IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 4, 2019, accepted April 13, 2019, date of publication April 29, 2019, date of current version May 10, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2913697

Adversarial Learning With Knowledge of Image Classification for Improving GANs

JAE-YONG BAEK, YONG-SANG YOO, AND SEUNG-HWAN BAE, (Member, IEEE)
Department of Computer Science and Engineering, Incheon National University, Incheon 22012, South Korea

Corresponding author: Seung-Hwan Bae (shbae@inu.ac.kr)

This work was supported in part by the Incheon National University (International Cooperative) Research Grant in 2018 and in part by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) under Grant NRF-2018R1C1B6003785.

ABSTRACT Generating realistic images with fine details are still challenging due to difficulties of training GANs and mode collapse. To resolve this problem, our main idea is that leveraging the knowledge of an image classification network, which is pre-trained by a large scale dataset (*e.g.* ImageNet), would improve a GAN. By using the gradient of the network (*i.e.* discriminator) with high discriminability during training, we can, therefore, guide the gradient of a generator gradually toward the real data region. However, excessive negative feedback of the powerful classifier often prevents a generator from producing diverse images. Based on the main idea, we design a GAN including the added discriminator and propose a novel energy function in order to transfer the pre-trained knowledge to a generator and control the feedback of the added discriminator. We also present an incremental learning method to prevent the density of the generator to be the low-entropy distribution when training our GAN with respect to the energy function. We incorporate our method to DCGAN and demonstrate the ability to enhance the image quality even in high resolution on several datasets compared to DCGAN. In addition, we compare our method with recent GANs for the diversity of generated samples on CIFAR-10 and STL-10 datasets and provide the extensive ablation studies to prove the benefits of our method.

INDEX TERMS Generative adversarial network, image classification network, image generation, generative model, deep learning, convolutional neural network.

I. INTRODUCTION

As a generative model, generative adversarial networks (GANs) produce synthetic images by approximating a model density $p_{model}(\mathbf{x})$ to a real data density $p_{data}(\mathbf{x})$. In general, a GAN consists of generator and discriminator networks. The generator produces a synthetic sample $G(\mathbf{z})$ for a given input vector $\mathbf{z}$ with random noise, but the discriminator identifies between real and synthetic samples generated from $p_{data}(\mathbf{x})$ and $p_{model}(\mathbf{x})$. Both adversarial networks are trained alternatively: the discriminator first learns to discriminate between samples from data and model distributions. Subsequently, the generator learns to generate samples close to real one by fitting its distribution to the real data distribution. This adversarial learning progresses based on the basis of Nash equilibrium [1], [2]. When the loss of GAN converges toward a minimum, $p_{model}(\mathbf{x})$ is close to $p_{data}(\mathbf{x})$, and the discriminator cannot distinguish between the two distribution any more [1], [3].

In many cases, training GANs often fails due to mode collapse and diminished gradient problems. To handle this problem, improved object functions [4]–[7], learning methods [2], [8], [9], and architectures [10]–[12] have been developed. To avoid the mode collapse, dual discriminator GAN [10] presents a loss function consisting of KL and reverse KL divergences between data and model distributions. WGAN-GP [7] improves the loss of WGAN [6] by penalizing the norm of gradient of the discriminator.

However, generating high-quality images in high resolution is still challenging since some local details of an object are more likely to be inaccurate or missed. As a result, the gradients are drastically amplified because a discriminator easily distinguishes between real and synthetic samples. In this case, the variability of the generated images is also reduced since a discriminator does not guide the generator to the diverse region of the data distribution.

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

In order to generate high quality and diverse image even in high resolution, our main idea is to leverage the knowledge of an image classification network. In general, a classification network (*e.g.* ResNet [13] and DenseNet [14]) has high generality and classification accuracy since it was trained with a large scale dataset (*e.g.* ImageNet). In general, the classification network is based on the deep architecture, and loses the details of spatial information of object parts due to many pooling operations [15], [16]. It means that it is likely to classify an object image by looking into the existence of some object details rather than the spatial locations. Therefore, to fool the discriminator having high classification accuracy, the image from a generator should contain the accurate local details of an object or object parts.

In this paper, we design a new generative adversarial network by incorporating DenseNet [14] into DCGAN [17]. However, the gradient divergence or mode collapse problems occur sometime when training our network using the standard GAN loss function and learning algorithm [3]. This is because the excessive negative feedbacks of the powerful classifier prevent a generator from producing diverse images. In other word, a generator is more likely to produce some specific images which can deceive the discriminator.[1]

To resolve this problems, we propose a new GAN object function to train the three players consisting of one generator and two discriminators in an adversarial manner. We define the loss of an added discriminator by including a scale factor in order to control the feedback of this discriminator during adversarial training. Furthermore, we present the incremental learning method to train a generator step-by-step. We first train both discriminator and generator of DCGAN by solving the two-player minimax game. We then train the added discriminator by minimizing the loss for real and fake samples. When the three players optimized individually, we jointly train both discriminators by maximizing the probabilities of assigning the correct labels to both real samples and fake samples of the generator. We simultaneously train the generator by maximizing the probabilities of both discriminators over generated fake samples. As a result, we first use a soft discriminator for learning basic image representation, and then use the strict classifier for enhancing the representation further.

We extensively evaluate our GAN method on several datasets such as CIFAR-10 [18], STL-10 [19] and CelebA-HQ [9]. We prove the ability of our GAN to generate images in high resolution by comparing DCGAN. In addition, we evaluate the inception score [20], [21] and Fréchet Inception Distance (FID) [22] to evaluate the quality and diversity of the generated samples by our GAN compared to other recent GAN methods. We also provide the ablation study to prove effects of our learning method.

To sum up, the main contribution of this paper can be summarized as follows:

- proposition of transfer learning to leverage the knowledge of a pre-trained network for improving the image generation ability of GANs.
- proposition of a new GAN loss function and incremental learning method for the transfer learning.
- analytic solution and theoretical analysis for the proposed GAN loss function.

From the extensive comparison and ablation study as shown in Fig. 4-9, we show the superiority of our method compared to the recent GAN methods, and prove effects of the proposed methods. In addition, we provide the experimental results on CIFAR-100, and these results show that our method is useful to improve the ability of generating diverse images for many different categories.

II. RELATED WORKS

Unsupervised learning [23], [24] to learn general representations and relationship between samples of a dataset is an important problem in machine learning. One of the powerful approach for unsupervised learning uses deep neural networks [1], [25]–[27]. In particular, the generative adversarial networks (GANs) [3] show the promising results among deep learning-based methods. GANs consist of two neural networks for generation and discrimination, and learn both networks simultaneously based on a minimax two-player game. To make the GANs suitable for image generation more, [17] has extended GANs to the deep convolutional GAN (DCGAN). By using the batch normalization and replacing pooling layers with strided convolutions, DCGAN mitigates poor initialization and gradient vanishing problems.

Moreover, GANs still suffer from mode collapse [2] and training instability. As a result, it fails to achieve the diversity of a data distribution. To address these problems, some methods [4]–[7] have been developed. The common idea of those methods is to improve the original GAN loss [3] by inducing different distance measures between sample and model distributions. Reference [4] derives the GAN training objectives for all f-divergences, and applies a variational divergence estimation method for GAN training with arbitrary $f-$ divergences. In the least square GAN [5], the GAN objective function is modeled with the least square loss, and it is shown that minimizing the designed function is to minimize the Pearson χ^2 divergence. In Wasserstein GAN (WGAN) [6], a GAN function is defined with the Wasserstein distance which is the approximation of the EM distance. To improve the training stability of WGAN further, [7] adds a gradient penalty to constrain the gradient norm of a discriminator. To train GANs for high resolution image generation, [9] presents a progressive training method in order to increase depths of networks step-by-step.

On the other hand, some methods use the multiple discriminators for handling the mode collapse. Dual discriminator GAN (D2GAN) [10] extends the common two-player minimax game to a three-player minimax game between a generator and two discriminators, and shows that the optimizing the generator is to reduce the KL and reverse

[1]We discuss the impact of feedbacks of the added discriminator in Sec. IV-E and Fig. 8.

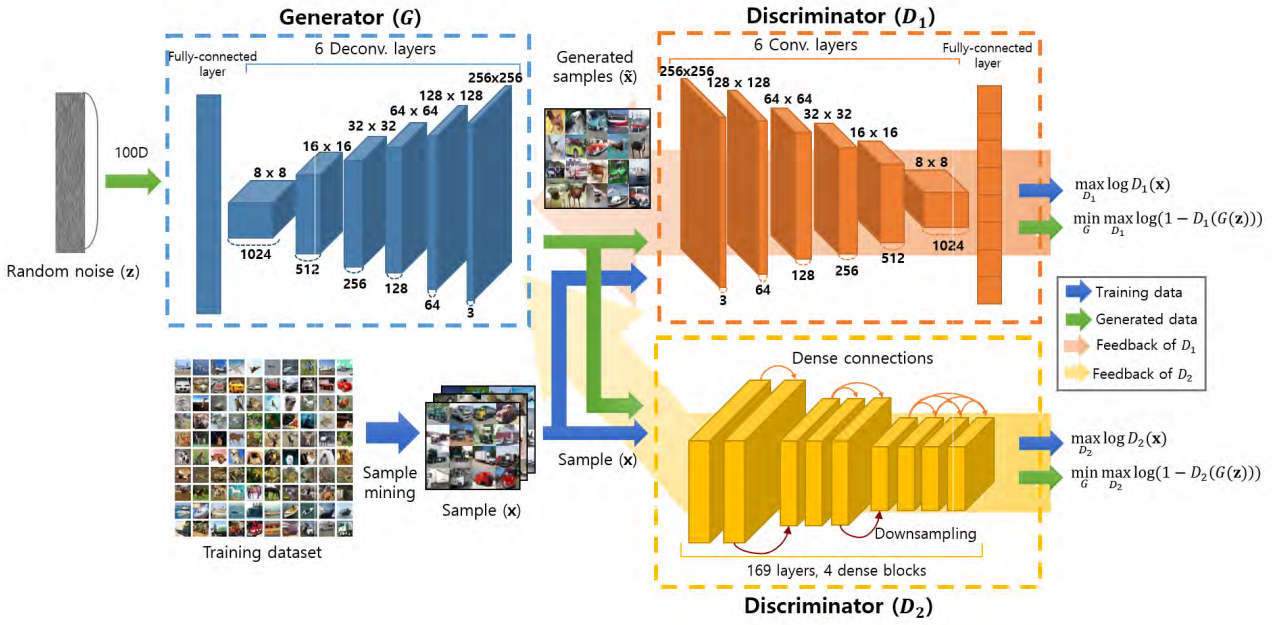

FIGURE 1. The proposed IDGAN consists of a generator G, a discriminator D_1 and an image classification network D_2 pre-trained with ImageNet. Given an input vector z disturbed by a random noise, G generates a sample image $\tilde{x}$ by $G(z)$. D_1 and D_2 are trained with real samples x of a training set and generated $\tilde{x}$. Then, D_1 and D_2 compute the losses Eq. (2) and gradients Eq. (3) for updating their parameters. From the feedbacks of D_1 and D_2, the loss and gradients of G are computed by Eq. (2) and Eq. (4), and then parameters of G are updated with the gradients.

KL divergences together. Auxiliary classifier GAN (AC-GAN) [12] for semi-supervised GANs uses a pre-trained classifier to predict sample labels, and maximizes the log likelihoods for sample sources and class labels. Therefore, when training GANs, AC-GAN uses sample labels as inputs of the losses, whereas our IDGAN does not use them. In addition, [28] defines a modified GAN loss to produce samples in the low density area of the data distribution, and uses the generated samples to train a more confident classifier. In this paper, we also use multiple discriminators for improving GANs. However, our method more focuses on improving unsupervised GANs compared to other methods for enhancing semi-supervised GANs [12] and image classification [28]. Generative Multi-Adversarial Network (GMAN) [11] extends GAN to multiple discriminators. In order to reduce the impact of too harsh a discriminator, the ensemble method makes multiple discriminators provide uniform feedbacks to a generator. In addition, GAN methods [10]–[12] focus on improving GANs by training generator and discriminators from scratch rather than exploit the transferred knowledge for GAN training.[2]

Although [29]–[33] present transfer learning methods for GANs, our transfer learning method is different from those works. A main difference is that they pre-train a generator and a discriminator, whereas we pre-train a discriminator only with a source dataset. In addition, our GAN frameworks are different from [29]–[33] because we use two discriminators instead of using one discriminator. As a result, our method can transfer the knowledge of source data to a target task without pre-training a generator.

III. JOINT TRAINING WITH GANS AND IMAGE CLASSIFICATION NETWORK

When source and target domains have some similarity, leveraging the pre-trained knowledge usually is beneficial for a target task [34]. To apply this transfer learning for GANs, our key idea is to leverage the precise feedback of an image classification network trained with a large dataset (*e.g.* ImageNet) when learning a generator. Since the auxiliary discriminator has much higher classification accuracy than a common discriminator in general, the generator should produce more diverse and higher quality images to make the well-trained discriminator fool. However, applying the conventional GAN framework [17] and loss [1] for training these networks in an alternative training manner,[3] we found that the GAN performance decreases as shown Fig. 3.

To resolve this problem, we present a new GAN loss and theoretical analysis of the loss to show that optimizing a generator minimizes the Jensen-Shannon divergences between a generator and discriminators. In addition, we propose the incremental learning to learn a generator step-by-step with the knowledge of discriminators. To this end, we first train a standard GAN and an auxiliary discriminator individually, and train all networks jointly by maximizing and minimizing the proposed loss.[4] In Fig. 1, we present a GAN

[2]More discussions of GANs with multiple discriminators are given in Sec. III-F.

[3]It means that we alternatively train the pair of G and D_1, and the pair of G and D_2 per iteration.

[4]Even though we can train networks jointly for our loss, the performance is also degraded as shown in Table 3.

framework (IDGAN) consisting of a generator (G), a discriminator (D_1), and an auxiliary discriminator (D_2). Although we represent D_2 as the DenseNet-169 [14] can be any image classification network.[5]

A. GENERATIVE ADVERSARIAL NETWORKS

In this section, we revisit the traditional GAN [3] before discussing our methods. The GAN consists of generator G and discriminator D. To learn the generator's distribution p_G over data $\mathbf{x}$, the generator G is defined as a mapping function $G(\mathbf{z}; \theta_\mathbf{G})$ that maps $\mathbf{z}$ sampled from a prior noise distribution $p_z(\mathbf{z})$ to a point $\mathbf{x}$ in data space. The discriminator $D(\mathbf{x}; \theta_\mathbf{D})$ outputs a probability that $\mathbf{x}$ cames from the data rather than p_G. Here, θ_G and θ_D are parameters of a generator and a discriminator, respectively. G is trained to minimize $\log(1 - D(G(\mathbf{z})))$, and D is simultaneously trained to maximize the probability of assigning the correct label to both training examples and samples from G. As a result, the generator and discriminator compete in the two-player minimax game with value function $V(G, D)$:

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

B. ADVERSARIAL LEARNING WITH KNOWLEDGE OF IMAGE CLASSIFICATION

To exploit the feedbacks of both discriminators D_1 and D_2 for training G, we simply consider the discriminator D in Eq. (1) as D_1. Then, the loss function for G and D_1 is Eq. (1). Since D_2 also outputs the probability of $\mathbf{x}$ comes from a real sample as D_1, we can similarly define the loss for G and D_2 with $\log D_2(\mathbf{x})$ and $\log(1 - D_2(G(\mathbf{z})))$ over training examples $\mathbf{x}$ and samples $\mathbf{z}$ from G. However, at early training stage the gradients of G could be led away from the data region. A main reason is that G is still difficult to generate a sample which can be regarded as "realistic" one by pre-trained D_2 with high discrimination accuracy. Thus, a generator is likely to receive many negative feedbacks from D_2. As noted in [11], too many negative feedbacks guide the distribution of G to the other regions of the data domain. In addition, the number of negative feedbacks could be increased since D_1 also is likely to give the negative feedbacks as D_1 gets discriminative.

To alleviate this problem, our method is to build up the generation ability of G with the feedbacks of D_1 only, and improves G with the feedbacks of D_2 gradually. We present more details of this learning method in Sec. III-D. Furthermore, we can reduce the feedback of D_2 with a scaling factor λ, where $0 < \lambda \leq 1$, and design a new loss function for training G, D_1, and D_2 jointly as follows:

$$\begin{aligned}\min_G \max_{D_1,D_2} V(G, D_1, D_2) &= \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D_1(\mathbf{x})] \\ &+ \mathbb{E}_{\mathbf{z}\sim p_\mathbf{z}(\mathbf{z})}[\log(1 - D_1(G(\mathbf{z})))] \\ &+ \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D_2(\mathbf{x})] \\ &+ \lambda\mathbb{E}_{\mathbf{z}\sim p_\mathbf{z}(\mathbf{z})}[\log(1 - D_2(G(\mathbf{z})))] \quad (2)\end{aligned}$$

We provide the comparison results with different λ in Figure 3 and 7, and set λ to 0.6 in other evaluations.

From our loss Eq. (2), we present the following gradient update equations of D_1, D_2, and G:

$$\begin{aligned}\theta_{d_1} &\leftarrow \theta_{d_1} - \eta_1\nabla\theta_{d_1}\frac{1}{m}\sum \log D_1(\mathbf{x}) + \log(1 - D_1(G(\mathbf{z}))) \\ \theta_{d_2} &\leftarrow \theta_{d_2} - \eta_2\nabla\theta_{d_2}\frac{1}{m}\sum \log D_2(\mathbf{x}) + \log(1 - D_2(G(\mathbf{z}))) \quad (3)\end{aligned}$$

$$\begin{aligned}\theta_g \leftarrow \theta_g - \eta_1\nabla\theta_g\frac{1}{m}\sum &\log(1 - D_1(G(\mathbf{z}))) \\ &+ \lambda\log(1 - D_2(G(\mathbf{z}))) \quad (4)\end{aligned}$$

where η_1 is a learning rate of G and D_1, and η_2 is a learning rate of D_2. Note that λ is used only for updating G to reduce the feedback of D_2.

C. THEORETICAL ANALYSIS

For the proposed 2, we provide theoretical analysis. When G, D_1 and D_2 have enough capacity (*i.e.* nonparametric limit), optimal G can be found by minimizing both Jensen-Shannon divergences between model p_{model} and data p_{data} distributions. To proof this, we first find the optimal D_1^* and D_2^* given a fixed G.

Proposition 1: Given a fixed G, the optimal D_1^* and D_2^* which maximize $V(G, D_1, D_2)$ are

$$\begin{aligned}D_1^* &= \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + p_G(\mathbf{x})}, \\ D_2^* &= \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + \lambda p_G(\mathbf{x})} \quad (5)\end{aligned}$$

Proof. By the induced measure theorem [10], we know $\mathbb{E}_{\mathbf{z}\sim p_\mathbf{z}(\mathbf{z})}[f(G(\mathbf{z}))] = \mathbb{E}_{\mathbf{x}\sim p_\mathbf{G}}[f(\mathbf{x})]$, where f can be $D_1(\mathbf{x})$ or $D_2(\mathbf{x})$. Then, the value function $V(G, D_1, D_2)$ can be represented as

$$\begin{aligned}&V(G, D_1, D_2) \\ &= \int_\mathbf{x} [p_{data}(\mathbf{x})\log D_1(\mathbf{x}) + p_G(\mathbf{x})\log(1 - D_1(\mathbf{x})) \\ &\quad + p_{data}(\mathbf{x})\log D_2(\mathbf{x}) + \lambda p_G(\mathbf{x})\log(1 - D_2(\mathbf{x}))]\mathrm{d}\mathbf{x} \quad (6)\end{aligned}$$

For the function $\Psi(\mathbf{x})$ inside the integral, we can find D_1^* and D_2^* to maximize this function by setting the derivatives of the function w.r.t D_1 and D_2 as

$$\begin{aligned}\frac{\partial\Psi(\mathbf{x})}{\partial\mathbf{D}_1} = 0 \Rightarrow \frac{\partial\Psi(\mathbf{x})}{\partial\mathbf{D}_1} &= \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + p_G(\mathbf{x})} \\ \frac{\partial\Psi(\mathbf{x})}{\partial\mathbf{D}_2} = 0 \Rightarrow \frac{\partial\Psi(\mathbf{x})}{\partial\mathbf{D}_2} &= \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + \lambda p_G(\mathbf{x})} \quad (7)\end{aligned}$$

Since $\partial\Psi(\mathbf{x})/\partial\mathbf{D}_1^2 = -p_{data}(\mathbf{x})/D_1^2(\mathbf{x}) - p_G(\mathbf{x})/(1 - D_1(\mathbf{x}))^2$ and $\partial\Psi(\mathbf{x})/\partial\mathbf{D}_2^2 = -p_{data}(\mathbf{x})/D_2^2(\mathbf{x}) - \lambda p_G(\mathbf{x})/(1 - D_2(\mathbf{x}))^2$,

[5]The comparison of different D_2 is given in Table. 3.

the second derivatives are non-positive. Therefore, verifying that Eq. (7) achieves the maximum solution and concluding the proof.

Theorem 2: Given D_1^* and D_2^*, the Nash equilibrium point $\left(D_1^*, D_2^*, G^*\right)$ for the minimax game is achieved if and only if $p_G = p_{data}$. At that point, the minimum value is -log8.

Proof. By substituting D_1^* and D_2^* from (7) into the object function Eq. (2) of the minimax optimization problem, we have

$$\begin{aligned} V(G, D_1, D_2) = &\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left[\log \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + p_G(\mathbf{x})}\right] \\ &+ \mathbb{E}_{\mathbf{x}\sim p_G(\mathbf{x})}\left[\log \frac{p_G(\mathbf{x})}{p_{data}(\mathbf{x}) + p_G(\mathbf{x})}\right] \\ &+ \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left[\log \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + \lambda p_G(\mathbf{x})}\right] \\ &+ \mathbb{E}_{\mathbf{x}\sim p_G(\mathbf{x})}\left[\log \frac{\lambda p_G(\mathbf{x})}{p_{data}(\mathbf{x}) + \lambda p_G(\mathbf{x})}\right] \end{aligned} \tag{8}$$

We can represent it with the Kullback–Leibler divergence as

$$\begin{aligned} V(G, D_1, D_2) = &KL\left(p_{data}(\mathbf{x})||\frac{p_{data}(\mathbf{x}) + p_G(\mathbf{x})}{2}\right) \\ &+ KL\left(p_G(\mathbf{x})||\frac{p_{data}(\mathbf{x}) + p_G(\mathbf{x})}{2}\right) \\ &+ KL\left(p_{data}(\mathbf{x})||\frac{p_{data}(\mathbf{x}) + \lambda p_G(\mathbf{x})}{2}\right) \\ &+ KL\left(\lambda p_G(\mathbf{x})||\frac{p_{data}(\mathbf{x}) + \lambda p_G(\mathbf{x})}{2}\right) - \log 8 \end{aligned} \tag{9}$$

Also, we recognize this expression is the sum of the Jensen-Shannon divergences between $p_{data}(\mathbf{x})$ and $p_G(\mathbf{x})$ as

$$\begin{aligned} V(G, D_1, D_2) = &2 \cdot JSD\left(p_{data}(\mathbf{x})||p_G(\mathbf{x})\right) \\ &+ 2 \cdot JSD\left(p_{data}(\mathbf{x})||\lambda p_G(\mathbf{x})\right) - \log 8 \end{aligned} \tag{10}$$

These Jensen-Shannon divergences are nonnegative always and zero only if two distributions are equal . Therefore, we have proved that $-\log(8)$ is the global minimum of V when $p_G = p_{data}$ and $p_G = \frac{p_{data}}{\lambda}$. Since λ is a scaling factor, the distribution of p_{data} is not changed by λ. Therefore, the solution can be $p_G = p_{data}$. In other word, the distribution p_G^* generated from a generator G is identical to the data distribution p_{data}, and D_1 and D_2 fail to identify the real and fake samples since both discriminators produce the same score to 1/2 for any sample.

D. INCREMENTAL LEARNING

As mentioned in Sec. III-B, the excessive negative feedback from discriminators is unlikely to guide the distribution of a generator toward the data distribution. The negative feedback is because a generator does not produce 'realistic' samples which can mislead decisions of discriminators. Without the sufficient training of G, the joint training of all the G, D_1 and D_2 degrades the inception score as shown in Table 3.

Algorithm 1 Proposed Incremental Learning

Input : Mini batch of noise samples $\{\mathbf{z}^{(i)}\}_{i=1}^m$ and data samples $\{\mathbf{x}^{(i)}\}_{i=1}^m$

```
//Step 1 Training G and D1
for number of training iterations do
    Update D1:
    θ_d1 ← θ_d1 − η1 ∇θ_d1 (1/m) Σ log D1(x)
    + log(1 − D1(G(z)))
    for k iterations do
        Update G:
        θ_g ← θ_g − η1 ∇θ_g (1/m) Σ log(1 − D1(G(z)))
    end
end
//Step 2 Retraining D2
for number of training iterations do
    Update D2:
    θ_d2 ← θ_d2 − η2 ∇θ_d2 (1/m) Σ log D2(x)
    + log(1 − D2(G(z))
end
//Step 3 Jointly training of G, D1, and D2
Decrease η1
for number of training iterations do
    Update D1:
    θ_d1 ← θ_d1 − η1 ∇θ_d1 (1/m) Σ log D1(x)
    + log(1 − D1(G(z)))
    Update D2:
    θ_d2 ← θ_d2 − η2 ∇θ_d2 (1/m) Σ log D2(x)
    + log(1 − D2(G(z))
    for k iterations do
        Update G:
        θ_g ← θ_g − η1 ∇θ_g (1/m) Σ log(1 − D1(G(z)))
        +λ log(1 − D2(G(z)))
    end
end
```

Therefore, our main idea is that we first train G with the feedback of D_1 only, and then train G with feedbacks of both D_1 and D_2. This incremental learning means that we first teach the basic knowledge of generating samples to G with the lenient D_1, and improve the generation ability of G with the knowledge of superior D_2.

Figure 2 shows the overall procedure of our incremental learning. In the step 1, we train G and D_1 using the common adversarial learning [17]. In the step 2, we fine-tune D_2, which is image classification network (*e.g.* ResNet, DenseNet). To this end, we alter the 1000 outputs of D_2 with 2 units suitable for discriminating real and fake samples, and re-train D_2 from end-to-end learning with data samples and generated samples from G trained in the step 1. Subsequently, the mature G is re-trained from the feedbacks of D_1 and D_2 using joint learning to expend its knowledge. We summarize this incremental learning method in Algorithm 1.

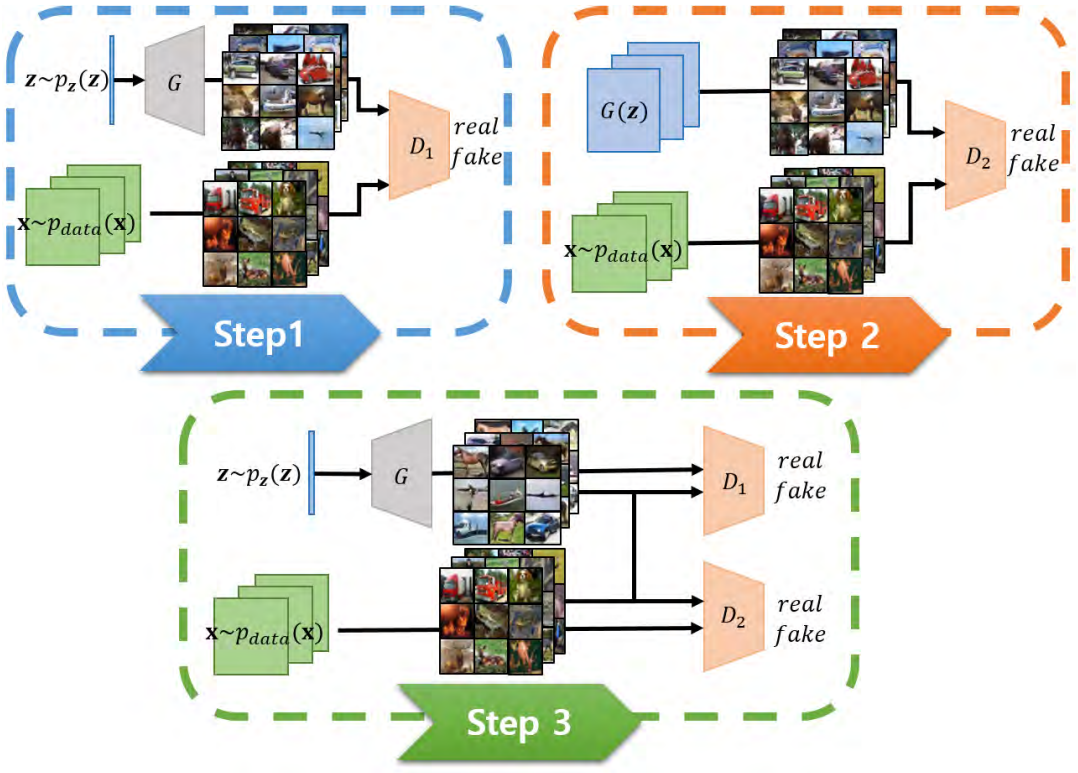

FIGURE 2. The proposed incremental learning method to extend the knowledge of G step-by-step. In the step 1, we learn trainable parameters of G and D_1 using adversarial learning with the loss Eq. (1), and fine-tune D_2 with data samples and generated samples from G in the step 2. In the step 3, all networks are trained together using Eq. (3) and (4).

E. NETWORK ARCHITECTURE

We present the network architecture for generating 256×256 resolution images based on the DCGAN. The dimension of a latent vector $\mathbf{z}$ is 100, and each network consists of a fully connected layer and 6 convolutional or deconvolutional layers. G uses the ReLU as an activation function at 1-5 deconvolution layers, and the tanh at the output layer to generate images. All deconvolutional layer use 5×5 filters with 2 strides.

On the other hand, the discriminator uses the Leaky ReLU ($\alpha = 0.2$) as an activation function. At 1-5 convolutional layers, convolutional filters with the size of 5×5 filters and 2 strides are applied. At the last convolutional layer, we use a sigmoid function.

When generating 32×32 low resolution images, we use a 2×2 filter at the first deconvolutional layer of G, and use 4×4 filters at the other layers. Also, 4×4 filters are applied at 1-5 convolutional layers, but a 2×2 filter is used at 6 convolutional layers.

We use DenseNet [14] or ResNet [13] as D_2. They show the impressive classification results by using skip and dense connections. To use those as D_2, we randomly initialize the weights of the fully connected layers while keeping the pre-trained weights of other layers.

F. DISCUSSION: GANS WITH MULTIPLE DISCRIMINATORS

In this section, we discuss the differences of our IDGAN and other GANs with multiple discriminators. Even though IDGAN and D2GAN contain outputs of two discriminators in the loss functions, both loss functions are significantly different since the usage of the additional discriminator (D_2) is different. In D2GAN, D_2 is used to minimize the reverse KL divergence. As a result, optimizing a generator minimizes forward and reverse KL divergences in D2GAN. On the other hand, our IDGAN exploits D_2 in order to transfer the pre-trained knowledge of D_2 to a generator. Therefore, optimizing a generator is KL divergences as shown in Eq. (10). This means that global solutions between IDGAN and D2GAN are clearly different each other.

Auxiliary classier GAN (AC-GAN) [12] also uses a pre-trained classifier as an auxiliary classifier, but the target domain is different with ours. They consider a semi-supervised learning problem which generates images with class labels, and use the labels as inputs of the loss functions. On the other hand, our IDGAN considers an unsupervised learning problem which generates images without sample labels, and then our loss function does not use sample labels as inputs. Therefore, the loss functions of our IDGAN and AC-GAN are different since target problems are different each other.

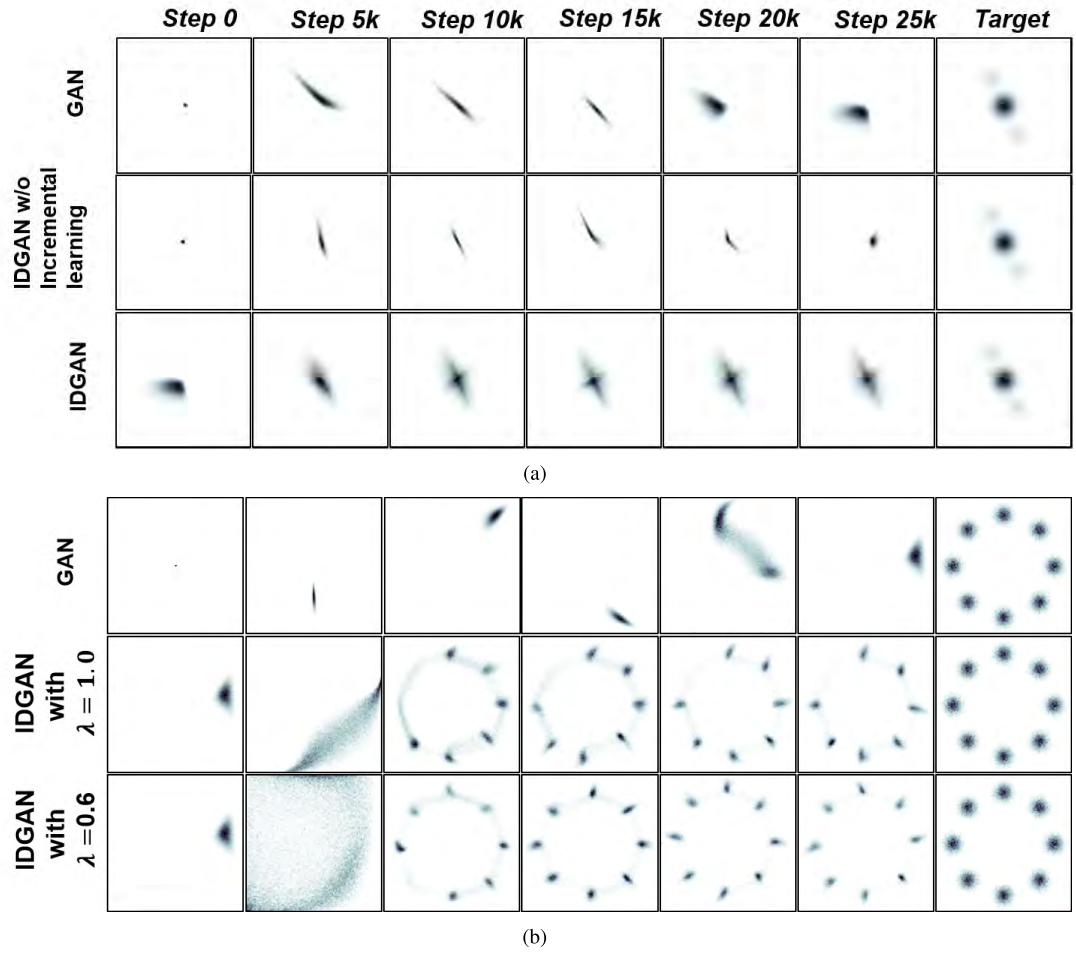

FIGURE 3. The evaluations of different learning methods and loss functions on 2D mixtures of Gaussian distributions. (a) Evaluation of different learning methods on mixtures of 3 Gaussian distributions. (b) Evaluation of different loss functions on mixtures of 8 Gaussian distributions.

In addition, when using a trained classifier with accuracy in GAN training, an excessive negative feedback from the accurate classifier is one of the challenge problem in GAN as mentioned in [11]. AC-GAN does not handle this problem. In fact, GMAN [11] addresses this problem by combining feedbacks of soft-discriminators and feeding the uniform feedbacks to a generator. On the other hand, we resolve this problem by migrating the feedback of D_2 with a scale factor λ and using our incremental learning efficiently. Therefore, our IDGAN can use the knowledge of the powerful discriminator for generator's learning unlike GMAN. In Sec. IV-E and Sec. IV-G, we have shown the effectiveness of our methods to handle the excessive negative feedback.

IV. EXPERIMENTS

In this section, we prove the effects and benefits of our methods on a synthetic 2D mixture of Gaussian, CIFAR-10, CIFAR-100 [18], STL-10 [19] and CelebA-HQ [9] datasets.

When training G and D_1, we use the Adam optimizer [35] with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. We set a learning rate to 0.0002. However, for training DenseNet-169 ($D2$) we use the Nesterov momentum optimizer [36] with $\gamma = 0.9$. Here, we set a learning rate to 0.1. We use the Tensorflow [37]. All our experiments are conducted on a NVIDIA TITAN Xp GPU and an Intel Xeon E5-2640-v4 CPU.

A. DISCUSSION: ROLES OF D_1 AND D_2

Because D_1 and D_2 are heterogeneous for the architectures and training manners, we presume that D_1 and D_2 differently contribute to the G training. To find out the behaviors of G over D_1 and D_2, we evaluate how well G trained with D_1 and D_1&D_2 can approximate the data distributions with several modes.

For this evaluation, we generate synthetic samples from two mixtures of Gaussians. We first generate mixtures of 3 Gaussian distributions with different means μ but the same variance (0.01^2). As shown in Fig. 3, 40k samples are generated with $\mu = (0, 0)$, and 5k samples are produced with $\mu = (0.2, 0.4)$ and $\mu = (-0.2, -0.4)$, respectively. G and D_1 have only a fully connected layer with 128 units, and use ReLU and Leaky ReLU activations. For D_2, we add 2 fully connected layers. In the second Gaussian mixture dataset, 8 Gaussian

TABLE 1. Comparison with other GAN methods: The inception and FID scores on CIFAR-10.

Method	Generator	Discriminator	Inception score	FID
Real data			11.24 ± 0.12	7.8
D2GAN	Standard CNN	Standard CNN	6.51 ± 0.08	61.4
GMAN	Standard CNN	Standard CNN	5.97 ± 0.06	62.7
SN-GAN [39]	Standard CNN	Standard CNN	7.42 ± 0.08	29.3
WGAN-GP [7]	ResNet-101	ResNet-101	6.68 ± 0.06	40.2
DCGAN-RF [40]	Standard CNN	Standard CNN	6.63	40.2
DCGAN	Standard CNN	Standard CNN	6.51 ± 0.01	53.7
WGAN	Standard CNN	Standard CNN	5.98 ± 0.10	49.9
IDGAN(ours)	Standard CNN	Standard CNN	7.32 ± 0.07	32.8
ID-WGAN(ours)	Standard CNN	Standard CNN	6.38 ± 0.05	41.2

distributions with the variance 0.005^2 are combined as in [2]. All the networks have one more fully connected layer compared to previous evaluation. In addition, we train the IDGAN without pre-training on other datasets.

Figure 3 compares different learning methods and loss functions. In the first evaluation, our proposed method using incremental learning approximates the Gaussian mixture distribution better than GAN [3]. When not exploiting incremental learning, the data distribution for the low density is also not captured. This evaluation shows that our IDGAN with incremental learning is effective to make the generator capture the multi modal distribution.

In the second evaluation, exploiting the GAN losses causes the mode collapse problem. When using $\lambda = 1$, a few modes are captured collapsed. This evaluation shows that our IDGAN with proposed loss and learning method alleviates the mode collapse problem because D_1 and D_2 differently contribute to the G training. Furthermore, this evaluation shows that reducing a feedback of D_2 by using $\lambda = 0.6$ leads to more stable training of G than using high λ. We provide more details of the affect of λ in Sec. IV-E and Fig. 7.

B. EVALUATION METRICS

For quantitative evaluation, we adopt the inception score [20] and Fréchet Inception Distance (FID) [22] as evaluation metric. The inception score can evaluate generated images of quality and diversity together. The inception score is evaluated with the outputs (*or* probabilities for classes) of a pre-trained Inception-v3 network [38] on ImageNet. We define the inception score function $IS(G)$ as follows:

$$IS(G) = \exp\left(\mathbb{E}_{\mathbf{z}\sim p_G} KL(p(\mathbf{y}|\mathbf{x})\|p(\mathbf{y}))\right) \tag{11}$$

$$p(\mathbf{y}) = \int_{\mathbf{x}} p(\mathbf{y}|\mathbf{x}) p_G(\mathbf{x}) \tag{12}$$

The high inception score indicates that each generated image contains a clear single object and generative model should output high density of images for all classes of ImageNet [20], [21].

A disadvantage of the inception score is that it does not use the statistics of real world samples. To compare the statistics of synthetic samples with those of real world samples, we use the FID metric with the pre-trained Inception-v3 network. FID uses the features from the last hidden layer, which is the last pooling layer in general. m and m_w are the means of output features extracted from the synthetic and real data samples, and C and C_w are the covariance of the output features from synthetic and real samples. Using this statistics Fréchet distance function d^2 can be evaluated as follows [22]:

$$d^2((m, C), (m_w, C_w)) = \|m - m_w\|_2^2 + Tr(C + C_w - 2(CC_w)^{\frac{1}{2}}), \tag{13}$$

where $\|\cdot\|_2$ is the Euclidean L_2 norm, and $Tr(M)$ is the trace of a matrix (M). The lower FID score means better performance since a generative model captures more information for a training dataset.

C. COMPARISONS WITH STATE-OF-THE-ART GANS ON CIFAR-10 AND STL-10

In this section, we demonstrate that IDGAN improves the image quality as well as image the diversity. As evaluation metrics, we use the inception score [20] and Fréchet Inception Distance (FID) [22]. On CIFAR-10, CIFAR-100 [18] and STL-10 [19] datasets, we generate images with unsupervised learning. In the CIFAR-10 and CIFAR-100 dataset, 60k images of the 32×32 resolution for 10 and 100 classes are included, respectively. On the other hand, the STL-10 dataset, which is a subset of ImageNet, contains 100k unlabeled images. We resize images of STL-10 to 48×48 resolution as done in other works [39].

For evaluating the quality and diversity of images produced by our IDGAN, we compare images from IDGAN with those of DCGAN on CIFAR-10. Then, we compute the inception score for 50k images generated from the IDGAN and DCGAN. Subsequently, we implement ID-WGAN by applying our architecture and incremental learning methods for WGAN while maintaining its loss. To compare the different normalization methods, we train our IDGAN with layer normalization [42] (our GANs without the layer normalization employ the batch normalization). On the STL-10 dataset, we also generate images by using our IDGAN, and compute the inception and FID scores to compare other methods.

TABLE 2. Comparison with other GAN methods: The inception and FID scores on STL-10.

Method	Discriminator	Generator	Inception score	FID
Real data			26.08 ± 0.26	7.9
DCGAN	Standard CNN	Standard CNN	7.43 ± 0.13	93.3
GMAN	Standard CNN	Standard CNN	7.67 ± 0.07	49.9
WGAN-GP [7]	ResNet-101	ResNet-101	8.42 ± 0.13	55.1
SNGAN [39]	Standard CNN	Standard CNN	8.28 ± 0.09	53.1
M-BGAN [41]	Standard CNN	Standard CNN	8.7	51
IDGAN(ours)	Standard CNN	Standard CNN	7.80 ± 0.10	48.1

TABLE 3. Ablation experiments: The inception and FID scores on CIFAR-10.

Method	Inception score	FID
-normalization-		
Batch norm.	7.32 ± 0.07	32.8
Layer norm.	7.08 ± 0.07	33.3
-Training manner-		
Proposed incremental learning	7.32 ± 0.07	32.8
Joint training	6.90 ± 0.08	55.8
Alternative training	6.70 ± 0.07	56.9
-D_2 architecture-		
Standard CNN	6.63 ± 0.08	47.3
DenseNet-121	6.89 ± 0.07	32.4
DenseNet-169	7.32 ± 0.07	32.8
ResNet-50	6.80 ± 0.08	32.5
ResNet-152	$6.86 \pm 0.$	33.4

In Table 3 and 2, we compare our IDGANs with other GANS. Compared to the DCGAN, IDGAN improves both inception and FID scores for all the evaluations. In particular, the FID score is significantly improved. Since the FID score measures the produced image quality, our IDGAN can enhance the quality. When comparing between WGAN and ID-WGAN, exploiting the auxiliary discriminator and our incremental learning improves the scores. However, we verify that our loss Eq. (2) is more appropriate for leveraging the knowledge of the ImageNet discriminator when comparing the results of IDGAN and ID-WGAN. The layer normalization does not provide the score gain for our IDGAN in our case.

In Fig. 4, we compare the inception scores per epoch. Even though we can improve the scores with more training gradually, employing our loss and learning methods increases the score more rapidly.

Compared to the state-of-the-art GAN methods shown in Table 3 and 2, our IDGAN achieves the better performance for both metrics than other methods. In particular, our IDGAN produces the best FID score on the STL-10 dataset.

D. COMPARISONS WITH GANS WITH MULTIPLE DISCRIMINATORS

To compare with the other recent methods using multiple discriminators, we implement D2GAN [10] and GMAN [11] using the public available official codes of both methods.[6] However, both networks are implemented on the same architecture used in IDGAN. The details of the network architecture are given in Sec. III-E. In our implementation, GMAN uses 5 discriminators. We train the D2GAN and GMAN

[6]Code is available at https://github.com/tund/D2GAN, GMAN code is available at https://github.com/iDurugkar/GMAN

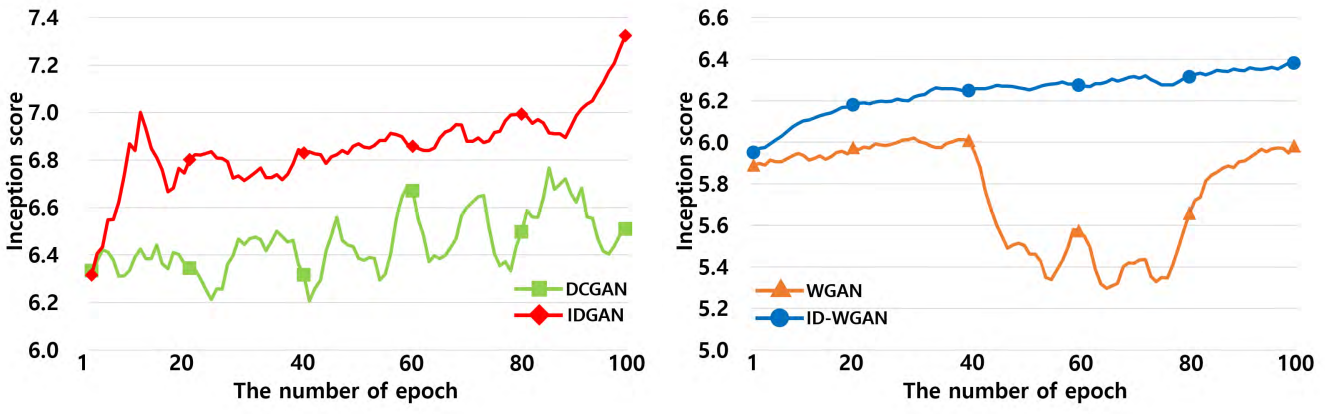

FIGURE 4. On the CIFAR-10 dataset, the inception score curves per epoch at the step 3. At the step 1, we first train DCGAN and WGAN for 200 epochs. At step 2, we train D_2 (DenseNet-169) during 10 epochs. We then train all the GANs for 100 epochs at the step 3. (a) The inception score curves of IDGAN and DCGAN. (b) The inception score curves of ID-WGAN and WGAN.

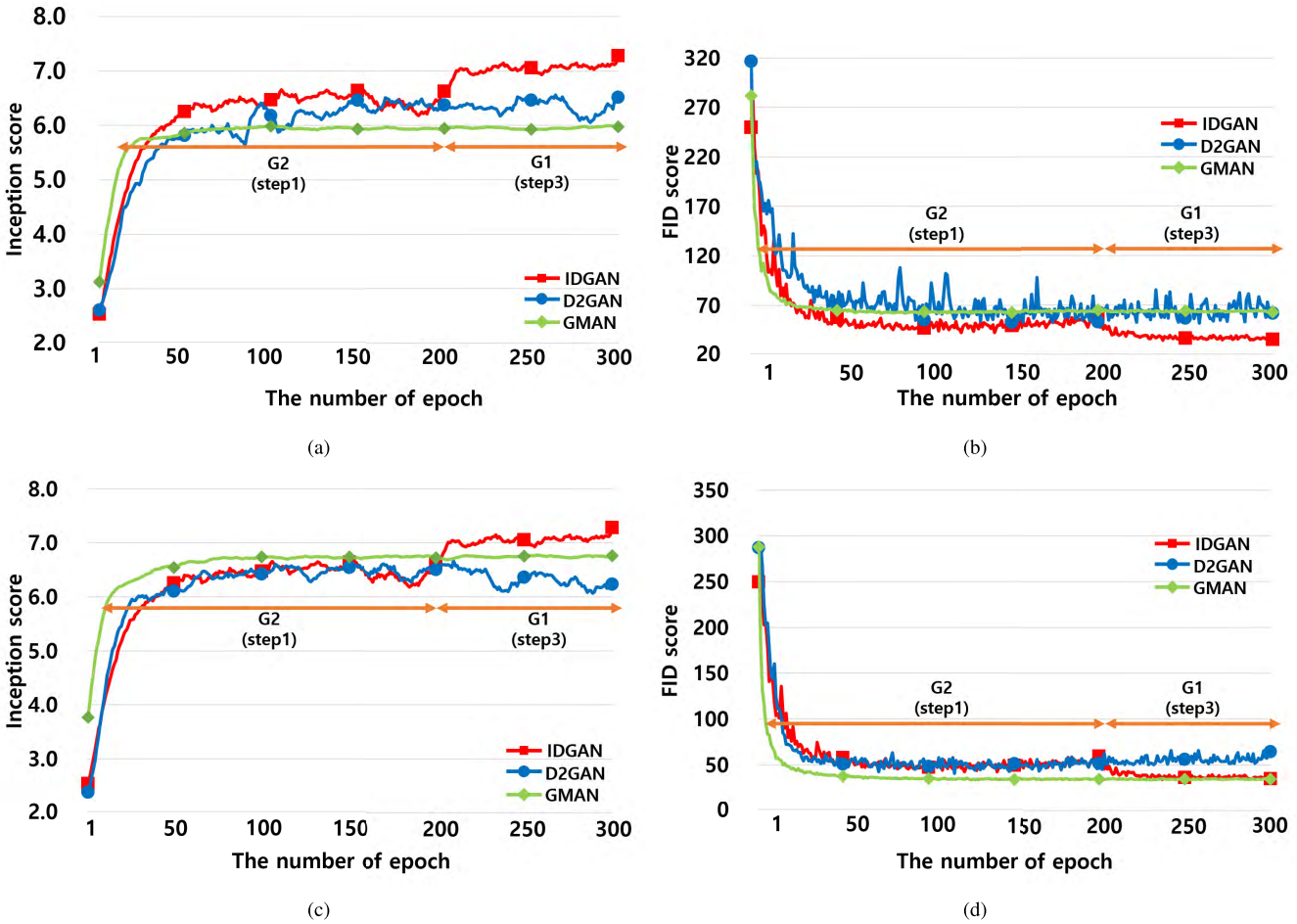

FIGURE 5. On the CIFAR-10 and CIFAR-100 dataset, the inception and FID scores per epoch. We first train the IDGAN, D2GAN and GMAN for 300 epochs. In the curves of IDGAN, the inception and FID scores before and after 200 epochs represent the IDGAN performance during step 1 and step 3, respectively. (a) The inception score curves of IDGAN, D2GAN and GMAN on CIFAR-10. (b) The FID score curves of IDGAN, D2GAN and GMAN on CIFAR-10. (c) The inception score curves of IDGAN, D2GAN and GMAN on CIFAR-100. (d) The FID score curves of IDGAN, D2GAN and GMAN on CIFAR-100.

during 300 epochs on CIFAR-10 and CIFAR-100, but train them during 200 epochs on STL-10. For evaluation, we compute the inception and FID scores of methods for 50k images per epoch.

In Fig. 5, we compare IDGAN with D2GAN and GMAN on CIFAR-10 and CIFAR-100. Note that the inception and FID scores of IDGAN are shown after 200 epoch starting out the step 3. Here, we verify that the performance can

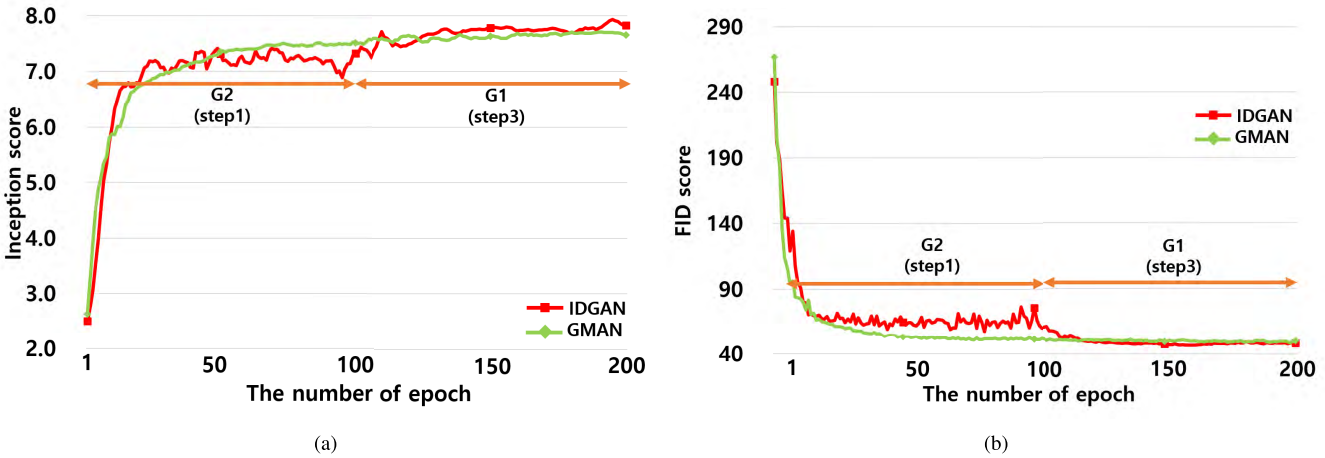

FIGURE 6. On the STL-10 dataset, the inception and FID scores per epoch: We train IDGAN and GMAN for 200 epochs. In the curves of IDGAN, the inception and FID scores before and after 100 epochs represent the IDGAN performance during step 1 and step 3, respectively. (a) The inception score curves of IDGAN and GMAN on STL-10. (b) The FID score curves of IDGAN and GMAN on STL-10.

TABLE 4. On several target datasets, inception and FID scores of IDGANs trained with different source datasets.

Target domain	Pre-trained model	Source domain			
CelebA-HQ (FID)	Inception-ResNet-v1	Scratch	ImageNet	VGG Face	CASIA-WebFace
		28.61	30.44	30.05	30.44
CIFAR-10 (FID/IS)	DenseNet-169	Scratch	ImageNet		
		31.85 / 6.98 ± 0.07	30.14 / 7.32 ± 0.07		
CIFAR-100 (FID/IS)	DenseNet-169	Scratch	ImageNet		
		37.5 / 7.30 ± 0.08	29.96 / 7.57 ± 0.09		

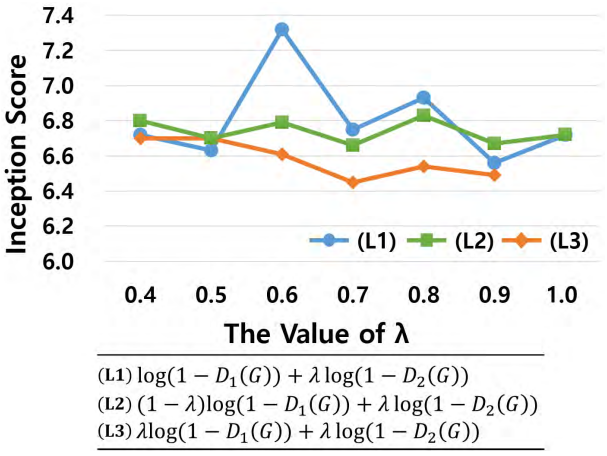

FIGURE 7. The inception scores of our IDGAN with different losses and λ on the CIFAR-10 dataset.

be boosted further by this step. Compared to the D2GAN and GMAN, our IDGAN show the better performance in terms of both metrics. On CIFAR-100 including many object classes, GMAN shows the impressive performance. However, our IDGAN also achieves the similar FID score to that of GMAN. In particular, the inception score of IDGAN are much higher than that of GMAN. Remarkably, IDGAN achieves the performance with 2 discriminators only while GMAN uses 5 discriminators. This proves the effectiveness and usefulness of our learning methods.

For the more comparison, we compare our IDGAN with the GMAN on STL-10 in Fig. 6.[7] Similarly, the inception and FID scores after 100 epochs represent the performance of IDGAN at step 3. On this dataset, our IDGAN shows the better inception and FID scores than those of GMAN. From these comparisons, we verify that the our main idea of transferring the knowledge of image classification network to GANs is indeed beneficial for image generation. In addition, the proposed GAN framework, loss function, and incremental learning are suitable for the transfer learning. Furthermore, they make GAN scale up to generate images for many different object classes.

[7]The implemented D2GAN is not trained well on this dataset.

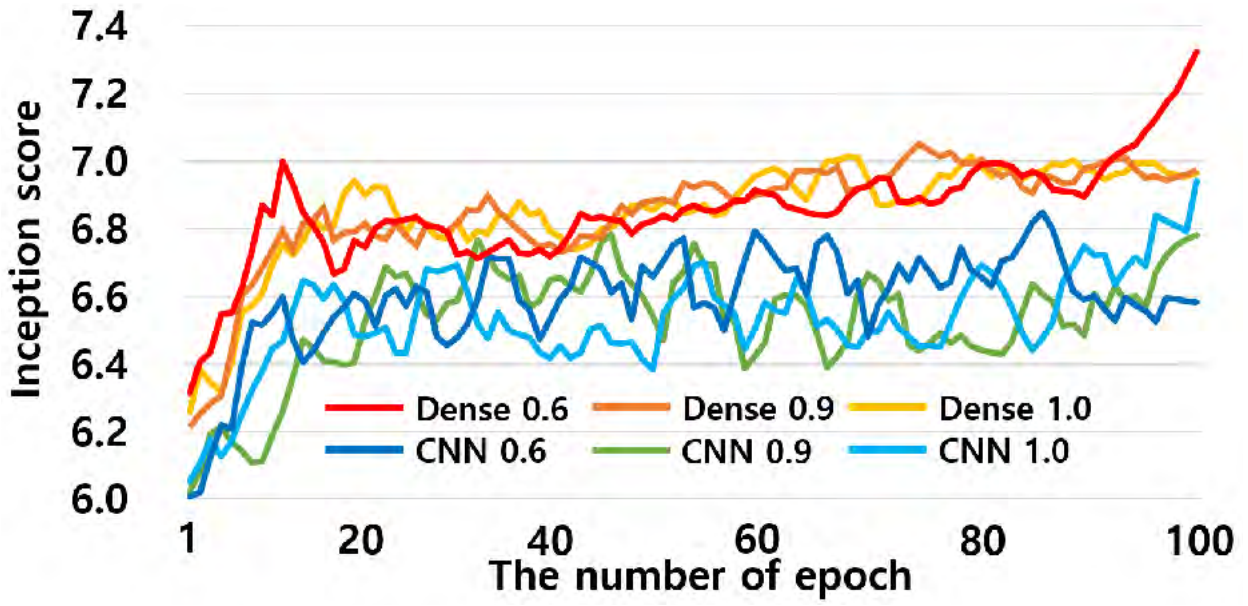

FIGURE 8. On the CIFAR-10 dataset, the inception scores of IDGANs per epoch: We use the DenseNet-169 (Dense) and standard CNN (CNN) as D_2, and evaluate IDGANs by applying different scaling factor ($\lambda = [0.6, 0.9, 1.0]$) for the losses of D_2. More discussion can be found in Sec. IV-E.

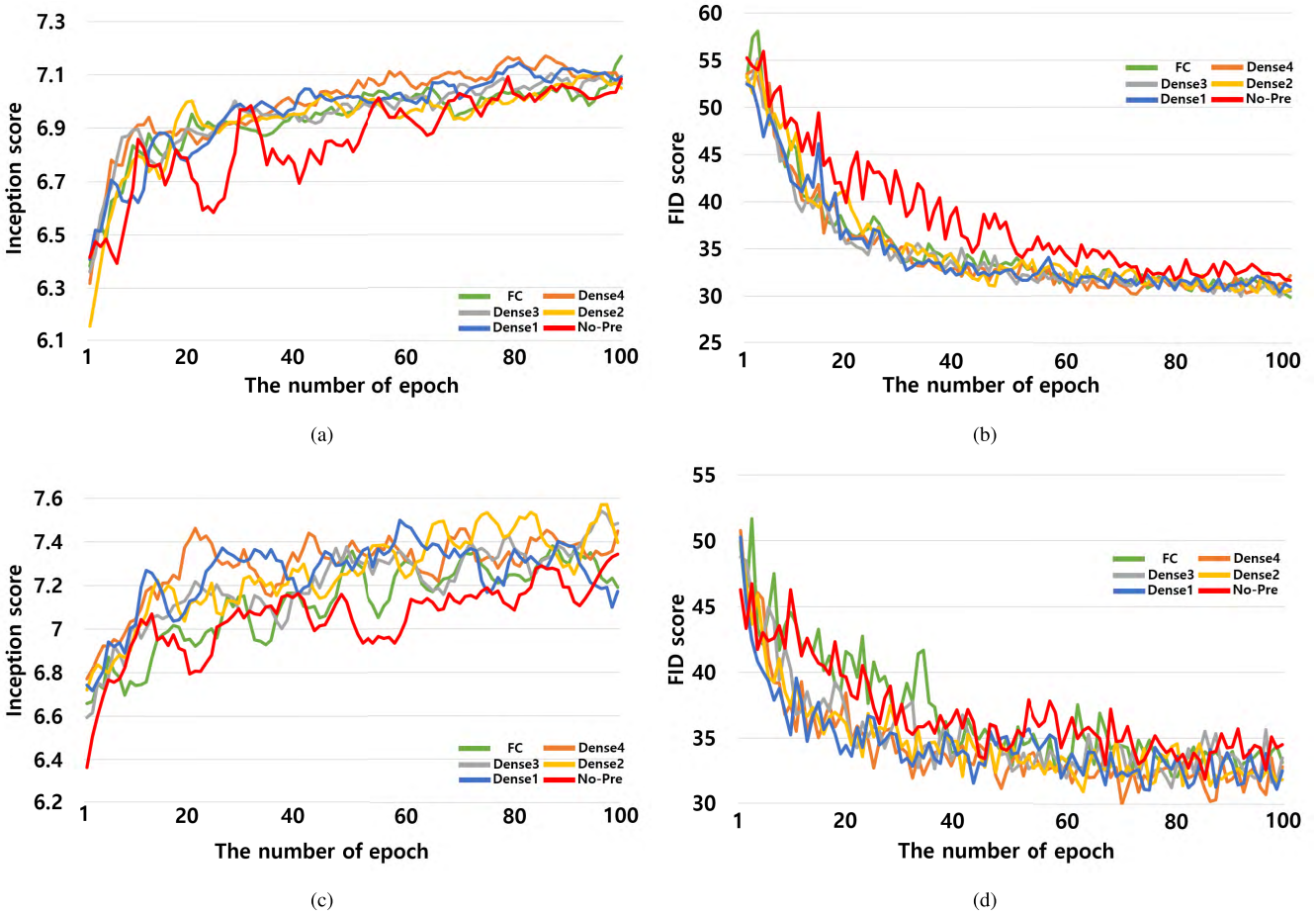

FIGURE 9. On the CIFAR-10 and CIFAR-100 datasets, the inception and FID scores of IDGANs per epoch: We use the DenseNet-169 as D_2 and evaluate the IDGANs by applying these different transfer learning ways for D_2. More details can be found in Sec. IV-F. (a) The inception score curves on CIFAR-10. (b) PThe FID curves on CIFAR-10. (c) The inception score curves on CIFAR-100. (d) The FID curves on CIFAR-100.

E. ABLATION EXPERIMENTS

In this section, we evaluate the effectiveness of our proposed loss function, learning method, and network architecture. We first compare the different loss functions with a scaling factor λ, where $0 < \lambda \leq 1$, as follows:

(L1) $\log(1 - D_1(G)) + \lambda \log(1 - D_2(G))$

(L2) $(1 - \lambda)\log(1 - D_1(G)) + \lambda \log(1 - D_2(G))$

(L3) $\lambda \log(1 - D_1(G)) + \lambda \log(1 - D_2(G))$

(L1) is proposed one, (L2) is the sum of the factors for D_1 and D_2 to be one, and (L3) applies the same λ for both networks. We change $\lambda \in [0.3, 0.4, \ldots, 1.0]$. Given that G, D_1 and D_2 are trained by the step 1 and step 2, we jointly train them with with the losses (L1)-(L3). Figure 7 shows the inception scores when different negative feedbacks are fed to our IDGAN. As λ increases, the feedback of D_2 affects the proposed loss (L1) more. When using $\lambda = [0.9, 1]$ for (L1), the inception score gets lower than using $\lambda = 0.6$. (L2) and (L3) which controls feedbacks in different manners also show much lower scores than (L1).

However, the effects of feedbacks of discriminators can be different according to the accuracy of the discriminator. This is because discriminators with high accuracy could provide more strict and higher negative feedbacks to the fake images generated by G. Therefore, we compare IDGANs with different D_2 as shown in Fig. 8. We use the DenseNet-169 (Dense) and standard CNN (CNN) as $D2$. Here, we apply different $\lambda = [0.6, 0.9, 1.0]$ for the loss term of the Dense and CNN.

As can be seen, DenseNet-169 using $\lambda = 0.6$ (Dense 0.6) shows higher inception scores than $\lambda = 0.9$ (Dense 0.9) and $\lambda = 1.0$ (Dense 1.0). However, the score of standard CNN using $\lambda = 0.6$ (CNN 0.6) is almost similar as those of $\lambda = 0.9$ (CNN 0.9) and $\lambda = 1.0$ (CNN 1.0). These results reflect that the excessive negative feedback can be caused by a strict discriminator, but not caused by a gentle discriminator. Also, the excessive feedbacks can degrade GAN performance as shown. However, our loss function can resolve this problem by controlling the feedback of D_2.

In addition, we change the D_2 with other ImageNet networks. As shown in Table 3, using networks with deeper layers improves the metric scores, and we achieve the best rates by using DenseNet-169 as D_2. We also compare different training methods in Table 3. Here, the joint training indicates that G, D_1 and D_2 are trained simultaneously without the previous learning of the step 1 and step 2 in Fig. 2. In the alternative training, we train G with the feedback of D_1 or D_2 alternatively at the step 3. From this comparison, we prove

FIGURE 10. Generated 256 × 256 resolution images by the DCGAN and IDGAN on the CelebA-HQ dataset. Each pair of images is generated by using the same latent vector as an input.

that our incremental learning is much more effective for IDGAN.

F. TRANSFER LEARNING

To show the effects of our transfer learning, we compare the performance of our GAN with/without transfer learning on several target datasets such as CelebA-HQ, CIFAR-10, and CIFAR-100. On the CelebA-HQ evaluation, we use the Inception-ResNet-v1 models trained on VGG-Face [43], CASIA-WebFace [44], and ImageNet datasets[8] as D_2. In addition, we use the DensetNet-169 model pre-trained on ImageNet as D_2 on CIFAR-10 and CIFAR-100. Table 4 shows the Inception and FID scores of our IDGAN with the pre-trained models on the several target datasets. Since

[8]The models are provided from the FaceNet.

the face class is not included in 1000 classes of ImageNet, the inception score cannot be evaluated for CelebA-HQ.

On CIFAR-10 and CIFAR-100 sets, our IDGAN with the pre-trained models shows the better scores than IDGAN trained from scratch. In particular, the FID score dramatically increases on CIFAR-100. However, the performance of IDGAN does not increase on CelebA-HQ even though we apply several Inception-Resnet-v1 networks trained with different source datasets. From these experimental results, we confirm that our transfer learning can be effective when a target domain contains multiobject classes. In particular, the more performance improvement can be achieved by our transfer learning as object classes increase. However, in the case of a single object class we consider that IDGAN without pre-trained D_2 could provide the better performance than using pre-trained D_2.

To find out the best way when re-training D_2 on a target dataset, we train DenseNet-169 in different manners. Here, it was pre-trained with ImageNet and consists of 1 fully connected layer, 4 dense blocks and 1 convolutional layer above the input layer from top to bottom. We re-train (FC) the fully connected layer only, but (Dense4/Dense3/Dense3/Dense1) from the fully connected layers to the indicated dense block while remaining parameters of other layers. (No-Pre) means that parameters of all layers are learned from scratch. On CIFAR-10 and CIFAR-100, we evaluate the IDGANs by applying these different transfer learning ways for D_2.

In Fig. 9, we compare the inception and FID scores of the IDGANs per epoch. When comparing (No-Pre) and the others, we find that our transfer learning is indeed beneficial for improving GANs on both datasets In particular, the performance gain from our transfer learning is much higher in CIFAR-100 dataset. This means that our method is more beneficial for the dataset containing diverse modes. (Dense2) achieves the best inception and FID scores on both datasets. This indicates that learning semantic features of higher layers is also effective for transfer learning of GANs while remaining generic features in lower layers.

G. EFFECTS OF DISCRIMINATORS D_1 AND D_2

To analyze the effects of D_1 and D_2, we should compare the following IDGANs with different discriminators:
(G1) IDGANs with $D1$ and $D2$;
(G2) IDGANs with D_1;
(G3) IDGANs with D_2;
where (G1) is the IDGAN trained at step 3 as shown in Fig. 2. (G2) is a GAN trained at step 1. In (G3), D_2 is a pre-trained discriminator on a dataset. In case of (G3), we however find that G is not trained well. The reason is that the excessive negative feedback at early stage could guide the distribution of a generator to the other data region as mentioned in Sec. III-B.

However, in order to compare (G1) and (G2) more clearly, we evaluate the (G1) and (G2) performance per epoch on CIFAR-10, CIFAR-100, and STL-10 as shown in Fig. 5-6. Here, we evaluate the (G1) and (G2) performance in terms of inception and FID scores. As can be seen, D2 indeed contributes to the performance enhancement of IDGAN.

H. CELEBA-HQ EVALUATION

To show the ability of generating high resolution images of our IDGAN, we use the CelebA-HQ dataset consisting of 30k face images of 1024×1024 resolution. In our experiment, we resized CelebA-HQ to 256×256 resolution. Therefore, we design the network architecture as described in Sec. III-E. However, we train our IDGAN without transfer learning since it provides the better results as described in Sec. IV-F. Figure 10 compares the generated images from DCGAN and IDGAN. We also compute the FID scores of the DCGAN and IDGAN. For 50k images, DCGAN and IDGAN achieve 49.97 and 32.08. From the results, we know that our IDGAN generates more realistic images than DCGAN.

V. CONCLUSION

In this paper, we have proposed the IDGAN for improving a GAN. IDGAN uses an image classification network as an auxiliary discriminator. As a result, it can produce more diverse and better quality image by using the knowledge of the classification network. In order to train three networks adversarially, we have designed the new loss function, and provided the theoretical analysis to verify optimality of the loss. For more stable training of IDGAN, we present the incremental learning method to train the generator gradually.

From various evaluations, we have verified the effectives and usefulness of the proposed methods. Compared to the recent GANs, our IDGAN shows the better inception and FID scores on CIFAR-10 and STL-10 datasets. In addition, we show the image generation ability of IDGAN in high resolution on the CelebA-HQ dataset. We believe that our loss and learning method can be compatible with other GANs since it does not depend on the GAN architecture. For instance, we have implemented the improved WGAN with two discriminators by adding the WGAN loss of the added discriminator. In a similar way, IDGAN uses two discriminators only, but it can use more discriminators by adding the losses of auxiliary discriminators.

REFERENCES

[1] I. J. Goodfellow. (Apr. 2017). "NIPS 2016 tutorial: Generative adversarial networks." [Online]. Available: https://arxiv.org/abs/1701.00160
[2] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled generative adversarial networks," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, Apr. 2017.
[3] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, Dec. 2014, pp. 2672–2680.
[4] S. Nowozin, B. Cseke, and R. Tomioka, "f-GAN: Training generative neural samplers using variational divergence minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, Dec. 2016, pp. 271–279.
[5] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2813–2821.
[6] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, Sydney, NSW, Australia, Aug. 2017, pp. 214–223.
[7] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 5767–5777.

[8] D. Berthelot, T. Schumm, and L. Metz. (Mar. 2017). "BEGAN: Boundary equilibrium generative adversarial networks." [Online]. Available: https://arxiv.org/abs/1703.10717
[9] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)* 2018, Vancouver, BC, Canada, Apr./May 2018.
[10] T. Nguyen, T. Le, H. Vu, and D. Phung, "Dual discriminator generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 2670–2680.
[11] I. P. Durugkar, I. Gemp, and S. Mahadevan, "Generative multi-adversarial networks," *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, Apr. 2017.
[12] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, Sydney, NSW, Australia, Aug. 2017, pp. 2642–2651.
[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[14] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, Honolulu, HI, USA, Jul. 2017, pp. 2261–2269. doi: 10.1109/CVPR.2017.243.
[15] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 3859–3869.
[16] R. Saqur and S. Vivona. (Jun. 2018). "CapsGAN: Using dynamic routing for generative adversarial networks." [Online]. Available:https://arxiv.org/abs/1806.03968
[17] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, 2016, San Juan, Puerto Rico, May 2016.
[18] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009. [Online]. Available: https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf
[19] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. 14th Int. Conf. Artif. Intell. Statist. (AISTATS)*, Fort Lauderdale, FL, USA, Apr. 2011, pp. 215–223.
[20] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, Dec. 2016, pp. 2234–2242.
[21] S. Barratt and R. Sharma. (Jan. 2018). "A note on the inception score." [Online]. Available: https://arxiv.org/abs/1801.01973
[22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 6626–6637.
[23] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics). Springer, 2006, pp. 196–203.
[24] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Vancouver, BC, Canada, Dec. 2001, pp. 841–848.
[25] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada, Apr. 2014.
[26] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
[27] A. Van Den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proc. 33nd Int. Conf. Mach. Learn. (ICML)*, New York City, NY, USA, Jun. 2016, pp. 1747–1756.
[28] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," *CoRR*, Nov. 2017.
[29] A. Antoniou, A. J. Storkey, and H. Edwards. (Nov. 2017). "Data augmentation generative adversarial networks." [Online]. Available: https://arxiv.org/abs/1711.04340
[30] H. Shi, L. Wang, G. Ding, F. Yang, and X. Li, "Data augmentation with improved generative adversarial networks," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Beijing, China, Aug. 2018, pp. 73–78.
[31] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa, "Generate to adapt: Aligning domains using generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, Jun. 2018, pp. 8503–8512.
[32] J. Choe, S. Park, K. Kim, J. H. Park, D. Kim, and H. Shim, "Face generation for low-shot learning using generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Venice, Italy, Oct. 2017, pp. 1940–1948.
[33] Y. Li and L. Shen, "cC-GAN: A robust transfer-learning framework for HEp-2 specimen image segmentation," *IEEE Access*, vol. 6, pp. 14048–14058, 2018.
[34] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015.
[36] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. 30th Int. Conf. Mach. Learn. (ICML)*, Atlanta, GA, USA, Jun. 2013, pp. 1139–1147.
[37] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, Savannah, GA, USA, Nov. 2016, pp. 265–283.
[38] 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. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 2818–2826.
[39] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, Vancouver, BC, Canada, Apr./May 2018.
[40] D. Bang and H. Shim, "Improved training of generative adversarial networks using representative features," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*. Stockholm, Sweden: Stockholmsmässan, Jul. 2018, pp. 442–451.
[41] T. Lucas, C. Tallec, Y. Ollivier, and J. Verbeek, "Mixed batches and symmetric discriminators for GAN training," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)* 2018, Stockholm, Sweden: Stockholmsmässan, Jul. 2018, pp. 2850–2859.
[42] L. J. Ba, R. Kiros, and G. E. Hinton. (Jul. 2016). "Layer normalization." [Online]. Available: https://arxiv.org/abs/1607.06450
[43] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, Swansea, U.K., Sep. 2015, pp. 41.1–41.12.
[44] D. Yi, Z. Lei, S. Liao, and S. Z. Li. (Nov. 2014). "Learning face representation from scratch." [Online]. Available: https://arxiv.org/abs/1411.7923

JAE-YONG BAEK received the M.S. degree in computer engineering from Incheon National University, South Korea, in 2019. His current research interests include generative adversarial networks, generative model, image classification, and deep learning.

YONG-SANG YOO is currently pursuing the B.S. degree with the Department of Computer Science and Engineering, Incheon National University, South Korea. His current research interests include generative adversarial networks, objection detection, machine learning, multi-object tracking, and deep learning.

SEUNG-HWAN BAE (M'18) received the B.S. degree in information and communication engineering from Chungbuk National University, in 2009, and the M.S. and Ph.D. degrees in information and communications from the Gwangju Institute of Science and Technology (GIST), in 2010 and 2015, respectively. He was a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), South Korea, from 2015 to 2017. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Incheon National University, South Korea. His research interests include multi-object tracking, object detection, deep learning, dimensionality reduction, medical image analysis, and generative adversarial networks.