Collaborative Learning of Generative Adversarial Networks

Takuya Tsukahara[1], Tsubasa Hirakawa[1], Takayoshi Yamashita[1] and Hironobu Fujiyoshi[1]
[1]*Chubu University, 1200 Matsumoto-cho, Kasugai, Aichi, Japan*
{*tsukahara, hirakawa*}*@mprg.cs.chubu.ac.jp,* {*takayoshi, fujiyoshi*}*@isc.chubu.ac.jp*

Keywords: Generative adversarial networks, Deep mutual learning, Deep learning, Convolutional neural network

Abstract: Generative adversarial networks (GANs) adversarially train generative and discriminative and generate a nonexistent images. Common GANs use only a single generative model and discriminant model and are considered to maximize their performance. On the other hand, in the image-classification task, recognition accuracy improves by collaborative learning in which knowledge transfer is conducted among several neural networks. Therefore, we propose a method that involves using GANs with multiple generative models and one discriminant model to conduct collaborative learning while transferring information among the generative models. We conducted experiments to evaluate the proposed method, and the results indicate that the quality of the images produced by the proposed method is improved and increased in diversity.

1 INTRODUCTION

Generative adversarial networks (GANs) adversarially train generative and discriminative models (Goodfellow et al., 2014). In the discriminator, which is a discriminative model, learning is conducted so that the real image and image generated with the generative model can be distinguished. The generator, which is a generative model, generates an image using latent variables and is trained so that the generated image can be distinguished from the real image by using the discriminative model. GANs that can generate high-definition images include deep convolutional GANs (DCGANs), progressive growing GANs (progressive GANs), and self-attention GANs (SAGANs) (Radford et al., 2016; Karras et al., 2018; Zhang et al., 2019). DCGANs use convolutional neural networks (CNNs) (LeCun et al., 1998), progressive GANs deepen the layers of the generator and discriminator as learning progresses, and SAGANs improve the quality of the generated image by introducing a self-attention mechanism. Methods using these models generate high-definition images by devising the structure of the neural network, but the image generated from the generator is over-optimized to trick the discriminator, resulting in mode collapse (Goodfellow, 2016).

Various methods have been proposed to suppress mode collapse, and one involves using multi-agent diverse GANs (MAD-GANs). MAD-GANs are composed of multiple generators and one discriminator (Ghosh et al., 2018). If the image generated by one generator is over-optimized to trick the discriminator, another generator can compensate to prevent mode collapse. However, the problem is that each generator loses the diversity of the generated images. Therefore, when there are multiple generators, it is thought that by conducting collaborative learning among the generators and transferring knowledge among them, each generator can acquire more knowledge and increase the diversity of the generated images. We also believe that learning that makes the most of the strengths of each generator is possible.

In the image-classification task, recognition accuracy improves compared with ordinary supervised learning by conducting learning in which knowledge is transferred to each of several neural networks (Hinton et al., 2014; Romero et al., 2015; Zhu et al., 2018; Song and Chai, 2018; Zhang et al., 2018). Knowledge-transfer methods for neural networks include knowledge distillation (KD) (Hinton et al., 2014) for unidirectional transmission and deep mutual learning (DML) (Zhang et al., 2018) for bidirectional transmission. KD is a method for learning a student network, which is a lightweight neural network, using a teacher network, which is a large-scale, pre-trained neural network with excellent recognition accuracy. Thus, superior performance can be obtained compared to the student network in which normal supervised learning is conducted. DML conducts mutual learning between student networks and is effective not only for learning by combining student networks with different structures but also for improving recognition accuracy between networks with the same structure. It is also possible to train three or more neural networks at the same time.

Similar to the image classification task, GANs can maximize the performance of neural networks by conducting learning in which knowledge is transferred to each of several neural networks. Therefore, we propose a method with which learning is conducted using multiple generators, and the generators conduct joint learning while transferring knowledge to one another. Learning that uses the characteristics of each generator and increases diversity of the generated images are achieved. The performance of a neural network is also maximized by optimizing the method of transmitting knowledge among generators. We conducted experiments to evaluate the proposed method, and the results indicate the effectiveness of the method in transferring knowledge.

1.1 Contributions

The contributions reported here are summarized as follows.

- GANs research has resulted in the proposal of many methods of generating high-resolution images by devising the structure of neural networks, but they do not achieve maximum performance because it does not conducting learning in which knowledge is transferred to each of several neural networks. With the proposed method, multiple generators conduct collaborative learning and transfer knowledge to one another to conduct learning that makes the best use of the strengths of each generator.
- MAD-GANs, which use multiple generators, have been proposed to suppress mode collapse, but MAD-GANs lose the diversity of images generated by each generator. With the proposed method, multiple generators conduct collaborative learning, transfer knowledge to one another, and increase the diversity of the generated images.

2 RELATED WORK

A knowledge transfer graph is used to illustrate the collaborative learning of image-classification tasks and express various knowledge-transfer methods including KD and DML (Minami et al., 2019). GANs that use multiple generators have also been proposed (Ghosh et al., 2018; Hoang et al., 2018; Nguyen et al., 2017; Durugkar et al., 2017), i.e., MAD-GANs and

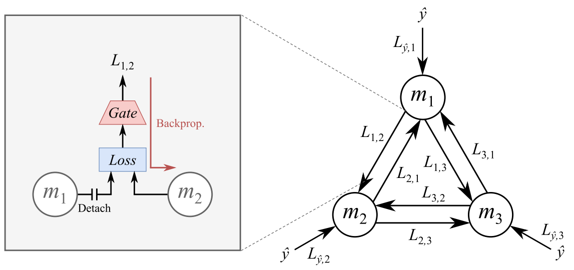


Figure 1: Knowledge transfer graph ($k = 3$) (Minami et al., 2019)

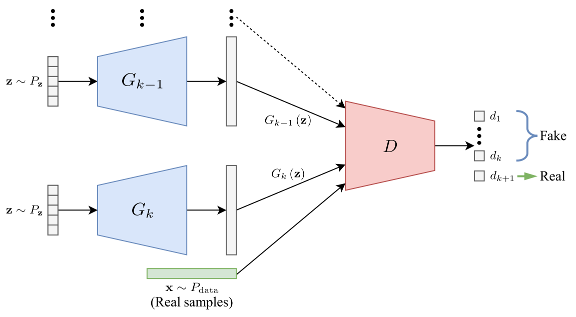

$\mathbf{z} \sim P_{\mathbf{z}}$
G_{k-1}
$G_{k-1}(\mathbf{z})$
$\mathbf{z} \sim P_{\mathbf{z}}$
G_k
$G_k(\mathbf{z})$
D
d_1
d_k
d_{k+1}
Fake
Real
$\mathbf{x} \sim P_{\text{data}}$
(Real samples)

Figure 2: Network structure of MAD-GANs (Ghosh et al., 2018)

mixture GANs (MGANs) (Hoang et al., 2018). On the other hand, dual discriminator GANs (D2GANs) (Nguyen et al., 2017) conduct learning using two discriminators and generative multi-adversarial networks (GMANs) (Durugkar et al., 2017) conduct learning using multiple discriminators. We focus on the knowledge transfer graph and MAD-GANs, which are highly relevant to our research.

2.1 Knowledge transfer graph

Figure 1 shows a knowledge transfer graph when three neural networks are used. If the number of neural networks used for learning is k, each neural network is denoted as $m_1, \ldots, m_k$ and represented as a node. In addition, two edges are defined between each node and represented by directed graphs with different orientations. This edge represents the direction in which gradient information is transmitted during learning. An individual loss function is defined for each edge, and the propagated loss is controlled by introducing a gate function inside the loss function. By using such expressions, it is possible to express KD and DML. Various learning methods can also be expressed by combining selected models and loss functions.

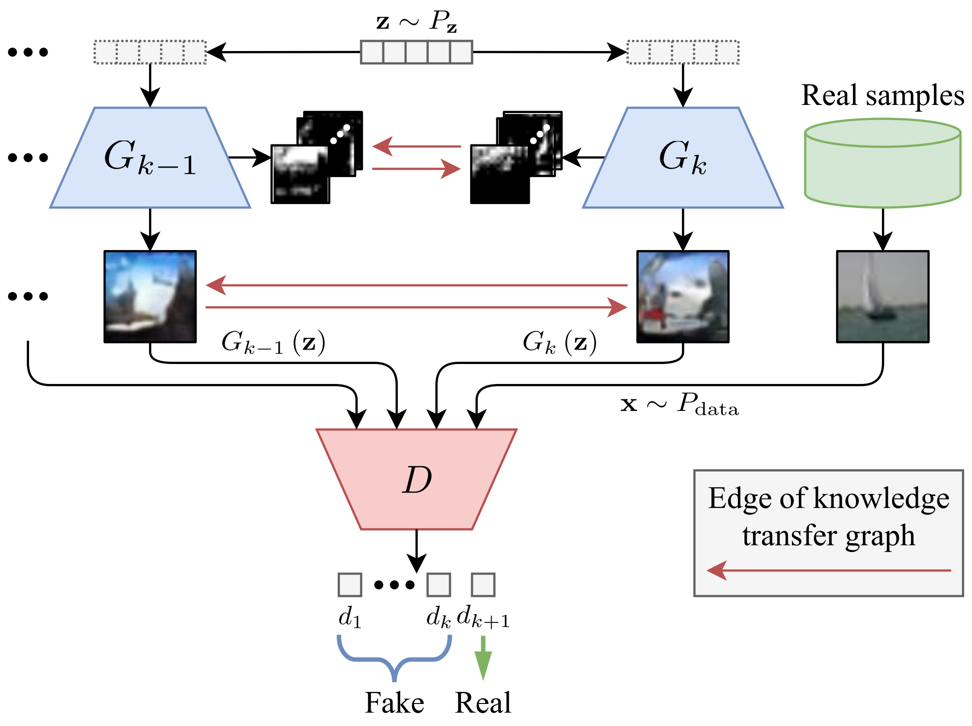

$\mathbf{z} \sim P_{\mathbf{z}}$
Real samples
G_{k-1}
G_k
$G_{k-1}(\mathbf{z})$
$G_k(\mathbf{z})$
$\mathbf{x} \sim P_{\text{data}}$
D
Edge of knowledge transfer graph
d_1
d_k
d_{k+1}
Fake
Real

Figure 3: Network structure of the proposed method

2.2 Multi-agent diverse GANs

As mentioned above, MAD-GANs are composed of multiple generators and one discriminator. Figure 2 shows the network structure of MAD-GANs. Each generator uses latent variables to generate an image, and the generated image is trained so that it can be identified as a real image by the discriminator. The discriminator not only learns to distinguish between the real and generated images but also learns from which generator the image was generated. In general GANs learning, the image generated from the generator is over-optimized to trick the discriminator, resulting in mode collapse. In MAD-GANs, if the image generated by one generator is over-optimized to trick the discriminator, another generator can compensate to prevent mode collapse.

3 PROPOSED METHOD

Our proposed method enables collaborative learning of GANs in which learning is conducted using multiple generators that learn while transferring each other's knowledge. The method of constructing the proposed method, method of introducing the knowledge transfer graph, and learning algorithm are explained below.

3.1 Construction of proposed method

The proposed method involves using GANs with multiple generators and a single discriminator. We also need to make changes to the base GANs. Figure 3 shows the network structure of the proposed method.

If the number of generators used for learning is k, each generator is represented as $G_1, \ldots, G_k$, as shown in Figure 3. Each generator uses the same latent variable $\mathbf{z} \sim P_{\mathbf{z}}$ to generate an image. Therefore, k generated images $G_1(\mathbf{z}), \ldots, G_k(\mathbf{z})$ can be obtained. In addition, by transferring the feature map or generated image of each generator using a knowledge transfer graph, knowledge can be transferred between generators.

As shown in Figure 3, the discriminator is represented as D and needs to not only distinguish between the real and generated images but also which generator the generated image is from. Therefore, the number of units of the output value of the final layer is $k+1$, and the class score $d_1, \ldots, d_{k+1}$ is output using the softmax function. Score $d_1, \ldots, d_k$ represents the probability of being an image generated from each generator, and score d_{k+1} represents the probability of being a real image.

Batch normalization (Ioffe and Szegedy, 2015) is often used inside a network to stabilize learning. However, batch normalization is insufficient to stabilize learning in GANs. Therefore, with the proposed

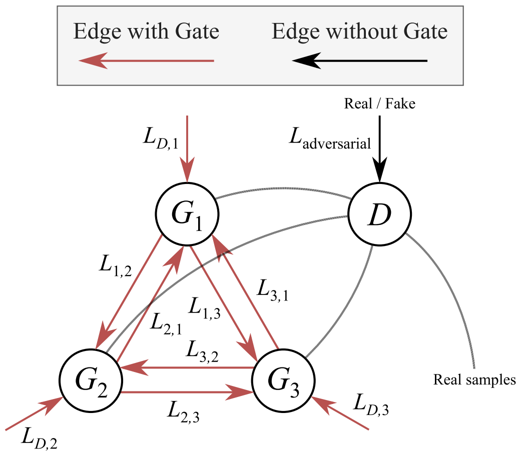


Figure 4: Introduction of knowledge transfer graph with proposed method ($k = 3$)

method, spectral normalization (SN) (Miyato et al., 2018) is introduced into the discriminator. If the base GANs use batch normalization for the discriminator, batch normalization changes to SN. By introducing SN into the discriminator, it is possible to set the constraint of satisfying Lipschitz continuity. Therefore, GANs mode collapse can be suppressed in the same manner as with methods such as those using Wasserstein GANs (WGANs) (Arjovsky et al., 2017) and WGANs-GP with a gradient penalty (Gulrajani et al., 2017). Also, SN can be introduced without changing the loss function.

3.2 Introduction of knowledge transfer graph

The proposed method introduces a knowledge transfer graph, which enables each generator to learn while transferring its knowledge to the others. Figure 4 shows the proposed method with a knowledge transfer graph introduced when three generators are used. Assuming that the number of generators used for learning is k, each generator is $G_1, \ldots, G_k$ and discriminator is D, which are represented as nodes. Two edges are defined between the nodes of each generator and represented by directed graphs with different orientations. The orientation of each edge represents the direction in which gradient information is transmitted during learning. The edges between the nodes of each generator represent the transfer of knowledge between the generators. Edge $L_{D,i}$ of generator G_i represents the transmission of information as to whether the generated image generated from G_i was determined by D to be a real image. Edge $L_{\text{adversarial}}$ represents the transmission of information as to whether the D identification result is correct. A separate loss function is defined for each edge, and a gate function is introduced for the loss function of the edge facing the direction of each generator to control the propagated loss.

The proposed method uses, through, cutoff, negative linear, and positive linear gates as gate functions. The through gate is a function that passes the loss as it is. The cutoff gate is a function that sets the loss to 0, does not calculate the loss, and can cut any edge. The negative linear gate is a function that reduces the loss as learning progresses, and the early stage of learning is the through gate and the final stage is the cutoff gate. The positive linear gate is a function that increases the loss as learning progresses, and the early stage of learning is the cutoff gate and the final stage is the through gate.

3.3 Learning algorithm

The proposed method is trained by repeating the training of the discriminator and generators. Training is repeated until all actual images are trained. When training is complete, 1 epoch of learning is completed. The training of the discriminator and generators are explained below.

3.3.1 Training of discriminator

The discriminator learns to identify whether the input image is a real image or an image generated from a generator. Assuming that the number of Generators used is k, the identification result when the actual image or image generated from the generator is input to the discriminator is represented as $\mathbf{d} = d_1, \ldots, d_{k+1}$. The correct label is represented as $\mathbf{t}^{\mathrm{d}} = t^{\mathrm{d}}_1, \ldots, t^{\mathrm{d}}_{k+1}$. The $\mathbf{t}^{\mathrm{d}}$ when the image generated from generator G_i is input to the discriminator is a vector with t^{d}_i being 1 and others being 0. The $\mathbf{t}^{\mathrm{d}}$ when the real image is input to the discriminator is a vector with t^{d}_{k+1} being 1 and others being 0. The loss function $L_{\text{adversarial}}$ used for training of the discriminator is expressed as

$$L_{\text{adversarial}} = -\sum_{m=1}^{k+1} t^{\mathrm{d}}_m \log_e d_m \tag{1}$$

The discriminator learns in three steps:

Step 1 The actual image is input to the discriminator, and the gradient of the discriminator is updated using the loss function $L_{\text{adversarial}}$.

Step 2 The image generated from generator G_i is input to the discriminator, and the gradient of the discriminator is updated using $L_{\text{adversarial}}$. At this time, the latent variables input to each generator are different for each generator.

Step 3 By repeating steps 1 and 2 k times, learning is conducted on the images generated by all generators.

3.3.2 Training of generators

Each generator learns to generate an image that is identified by the discriminator as a real image. With the proposed method, each generator learns while transferring knowledge to the others according to the knowledge transfer graph. The edges between the nodes of each generator represent the transfer of knowledge among the generators. Edge $L_{D,i}$ of generator G_i represents the transmission of information as to whether the image generated from G_i was determined by the discriminator to be a real image. Let $L_{t,s}$ be the loss function when transferring the knowledge of generator G_t to generator G_s at the edge between the nodes of the generator. Also, let $L_{D,i}$ be the loss function that indicates whether the image generated from G_i is judged to be a real image by the discriminator. Let $Gate(\cdot)$ be the gate function at each edge. In generator learning, the input latent variables are the same for all generators. The complete loss function for each generator is calculated from the sum of $L_{t,s}$ and $L_{D,i}$ taken from the edge.

Assuming that the number of generators used is k, the identification result when the image generated from G_i is input to the discriminator is represented as $\mathbf{d}^{(i)} = d_1^{(i)}, \ldots, d_{k+1}^{(i)}$. The correct label $\mathbf{t}^{\mathrm{g}} = t_1^{\mathrm{g}}, \ldots, t_{k+1}^{\mathrm{g}}$ is a vector with t_{k+1}^{g} being 1 and the others being 0. At this time, the loss function $L_{D,i}$ is expressed as

$$L_{D,i} = Gate\left(-\sum_{m=1}^{k+1} t_m^{\mathrm{g}} \log_e d_m^{(i)}\right) \quad (2)$$

There are two types of knowledge-transfer methods that the proposed method uses, one is for transferring the feature map (two versions: close to the input-layer side and close to the output-layer side) and the other is for transferring the generated image, and the loss function $L_{t,s}$ is different for each method.

If the channel number of the feature map is $c = 1, \ldots, C$, the position of the layer of the feature map is $m = 1, \ldots, M$, and the vector number of the feature map is $n = 1 \ldots, N$, the feature map of G_i is represented as $F_{i,m,c,n}$. If the hyperparameter is α, the $L_{t,s}$ of the method for transferring the feature map is expressed as

$$Q_i^{(m,n)} = \frac{1}{C}\sum_{c}^{C} F_{i,m,c,n}^2 \quad (3)$$

$$L_{t,s} = Gate\left(\frac{\alpha}{MN}\sum_m^M \sum_n^N \left(\frac{Q_s^{(m,n)}}{\left\|Q_s^{(m,n)}\right\|_2} - \frac{Q_t^{(m,n)}}{\left\|Q_t^{(m,n)}\right\|_2}\right)^2\right) \quad (4)$$

When the vector number of the generated image is $n = 1 \ldots, N$ and the latent variable $\mathbf{z} \sim P_{\mathbf{z}}$ is input to G_i, let $G_i^{(n)}(\mathbf{z})$ be the generated image. If the hyperparameter is α, the $L_{t,s}$ of the method for transferring the generated image is expressed as

$$L_{t,s} = Gate\left(\frac{\alpha}{N}\sum_n^N \left(\frac{G_s^{(n)}(\mathbf{z})}{\left\|G_s^{(n)}(\mathbf{z})\right\|_2} - \frac{G_t^{(n)}(\mathbf{z})}{\left\|G_t^{(n)}(\mathbf{z})\right\|_2}\right)^2\right) \quad (5)$$

4 EXPERIMENTS

We evaluated the overall effectiveness of the proposed method and its effectiveness in introducing a mechanism for learning while transferring knowledge. We also investigated the optimal knowledge-transfer method by introducing a knowledge transfer graph into the proposed method.

4.1 Experimental conditions

The CIFAR-10 dataset (Krizhevsky, 2009) was used for the experiment. This dataset has $50,000$ images for training and $10,000$ images for evaluation and consists of 10 classes of color images of common objects. Only $50,000$ images training samples were used in this experiment. The image size of the CIFAR-10 dataset is 32×32 pixels, but it was resized to 64×64 pixels for our use.

We used DCGANs as the base GANs and set the number of trials of the gate function of the knowledge transfer graph to $1,500$ to search for a combination of gate functions that maximizes the maximum inception score (IS) (Salimans et al., 2016) of each generator. During training, the batch size was 256, number of dimensions of the latent variable input to each generator was 100, and hyperparameter α was $1,000$.

The IS and fréchet inception distance (FID) (Heusel et al., 2017) were used as indexes to evaluate the quality of the generated image. IS is easy to classify using a neural network, and the more classes of generated images, the higher the quality of generated images. The higher the IS, the higher the quality of the generated image. FID mitigates the drawback that IS does not take into account the distribution of real image, and the quality of the generated image is evaluated by the distance between the distribution of the real image and generated image. The lower the FID, the higher the quality of the generated image.

4.2 Investigating effectiveness of knowledge transfer

We then investigated the effectiveness of knowledge transfer. In this experiment, using DCGANs and

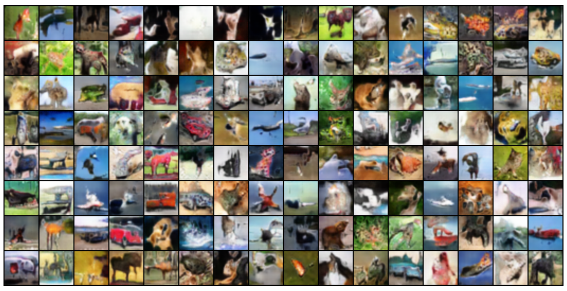
Table 1: Comparison of IS and FID by various methods

	SN	Transfer	IS ↑	FID ↓
DCGANs		–	4.57 ± 0.03	25.79
	✓	–	4.77 ± 0.05	**21.17**
MAD-GANs	✓		4.81 ± 0.05	27.02
Proposed	✓	✓	$\mathbf{5.10 \pm 0.07}$	21.31
Real images	–	–	8.70 ± 0.14	–

MAD-GANs and the proposed method were compared. Using DCGANs with and without SN was compared with using MAD-GANs and the proposed method. SN was introduced to the MAD-GANs, which used two generators, and the proposed method, which used. The proposed method used the method for transferring the generated image. The gate function was not searched, and only the through gate was used. The number of learning epochs was set to 200, and the evaluation was conducted using the model parameters at the highest point of IS. Table 1 lists the results of comparing IS and FID. The quality of the generated image with the proposed method was the highest using IS. In the evaluation using FID, the highest-quality image was generated using DCGANs with SN introduced, but the proposed method also generated an image of the same quality. The structure of the proposed method is the same as that of using MAD-GANs, but a significant improvement can be achieved by transferring the generated image and conducting learning. Therefore, the transfer of knowledge among generators can improve the quality of generated images in GANs.

We visualized the generated image when 128 identical latent variables $\mathbf{z} \sim P_{\mathbf{z}}$ were input to generators G_1 and G_2 of the proposed method. Figure 5 shows the results . The image corresponding to the same position was the generated image when the same latent variable was input. By comparing the generated images from G_1 and G_2, the images when the same latent variable was input were similar. This is probably because the generators have similar characteristics because they are trained to generate similar images by transferring the generated images among each other.

We used t-Distributed Stochastic Neighbor Embedding (t-SNE) (Maaten and Hinton, 2008) to visualize the distribution of images generated when entering 1,000 latent variables $\mathbf{z} \sim P_{\mathbf{z}}$ into G_1 and G_2. Figure 6 shows the results. Since the distributions of G_1 and G_2 of the proposed method overlap, the images when the same latent variable was input were similar. In addition, since the distribution of the proposed method is wider than that of using MAD-GANs, the diversity of the generated images can be increased by transferring the knowledge of the generators.


(a) Image generated by generator G_1

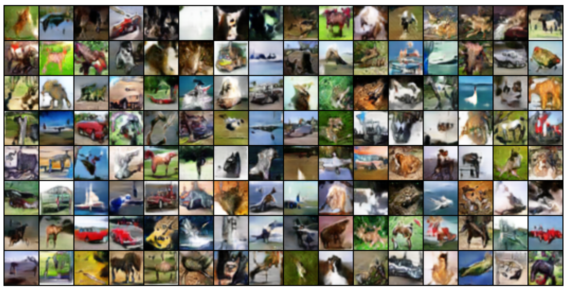

(b) Image generated by generator G_2

Figure 5: Visualization of generated images

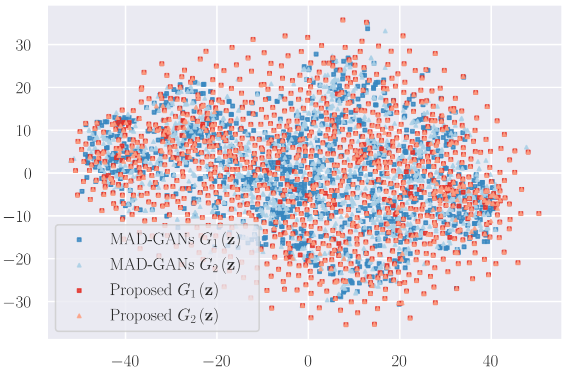


Figure 6: Visualization of the distribution of generated images by using t-SNE

4.3 Investigation of differences in transfer methods

Using the proposed method that introduces a knowledge transfer graph, we investigated when the knowledge-transfer method is changed. The proposed method used three generators for this experiment. We investigated the method for transferring the generated image, that for transferring the feature map close to the input-layer side (2nd and 3rd layers of DCGANs), and that for transferring the feature map close to the output-layer side (3rd and 4th layers of DCGANs). Through and cutoff gates were used as gate func-

Table 2: Comparison of IS and FID with different knowledge-transfer methods

Transfer method	**IS** ↑	**FID** ↓
Generated image	4.42 ± 0.04	46.06
Feature map (Input layer side)	4.46 ± 0.04	50.09
Feature map (Output layer side)	$\mathbf{4.48 \pm 0.05}$	**40.93**

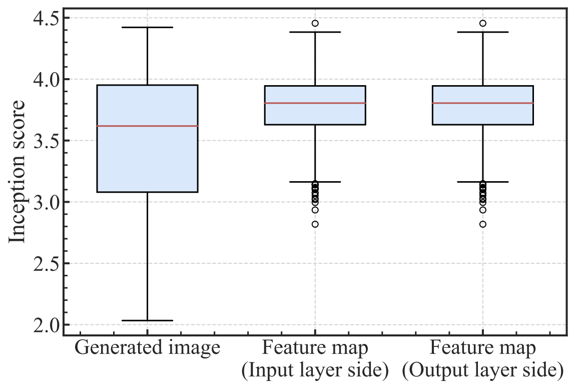


Figure 7: Tendency of inception score with different knowledge-transfer methods

tions, and the search was conducted using a knowledge transfer graph. The number of learning epochs was set to 50, and the evaluation was conducted using the model parameters at the end of learning. Table 2 lists the results of comparing IS and FID. The proposed method using the method for transferring the feature map close to the output layer side had the highest quality of the generated image for both IS and FID. Therefore, learning using this transfer method is effective in improving the quality of generated image.

The tendency of IS was investigated for each knowledge-transfer method of the proposed method when the search for the gate function using the knowledge transfer graph was attempted $1,500$ times. Figure 7 shows a boxplot of IS in each trial. The method for transferring the feature map had a higher IS value than that for transferring the generated image, and learning tended to be stable. The methods for transferring feature maps closer to the input layer side and closer to the output layer side tended to be similar.

4.4 Knowledge-transfer-graph optimization

We next investigated knowledge-transfer-graph optimization by comparing using DCGANs with and without SN) and MAD-GANs and the proposed method. SN was introduced to the MAD-GANs, which used three generators and the proposed

Table 3: Comparison of IS and FID by introducing knowledge transfer graph

	SN	**Transfer**	**IS** ↑	**FID** ↓
DCGANs		–	4.57 ± 0.03	25.79
	✓	–	4.77 ± 0.05	**21.17**
MAD-GANs	✓		4.77 ± 0.03	28.76
Proposed	✓	✓	$\mathbf{5.37 \pm 0.09}$	23.77
Real images	–	–	8.70 ± 0.14	–

method, which also used three generators as well as the method of transferring the feature map close to the output layer side (3rd and 4th layers of DCGANs). Through, cutoff, negative linear, and positive linear gates were used as gate functions, and the search was conducted using the knowledge transfer graph. The number of learning epochs was set to 200, and the evaluation was conducted using the model parameters at the end of learning. Table 3 shows the results of comparing IS and FID.

With the proposed method, the search for the gate function using the knowledge transfer graph was attempted $1,500$ times, and the knowledge transfer method was optimized. Figure 8 shows the combination of gate functions that maximizes the maximum IS of each generator, which is the target of optimization. The red edge represents the positive linear gate and black edge represents the through gate. In this structure, the IS of generator G_1 was 5.37 ± 0.09, that of generator G_2 was 4.70 ± 0.07, and that of generator G_3 was 1.33 ± 0.03. Since the IS of G_1 was higher than that of G_2, the transfer of knowledge was effective in improving the quality of the generated image. Also, it was not necessary to transfer knowledge among the generators at the beginning of learning, and it was effective to transfer knowledge as learning progresses. Generator G_3 contributed to the stabilization of training of G_1 by acting like a buffer of G_1.

5 CONCLUSION

We proposed a method of collaborative learning that involves GANs consisting of multiple generative models and one discriminator model for transferring knowledge among the generative models. Through evaluation experiments, the proposed method improved the quality of the generated images and increased diversity. In the future, we will use multiple generators and discriminators, introduce knowledge transfer graphs into each generator and discriminator,

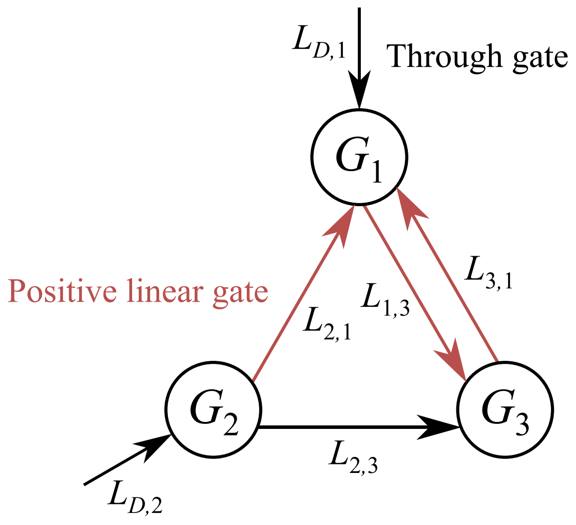


Figure 8: Obtained combination of gate functions

and search for the optimal knowledge transfer method to further improve the method's performance.

REFERENCES

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, volume 70, pages 214–223.

Durugkar, I., Gemp, I., and Mahadevan, S. (2017). Generative multi-adversarial networks. In *International Conference on Learning Representations*.

Ghosh, A., Kulharia, V., Namboodiri, V. P., Torr, P. H., and Dokania, P. K. (2018). Multi-agent diverse generative adversarial networks. In *Computer Vision and Pattern Recognition*, pages 8513–8521.

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Neural Information Processing Systems*, pages 2672–2680.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Neural Information Processing Systems*, pages 5767–5777.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Neural Information Processing Systems*, pages 6626–6637.

Hinton, G., Vinyals, O., and Dean, J. (2014). Distilling the knowledge in a neural network. *NIPS Deep Learning and Representation Learning Workshop*.

Hoang, Q., Nguyen, T. D., Le, T., and Phung, D. (2018). Mgan: Training generative adversarial nets with multiple generators. In *International Conference on Learning Representations*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, volume 37, pages 44–456.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2018). Progressive growing of gans for improved quality, stability, and variation. *International Conference on Learning Representations*.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical Report, University of Tront.

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

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605.

Minami, S., Hirakawa, T., Yamashita, T., and Fujiyoshi, H. (2019). Knowledge transfer graph for deep collaborative learning. *arXiv preprint arXiv:1909.04286*.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*.

Nguyen, T., Le, T., Vu, H., and Phung, D. (2017). Dual discriminator generative adversarial nets. In *Neural Information Processing Systems*, pages 2670–2680.

Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations*.

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. (2015). Fitnets: Hints for thin deep nets. *International Conference on Learning Representations*.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Neural Information Processing Systems*, pages 2234–2242.

Song, G. and Chai, W. (2018). Collaborative learning for deep neural networks. In *Neural Information Processing Systems*, pages 1832–1841.

Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. (2019). Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pages 7354–7363.

Zhang, Y., Xiang, T., Hospedales, T. M., and Lu, H. (2018). Deep mutual learning. In *Computer Vision and Pattern Recognition*, pages 4320–4328.

Zhu, X., Gong, S., et al. (2018). Knowledge distillation by on-the-fly native ensemble. In *Neural Information Processing Systems*, pages 7517–7527.