Anime dataset produce few recognizable objects, as shown in Fig. 1(b).

It is a considerable challenge to balance the convergence of the D and G. Training one of them too well results in instability. In practice, D tends to be overly trained during the training process, because D wins too easily in the beginning.

We estimated and tracked the training curves of MNIST and the diverse dataset, which are shown in Fig. Fig. 2. Specifically, Fig. 2 plots the curves of probabilities of real and synthesized data (P_r and P_g, respectively) estimated by D during the training process. Fig. 2(a) shows GANs achieve good stability when training on the narrow dataset MNIST. Note that in the beginning, a large gap exists between P_r and P_g. This gap gradually narrows during the training and finally converges at approximately 0.5. The good stability on MNIST is also reflected in the considerable quality of synthesized images, as shown in Fig. 1(a). Likewise, the poor visual results on the diverse dataset can best be explained by the constantly unstable training curves (their instability), as shown in Fig. 2(b). In other words, practical implementations dictate that the original GAN achieves a stable balance between G and D iterations and synthesizes images with higher quality on relatively simple datasets such as MNIST than on the diverse dataset.

D. RELATED WORK

To overcome the problem of *mode collapse*, several researchers have proposed to penalize the appearance of near-duplicate images during each training batch using heuristic-based approaches [13], [21]. Salimans *et al.* [13] sought to address this problem by training the discriminator in a semi-supervised fashion, granting the discriminator's internal representations knowledge of the class structure of the training data. Although this method increases synthesizing quality, it is less appealing as a tool for unsupervised learning. *Energy-based* GAN [21] has proposed modeling the D as an energy function, which enables the use of a wide variety of architectures and loss functions in addition to the binary classifier, making the model easy to train and more stable.

E-GAN [26] combines the GAN training with an evolutionary algorithm, which evolves a set of generators to adapt the discriminator and preserves the best generator for further training. PG-GAN [27] achieves high-resolution images by training GAN model in a progressive growing way to increase the resolution. SAGAN [28] imports the self-attention block into deeper GAN architecture to capture the global structure of images. Based on SAGAN, BigGAN [29] also uses different *residual block* for G and D, but trains the models with larger batches, larger parameters, and the *truncation trick*, achieving a new level of performance on ImageNet. These studies have shown the effects of increase in the depth of networks on some diverse datasets. However, our work focuses on adaptive control of the training process, which is orthogonal to the work of the increase of depth. For future work, it would be interesting to explore how to combine the adaptive control, very deep nets, and other techniques on complex datasets.

Consistent with our work, some authors have tried to use an adaptive approach [15], [18] to improve GANs. The AdaGAN [18] addresses the problem of *mode collapse* by incrementally adding new generators, which are trained using reweighted data at every step. Thus, the data generated by the mixture model can progressively cover all the modes of datasets. However, the generator of the mixture model is a mixture of single generator networks, and thus the latent representation becomes more complex with training.

Of particular interest to us is the work of ABC-GAN [15]. Its adaptive idea is also achieved by adjusting G and D steps based on a function of losses of D. A major downside of ABC-GAN is that it uses a manually fixed probability value, from which the ratio is sampled each time.

III. AN ADAPTIVE TRAINING PROCEDURE FOR GANS

To address the problem that G and D do not balance well on diverse datasets, we explore an adaptive training procedure to encourage convergence of GANs. Our method is similar in spirit to ABC-GAN [15] in the sense that it balances the generation capacities of GAN models by adjusting the ratio of D to G. Different from the adaptive controller used in ABC-GAN, which is based on a manually fixed probability value, our method automatically adjusts k by matching a set of well-trained learning curves (with different metrics) derived from relatively narrow datasets (e.g. MNIST). This section first explains the core idea and then describes the adaptive method in detail.

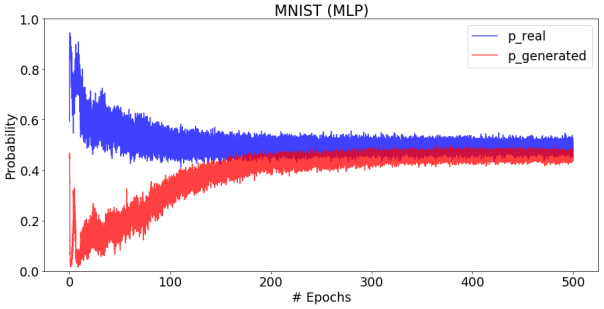
A. BASIC IDEA

Our main goal is to achieve a convincing performance on distinct datasets by matching the well-trained learning curves. We first describe the manner in which to obtain well-trained learning curves, and then present an algorithm that constrains the difference between the current value of control variables (e.g., posterior probabilities P_r and P_g, or loss values L_G and L_D) and the criterion (probabilities or loss values of the well-trained learning curve).

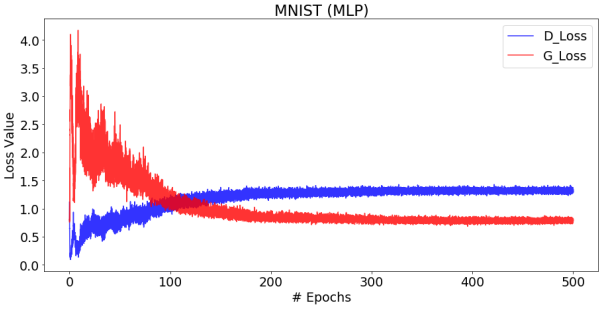
Because the original GAN object function is defined as a two-player game, determining the convergence of G and D is difficult, as the loss of G increases when that of D decreases. The learning curves are typically the efficient means to alert us as to how training is progressing. Therefore, we use the aforementioned learning curves as prototypes [22], which are considered to be more typical and central than others in the set of led-to-convergence curves. As shown in Fig. 3, the criterion learning curves consist of four measures: the posterior probabilities P_r and P_g when trained on MNIST dataset (expressed by P_{mr} and P_{mg}, respectively) and the loss costs L_G and L_D when trained on MNIST (represented by L_{mG} and L_{mD}, respectively). We use the ratio (k) of the training steps of D to G as the control variable. Instead of setting k to 1 as in the conventional GAN, we dynamically adjust the value of k to satisfy the constraints during the training on diverse datasets. k is controlled using the following inequality constraint.

$$\left|\frac{V_c - V_m}{V_m}\right| < \alpha \ (or\ \beta) \tag{4}$$

where V_c is P_r, P_g, L_G, or L_D obtained from the current training data, correspondingly, V_m is P_{mr}, P_{mg}, L_G or L_{mD} obtained from MNIST. Different from our previous work, we propose other algorithms which are controlled by P_r or L_D. α and β are thresholds that prescribe a limit to the range in which current values deviate from the criterion value.



(a) Learning curves for probabilities of real and generated data



(b) Learning curves for losses of the generator and discriminator

FIGURE 3. Learning curves for stable training.

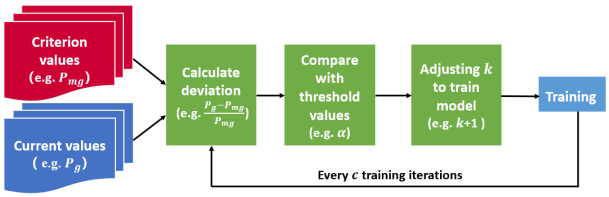
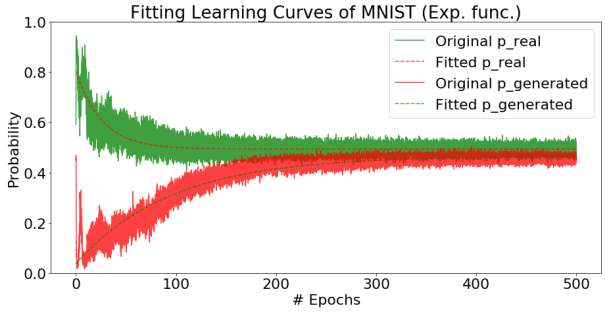


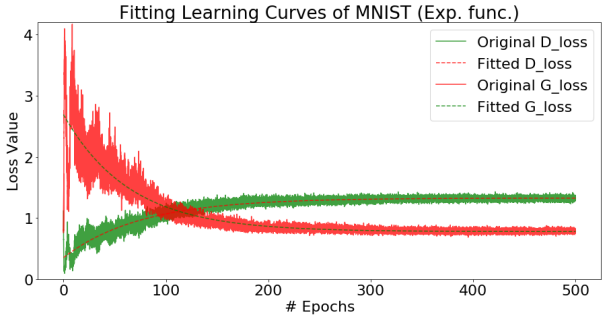
FIGURE 4. The adaptive control procedure of Ak-GAN.

Since the criterion learning curves for P_g or L_D show the same trend during the training process (as shown in Fig. 3), we use the same threshold α to limit the deviation. Similarly, we use another threshold β for updating P_r or L_G. We experimented with different α and β values and found that α to 0.2 and β to 0.05 achieve the best results.

We next provide an intuitive explanation of our adaptive methods before proceeding to the implementation details, as illustrated in Fig. 4. First, The Ak-GAN model is trained with the initial state of k being 1. Then, we pick the criterion values (for instance, P_{mg}) from the well-trained learning models and compare with the current values (in this case, P_g) every c iteration (c is set to 1 by default; we explore different c values in experiments), and then examine whether the deviation exceeds the constraint condition (in this case, $\left|\frac{P_g - P_{mg}}{P_{mg}}\right| < \alpha$). If it does, we update k in such a manner as to bias the next training toward the "weaker" of either G or D. Our experimental results indicate that the proposed procedure yields more stable training and improves sample generation. We designed two kinds of adaptive methods to adjust the k value: either with or without momentum. The adaptive method with momentum progressively adjusts the



(a) Fitted curves (power exponential functions) of probabilities of the real data (dotted line) and generated data (solid line)



(b) Fitted curves (power exponential function) of losses of the generator (dotted line) and discriminator (solid line)

FIGURE 5. **Regression curves (power exponential functions) of stable training.**

TABLE 2. **Coefficients of the exponential functions for fitting criterion curves of P_{mr}, P_{mg}, L_{mD}, and L_{mG}.**

	P_{mr}	P_{mg}	L_{mD}	L_{mG}
a	0.31068242	0.42990933	0.97477936	1.91548562
b	0.03853642	0.01056199	0.01321183	0.01552848
d	0.49340969	0.46643536	1.32625175	0.77667221

k value, and that without momentum changes the k value immediately, thus reducing the fluctuation of learning curves. In Appendix, we provide a more detailed description of the two methods, specifically on how to update the k value for the next iteration and the conditions of the update.

B. ADAPTIVE CONTROL USING FITTED CURVES

Note that the criterion learning curves shown in Fig. 5 fluctuate wildly during training. Instead of directly matching the curves, we used an approach to draw regression lines as guiding. In this approach, we employed two exponential functions, Eq. (5) is used to fit the curves of P_{mr} and L_{mG}, whereas Eq. (6) is used to fit curves of P_{mg} and L_{mD}. The coefficients of these functions are shown in Table 2.

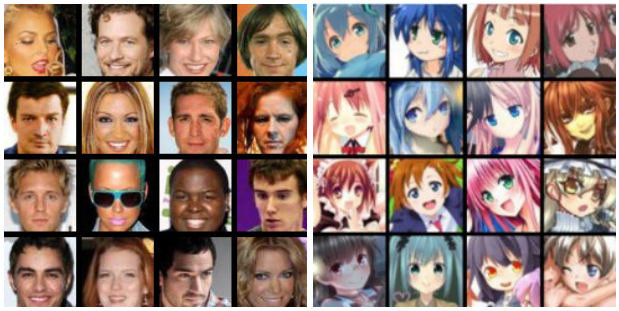
$$y = a * e^{-bx} + d \tag{5}$$

$$y = a * -(e^{-bx}) + d \tag{6}$$

IV. EXPERIMENTS

A. DATASETS

We performed our experiments on two datasets: CelebA [8], and the Anime face dataset. Details of each dataset are given as follows. The samples of two datasets are shown in Fig. 6.



(a) Samples from CelebA (b) Samples from Anime dataset

FIGURE 6. **Samples of two datasets.**

TABLE 3. **Architecture of a MLP-based generator.**

	Operation	Size
Input	Random noise	1×100
Feature	Fully connected layer, ReLU	1×150
Feature	Fully connected layer, ReLU	1×300
Generated image	Fully connected layer, tanh	$3\times$ image_h $\times$ image_w

1) CELEBA

The large-scale celeb faces attributes (CelebA) dataset contains 202,599 facial images of 10,177 celebrities. The images in this dataset are RGB color images containing diverse pose variations and cluttered backgrounds. In our experiments, each image was resized to 64×64.

2) ANIME FACES

We derived anime faces from an image board site for anime wallpapers and cropped the images to contain only faces. These images were sized to 96×96. This dataset consists of 51,223 color images.

B. IMPLEMENTATION

Our model was coded in Python and all experiments were conducted in the Tensorflow framework. We tested the model with two architectures: GANs with G and D comprising fully connected layers (MLP), and GANs with convolutional architectures (DCGAN [12]), both of which are described in the following subsections.

1) MLP-BASED GAN

The first GAN architecture we experimented with consisted of two MLP nets. The parameters of the networks were updated to minimize the loss function, with Adam [6] used as the optimizer function. A 100-dimension vector sampled from a Gaussian distribution is transformed into an image size using a series of fully connected layers with non-linearity activation functions (ReLU and tanh). The architecture of G is given in Table 3. D consisted of three fully connected layers, the first two of which with dropout. The penultimate layer was connected to a single sigmoid layer, as shown in Table 4.

TABLE 4. Architecture of a MLP-based discriminator.

	Operation	Size
Input	Real data + Generated data	3× image_h × image_w
Feature	Fully connected layer, dropout	1 × 300
Feature	Fully connected layer, dropout	1 × 150
Output	Fully connected layer, sigmoid	1

TABLE 5. Architecture of a DCGAN-based generator.

	Operation	Size
Input	Random noise	100
Feature	Linear connection layer, reshape, batch normalization, ReLU	4 × 4 × 1024
Feature	Fractional-strided convolutions (filter: 5 × 5, stride: 2), batch normalization, ReLU	8 × 8 × 512
Feature	Fractional-strided convolutions (filter: 5 × 5, stride: 2), batch normalization, ReLU	16 × 16 × 256
Feature	Fractional-strided convolutions (filter: 5 × 5, stride: 2), batch normalization, ReLU	32 × 32 × 128
generated image	Fractional-strided convolutions (filter: 5 × 5, stride: 2), tanh	64 × 64 × 3

2) DCGAN-BASED GAN

We also performed our adaptive methods using the architecture of DCGAN [12]. The generator networks (Table 5) have five layers and include one linear connection layer and four fractional-strided convolutional layers with a filter size of 5 × 5 and a stride of 2. All the layers except the final layer are followed by batch normalization and the ReLU activation function. The final layers consist of a fractional-strided convolutional layer and the tanh function. The input is also a 100-dimension vector sampled from a Gaussian distribution. The discriminator (Table 6) has five layers: four convolutional layers, the first of which is a convolutional layer with Leaky ReLU, the other three being convolutional layers followed by batch normalization and Leaky ReLU; and a final hidden layer that uses a full connection layer and sigmoid to transform the feature into a scalar, which is the probability that an input image belongs to a set of real data.

V. RESULTS

A. ADAPTIVE CONTROL ON DIVERSE DATASETS USING MLP AND DCGAN

In Fig. 7, we show the results of the proposed Ak-GAN using MLP on diverse datasets. Compared with the samples generated by the conventional GAN models, the adaptive control techniques enabled GANs to capture the recognizable features of faces, such as facial contours and eyes. However, these samples also had frequent noise. In addition, the image acuity still showed room for improvement. We applied the proposed adaptive control procedure to the DCGAN architecture to further improve visual quality. We also compared the results with the dataset using the DCGAN architecture. In Fig. 8 (D01-D03 on CelebA dataset, D07-D09 on Anime dataset), we show that Ak-GAN model generates images with higher quality and alleviates the mode collapse as compared with the conventional DCGAN.

TABLE 6. Architecture of a DCGAN-based discriminator.

	Operation	Size
Input	Real data + Generated data	64 × 64 × 3
Feature	Convolutional layer (filter: 5 × 5, stride: 2), Leaky ReLU	32 × 32 × 128
Feature	Convolutional layer (filter: 5 × 5, stride: 2), batch normalization, Leaky ReLU	16 × 16 × 256
Feature	Convolutional layer (filter: 5 × 5, stride: 2), batch normalization, Leaky ReLU	8 × 8 × 512
Feature	Convolutional layer (filter: 5 × 5, stride: 2), batch normalization, Leaky ReLU	4 × 4 × 1024
Output	Reshape, linear connection layer, sigmoid	1

B. PERFORMANCE MEASURE

The *Inception Score* [13] [16] is an approach to evaluate the performance of GANs quantitatively. This approach correlates well with human judgment. The *Inception Score* uses an Inception v3 network [17], which is designed for classification tasks pre-trained on ImageNet [3], to calculate statistics of the network's output when generated samples are feed into.

$$IS(G) = exp(\frac{1}{N}\sum_{i=1}^{N} D_{KL}(p(y \mid x) \parallel p(y))) \tag{7}$$

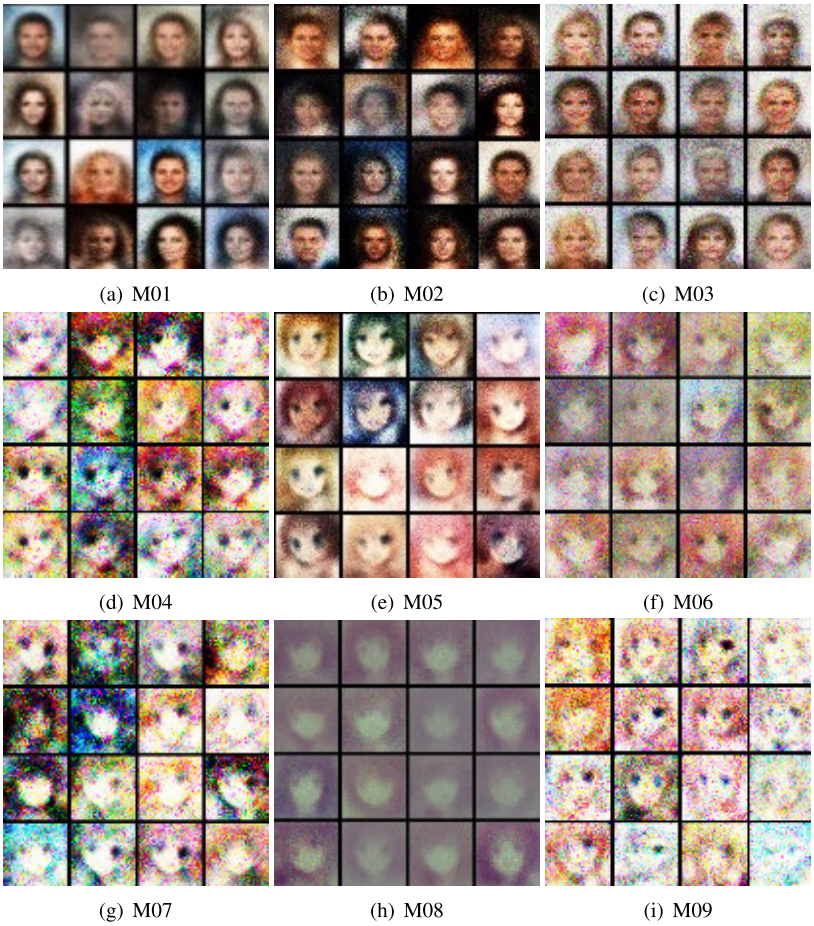
where x is an image sampled from generation distribution P_g, $p(y|x)$ is the conditional class distribution predicted by Inception v3 network given the sample x, $p(y)$ is the marginal distribution of class y. $D_{KL}(p(y|x) \parallel p(y))$ is the KL-divergence between the distribution $p(y|x)$ and $p(y)$. For exponential operation, we use base e in our experiments to make the value easier to compare. The *Inception Score* satisfies two desirable qualities of the generative models: the generated images should contain clear objects (visually recognizable), that means $p(y|x)$ should have low entropy. On the other hand, the models should generate diverse samples, that is $p(y)$ should have high entropy. Therefore, we want a large *Inception Score*, which means the KL-divergence between the distribution $p(y|x)$ and $p(y)$ is large.

Table 7 shows the *Inception Scores* obtained from 50k generated samples of Ak-GAN models and the conventional GAN model [4] on Anime and CelebA datasets, and both models use the MLP architecture, the corresponding samples are shown in Fig. 7. The highest scores are achieved by Ak-GAN models on both two datasets. The best-performed models are based on adaptive controls over P_r without momentum. We also tested with different c values, while other parameters being fixed. Results indicate that a proper value of c will increase the *Inception Score*. Moreover,

TABLE 7. ***Inception Scores*** **for samples generated by the conventional GAN models and A*k*-GAN models with the MLP architecture.**

Experiment ID	Model	Dataset	Optim	LR	Blur [a]	Base	c	IS(score±std)
M01	**Ak-GAN**	**CelebA**	**Adam**	1×10^{-4}	**None**	**Pr_fit**	**3**	$\mathbf{3.74 \pm 0.15}$
M02	Ak-GAN	CelebA	Adam	1×10^{-4}	None	Pr_fit	5	3.39 ± 0.12
M03	GAN	CelebA	Adam	1×10^{-3}	None	None	None	2.31 ± 0.34
M04	Ak-GAN	Anime	Adam	5×10^{-5}	hyperbolic	Pr_fit	4	2.18 ± 0.26
M05	**Ak-GAN**	**Anime**	**Adam**	1×10^{-4}	**None**	**Pr_fit**	**1**	$\mathbf{3.19 \pm 0.09}$
M06	Ak-GAN	Anime	Adam	1×10^{-4}	Reg_lin	Pg_fit	4	1.64 ± 0.22
M07	Ak-GAN	Anime	Adam	5×10^{-5}	Reg_lin	Pr_fit	4	2.59 ± 0.08
M08	GAN	Anime	Adam	5×10^{-5}	3×3	None	None	1.74 ± 0.11
M09	GAN	Anime	Adam	2×10^{-5}	Reg_lin	None	None	2.12 ± 0.12

[a] The blur [15] is a set of filter functions applied to the input images (real ones only) of the Discriminator to reduce the difference in the image quality between reals and samples, especially in the early stage of training process. In this paper, we tried three different blurring strategies: a Gaussian blur with a fixed kernel size of 3×3 (Fixed_blur) and two regressive strategies, linear (Reg_lin) and hyperbolic (Reg_hyper) blurs.



(a) M01 (b) M02 (c) M03

(d) M04 (e) M05 (f) M06

(g) M07 (h) M08 (i) M09

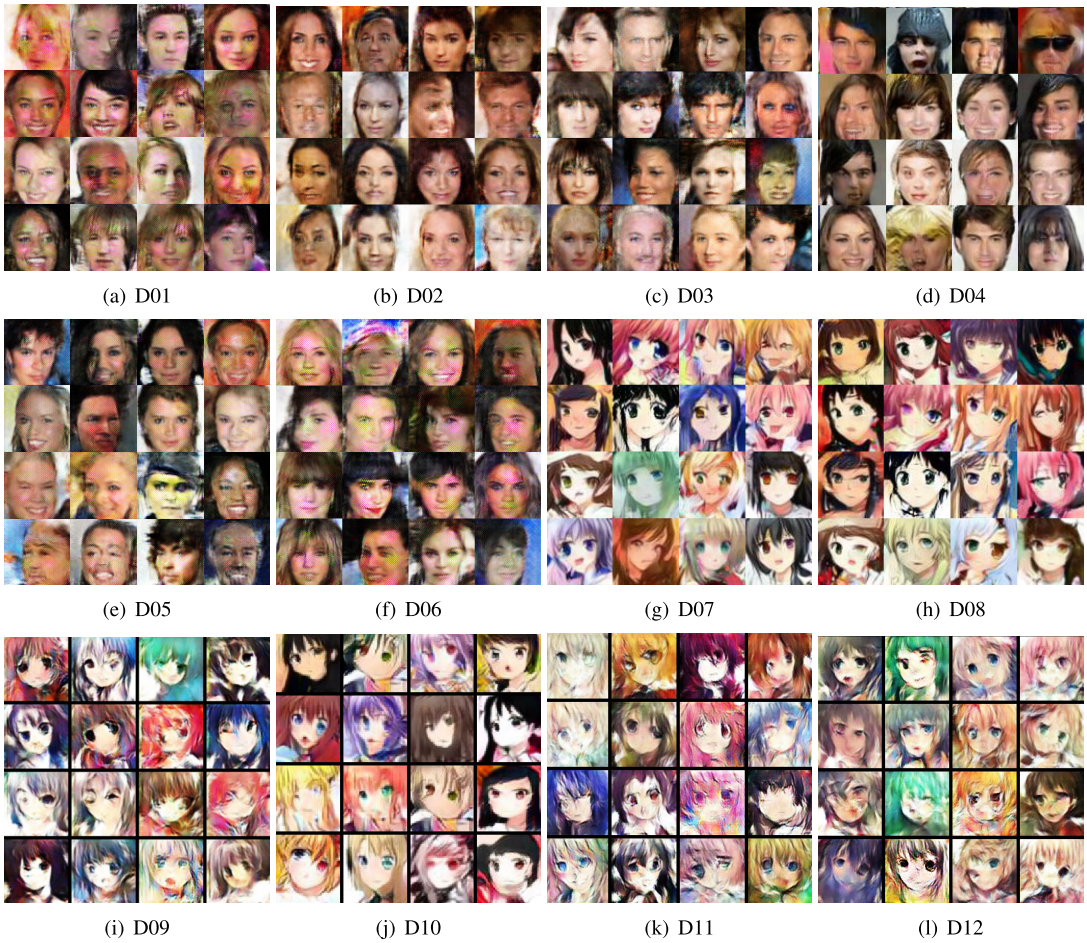
FIGURE 7. **The corresponding samples for the *Inception Scores* presented in Table 7.**

the choice of c values depends on the dataset used, e.g., the best score is achieved for the Anime dataset with c being 1, whereas for CelebA, c being 3.

Table 8 presents the *Inception Scores* for samples generated by Ak-GAN models with the DCGAN architecture, the conventional DCGAN model [12], and some state-of-the-art GAN models: WGAN [23], ABC-GAN [15], and SAGAN [28]. WGAN [23] used a more effective loss function, ABC-GAN [15] used an adaptive control method to stable the training of D and G, and SAGAN [28] investigated a much deeper network and applied the self-attention mechanism to capture long-range, multi-level dependencies features. In general, DCGAN-based models produce more vivid and recognizable samples compared to MLP-based models, as shown in Fig. 8. However, on the CelebA dataset, the MLP-based models achieved higher scores. This interesting result may suggest that the *Inception Score* not only depends on the dataset used, and also depends on the

TABLE 8. ***Inception Scores*** **for samples generated by A*k*-GAN models with the DCGAN architecture and some state-of-the-art GAN models.**

Experiment ID	Model	Dataset	Optim.	LR	Base	c	IS(score±std)
D01	Ak-GAN	CelebA	Adam	2×10^{-4}	Pg_fit	4	2.23 ± 0.14
D02	Ak-GAN	CelebA	Adam	2×10^{-4}	Pg_fit	3	2.13 ± 0.11
D03	DCGAN	CelebA	Adam	2×10^{-4}	None	None	2.34 ± 0.17
D04	SAGAN	CelebA	Adam	2×10^{-4}	None	None	1.00 ± 0.00
D05	**WGAN**	**CelebA**	**Adam**	2×10^{-4}	**None**	**None**	$\mathbf{2.43 \pm 0.21}$
D06	ABC-GAN	CelebA	Adam	2×10^{-4}	None	None	1.00 ± 0.00
D07	**Ak-GAN**	**Anime**	**rmsprop**	1×10^{-4}	**Pr_fit**	**1**	$\mathbf{4.55 \pm 1.16}$
D08	Ak-GAN	Anime	rmsprop	5×10^{-5}	Pg_fit	1	4.24 ± 0.12
D09	DCGAN	Anime	rmsprop	5×10^{-5}	None	None	3.43 ± 3.28
D10	SAGAN	Anime	rmsprop	5×10^{-5}	None	None	1.00 ± 0.00
D11	WGAN	Anime	rmsprop	5×10^{-5}	None	None	3.63 ± 0.46
D12	ABC-GAN	Anime	rmsprop	5×10^{-5}	None	None	3.36 ± 0.32



(a) D01 (b) D02 (c) D03 (d) D04

(e) D05 (f) D06 (g) D07 (h) D08

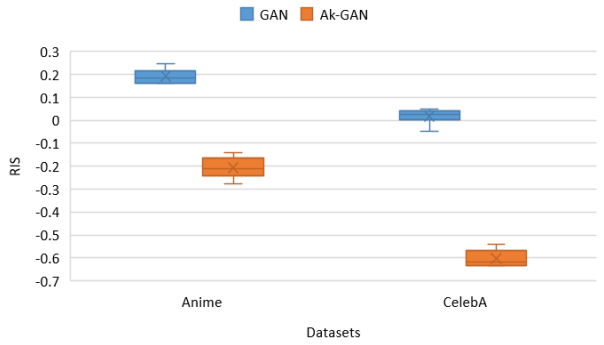
(i) D09 (j) D10 (k) D11 (l) D12

FIGURE 8. **The corresponding samples for the *Inception Scores* presented in Table 8.**

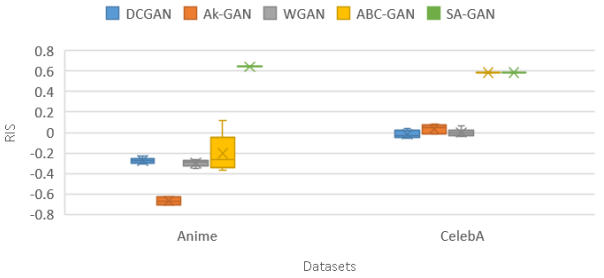
underlying models, that is, the absolute value of the *Inception Score* is unimportant between MLP- and DCGAN-based GANs on the same dataset. It would be better to derive a more general scoring mechanism, therefore, we leave the further investigation of this matter for future work. As shown in Table 8, WGAN slightly outperforms Ak-GAN on the CelebA dataset. Whereas, the best *IS* is achieved by Ak-GAN on the Anime dataset compared to other state-of-the-art GAN models. Relatively higher scores on most of test cases suggest that the proposed Ak-GAN models effectively improve the generation quality of GAN models.

To more accurately evaluate the performance of GAN models, we use the *relative inverse Inception Score* (*RIS*) [24]:

$$RIS = 1 - \frac{IS_{g_{avg}}}{IS_{r_{avg}}} \tag{8}$$



(a) The $RISs$ for models based on MLP architecture.



(b) The $RISs$ for Ak-GAN models with the DCGAN architecture and some state-of-the-art GAN models.

FIGURE 9. Range of the *relative inverse Inception Scores* (*RISs*) of models with each architecture, on each dataset. The cross mark in each range is the average of 10 *RISs*. The whiskers indicate the minimum and maximum of all of the *RISs*. The boxes display the variation of *RISs*.

$$IS_{g_{avg}} = \frac{1}{n}\sum_{i=1}^{10} IS_{g_i} \tag{9}$$

$$IS_{r_{avg}} = \frac{1}{n}\sum_{i=1}^{10} IS_{r_i} \tag{10}$$

where IS_{g_i} and IS_{r_i} are the *Inception Scores* for generated and real images, respectively. Whereas, $IS_{g_{avg}}$ and $IS_{r_{avg}}$ are the average *Inception Scores* for IS_{g_i} and IS_{r_i}, respectively. In practice, out of 500k samples, 5k real or generated ones are randomly chosen to obtain IS_{g_i} and IS_{r_i}, respectively. Then we repeated this process 10 times in total in order to compute average $IS_{g_{avg}}$ and $IS_{r_{avg}}$ based on equations (9) and (10). *RIS* can be obtained using equation (8). Obviously, *RIS* is inversely proportional to *IS*: the lower *RIS* value, the better the model.

As shown in Fig. 9, we compare the *RISs* of Ak-GAN with other models using the boxplot. The Ak-GAN based on MLP architecture obtains smaller scores on both two datasets compared to the original GAN. We also compare Ak-GAN with four state-of-the-art models: DCGAN [12], WGAN [23], ABC-GAN [15], and SAGAN [28]. The Ak-GAN shows the best performance on Anime dataset. However, for CelebA dataset, the *RISs* of all the models are basically same except ABC-GAN and SAGAN. Despite its deeper network architecture, SAGAN fails to show the superiority on the task at hand. For future work, we will explore the performance of the models that combine the adaptive control and attention mechanisms on diverse datasets.

Algorithm 1 Adaptive Control Over k Values Based on P_g or L_D (With Momentum)

Input: k, P_g or L_D
Output: k'

if $k > 1$ **then**
  if $\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ (or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$) **then**
    $k' = k + 1$
  else if $\frac{P_{mg} - P_g}{P_{mg}} > \alpha$ (or $\frac{L_{mD} - L_D}{L_{mD}} > \alpha$) **then**
    $k' = k - 1$
  else
    $k' = k$
  end if
else if $k = 1$ **then**
  if $\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ (or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$) **then**
    $k' = k + 1$
  else if $\frac{P_{mg} - P_g}{P_{mg}} > \alpha$ (or $\frac{L_{mD} - L_D}{L_{mD}} > \alpha$) **then**
    $k' = \frac{1}{k+1}$
  else
    $k' = k$
  end if
else
  if $\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ (or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$) **then**
    $k' = \frac{1}{\frac{1}{k} - 1}$
  else if $\frac{P_{mg} - P_g}{P_{mg}} > \alpha$ (or $\frac{L_{mD} - L_D}{L_{mD}} > \alpha$) **then**
    $k' = \frac{1}{\frac{1}{k} + 1}$
  else
    $k' = k$
  end if
end if
return k'

VI. CONCLUSION AND FUTURE WORK

In this paper, a family of adaptive control-based GAN models called Ak-GANs are proposed to stabilize the GAN model when training on the diverse datasets, and thus improve the quality of synthesized images to a certain degree. The proposed method is achieved by directing the training process using the well-trained learning curves (prototypes) under relatively narrow datasets. Compared with the previous work [25], the proposed models cover some various Ak-GANs in which k depends on more sophisticated criteria, and with or without momentum. We also investigated the Ak-GAN models with both MLP and DCGAN architectures. Compared to MLP-based regular GAN models, the Ak-GAN model has improved the *Inception Score* by 61% and 83% on Anime and CelebA, respectively. Whereas, the DCGAN-based Ak-GAN model has improved 32% on Anime and showed a slight decrease (9%) on CelebA compared to WGAN [23]. In addition, the Ak-GAN model significantly outperformed the previous state-of-the-art models

Algorithm 2 Adaptive Control Over k Values Based on P_r or L_G (With Momentum)

Input: k, P_r or L_G
Output: k'

if $k > 1$ **then**
  if $\frac{P_r - P_{mr}}{P_{mr}} > \beta$ (or $\frac{L_G - L_{mG}}{L_{mG}} > \beta$) **then**
    $k' = k - 1$
  else if $\frac{P_{mr} - P_r}{P_{mr}} > \beta$ (or $\frac{L_{mG} - L_G}{L_{mG}} > \beta$) **then**
    $k' = k + 1$
  else
    $k' = k$
  end if
else if $k = 1$ **then**
  if $\frac{P_r - P_{mr}}{P_{mr}} > \beta$ (or $\frac{L_G - L_{mG}}{L_{mG}} > \beta$) **then**
    $k' = \frac{1}{k+1}$
  else if $\frac{P_{mr} - P_r}{P_{mr}} > \beta$ (or $\frac{L_{mG} - L_G}{L_{mG}} > \beta$) **then**
    $k' = k + 1$
  else
    $k' = k$
  end if
else
  if $\frac{P_r - P_{mr}}{P_{mr}} > \beta$ (or $\frac{L_G - L_{mG}}{L_{mG}} > \beta$) **then**
    $k' = \frac{1}{\frac{1}{k}+1}$
  else if $\frac{P_{mr} - P_r}{P_{mr}} > \beta$ (or $\frac{L_{mG} - L_G}{L_{mG}} > \beta$) **then**
    $k' = \frac{1}{\frac{1}{k}-1}$
  else
    $k' = k$
  end if
end if
return k'

on the *relative inverse Inception Score* (*RIS*), reducing *RIS* by around 21% and 60% on Anime and CelebA, respectively. Quantitative and qualitative results showed that the proposed procedure could indeed improve the stability of GAN training and generate compelling images under various models and diverse datasets, compared to some state-of-the-art models such as WGAN [23], ABC-GAN [15], and SAGAN [28].

For future work, we plan to investigate a more effective criteria to direct the training, which may encourage the convergence of GANs. We are also going to experiment with more objective functions, such as *f*-divergence [10], to measure the difference in the divergence between the distributions of real and generated data.

ACKNOWLEDGMENT

(Xiaohan Ma and Rize Jin contributed equally to this work.)

APPENDIX

A. ADAPTIVE METHOD WITH MOMENTUM

We next introduce the adaptive methods with momentum based on P_g or L_D and on P_r or L_G.

Algorithm 3 Adaptive Control Over k Values Based on P_g or L_D (Without Momentum)

Input: k, P_g or L_D
Output: k'

if $k > 1$ **then**
  if $\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ (or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$) **then**
    $k' = k + 1$
  else if $\frac{P_{mg} - P_g}{P_{mg}} > \alpha$ (or $\frac{L_{mD} - L_D}{L_{mD}} > \alpha$) **then**
    $k' = \frac{1}{2}$
  else
    $k' = k$
  end if
else
  if $\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ (or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$) **then**
    $k' = 2$
  else if $\frac{P_{mg} - P_g}{P_{mg}} > \alpha$ (or $\frac{L_{mD} - L_D}{L_{mD}} > \alpha$) **then**
    $k' = \frac{1}{\frac{1}{k}+1}$
  else
    $k' = k$
  end if
end if
return k'

1) ADAPTIVE CONTROL BASED ON P_g OR L_D

Because the criterion learning curves for P_g or L_D show the same trend during the training process, we use the same method to control them. In Algorithm 1, when k is greater than one, which means D is trained k steps with G one step. If the current P_g or L_D exceeds the allowable deviation defined by threshold α, we increase or decrease the D steps (represented by k) by 1 based on one of the following resulting cases: positive deviation ($\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$) or negative deviation ($\frac{P_{mg} - P_g}{P_{mg}} > \alpha$ or $\frac{L_{mD} - L_D}{L_{mD}} > \alpha$). When k is one, which denotes that G and D is alternately trained with one step, respectively. If positive deviation, we increase D steps by making k more than one ($k = k + 1$); if negative deviation, we increase G steps by making k less than one ($k = \frac{1}{k+1}$). When k is less than one, which indicates that G is trained $\frac{1}{k}$ steps with D one step. If the positive deviation, we decrease the G steps by 1; else, we increase the G steps by 1.

2) ADAPTIVE CONTROL BASED ON P_R OR L_G

Different from the aforementioned method, Algorithm 2 is based on P_r or L_G. Since the curves of P_r and L_G grow in opposite direction with learning curves for P_g and L_D, the adjustment of k is the extreme opposite of the previous one. That is, if there is a positive deviation ($\frac{P_r - P_{mr}}{P_{mr}} > \beta$ or $\frac{L_G - L_{mG}}{L_{mG}} > \beta$), G steps increase or D steps decrease; otherwise, G steps decrease or D steps increase.

Algorithm 4 Adaptive Control Over k Values Based on P_r or L_G (Without Momentum)

Input: k, P_r or L_G
Output: k'

```
if k > 1 then
  if (P_r − P_mr)/P_mr > β (or (L_G − L_mG)/L_mG > β) then
    k' = 1/2
  else if (P_mr − P_r)/P_mr > β (or (L_mG − L_G)/L_mG > β) then
    k' = k + 1
  else
    k' = k
  end if
else
  if (P_r − P_mr)/P_mr > β (or (L_G − L_mG)/L_mG > β) then
    k' = 1/(1/k + 1)
  else if (P_mr − P_r)/P_mr > β (or (L_mG − L_G)/L_mG > β) then
    k' = 2
  else
    k' = k
  end if
end if
return k'
```

B. ADAPTIVE METHOD WITHOUT MOMENTUM

In practical implementations, the progressive adjustment methods improve the stability of models. However, the training is slow. Therefore, we explore an variant which removes the momentum. This means that the k value is immediately changed to an opposite number ($k = \frac{1}{2}$ when $k \geq 1$, $k = 2$ when $k < 1$) when control variables are out of the constraint range.

When k is greater than or equal to one, which means D is trained k steps with G one step. In cases of positive deviation ($\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$), where P_g or L_D is greater than the criterion value, indicating that G is overly trained, we must increase k by 1. If the negative deviation ($\frac{P_g - P_{mg}}{P_{mg}} > \alpha$ or $\frac{L_D - L_{mD}}{L_{mD}} > \alpha$), in which D is overly trained, we immediately makes k to be $\frac{1}{2}$, which means G is trained 2 steps with D one step, to bias toward G. When k is less than one, where G is trained $\frac{1}{k}$ steps with D one step, if the positive deviation, we immediately make k to be 2 to bias to training D. If the negative deviation, we increase the G steps by 1 ($k' = \frac{1}{\frac{1}{k}+1}$). We modify k for the cases of algorithm based on P_r or L_G in an opposite manner with the algorithm based on P_g or L_D. These two algorithms are shown in Algorithm 3 and Algorithm 4, respectively.

REFERENCES

[1] Martin Arjovsky and Léon Bottou, "Towards principled methods for training generative adversarial networks," 2017, *arXiv:1701.04862*. [Online]. Available: https://arxiv.org/abs/1701.04862

[2] A. Dash, J. C. B. Gamboa, S. Ahmed, M. Liwicki, and M. Z. Afzal, "TAC-GAN-text conditioned auxiliary classifier generative adversarial network," 2017, *arXiv:1703.06412*. [Online]. Available: https://arxiv.org/abs/1703.06412

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[5] D. J. Im, C. D. Kim, H. Jiang, and R. Memisevic, "Generating images with recurrent adversarial networks," 2016, *arXiv:1602.05110*. [Online]. Available: https://arxiv.org/abs/1602.05110

[6] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[7] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[8] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 3730–3738.

[9] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," 2016, *arXiv:1605.07725*. [Online]. Available: https://arxiv.org/abs/1605.07725

[10] S. Nowozin, B. Cseke, and R. Tomioka, "f-gan: Training generative neural samplers using variational divergence minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 271–279.

[11] S. Pascual, A. Bonafonte, and J. Serra, "Segan: Speech enhancement generative adversarial network," 2017, *arXiv:1703.09452*. [Online]. Available: https://arxiv.org/abs/1703.09452

[12] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*. [Online]. Available: https://arxiv.org/abs/1511.06434

[13] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2234–2242.

[14] F. Y. Y. Z. S. Song and A. S. J. Xiao, "Construction of a large-scale image dataset using deep learning with humans in the loop," 2015, *arXiv:1506.03365*. [Online]. Available: https://arxiv.org/abs/1506.03365

[15] I. Susmelj, E. Agustsson, and R. Timofte, "Abc-gan: Adaptive blur and control for improved training stability of generative adversarial networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 5, 2017.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2818–2826.

[17] L. Theis, A. van den Oord, and M. Bethge, "A note on the evaluation of generative models," 2015, *arXiv:1511.01844*. [Online]. Available: https://arxiv.org/abs/1511.01844

[18] I. O. Tolstikhin, S. Gelly, O. Bousquet, C.-J. Simon-Gabriel, and B. Schölkopf, "Adagan: Boosting generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5430–5439.

[19] D. Warde-Farley and Y. Bengio, "Improving generative adversarial networks with denoising feature matching," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, Apr. 2017.

[20] H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas, "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5907–5915.

[21] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," 2016, *arXiv:1609.03126*. [Online]. Available: https://arxiv.org/abs/1609.03126

[22] Rosch, Eleanor, "Cognitive representations of semantic categories," *J. Exp. Psychol., Gen.*, vol. 104, p. 192, Sep. 1975.

[23] M. Arjovsky, S. Chintala, and L. Bottou, "An empirical study on evaluation metrics of generative adversarial networks," 2017, *arXiv:1701.07875*. [Online]. Available: https://arxiv.org/abs/1701.07875

[24] Q. Xu, G. Huang, Y. Yuan, C. Guo, Y. Sun, F. Wu, and K. Weinberger, "An empirical study on evaluation metrics of generative adversarial networks," 2018, *arXiv:1806.07755*. [Online]. Available: https://arxiv.org/abs/1806.07755

[25] X. Ma, R. Jin, K. A. Sohn, J. Y. Paik, J. Sun, and T.-S. Chung, "Improving generative adversarial networks with adaptive control learning," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Dec. 2018, pp. 1–4.

[26] C. Wang, C. Xu, X. Yao, and D. Tao, "Evolutionary generative adversarial networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 6, pp. 921–934, Dec. 2019.

[27] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.
[28] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019.
[29] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," 2018, *arXiv:1809.11096*. [Online]. Available: https://arxiv.org/abs/1809.11096

XIAOHAN MA received the B.S. degree from the Department of Electronic Engineering, Nanyang Institute of Technology, in 2014. She is currently pursuing the Ph.D. degree with the Department of Computer Science, Ajou University, Suwon, South Korea. Her current research interests include machine learning, natural language processing, and computer vision.

RIZE JIN received the M.S. and Ph.D. degrees in computer engineering from Ajou University (AU), South Korea, in February 2011 and February 2015, respectively. He was an Assistant Professor with the Software Department, AU. Before joining AU, he was a Postdoctoral Researcher with the Department of Computer Engineering, Korea Advanced Institute of Science and Technology (KAIST), South Korea. He is currently a Professor with the School of Computer Science and Technology, Tianjin Polytechnic University, China. His research interests include flash-based DB, NoSQL, natural language processing, and deep learning.

KYUNG-AH SOHN received the B.S. degree in mathematics and the M.S. degree in computer science from Seoul National University, Seoul, South Korea, in 2003, and the Ph.D. degree in computer science from Carnegie Mellon University, Pittsburgh, PA, USA. From 2003 to 2005, she was a Research Assistant Professor with the College of Medicine/Systems Biomedical Informatics Research Center, Seoul National University, South Korea. She is currently an Associate Professor with the Department of Software and Computer Engineering, Ajou University, South Korea. Her current research interests include machine learning, biomedical informatics, and social media data mining.

JOON-YOUNG PAIK received the B.S., M.S., and Ph.D. degrees in computer science and engineering from Chungnam National University, South Korea, in 2008, 2010, and 2013, respectively. He is currently an Associate Professor with the School of Computer Science and Technology, Tianjin Polytechnic University, China. His current research interests include flash memory storage, storage security, and deep learning.

TAE-SUN CHUNG received the B.S. degree from KAIST, Daejeon, South Korea, in 1995, and the M.S. and Ph.D. degrees from Seoul National University, Seoul, South Korea, in 1997 and 2002, respectively, all in computer science. He is currently a Professor with the Department of Computer Engineering, Ajou University, Suwon, South Korea. His current research interests include flash memory storage, XML databases, and database systems.