Master Thesis

Study on Performance Improvement Techniques and Applications of Deep Learning

June 2020

Hansung University

Major in Electronic Information Engineering

Dept. of Electronic Information Engineering

Choi, Seoung-Ho

Master Thesis
Advisor Professor SungHoon Jung

Study on Performance Improvement Techniques and Applications of Deep Learning

– 딥 러닝의 성능 향상 기법 및 응용에 관한 연구 –

June 2020

The Graduate School of Hansung University

Major in Electronic Information Engineering

Dept. of Electronic Information Engineering

Choi, Seoung-Ho

Master Thesis
Advisor Professor SungHoon Jung

Study on Performance Improvement Techniques and Applications of Deep Learning

– 딥 러닝의 성능 향상 기법 및 응용에 관한 연구 –

Submit the above thesis as a master's thesis

June 2020

The Graduate School of Hansung University

Major in Electronic Information Engineering

Dept. of Electronic Information Engineering

Choi, Seoung-Ho

Approved Choi, Seoung-Ho Master Thesis in Electronic Inforamtion Engineering

June , 2020

Judge Chair _____ (Sign)

Judge _____ (Sign)

Judge _____ (Sign)

Abstract

Study on Performance Improvement Techniques and Applications of Deep Learning

Seoung-Ho Choi
Major in Electronic Information Engineering
Dept. of Electronic Information Engineering
The Graduate School
Hansung University

Deep learning currently shows high performance in many real-life applications and has been applied to various environments, and research has been conducted. However, since deep learning is a black-box model, it is difficult to interpret it, and it is difficult to understand why it is getting better. Therefore, we will look at the research contents on the fields of application of deep learning performance improvement technology and deep learning to improve the performance of existing deep learning. To improve the performance of deep learning, we proposed the contents of the improvement of deep learning technology, which is improved by grasping where the problem is.

Improved performance of deep learning technology, we look at seven types of technology. First, the bi-activation function: this is an

enhancement activation function that is enhanced in the convolution neural network. Second, it is the loss of a neural network that is continuously occurring, and the continuous correction cascade loss occurs through continuous online learning. Third, non-linear regularization: This is an improved regularization version for the regularization method. Fourth, it is a new auxiliary component for optimizing the deep learning model. The fifth is ensemble normalization for stable learning. The sixth is the similarity analysis of actual fake fingerprints and fake fingerprints generated by DCGAN. Seventh is a multi-path decoder scheme with error reduction embedding in one-hot bi-directional Seq2Seq with adaptive regularization for music composition.

In addition, deep learning that expresses high performance will be introduced to technology applied to real life. In technology using deep learning, we will look at four types of technology. The first is the importance of adaptive seeding. The second is the module comparison study in the image captioning. Third, the visualization of outlier data. Fourth, it is a stable and fine-grained segmentation that uses batch normalization and focal loss and L1 regularization in the U-Net structure.

Through this, we will create a new deep learning theory using deep learning to improve the performance of the deep learning model and proceed to future research in the new research field.

【Keyword】 deep learning. performance, improvement, application, technology

Contents

Index of Table

Index of Figure

Chapter 1 Introduction

1.1 Research objectives

Deep Learning currently shows high performance in many real-life applications and has been applied to various environments and research has been conducted. However, since deep learning is a black-box model, it is difficult to interpret it, and it is difficult to understand why it is getting better. So, we show high performance as an advantage of the deep learning model.

There are three ways to improve the existing deep learning performance in terms of data. First, more data is obtained. This is due to the increase in the amount of data information judged by the deep learning model, which improves performance. At this time, if the quality of the acquired data is poor, rather, it shows low performance. Therefore, it is important to augment good quality data. Second, adjust the data scale. The process of adjusting the data scale has three-step. The first step normalized to 0 to 1. The second step rescaled to -1 to 1. Third step standardized.

In terms of deep learning model improvement, technologies such as hyper parameter optimization, batch size and adjustment, early stop, regularization, dropout, and network search methods are known as ways to improve the performance of deep learning models. However, even though many studies have been conducted, various issues still exist that require new problems and solve existing problems. We would like to suggest several ways to improve performance in the deep learning model. We explain in the next section.

1.2 Research contents

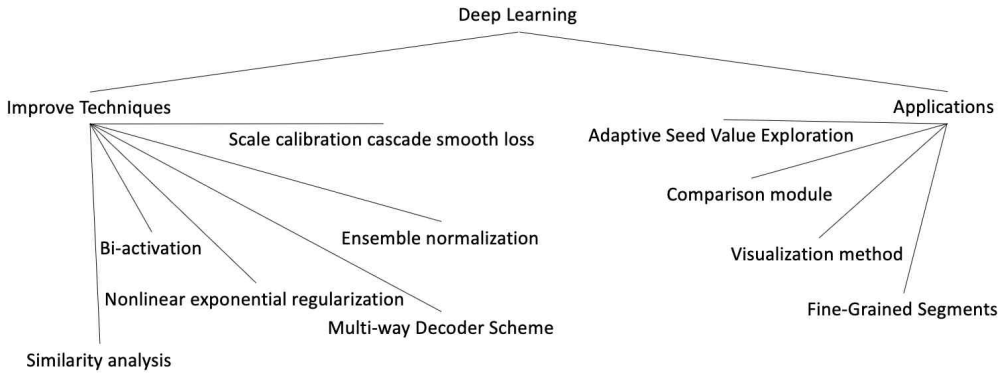


Figure 1. Taxonomy of contents of my thesis

We intend to reduce the problems and obtain an opinion to improve the performance of the deep learning model. Figure 1 is a taxonomy the proposed methods. We obtain high performance using deep learning. We describe the contents of the research conducted by the technique of improving the performance of the existing deep learning model. The following are research on the performance improvement techniques from the previous research.

3.1 The bi-activation function: an enhanced version of an activation function in convolution neural network, 3.2 Scale calibration cascade smooth loss of generative adversarial networks with online continual task learning, 3.3 Nonlinear exponential regularization: an improved version of regularization for deep learning model, 3.4 Novel auxiliary components to help optimize deep learning model, 3.5 Ensemble normalization for stable training, 3.6 Multi way decoder scheme with error reduction embedding on one-hot bi-directional seq2seq with adaptive regularization for music composition, and 3.7 Similarity analysis of actual fake fingerprints and generated fake fingerprints by DCGAN.

To improve the performance of the existing deep learning model, we will

describe application research using deep learning in addition to the research conducted. The following is an application study using existing deep learning. 4.1 Study on the importance of adaptive seed value exploration, 4.2 Stable acquisition of fine-grained segments using batch normalization and focal loss with l1 regularization in U-Net structure 4.3 Visualization techniques for outlier data, 4.4 Component-based comparative analysis of each module in image captioning

In conclusion, the contributions of this thesis will explain the limitations and additional directions for research and existing methods related to methods for improving the performance of the deep learning model and application research.

Chapter 2 Basics of Deep Learning Techniques and Application

Recently, deep learning has been making achievements that have been planned in several fields. It has been successfully applied, such as showing better results than a person. Therefore, to understand the contents described, the background knowledge of deep learning will be described. Deep Learning is composed of three categories. The first is supervised learning, the second is unsupervised learning, and the third is reinforced learning. Supervised learning is a method of learning a model with a correct answer and a label for a correct answer. Unsupervised learning finds input characteristics only in the input state. Reinforcement learning is a method of updating through compensation according to the current state. Deep learning consists of the convolution model, sequential model, generative adversarial model, reinforcement learning, and transfer learning. The advantage of deep learning is high performance. The disadvantage of deep learning is that it has a large amount of calculation in the course of learning. In addition, deep learning requires a large amount of data. Because if you enter the wrong quality data in the deep learning model, the wrong garbage result will occur. The existing research will be described based on the type of deep learning.

2.1 Techniques

2.1.1 Convolution Neural Networks (CNN)

CNN uses a local filter to calculate by sharing the filter in the image. CNN is

strong in regional characteristics because it uses local filters to perform calculations. The following are the contents studied in the convolution method to improve music composition performance. H.-M. Sulfur *et al*. [1] used the characteristics of the Mel-frequency spectrum coefficient, and the discrete Fourier transform coefficient, and the raw PCM (Pulse Code Modulation) sample as the CNN input. J. Lee and J. Nam [2] improved multi-level and multi-scale functions from CNN and improved CNN performance through aggregation to improve music composition performance.

2.1.2 Recurrent Neural Network (RNN)

RNN recursively reflects information over time. RNN is a method of receiving data in a single direction and processing data for each state, and it is important to include time information in the input and to maintain each state well [3-6]. The improvement of the performance research through the combination of the existing RNN and various models is as follows. A. Huang and R. Wu [4] tried to improve the music composition performance by using the combination of RBM and RNN to improve memory. R. Vohra *et al.* [5] proposed to successfully apply harmonic music generation by combining DBN (Deep Belief Network) and LSTM (Long Short Term Memory) to improve memory. Q. Lyu *et al.*, [6] proposed a method of combining RRTBM to improve the memory of LSTM. The combination of RBM, DBN, and RTRBM in a continuous model was intended to improve the performance of music composition by improving the memory ability of the existing RNN.

2.1.3 Sequence to Sequence

Seq2Seq is composed of an encoder-decoder model, and has the feature of reflecting information about the previous output in the decoder model section. The advantage of Seq2Seq is that the encoder-decoder learns at the same time so that Seq2Seq is more useful than RNN in evaluating the length problem of the generated result and generating the continuously generated result. As a disadvantage, only the information of a short reflection is considered, so causality based on sequential input is not considered. The research using the existing Seq2Seq is as follows. D. Spital [7] proposed a new data expression system to transfer the characteristics of one composer to another composer and confirmed that it is possible to transmit music compositions using the Seq2Seq at the composer level.

2.1.4 Generative adversarial network (GAN)

GAN improves production performance through alternative learning. Research using GAN consists of two types. The first subsection is the static creation of music, and the next subsection is the creation of a music sequence.

The research on the static music creation network is as follows. A. Knower *et al*. [9] proposed a GAN-based model for removing the staff line. The removal of the staff line referred to here is an important preprocessing step in optical music recognition. It is a difficult task to reduce noise in the existing music score image and maintain the quality of the music symbol context at the same time. In order to remove the staff line, GAN was used to improve music composition performance.

The research on how to create a sequence type network is as follows. In

SeqGAN [10], there is a limit in the case of creating a discrete sequence in the existing GAN. Because the generation model was not easy to update information from the discriminator to the creator, the information was not updated. Therefore, the concept of compensation was introduced by Reinforcement Learning (RL) to the creator and applied to the continuous generation process. Through this, the balance between the current score and the future score was obtained, and the result that the entire sequence was generated was obtained. S.-g. Lee *et al*. Proposed the application of SeqGAN to create a polyphonic music sequence [11].

2.1.5 Conditional GAN

H.-M. Liu and Y.-H. Yang [12] is a method of improving the existing musical notation to improve the performance of music production, and to improve the existing musical expression, by entering the instrument information into the generator to extract information from the first stage of extraction. During the creation process, it was confirmed that it was sufficiently reflected and generated. In addition, we improved the performance of multi-instrument music production by inputting it as a condition in the process of generating music as an input condition. R. Manzelli [13] shows that the GAN model learning was generated through more reliably reflecting information through the input of additional conditions to address the nuances of primitive audio generation and the part that does not understand the richness of expression. After all, conditional learning can confirm that it reflects various information more stably.

2.1.6 Auto encoder

The study of music composition using Auto Encoder is as follows. G. Brunner *et al*. [16] calculates the information of μ and σ to create a new potential space by reflecting noise and style classification information. This improved the performance of music composition by improving the area between different data spaces in the potential variable space. A. Tikhonov and I. Yamshchikov [17] calculated the average σ during the process of moving from the encoder to the decoder and used the variable Bayesian noise along with the new Bayesian noise as input to the decoder. Through Bayesian noise, we improved the music composition performance through more robust learning. Finally, Y.-A. Wang *et al*. [18] designed the VAE(Variational Auto Encoder) model using a modular model to model music compositions as a domain in which the modular encoder encodes potential information and provides greater flexibility in data expression. Through this, data performance was efficiently expressed to improve music composition performance.

2.1.7 Transfer learning

Transfer learning is advantageous in that it efficiently reflects common information of information. However, there is a tendency to depend on performance according to common information. Therefore, there is a point where important semantic information must be handed over.

2.1.8 Knowledge distillation

Knowledge Distillation is a way for the student model to learn through the information of the existing teacher model. This method enables efficient learning

in knowledge transfer and can be effectively applied in various areas. Y.-N. Hung [19] created more realistic and different music audio by receiving different information through each CNN. This is an improvement in the performance of music composition through the reflection of various music information.

2.1.9 Cycle Loss

Let's look at two methods of loss used in music composition and deep learning. The loss function consists of a cycle loss and a triple loss method. G. Brunner *et al*. [20] used the strong characteristics of domain transmission by using the cycle loss. G. Brunner *et al.* [20] applied cycle loss on GAN, which uses cycle loss for symbolic music domain transfer, in which research conducted using VAE (Variational Autoencoder) and GAN for image style and domain transmission. R. Lu *et al*. In [21], entering triple metrics is an important issue in applications that retrieve a lot of music information. To solve this, we learned the information extracted from metrics similar to the triple input. Triple input shows better performance than single input or two inputs. The triple input shows a more well-preserved result in a relatively similar part. In addition, performance is improved by four or more generalization experiments, but if the number of inputs increases, the cost increases.

2.1.10 Reinforcement learning

Reinforcement learning (RL) is a process in which an agent receives compensation from the environment and updates the new state whenever the environment has a new state. RL applied the policy gradient series algorithm of

PPO (Proximal Policy Optimization) to the continuous optimization of information, and before showing better results in music production, L. Yu *et al*. [10] proposed creating a data generator. In addition, the RL's updated update process rewarded GAN discriminators as a single reward. In addition, N. Kotecha [22] remembers the details of the past and solves the problem of having clear and consistent functions, and inputs the expected values and the existing music data from the Bi-axial LSTM that creates the music rules to enter the DQN. The following section describes how to apply deep learning music for music composition.

2.1.11 Hybrid model

The hybrid model has been extensively studied in the direction of complementing the advantages and disadvantages of the existing model. H.-M. Liu and Y.-H. Yang [14] uses the structure of the hybrid auto-encoder GAN. The existing automatic encoder cannot generate various types, and the GAN model generates various types of images, but the generated images do not reflect the existing input image well. However, if you use Hybrid Autoencoder GAN, various images are created using the characteristics that reflect the input image well. There are many studies to supplement existing shortcomings.

2.1.12 Style transfer

In style transition, various music-related studies on style transition related to music deep learning are being conducted because existing music information must be properly reflected and reflected in other styles. There is also a problem of

moving the characteristics of one composer to the song of another composer. In order to solve these problems, the following research has been conducted in the transmission of music styles. D. Coster and B. Mathieu [27] have traditionally used a limited number of codes for single or polyphonic music. However, additional research is needed on the creation of captain music through a style transition. The following describes the latest deep learning technology and application method.

2.1.13 Activation function

It is important for the activation of the existing deep learning model to pass meaningful information in the course of the model training. It is also important not to be destroyed in the process of learning it. In the existing activation function research, the RELU function [31] uses the RELU function to alleviate the problem that the weight is lost when the model is learned.

2.1.14 Loss function

The loss function, which is a rule for learning a deep learning model, is important to efficiently design and learn the loss function to learn about the information that the model does not judge correctly. In the existing loss function study, the cross-entropy function is mainly used to minimize the difference between the data distribution we have and the Gaussian distribution of the model. The cross-entropy function is mainly used.

2.1.15 Regularization

Deep learning is a supplementary regularization technique that helps the deep learning model not to fall into overfitting. In the existing regularization study, L1 regularization [32] reflects the absolute value of the size of the difference between the prediction of the model and the actual correct answer. L2 regularization [32] is reflected as the sum of squares of the differences between the prediction of the model and the actual correct answer.

2.1.16 Normalization

Normalization, which helps the deep learning model to learn stably, is important to learn through the stabilization of signals so that large values do not occur in the course of the deep learning model training. In the existing normalization study, the batch normalization [33] is normalized by using the variance and the mean to input the input to each layer of the model, and the model learns stably.

2.1.17 Measure

Since the deep learning model is used to calculate the difference between the predicted value and the actual correct answer, it is important to learn using any measure because the deep learning model is used to update the error in the process of learning deep learning. In the case of PSNR in the measurement study, it has been used to discriminate the generated image at the maximum signal-to-noise ratio. This was studied by SSIM measure to evaluate the quality

in three aspects: Luminance, contrast, and structural.

2.2 Application

2.2.1 Visualization

Deep learning shows good performance when good data is entered as input and low performance when bad quality data is entered. Therefore, it is necessary to visualize and judge the training data in advance. In the existing visualization study, the t-SNE [34] method is studied as a dimensional reduction and visualization methodology, and t-SNE is mainly used to visualize data.

2.2.2 Image captioning

Image captioning [35] is the process of viewing an image and automatically attaching the appropriate description. In the existing image captioning study, CNN was used to extract the characteristics of an image, and the extracted image feature was used as the initial hidden cell input of RNN to reflect the text information as an input, and after calculation in the cell, the description in the model appears as output.

Chapter 3 Performance Improvement Technique

Existing deep learning is composed of three categories. The first is supervised learning, the second is unsupervised learning, and the third is reinforcement learning. Supervised learning is a method of learning a model with a correct answer and a label for a correct answer. Unsupervised learning finds input characteristics only in the input state. Reinforcement learning is a method of updating through compensation according to the current state. Deep learning consists of convolution neural networks, sequential networks, generative adversarial networks, reinforcement learning, and transfer learning.

First, let's take a look at the researched method to improve performance in the method used as a basic component of deep learning.

3.1 The Bi-activation Function: an Enhanced Version of an Activation Function in Convolution Neural Networks

This research describes the bi-activation function that has been researched during the existing acquisition method. S.-H. Choi and K. Kim [6] description of Bi-activation is as follows.

Introduction: RELU, a function frequently used as an activation function, can be simply expressed : f(x)=max(0,x), when x is the input of a neuron. That is, the output is zero when x < 0 (called a blocking area) and the output is the same as the input when x >= 0 (called a linear output area). Those properties indicate that the only positive part of the existing activation function is reflected and it takes more time to train the model. In addition, it is a disadvantage that

it does not reflect generalized characteristics in the model because it does not reflect negative partial information.

To improve this problem, we propose a bi-activation function as an improved version of a activation function. To verify the performance of the bi-activation function, We extensively experimented on CNN with typical datasets such as MNIST, Fashion MNIST, CIFAR-10, and CIFAR-100 and compared the results with those of RELU and eLU. As a result, the bi-activation function shows better performance than RELU and eLU in all experiments.

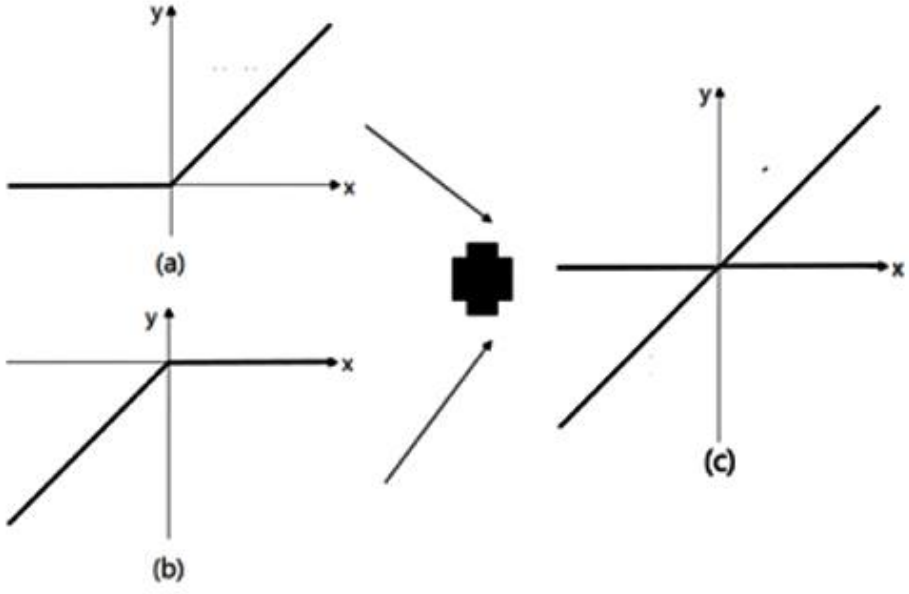


Figure 2 Process of bi-activation function: (a) Pos-activation function, (b) Neg-a ctivation function, (c) Bi-activation function.

Proposal Method: Bi-activation function consists of two types: pos-activation function and neg-activation function. In terms of RELU, the pos-activation function is the same as the existing RELU. That is, there is a blocking area when $x < 0$ and a linear output area when $x >= 0$. Neg-activation function has a blocking area when $x >= 0$ and a linear output area when $x < 0$. Simply

neg-activation function is expressed: fx=max(x,0). Figure 2(c) show bi-activation function, which consists of pos-activation function and neg-activation function. Pos-activation function has a blocking area, when x < 0 and a linear output area, when x >= 0 and neg-activation function has vice versa. CNN reflect the nonlinearity of inputs and outputs of training data using the activation function by properly selecting the linear output area.

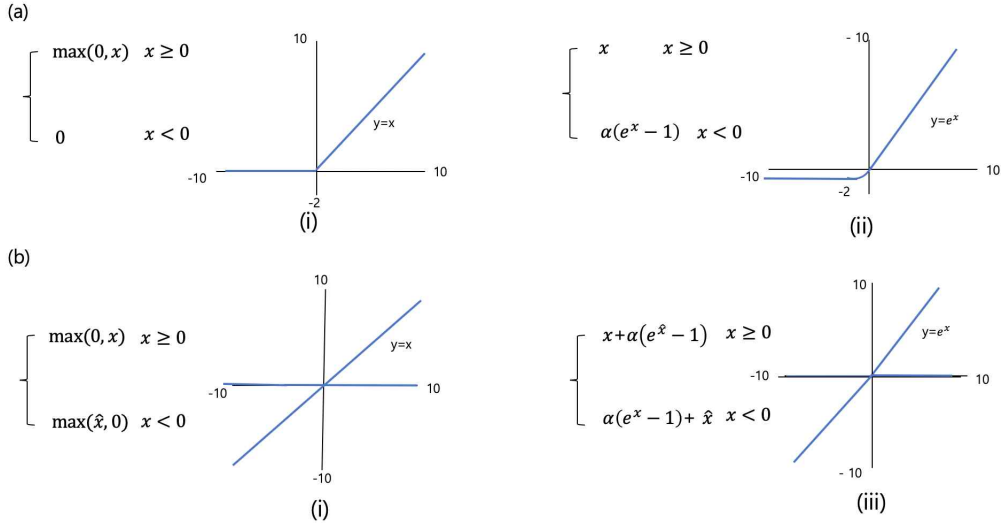


Figure 3 Experiment of activation function: (a) Existing method, (b) Proposal method, (a)i RELU, (a)ii eLU, (b)i bi-RELU, (b)ii bi-eLU.

Experimental Results: We tested the novel bi-activation function. The proposed method applied to existing functions. Figure 3 show the result of the experiments. Firstly, Figure 3(a) show RELU, and eLU and Figure 3(c)-(d) shows the result of applying the bi-activation function method to RELU and eLU, respectively. The CNN model used is intended to see the effect of a large-margin model prediction effect using Log SoftMax. When applying this log SoftMax, NLL Loss is generally applied. However, there is a problem that a non-convex effect causes the NLL loss. Therefore, cross-entropy is applied to generate the convex effect of the model. This is because the convex function

creates a unique solution and can be easily solved using the gradient method. Therefore, studies have been conducted to apply convex after applying cross-entropy to log SoftMax. The CNN model is defined as a validation model to verify that the activation function experiment group presented. Figure 3 is efficient even when there are few parameters. Four models are composed of Figure 4(a) CNN-Large, Figure 4(b) CNN-Middle, Figure 4(c) CNN-Small, and Figure 4(d) CNN-Little. The number of CNN-Large filter maps based. The model was designed based on the number of filter maps with CNN-Middle 0.75 times, the number of filter maps with CNN-Small 0.5 times, and the number of filters with CNN-Little 0.25 times. The loss used for the CNN model uses cross-entropy, and the optimization method is experimentally verified using the Adam optimization. We experimented with training datasets such as MNIST, Fashion MNIST, CIFAR-10, and CIFAR-100. When the proposed method is better than the activation function, it is indicated in bold.

(A) CNN-Large (Larger parameter number X 1.0)

	Input_features	Ouput_features	Kernel Size	Padding	Stride
Conv2d	Input_channel	**128**	3by3	1by1	1by1
Activation function.					
Conv2d	**128**	**128**	3by3	1by1	1by1
Activation function					
Conv2d	**128**	**128**	4by4	1by1	2by2
Activation function					
Conv2d	**128**	**256**	3by3	1by1	1by1
Activation function					
Conv2d	**256**	**256**	3by3	1by1	1by1
Activation function					
Conv2d	**256**	**256**	4by4	1by1	2by2
Activation function					

FC Layer (in_features=**12544**, out_features=512)
FC Layer (in_features=512, out_features=10)

(B) CNN-Middle (Large parameter number X 0.75)

	Input_features	Ouput_features	Kernel Size	Padding	Stride
Conv2d	Input_channel	**96**	3by3	1by1	1by1
Activation function.					
Conv2d	**96**	**96**	3by3	1by1	1by1
Activation function					
Conv2d	**96**	**96**	4by4	1by1	2by2
Activation function					
Conv2d	**96**	**192**	3by3	1by1	1by1
Activation function					
Conv2d	**192**	**192**	3by3	1by1	1by1
Activation function					
Conv2d	**192**	**192**	4by4	1by1	2by2
Activation function					

FC Layer (in_features=**9408**, out_features=512)
FC Layer (in_features=512, out_features=10)

(C) CNN-Small (Large parameter number X 0.5)

	Input_features	Ouput_features	Kernel Size	Padding	Stride
Conv2d	Input_channel	**64**	3by3	1by1	1by1
Activation function.					
Conv2d	**64**	**64**	3by3	1by1	1by1
Activation function					
Conv2d	**64**	**64**	4by4	1by1	2by2
Activation function					
Conv2d	**64**	**128**	3by3	1by1	1by1
Activation function					
Conv2d	**128**	**128**	3by3	1by1	1by1
Activation function					
Conv2d	**128**	**128**	4by4	1by1	2by2
Activation function					

FC Layer (in_features=**6272**, out_features=512)
FC Layer (in_features=512, out_features=10)

(D) CNN-Little ((Large parameter number X 0.25)

	Input_features	Ouput_features	Kernel Size	Padding	Stride
Conv2d	Input_channel	**32**	3by3	1by1	1by1
Activation function.					
Conv2d	**32**	**32**	3by3	1by1	1by1
Activation function					
Conv2d	**32**	**32**	4by4	1by1	2by2
Activation function					
Conv2d	**32**	**64**	3by3	1by1	1by1
Activation function					
Conv2d	**64**	**64**	3by3	1by1	1by1
Activation function					
Conv2d	**64**	**64**	4by4	1by1	2by2
Activation function					

FC Layer (in_features=**3136**, out_features=512)
FC Layer (in_features=512, out_features=10)

Figure 4 Experiment of model sample: (a) CNN-Large, (b) CNN-Middle, (c) C NN-Small, (d) CNN-Little.

We compare the experiments using the MNIST dataset with Seed 999, 500, and 1 to analyze the influence of each filter number. The experimental results are to verify results obtained from Table 1. That is different depending on the number of filters. First of all, linear activation showed a decrease in test value when there were a large number of filters. However, eLU with nonlinear features exhibited a performance increase and decreased as the number of filters decreased, which confirms that the appropriate number of filters should found when inferring through the Activation function in the model. Also, there is a problem of finding an appropriate number of filters, even when the bi-activation function is applied. When the bi-RELU is applied, the accuracy increases linearly and decreases as the number of filters decreases. The bi-eLU exhibits a nonlinear phenomenon in which the accuracy increases as the number of features decreases, then decreases and then increases. The performance varies depending on the number of filters and the nonlinearity of the activation function.

The proposal method receives both positive and negative information from the activation function and outputs less error value and improved performance than activation that seems to improve performance by making the feature a little clearer by processing both positive and negative information at the same time. The comparison of the same number of filters showed that most of them improved over a activation function.

Table 1. Comparison of influence according to the number of CNN model featur e maps (a) CNN-Large, (b) CNN-Middle, (c) CNN-Small, (d) CNN-Little.

	RELU		eLU		bi-RELU		bi-eLU	
(a)	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
Train	2.303	0.110	2.822	0.139	**1.961**	**0.353**	**2.503**	**0.260**
Test	X	0.062	X	0.125	X	**0.312**	X	**0.260**
(b)	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
Train	2.302	0.110	2.828	0.137	**1.912**	**0.361**	2.368	**0.291**

Test	X	0.125	X	0.031	X	**0.166**	X	**0.078**
(c)	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
Train	2.303	0.110	2.826	0.138	1.897	**0.371**	2.548	**0.235**
Test	X	0.125	X	0.0625	X	**0.5**	X	**0.1875**
(d)	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
Train	2.302	0.110	2.828	0.137	**1.912**	**0.361**	2.368	**0.291**
Test	X	0.125	X	0.031	X	**0.166**	X	**0.078**

We test the proposal method with the CNN-Small model and 2 layer CNN t o verify the effect of the proposal method in Figure 5.

	Input_features	Ouput_features	Kernel Size	Padding	Stride
Conv2d	Input_channel	**32**	3by3	1by1	1by1
Activation function.					
Conv2d	**32**	**64**	4by4	1by1	2by2
Activation function					
FC Layer (in_features=**16384**, out_features=512)					
FC Layer (in_features=512, out_features=10)					

Figure 5 Experiment of two layer CNN

To analyze the initial random influences of the proposed method, we conducted experiments about three seed values, Seed 999, Seed 500, and Seed 1. We perform a quantitative analysis of the experimental results using the average of three seed test results. In Table 2, the proposed method obtained an increase of 0.305 on average in the case of Train in bi-RELU in the MNIST dataset. The accuracy was reduced by 2.3%. The test resulted in a 4.1% improvement. In bi-eLU, the average loss was reduced by 0.307 during the train. Accuracy increased by 10.4%. W improved by 10.4%. The average of both methods loss results reduces by 0.001, a 4.05% improvement in the train, and a 7.29% accuracy increase in the test. In Fashion-MNIST, On average loss increased 0.011, accuracy increased 5.766% on the test, and the same on test in bi-RELU. On average, bi-eLU showed a loss reduction of 0.463 in loss, Improve accuracy 14.2% in the train, and 14.58% in the test. The average of two methods reduce loss 0.226 in the train, improves the accuracy by 10.0% in the train, and

improves the accuracy by 7.29% in the test. In CIFAR-10, in the case of bi-RELU, Loss shows a 0.0345 increase in a train, 6.96% accuracy improvement in a train and a 13.9% inaccuracy reduction in the test. In bi-eLU, loss decreases by 0.036, 1.46% on average accuracy improve, and the same in the test. The average of the two methods is 0.023 increase in loss, 4.2% accuracy improvement in a train, and a 6.9% accuracy reduction in the test. In the case of CIFAR-100, bi-RELU shows an average loss 0.486 increase in accuracy, 3.23% accuracy in trains, and the same result in the test. In bi-eLU, Loss decreases by 0.172 on average, improves accuracy by 0.67%, and is the same when testing. The average of both methods is 0.156 increase in loss, 1.95% increase of accuracy, and the same in the test. Finally, the average result of improvement and reduction of the four data sets shows that the bi-activation function method shows 0.018 reductions in train, 5.06% accuracy improves, and 1.8975% accuracy improves in test. In the case of MNIST and Fashion MNIST, the proposed method shows the improved results in Train and Test, which shows that the performance is improved by learning more effectively the positive information and negative information model and obtaining a more precise decision boundary. However, the complexity of data from CIFAR-10 increased as the number of image sizes, and image channels increased compared to the MNIST series. Nevertheless, the proposed method considers both positive information and negative information at the same time so that the average accuracy increased by classifying through clear boundaries in the train, but the generalized boundary not found in the test accuracy due to the poor performance. We can see that the decision boundary that led to the data found. This cause is seen to occur as the data size and data channel increase. In the case of data of CIFAR-100, the complexity increases as the number of classes in this model

increases, so it inferred that the data not adequately learned. That shows that the loss value is significantly higher than the data set result. Also, We can see that the proposed method decreases when the performance increases when tested with various seed values. This method is more affected by the influence of the initial value because the proposed method reflects bi-directional information confirm. Finally, the proposed method influenced by data and seed value, but it confirmed that reflecting positive and negative information helps to improve model learning and performance by maximizing the margin between model information.

Table 2. Train / Test accuracy and loss in CNN-Small according to Seed 999 on CNN small.

(a)	**MNIST**		**Fashion MNIST**		**Cifar10**		**Cifar100**	
Train	**Loss**	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**
RELU	2.303	0.110	2.304	0.101	2.304	0.103	4.611	0.01
eLU	2.826	0.138	2.820	0.139	2.841	0.136	6.918	0.035
bi-RELU	**1.897**	**0.371**	**1.737**	**0.397**	2.431	**0.161**	5.136	**0.041**
bi-eLU	**2.548**	**0.235**	**2.445**	**0.256**	**2.766**	**0.150**	**6.611**	**0.041**
Test	**Acc**		**Acc**		**Acc**		**Acc**	
RELU	0.125		0.1875		0.1875		0	
eLU	0.0625		0.125		0		0	
bi-RELU	**0.5**		**0.3125**		0.041		0	
bi-eLU	**0.1875**		**0.3125**		0		0	

Table 3. Train / Test accuracy and loss error in CNN-Small according to Seed 5 00 on CNN small.

(b)	**MNIST**		**Fashion MNIST**		**Cifar10**		**Cifar100**	
Train	**Loss**	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**
RELU[1]	2.303	0.111	2.304	0.102	2.304	0.102	4.610	0.009
eLU[2]	2.834	0.139	2.839	0.139	2.84	0.137	6.902	0.035
bi-RELU	**1.836**	**0.395**	**1.621**	**0.437**	2.407	**0.164**	5.071	**0.044**
bi-eLU	**2.524**	**0.241**	**2.296**	**0.301**	**2.807**	**0.154**	**6.705**	**0.04**
Test	**Acc**		**Acc**		**Acc**		**Acc**	
RELU	0.125		0.125		0.1875		0	
eLU	0.0625		0.0625		0.0625		0	
bi-RELU	**0.3125**		**0.4375**		0.0625		0	
bi-eLU	**0.1875**		**0.1875**		0.03125		0	

Table 4. Train / Test accuracy and loss error in CNN-Small according to Seed 1 on CNN small.

(c)	MNIST		Fashion MNIST		Cifar10		Cifar100	
Train	**Loss**	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**	**Loss**	**Acc**
RELU	0.142	0.973	0.41	0.863	2.304	0.102	4.61	0.010
eLU	2.843	0.139	2.834	0.14	2.84	0.137	6.923	0.035
bi-RELU	1.931	0.357	1.693	0.405	2.407	**0.164**	5.109	**0.041**
bi-eLU	**2.508**	**0.254**	**2.361**	**0.289**	**2.807**	**0.154**	**6.766**	**0.042**
Test	**Acc**		**Acc**		**Acc**		**Acc**	
RELU	0.875		0.875		0.1875		0	
eLU	0.0625		0.125		0		0	
bi-RELU	0.437		0.4375		0.0416		0	
bi-eLU	**0.125**		**0.25**		0		0	

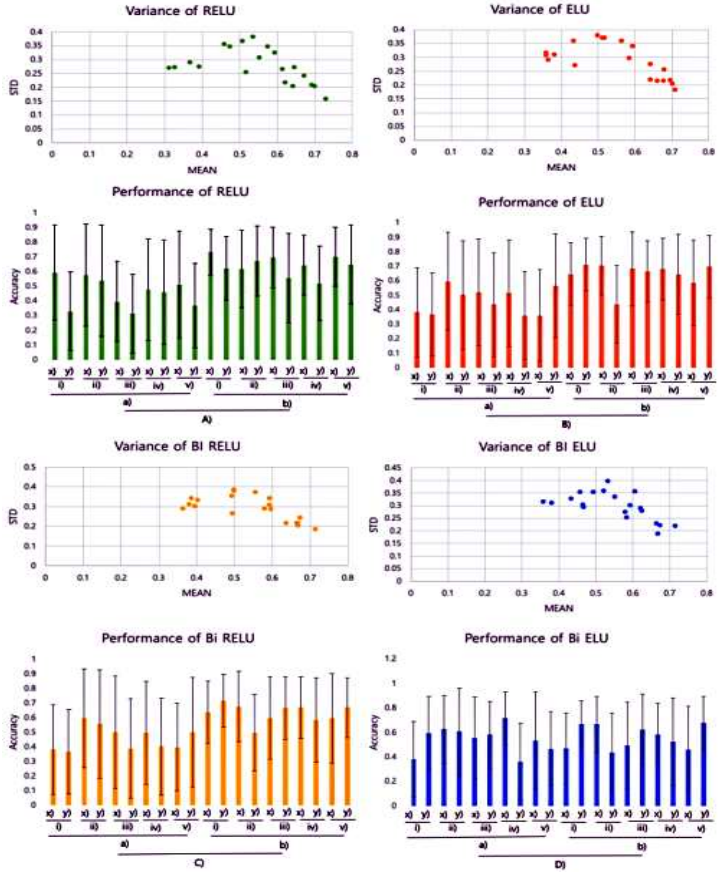


Figure 6 Performance analysis of proposal methods. x) small of CNN, y) two la

yer of CNN, a) Using MNIST dataset, b) Using Fashion MNIST, A) Activation function of RELU, B) Activation function of eLU, C) Activation function of bi-RELU, D) Activation function of bi-eLU, i) Seed 1, ii) Seed 250, iii) Seed 500, iv) Seed 750, v) Seed 999

Figure 6 is composed of 2 parts. The upper scatter plot is the result of the variance of the activation function. The lower histogram is the result of the performance of the activation function. We include the results of experiments with five seeds and two CNN models in Figure 6. The above experiment calculates the average for each activation function. The notation is written as (mu, std). mu is mean, std is the standard deviation. RELU is (0.54227, 0.2271). eLU is (0.543914, 0.29004). bi-RELU is performance analysis of proposal methods in Figure 5. x) small of CNN, y) two layer of CNN, a) Using the MNIST dataset, b) Using Fashion MNIST, A) Activation function of RELU, B) Activation function of eLU, C) Activation function of bi-RELU, and D) Activation function of bi-eLU. The results of Figure 5 show more clustered plots of eLU variance compared to RELU. This is because the nonlinear characteristics reflect more clustered results. In Figure 5, the results of experiments applied to each activation function for each of the two models show that some variation in performance occurs for each seed.

Conclusion: We have demonstrated an improved performance by the Bi-activation function. Bi-activation function combines pos-activation function and neg-activation function in a small number of parameter spaces. Compared with the existing activation function, the Bi-activation function considers bi-directional information to reflect generalization characteristics in the model through bi-directional information. As this reflects the bi-directional characteristic, it can be seen that the convergence speed is faster in the learning process than when

reflecting on the existing single characteristic. Because reflecting the information with the existing activation function has a high complexity in processing the information while processing through the bi-directional information, the complexity is somewhat lower, so the convergence speed seems to be faster. To show the advantage of the proposed activation function, We verified the effect of the initialization method on the input of the bi-directional information in a few parameters that show that bi-directional information is a little bit better when it is nonlinear. Since the weight of the model of the existing deep learning has a value between 0 and 1, the weight information of the deep learning model between 0 and 1 more effectively reflects the nonlinear characteristics through the activation function having a nonlinear characteristic. This property of the proposed transform effectively used for optimization or edge device deep learning.

Through this, this study examined the bi-activation study. Next, we describe the improved cascade loss in the loss function.

3.2 Scale Calibration Cascade Smooth Loss of Generative Adversarial Networks with Online Continual Task Learning.

This research describes the cascade loss that has been researched during the existing acquisition method. We description of cascade loss is as follows [37].

Introduction : Many deep learning models go through the process of learning [38]. The most influential part of this learning process is the loss, which tells how to learn. A lot of deep learning models are coming out, but most importantly, when a deep learning model receives new data and processes it newly, the loss formula determines how much new data is reflected and how

much existing data is reflected.

In the deep learning model, while loss outputs the information as one information from the last stage of the deep learning model, we found that three problems can occur directly. And indirectly from loss's point of view, We discovered a new problem in the process of learning deep learning models. First, the last layer of deep learning is feature vectors with independent features. These feature vectors have various view features. Since various feature vectors are implied, loss assumed that the information overlapping of the feature vectors might occur, and congestion may increase during the calculation. Also, since deep learning only uses a single loss, it was speculated that if the character is reflected in the wrong direction during the optimization of the single loss, the learning is conducted in the wrong direction. And if the value from loss has a skewed distribution to one side, it is assumed that it will be learned in the wrong direction. The above assumption is assumed to be a phenomenon that can occur directly in a deep learning loss. Indirectly, indirectly, it is assumed that the deep learning model can be learned in a good direction if it is sustainable in terms of loss, and there is a process of learning minutely by various tasks because existing deep learning models are three types of learning. i.e.,) offline learning, online learning, incremental learning, Offline learning is a batch-based method that does not change the approximation of a function until the initial training is completed. Online learning is a way to learn data by data without waiting for the initial training to complete. Finally, incremental learning is a method of learning incrementally. This deep learning model learning method has studied through three categories. However, there is a problem that does not depend on each method and is common to all three learning methods. This is a catastrophic forgetting problem that occurs in continuous learning. Catastrophic

forgetting problem means that learned information disappears without being maintained. We alleviate this catastrophic forgetting problem that is very important in the direction of life long deep learning model. We have assumed that the method of learning with precision is necessary for the deep learning model to be sustainable. In addition, these problems can often occur in the process of continuously learning the deep learning model when the actual deep learning model is deployed and operated in real service. The above problems need to be researched to improve deep learning to learn for a long time continuously. Therefore, the contribution is as follows.

- We novel defined direct three and indirect types of problems that arise from existing losses. We proved that the proposed direct problem could be alleviated by the proposed loss function.
- To prove, we defined three types of direct problems, we proposed Scale Correction Cascade Smooth Loss (SCCSL). The SCCSL consists of three-component. The first component is based on smooth loss. The second component is cascade component to make stronger judgment criteria by maintaining existing judgment rules to some extent and reflecting new judgment information on existing rules. The third component is a scale correction for learning by making the information acquired in the process of reflecting the information. Also, we propose the cause of the loss problem and prove the reason for the problem to be solved.
- We defined a new indirect problem of learning to mitigate problems that may arise indirectly. This is a learning method for continuously learning from the model. The definition of learning is the method for fine-grained learning on deep learning.

Proposal Methods: We defined three direct problems and indirect problems on the existing loss function.

Definition 1 *: The overlapping information in the lost inputs occurs, and errors occur, which prevents the deep learning model from learning properly.*

Description : We have proposed a smooth component that can resolve redundant information and errors to prove reliable learning. The final input of the existing model consists of various feature vectors (feature maps). We tried to prove that it is necessary to improve the error by proving that each feature map should be mapped with minimized error in the process of mapping into one space of the loss. For this purpose, this study proposes soft components. Details of the soft components are described in the subsections below. We can see that the problem defined is reduced by suggesting soft components

Definition 2 : *The method of learning using a single loss in a loss causes a problem in that the learning is not good because it leads the learning in a direction that is inaccurate.*

Description : We have proposed a cascade component to solve the problem of learning in the wrong direction in a single loss. This is to learn information from a single direction in the correct direction through error correction. The details of the cascade component are described in the subsections below. By suggesting the cascade component, we can reduce the problems defined.

Definition 3 *: Since the error in the loss occurs in an unbalanced state or balanced state, If there is a problem that the learning is difficult due to the unbalance information.*

We proposed a scale correction component to solve the unbalanced information in the loss. Detailed methods for the scale correction component are described in the subsections below. By suggesting a scale correction component, we can

alleviate the problems defined. This is because when learning in one tilted state, only the information inclined to one side in the model is reflected by learning only the information, which may cause a feature that does not have a generalized characteristic.

We propose SCCSL loss to improve the method proposed in this study through direct problem definition that can occur in the loss. Our SCCSL is composed of three components. First is the Smooth component, Second is the novel cascade component. The third is the novel loss of the Scale correction component. The novel proposal losses are as follows.

$$\text{SCCSL} := -\frac{1}{m}\sum_{n=1}^{m}\alpha(1-smooth\,component)^{\gamma}\,y_{true}\log(smooth\,component) \quad (1)$$

Eq. 1 is the loss proposed. The proposal loss is a basic loss based on smooth loss. Based on the base loss, the two losses are merged through the cascade component. It is composed of a scale correction method to unbalance information based on two merged losses. m is the number of data. y_{true} means ground truth. A detailed description of each method is given in the subsection below. We explain each proposed method to analyze this.

Smooth component for reduce co-adaptive information and error information: We propose a method using the smooth component as the basis of the loss. If the function is smooth and the Taylor series at all points is equal to the loss value, it has the advantage of being an analytic function.

$$Smooth\,component :=$$

$$pos = \sum y_{true} * y_{pred}$$

$$\text{neg} = \max(1 - y_{true}) * y_{pred}$$

$$\frac{1}{m}\sum_{n=1}^{m}\max(0, \max(\neg - pos + 1)) \quad (2)$$

The analysis of the cascade component is as follows. Experiments were conducted using three losses of smooth loss, focal loss, and cross-entropy loss. Eq. 2 proposed as a single loss is explained. y_{true} means the ground truth answer set for learning the GAN's discriminator. y_{pred} means the value predicted by the discriminator of the GAN. m is the value of the total iteration number sum. We calculated the positive value and negative value separately. In this case, the positive value denotes by pos, and the negative amount is indicated by neg. The pos expression $y_{true} * y_{pred}$ tries to express common information by multiplying. neg is reflected in y_{pred} through $1 - y_{true}$ to maximize the non-correct answer. The reason for calculating the pos and neg divided is to consider only positive value and to make the negative value as 0, divide it by using sum for pos and neg as 0. Also, we wanted to set this to 0 from neg-pos + 1. The reason for applying neg-pos is to reflect a clear difference by reflecting only scarce information. We applied max max at the same time to generate a huge amount of information from max. We filter it by using a specific value through the max function, so we set it as above to accurately reflect the value of one category. The Smooth loss design the MAX to extract a smooth value using the value calculated at zero. The smooth loss divides into the front part and the rear section. For the first part, the most influential role maximizes by pos-neg, and the most massive value extract through max through attention. And in the second term, the value of y_{true} - y_{pred} remains for the difference between them, and there is much value for y_{true} here. Subtracting the

last word in the preceding clause, we can deduce that y_{true} information in y_{pred} information is a strong part of y_{pred} through subtraction. This results in a large value for y_{pred} which is far from the number of y_{true}. It can assume that the value between y_{true} and y_{pred} maximize.

Cascade component for calibration information: The cascade component in Eq. 3 is a method of using the result value of the existing loss as a second loss input. This method has the advantage of learning in the right direction through calibration if the existing loss goes to the wrong case. In the process of correcting a certain value properly, it is better to produce a better effect, but in case of incorrect calibration, it may have an adverse effect.

$$Cascade\,component := Second\,loss\,(first\,loss) \tag{3}$$

Scale Correction Component for Unbalance Information:

$$\text{Scale Correction Component} := \alpha(1-prior_{loss})^{\gamma} * y_{true} * \log(prior_{loss}) \tag{4}$$

The scale correction method proposed is as follows. Eq. 4 is a novel scale correction for continuous acquisition information by scale interpolation. $prior_{loss}$ means an existing loss. α is the influence covariance of the scale term. And γ is the scale term the attenuation term for general acquisition influence of the existing loss. The impact depends on continuous task loss on the coefficients of the two parameters. We try to confirm the influence of scale correction through various scale index values. In the case of scale correction, in the case of the information rarely generated in the past, the correction is large, so that the rare information can be stably obtained. This helps to obtain fine-grained and accurate information stably and improve the overall information acquisition amount.

Qualitative analysis of the SCCSL: We propose an SCCSL that is composed of three components (i.e., Smooth component, Cascade component, and Scale correction component). We conducted a qualitative analysis by a section on the generation of loss results for the impact of SCCSL. We analyze the SCCSL by dividing the interval on the numerical value. First, the interval was defined by dividing it into positive infinity, positive value, zero, a negative value, and negative infinity. In the case of total loss at infinity, nan is generated because it is not learned. The positive value is a case where the sum of weights is less than one. In the above case, the pred is good. If 0, no learning is done. Negative values do not match pred. Negative infinity does not occur. The SCCSL does not learn when the positive value is zero. Learning is only learned in positive and negative cases.

The smooth component means that it can be differentiated, and the derivative is continuous, and the loss can be differentiated k times, and all of the derivative loss is continuous. The above situation can show that there is a continuous derivative of every order at every point x in the solid line off. For this reason, we propose to use the smooth loss as the basis of the loss.

The cascade component is a method of using the output of an existing loss as the input of a new loss. This method can be changed in the right direction when the existing information is wrong by applying the new method with the existing information partially preserved. However, there is a disadvantage that the existing information can be changed in the wrong direction. However, the cascade component tries to improve the effect of forgetting existing information when learning.

The scale correction component is a method of using the information imbalance, when data is acquired, an imbalance between types occurs according

to the ratio of the number of data. In this case, when the model learns the unbalanced information, it is more likely to misjudge one side. We tried to learn effectively through the calibration about the possibility of false judgment.

Pros and cons of proposal method: The proposed method has three pros. First, It can be confirmed that it helps to judge the information efficiently by maximizing the margin between the characteristics of the distribution. This helps to make a clear determination by eliminating duplicate information determined by the model. Second, If it comes out through one characteristic, it is calibrated using another characteristic, so it can have generalized characteristics by reflecting characteristics. Also, when learning in the wrong direction during the learning process in the model, the mixture is corrected using different characteristics, so it is in the generalized direction. Third, It can help to learn more stably in the learning process such as overfitting or underfitting by correcting the balanced information in the process of learning the imbalanced data.

The proposed method has five cons. First, only the method that can be processed in terms of signal processing in the model is described. This can show the influence of learning performance on signal stabilization, regulation, initial model state, and analysis based on Convex characteristics. For the proposed method, it is necessary to conduct other influence analysis on stabilization, regulation, the initial state of the model, and convex characteristics.

Second, The experiment was applied only to the model of Generation Adversarial Networks. However, in order to generalize the method proposed, it is necessary to apply the experiment to experiments such as segmentation, object detection, classification, etc.

Third, We experimented with various losses, two models, and two small image

data sets. It is necessary to experiment with various sizes and sizes, such as images with large resolution, fine images, and packet data.

Fourth, To clearly show the influence of the method by the slope, it is necessary to mathematically interpret the feature map in terms of learning flow to find out the direction indicated by the slope of the feature map.

Fifth, It is necessary to mathematically prove the influence of each method.

Sixth, We need to confirm the impact proposed through the domain adaptation experiment.

Seventh, It is necessary to apply it to various state of the art technologies. However, it seems necessary to apply the experiment to the lightweight model, the distillation model, and the distributed model. Because the lightweight model can handle different amounts of parameters, the performance may be different. Similarly, in the knowledge distillation model, it is necessary to experiment on the above because learning performance affects the content according to the quantity and quality of information. Finally, it is assumed that the method proposed can show a better effect because it receives and processes the characteristics of various models in distributed computing. Therefore, the above process should be tested and verified.

Novel Online Continual Task Learning: We define online continual task learning methods that explain how to learn at the aspect of the loss function is more fine-grained learning on deep learning models. To clarify problems, we define a method using the task information and continual learning. This is shown in Figure 7 briefly and clearly for the proposed method. In the previous researches, there have been many studies of online learning and continuous learning. However, in previous studies is not a precise method of learning a task about the online way. We think that the study that reflects more precisely in

online learning and continues learning is essential for the AI life long model. Therefore, we define a problem called online continual learning that considers both online learning and continuous learning issues at the same time. We define the more critical problems in the problems based on online continual learning and propose sub-research topics of related problems through pros and cons. First, We explain the contents based on online continual learning.

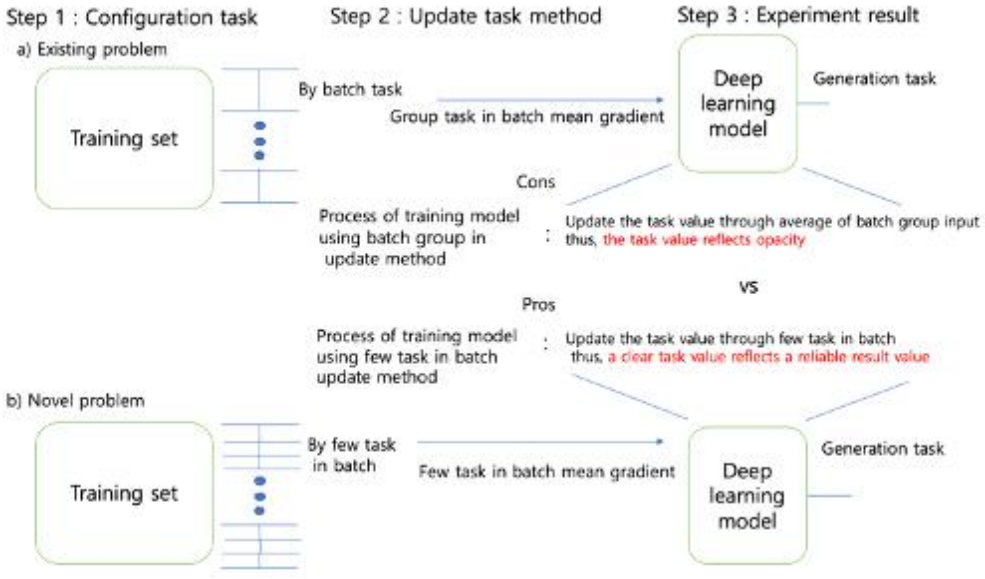


Figure 7 Novel problem that explain the direction to get a descriptive on few da ta learning at online continual task learning optimization

The term online continual learning is used [39]. However, looking at the episodic gradient memory for continual learning [40], a reference to the terminology used in [39], does not come out as online continual learning. Therefore, we define the term online continual learning.

Definition 4 : *online continual learning problem is to learn one by one in continual learning problems about new data at continual learning.*

We define this as Online continual learning in Definition 4. The problem .

Definition 5 *: online continual task learning problem is to learn more efficiently about more precise learning methods about new tasks at Online continual learning.*

We define this as online continual task learning in Definition 5. The problem proposed explains in more detail in Figure 6.

Why the research direction design by the above method is that in the process of learning continuous information, it is necessary to secure delicate information. That can lead to the model falling in the wrong direction when viewed as a whole, no matter how sensitive the information is. Therefore, the study conduct in consideration of both micro information and the overall tendency for the model.

The new problem explained in detail in Figure 7. Generally, to learn and evaluate the deep learning model, data is composed, and a part of data extract from the constructed information, and the task of the average value of the extracted data reflect in the deep learning model. You get the result of the task you want through the above learning process repeatedly. In the above process, learning the average information of the layout has a problem that the value of the task reflects unclearly. However, the new problem proposed demonstrates the task of each little data in the course of continuous learning of the data, so the model learns through the task that can be confirmed by the human. The method of acquiring through the assignment, which can be confirmed by a person like that, is advantageous in that it can reflect the task which is known by the person. However, the newly defined problem has four difficulties.

Firstly, there is a difficulty in obtaining information by efficiently storing the

current knowledge and reducing the vibration in the process of getting standard information continuously. The reason why we need to converge by reducing the vibration from the task is that there is a case where the task does not converge when the task reflects. Next, there is a difficulty in obtaining the size of the task unevenly in the course of collecting continuous task data. Therefore, the size of the task becomes unbalanced, which makes it difficult to reflect a rare task, and there is a problem that the information shifts to one side through uneven task processing.

Also, To obtain a steady task, it is difficult to efficiently reflect and maintain continuous task data in existing distributions for information to reflect on the existing distribution. If the present task information disappears, it makes an incorrect judgment about the data that you have already learned.

Finally, models that have a steep task and reflect inclination have difficulty in storing single information and generalized information. A model that efficiently processes only one information is not available in many fields.

We propose a sub research topic of online continual task learning, which is inspired by segmentation that is existing computer vision deep learning studies.

***Corollary**: Online continual task learning is a theory for learning more precisely, accurately, and explicitly the phenomena that occur in traditional Online continual learning. Online continual task learning is necessary for an explicit and accurate learning model. Therefore, Online continual task learning discussed in three areas: e.g.) Online instance task learning, b) Online semantic task learning, and c) Online panoptic task learning.*

Online continual task learning discusses in three areas in Corollary 6. i.e.) a)

Online instance task learning, b) Online semantic task learning, and c) Online panoptic task learning.

Firstly, Online continual instance task learning is a method of learning per instance task in an Online continual task learning environment. The advantage of this method is that you can make precise decisions about each instance task. The disadvantage is that much fluctuation occurs when a new instance task comes in.

Additionally, Online continual semantic task learning is a method of learning by the semantic task in an Online continual task learning environment. The advantage of the above method is that it can find semantic information quickly and can use for real-time information processing. The disadvantage of the above method is that the allocation of contiguous memory space is difficult when the semantic task is massive.

Finally, Online continual panoptic task learning combines the above method — online continual instance task learning with Online semantic task learning. An advantage of Online continual panoptic task learning is that you can consider both instance information and semantic information at the same time. The disadvantage of the Online panoptic task, learning is that it takes much computation to acquire.

Experimental result and discussion:

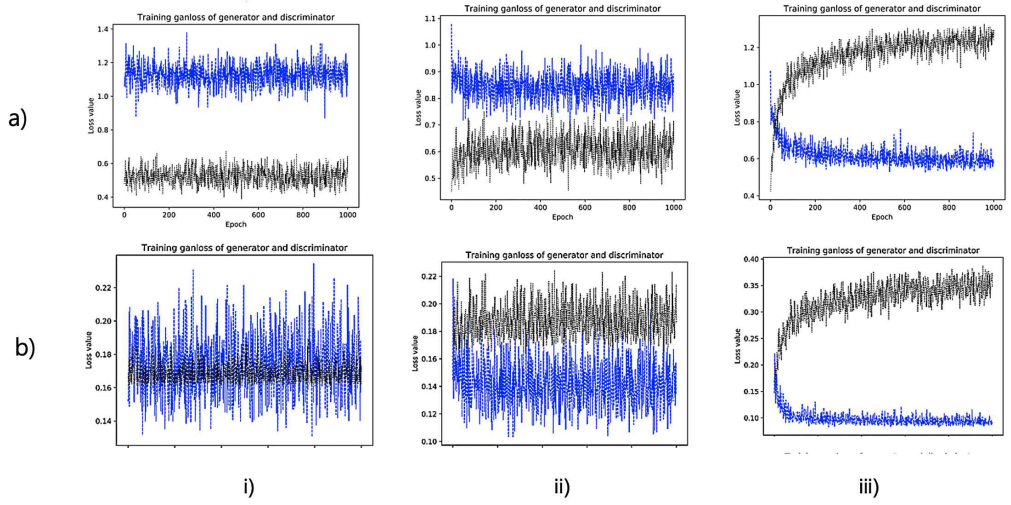


Figure 8. Compare of optimization methods on verified of proposal loss

Experimental verification of the proposed method performs on two image generation models, Valina-GAN and LSGAN at a single image generation task. The data used in the experiment verify by two datasets.

We first prove the problem due to the overlapping information in the loss; the deep learning model is not properly trained. We first try to verify the experimental results by using a method that can influence the successive acquisition of the continuous task information in a single loss. The influence of batch size on the single smooth loss was analyzed using weight decay. We can see that the variation of the loss reduces by applying the weight decay, and the discriminator loss and the generator loss maximize. As can be seen from this phenomenon, weight decay is less sensitive to batch size. However, if there is no weight decay, we can see that the batch size converges later, depending on the larger. Adam optimizer is well used and well used, but we analyzed the influence of three optimization methods to verify the proposed method. The result of the AdaDelta shows that the generator loss learns the cross-entropy and

focal loss with similar loss values. In the case of the smooth loss, it can seem that the loss value is somewhat higher than the two methods in obtaining the continuous task information stably. It assumes that the knowledge of the continuous task information gathered is not varied or corrected in the correct direction because only the max value obtains in smooth loss calculation. Smooth and focal loss analyzed according to three optimization methods and potential variable spaces that are to confirm the influence of generation performance according to the impact of the Z latent space size according to the input on the generation model. As the latent variable space increases, the convergence phenomenon appears later when the influence of the Z latent space influenced by the size of the potential variable area. It is essential to map and store the optimized values in the memory space through optimized parameters of the size of the storage space according to the data, which is necessary for model convergence through optimization of the model.

In summary, the smooth component can obtain smooth information. Also, we show that the model converges by acquiring knowledge reliably on a single loss, even if only a single loss used because of the correction obtained in the case of focal loss. However, there is some limitation in acquiring a piece of continuous task information in a single loss. The experimental result of the cascade component, which is a method to solve the constraint using the new environment, is as follows.

We secondly prove the problem of the method of learning using a single loss in a loss causes a problem in that the learning is not good because it leads the learning in a direction that is inaccurate. To verify the cascade component, we experimented about the proposed method when training with the Fashion MNIST dataset. Experimental results using the Cascade component are as follows.

We compare of optimization methods on verified of proposal cascade component that composed of 4 batch size, L1 0.0 regularization, L2 0.5 regularization, Z latent space size 100, and in Valina-GAN model using grayscale Fashion-MNIST dataset. i) Adadelta, ii) Adagrad, iii) Adam, a) Smooth loss, b) Smooth correction loss, i) Adadelta, ii) Adagrad, iii) Adam

Figure 8 is a comparison between Smooth and Smooth correction loss according to the optimization method. The result of applying the Cascade component has an error range of 0.5 to 1.4 when only a single component is applied, but the cascade component has a small error value of 0.2 to 0.4. This can be confirmed that when the existing loss is learning in the wrong direction, the loss reflected later is stably learned through the correction effect.

The cascade component has the advantage of being able to reuse a loss having the same characteristics and applying a loss having different characteristics. Since it is composed of a pipeline, it can be applied to parallelization, It is a way to apply multiple loss rather than apply. By applying a lot of loss, it is possible to make a loss that can be performed more precisely and to shorten learning time through loss. Cascade component has advantages such as performance enhancement, parallelism, reflection of characteristics, but there is a disadvantage that loss increases.

Third, it is assumed that the learning is difficult due to the imbalance of information because the error occurring in the loss occurs in an imbalanced state. A scale correction method is proposed. Experiments verifying the proposed method for scale correction are as follows. Adopted cascade component is the results of applying the scale correction component, correction correction component is a strong influence of Scale calibration, and showed only one learning or no learning at the time of confrontational learning. However, the

results of the smooth correction component show that it is possible to improve the existing learning by calibration through scale calibration. Scale correction shows better learning results. We prove this cascade component by applying scale calibration. To achieve stable acquisition for continuous task information through size calibration to obtain continuous task information, the result of verifying the novel scale correction method is as follows. To analyze the influence on the scale correction was explained in particular by the importance of the value of gamma and alpha in the scaling. We compare of the effect of test smooth calibration method on alpha and gamma coefficients at Learning rate 0.0007, Batch size 4, L1 0.25 regularization, L2 0.25 regularization, Z latent space size 100, Adam optimizer, Gumbel distribution in Valina-GAN (alpha 0.25, alpha 0.5, alpha 0.75, beta 2.0, beta 3.0, and beta 4.0). When the smooth calibration method, alpha was changed from 0.25 to 0.75 through 0.25 increments, and beta was changed from 1 to 4 through 1 increase. Experimental results show that the influence of attention on the impact of alpha and gamma does not affect the generation of the generated image. Scale term, the effect of the parameter in the scale is influential in creating the image in the Valina-GAN. These results show that, when the scale term enters, it is corrected for the acquisition of the continuous task information so that results obtained. However, it confirmed that the influence on the steady of the continuous task information acquisition is independent of the impact on gamma and alpha. Supplementary data were presented as a detailed basis for the proposed method.

Conclusion: We propose three direct problems and indirection problems that can occur in a loss and verify them through three new methods about three directions problem. In addition, we have defined a new learning method that can

alleviate problems that may occur indirectly in the loss. The new learning method is called online continual task learning. In our newly proposed loss, we confirmed that we could alleviate three problems that can occur in the existing loss that the novel smooth component, novel cascade component, novel scale correction component proposed to reflect task information.

First, the smooth component is smoothly displayed by maximizing the margin between feature information while various feature vectors reflect duplicate information through input, thereby reducing errors from various feature vectors. It is confirmed that the performance is improved.

Second, it shows the characteristic of the loss of parallelism from the cascade component and shows that it can apply the loss of features. The continuous task information synthesis by cascade component plays an essential role in the formation of new information. Also, the effect of regularization of the continuous task information may suppress the long term potentiation. However, the formation of memory through the continuous task information stored in the parameters of the model can withstand a massive amount of continuous task information stabilize inhibitor. Thus, it has shown that the synthesis of a continuous task may not be necessary for model parameter enhancement. It can see that the continuous task information synthesis using the cascade component of the required continuous task information synthesis is not unconditional.

Third, it can confirm that the learning is good by reflecting the task information stably through the scale interpolation. The scale of obtained the continuous task information has a distribution of scales, and it can confirm that the information between the continuous task information is obtained stably by balancing the continuous task information through the calibration of the scale of the continuous task information.

Finally, in order to continuously and precisely learn from the loss function, this study proposed a new online continual task learning method. This seems to allow us to move toward research where we can correct fine-grained information in the wrong part of the model through learning.

In the future, we will conduct research that can analyze and correct problems that can occur in deep learning models through fine-grained and precise learning methods. We will study the process of stably learning through the process of making a new loss using Bayesian theory.

Through this, this study examined the cascade loss with online learning study. Next, we describe the nonlinear exponential regularization [40]

3.3 Nonlinear Exponential Regularization : An Improved Version of Regularization for Deep Learning Model

Introduction : Deep learning models try to train the lowest points of cost functions. In the process of learning to the lowest target points, the direction should be given to go to the direction. To direction, the cost function, which tells how to learn, must be carefully devised with proper regularization. However, most existing regularizations of the linear combination of L1 and L2 regularization is a supplementary role to assist the loss. This regularization is responsible for driving the model in the right direction for learning. In the case of simple linear combinations, the simple linear combination of L1 and L2 regularization causes a problem that the characteristics of L1 and L2 regularization are not reflected well. We studied how to efficiently reach the target point by reflecting the influence of characteristics of L1 and L2 regularization efficiently. From extensive experiments, we can find that the nonlinear exponential combination of important loss in regularization considerably

reflects the characteristics of L1 and L2 regularization. Well-known regularization methods in deep learning models are L1 and L2 regularization [32]. L1 regularization is the absolute value of the error between the actual and predicted values, and L2 regularization [32] is defined as the sum of squares of errors. L1 regularization is more robust for outlier than L2 regularization because L2 regularization is calculated by the square of the error [2]. L2 regularization always gives a unique value for each vector [32]. This keeps low features and finds better predictions. In some cases, L1 regularization can output the same value. This makes it possible for the L1 regularization to be used as feature selection. As a result, this L1 regularization is suitable for sparse coding. However, there is a non-differentiation point. We want to well regulate the model through each of these L1 and L2 regularization. We can experiment by applying each regulation in various ways. For example, We generally use linear addition for regularization. The performance of the two regulatory methods is not reflected effectively because this linear combination of two regularizations is reflected symmetrically. From this observation, we propose a new method of regularization through nonlinear exponential regularization for better regulation performance by reflecting each performance efficiently.

$$-\frac{1}{m}\sum_{m=1}^{m}|y_{true}-y_{pred}|e^{-\frac{1}{m}\sum_{m=1}^{m}(y_{true}-y_{pred})^2} \quad (5)$$

OUR PROPOSAL : Eq. 5 is the nonlinear exponential regularization to find optimal solution and fast convergence. In Eq. 5, n is the class number, m is the total sum of n, y_{true} is the result of the ground truth, and y_{pred} :is the model prediction result. We use two terms for regularization. One is the average of absolute of the difference between the model predictions and the model ground

truth to make robustness to the outliers of the predicted values. The other is the average of the square of the difference between the model predictions and the model ground truth. We combined each regularization term through the exponential function. We used L1 regularization as a scaling term and L2 regularization as a phase term of the exponential function. We can also use L2 regularization as the scale term and L1 regularization as the phase term of the exponential function. However, we did not experiment with L2 regularization as the scale term because we only wanted to reflect the magnitude through the absolute value. Consequently, we conduct the nonlinear combinations of regularization and apply the exponential operation to combinations of existing regularization such as L1 and L2 regularization. The experimental method to verify the proposed regularization is as follows: a) L1 regularization, b) L2 regularization, c) linear combination of L1 and L2 regularization, d) nonlinear exponential combination of L1 and L2 regularization. To verify regularization effect, we experiment with four regularization coefficients, e.g.) 0.0, 0.25, 0.5, and 0.75. Also, we experimented on three-loss (Cross entropy, Focal loss, and Hinge loss) using Adam with Vanilla-GAN and LSGAN [41].

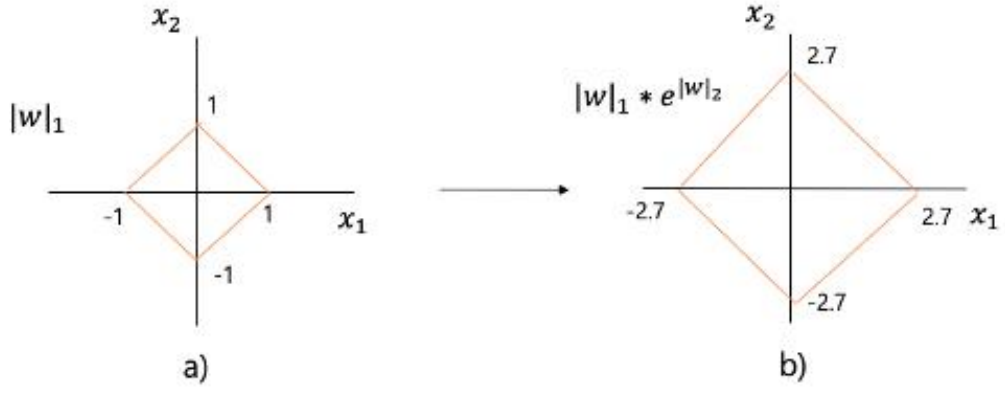


Figure 9. Effect of ours proposal for suboptimal training path gradients, a) L1 regularization b) Nonlinear exponential regularization (Our proposal)

Figure 9 shows the visualization of the effect of nonlinear exponential regularization on the L1 in terms of weight. Nonlinear exponential regularization results reduce the complexity of the weight by giving the scaling effect of L1 regularization. By lowering the complexity of the weights, the gradient in model moves efficiently to suboptimal optimization.

We conducted experiments based on the type, e.g.) Normal, Laplace, logistic, Gumbel, and size, e.g.) 100, 500, 1000 of latent variables. We checked the performance of the methods using MSE (Mean Square Error) and PSNR (Peak Signal to Noise Ratio).

Experimental image generation task results: We used three losses of focal, hinge, and cross-entropy to compare the images generated by Vanilla-GAN and LSGAN. As a result, the images created by the GAN with the hinge loss are clearer than those created by the GAN with the other loss. In the training process, the Vanilla-GAN using hinge loss with a smooth slop was found to be stable to generate images. Figure 10 shows the comparative analysis of proposed nonlinear exponential regularization and linear regularization. As a result, nonlinear exponential regularization generally has fewer errors than linearly combinational regularization on two datasets and two models.

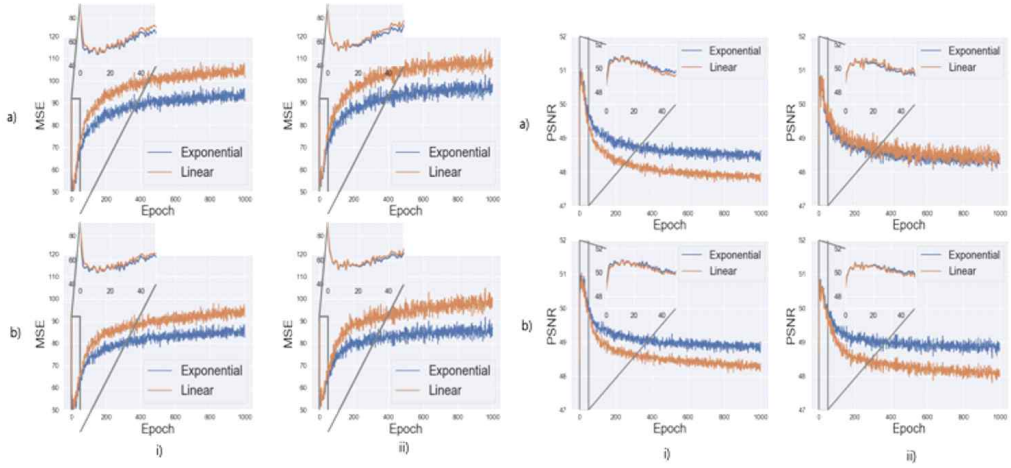


Figure 10. Comparative analysis of proposed methods to help fast convergence in training a) Vanilla-GAN, b) LSGAN, i) Cifar10, and ii) Cifar100

Table 5 Experimental results using two datasets on two models: top 5 values of PSNR and MSE

Measure	Model	Dataset	Existing	Proposed
MSE	Vaniila-GAN	Cifar 10	51.47±1.50	50.33±0.78
		Cifar 100	57.15±3.01	56.81±2.88
	LSGAN	Cifar 10	51.12±1.34	50.13±0.61
		Cifar 100	56.31±2.95	55.94±2.91
PSNR	Vanilla-GAN	Cifar 10	50.87±0.13	50.96±0.06
		Cifar 100	50.52±0.33	50.44±0.22
	LSGAN	Cifar 10	50.90± 0.11	50.97±0.05
		Cifar 100	50.48±0.23	50.51±0.23

We extracted the top 5 values from the experimental results as the 5 lowest valu es in the MSE and the highest values in the PSNR in Table 5. Our nonlinear e xponential regularization (proposed in table) shows lower MSE of 0.71 and better PSNR of 0.0275 than the linearly combined regularization of L1 and L2 (existin g in a table).

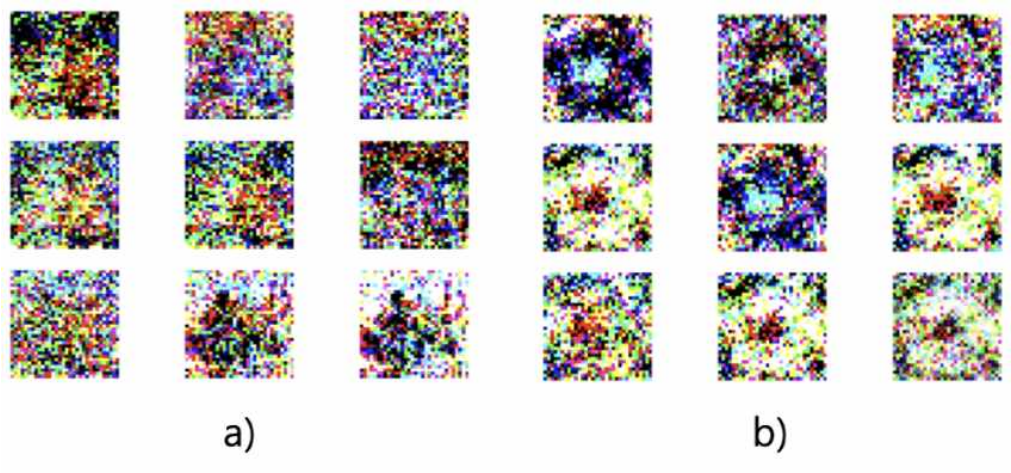


Figure 11 Generated images by a) linearly combined regularization and b) nonlinear exponential regularization

Figure 11 shows generated images by linearly combined regularization and nonlinear exponential regularization. Proposed regularization generates better images than the linearly combined regularization. This may be because proposed exponential regularization finds more optimal of gradients by focusing more important loss. The effects of nonlinear exponential regularization are applied to semantic segmentation and verified.

Table 6. Ablation study of our proposal in two models, a) loss, b) loss with linear combination of L1 and L2 regularization, c) loss with nonlinear exponential regularization of L1 and L2 regularization, d) loss with nonlinear exponential regularization of L1 and L2 regularization and linear regularization of L1 and L2 regularization

	DSC	**F1 Score**	**IOU**	**Precision**	**Recall**
a	0.7275	0.7275	0.7715	0.7275	0.7275
b	0.7255	0.7255	0.77	0.7255	0.7255
c	**0.728**	**0.728**	0.7715	**0.728**	**0.728**

d	0.7241	0.724	0.7689	0.724	0.724

Table 6 shows the experimental results by applying batch 2, learning rate 0.07, early stop, and seed 777 in Semantic Segmentation. Experiments using FCN [42] and U-Net [43] averaged five experiments. Percentages are expressed through the result of the division reference value after 100 times the difference between the reference value and the comparison value. To verify the regularization effect, Table 6(a) shows only cross-entropy. Table 6(b) shows the linear combination of L1 and L2 regularization in addition to cross-entropy, Table 6(c) shows the nonlinear exponential regularization in addition to cross-entropy, and Table 6(d) shows the ensemble regularization that is composed of linear combination of L1 and L2 regularization and nonlinear exponential regularization in addition to cross-entropy. Nonlinear exponential regularization improved DSC 0.06%, F1 score 0.06%, IOU 0%, Precision 0.06%, and Recall 0.06% than origin. Nonlinear exponential regularization improved DSC 0.34%, F1 score 0.34%, IOU 0.19%, Precision 0.34%, and Recall 0.34% than linear combination of L1 and L2 regularization. Nonlinear exponential regularization improved DSC 0.535%, F1 score 0.549%, IOU 0.337%, Precision 0.549%, and Recall 0.549% than ensemble regularization.

We conducted for further improvement. However, the performance did not improve. We experimented with an EMA (Exponential Moving Average) regularization. EMA is a first-order infinite impulse response filter that applies weighting factors which decrease exponentially. The weighting for each older datum decreases exponentially, never reaching zero. The graph at right shows an example of the weight decrease. The equation of EMA can be expressed as

Exponential Moving Average : =

Step 1. A × L1 *regularization* − (1 − A) × L1 *regularization* − B × L2 *regularization*

Step 2. $A \times Step1 + B$ *or* $(1 - A) \times$ *L2 regularization* (6)

Eq 6 is the exponential moving average. A is a coefficient of L1 regularization. B is a coefficient of L2 regularization. For verifying the performance of our proposal, we experimented based on cross-entropy using Adam optimization with FCN, U-Net, Deep Lab v3 [44] using VOC, ATRdataset. We checked the performance of the method using DSC, F1 Score, IOU, Loss, Precision, Recall for the quantitative analysis. Note that these results come out with 5 iterations of the experiment, batch size 2, seed 777, and early stopping used. Our experiment occurred in a desktop with GTX-1080ti as GPU and Ubuntu 18.04 as the operating system.

Table 7. Comparative test for verification of nonlinear exponential average moving in the VOC dataset. a) loss, b) loss with linear combination of L1 and L2 regularization, c) loss with exponential moving average linear combination of L1 and L2 regularization, i) FCN, ii) U-Net, iii) Deep lab v3

Model	Methods	DSC	F1 Score	IOU	Loss	Precision	Recall
i	a	0.733	0.733	0.776	1.125	0.733	0.733
	b	0.727	0.727	0.772	1.096	0.727	0.727
	c	0.730	0.730	0.774	1.254	0.730	0.730
ii	a	0.735	0.735	0.777	1.610	0.735	0.735
	b	0.733	0.733	0.776	2.742	0.733	0.733
	c	**0.739**	**0.739**	**0.780**	2.387	**0.739**	**0.739**
iii	a	0.706	0.706	0.759	2.516	0.706	0.706
	b	0.710	0.710	0.761	1.963	0.710	0.710
	c	0.698	0.698	0.754	1.740	0.698	0.698
average	a	0.724	0.724	0.767	1.75	0.724	0.724
	b	0.723	0.723	0.769	1.933	0.723	0.723
	c	0.723	0.723	0.769	1.793	0.722	0.722

Table 7 shows the result of the comparative test for verification of nonlinear exponential average moving in the VOC dataset. In this experiment, three regularization methods are tested: no regularization, linear combination, and exponential mo

ving average (EMA) with linear combination. Exponential moving average has lo wer performance among all methods when comparing the average of indexes. Co mpared with no regularization from our proposal, it has a 0.14% decrease in bot h DSC and F1 score, 0.28% decrease in precision, and recall. On the other han d, EMA has an increase in IOU and loss, with 0.26% and 2.40%, respectively. When compared with linear combination, the proposed method has the same valu e as linear on DSC, F1 score, an IOU, and less value on loss, precision, recall, which decrease rate is 7.81%, 0.14%, and 0.14% respectively.

Table 8. Comparative test for verification of nonlinear exponential average movin g in ATR dataset using linear coefficient a) loss, b) loss with linear combination of L1 and L2 regularization, c) loss with exponential moving average linear com bination of L1 and L2 regularization, i) FCN, ii) U- Net, iii) Deep labv3

Model	Methods	DSC	F1Score	IOU	Loss	Precision	Recall
i	a	0.718	0.718	0.764	1.054	0.721	0.721
	b	0.719	0.719	0.764	1.046	0.719	0.719
	c	0.717	0.717	0.763	1.203	0.717	0.717
ii	a	0.726	0.726	0.769	1.087	0.726	0.726
	b	0.723	0.723	0.767	1.113	0.723	0.723
	c	0.721	0.721	0.766	1.097	0.721	0.721
iii	a	0.767	0.767	0.797	0.773	0.767	0.767
	b	0.763	0.763	0.795	0.839	0.763	0.763
	c	0.766	0.766	0.797	0.933	0.766	0.766
average	a	0.737	0.737	0.776	0.971	0.738	0.738
	b	0.735	0.735	0.775	1.085	0.735	0.735
	c	0.734	0.734	0.775	1.077	0.734	0.734

Table 8 shows the results of the comparative test for verification of nonlinear ex ponential average moving in the ATR dataset. Three regularization methods are a lso tested: no regularization, linear combination of L1 and L2 regularization, and exponential moving average (EMA) with a linear combination of L1 and L2 regu larization. In general, EMA has low performance among all methods when comp aring the average of indexes. Compared with no regularization, EMA has a 0.4 1% decrease in both DSC and F1 score, 0.54% decrease in precision and recall, and 0.13% decrease in IOU. On the other hand, EMA has an increase in loss w

ith 9.84%. When compared with linear combination, EMA has the same value as linear on IOU, and less value on DSC, F1 score, loss, precision, recall, which d ecrease rate is 0.14%, 0.14%, 0.74%, 0.14%, and 0.14% respectively

We experiment with the exponential moving average combination of L1 and L2 r egularization functions for regularization. We can see that nonlinear exponential moving average combination lower the performance. Reflecting exponential movin g average through recurrent method efficiently move them to optimal in the mod el. The experimental method implanted the experiment in batch size 2 because th e larger batch size is limited to work on due to the out of memory in our deskt op device. Some cases turned out to get better performance but normally get low er performance because of the smaller batch size.

Table 9. Comparative test for verification of ours experiment in an average of A TR dataset and VOC dataset using convex coefficient a) loss, b) loss with linear combined of L1 and L2 regularization, c) loss with nonlinear exponential regulari zation, d) loss with exponential moving average regularization.

Methods	DSC	F1 Score	IOU	Loss	Precision	Recall
a	0.727	0.727	0.77	1.394	0.727	0.727
b	0.725	0.725	0.77	1.499	0.725	0.725
c	**0.728**	**0.728**	0.77	1.618	**0.728**	**0.728**
d	0.727	0.727	0.77	**1.357**	**0.729**	**0.729**

The results in Table 9 show that EMA may improve, but nonlinear exponential r egularization, which can be easily applied, performs better.

Conclusion : We propose the nonlinear exponential regularization of L1 and L2 f or termed exponential regularization. Also, an exponential moving average regular ization experiment was conducted. We can see that a nonlinear combination impr oves performance. Because it was confirmed that the nonlinear features helped th e model to go to the optimal that the model has efficiently. We experimented wi th nonlinear exponential regularization with fixed reflection and exponential avera ge moving regularization with dynamic reflection with dynamic reflection in the

process of making regularization.

In the future, we want to make sure that the hybrid regularization process is effe ctively reflected in the optimization process rather than the fixed and dynamic re gularization process. Also we will be conducted to the optimal in consideration o f general characteristics.

Through this, this study examined the nonlinear regularization study. Next, we describe the novel auxiliary components to help optimize deep learning model [45].

3.4 Novel Auxiliary Components to Help Optimize Deep Learning Model

Introduction : GAN has been widely applied in various fields. However, mode c ollapsing, vanishing gradient, and catastrophic forgetting are occurring in the train ing process in GAN. First, mode collapsing does not yield general results for un expected inputs because models trained with insufficient data only consider certai n characteristics. Secondly, the vanishing gradient problem occurs while the gener ative adversarial network is training through the distribution of latent variable inp ut. Thirdly, catastrophic forgetting forgets the information of existing data in the process of reflecting new data information.

- We propose three subsidiary component to solve the problems that can occur in this GAN. The three auxiliary components are illustrated in Figure 12.
- The first component is the hybrid regularization method.
- The second component is the hierarchical clustering method.
- The third method is to increase width of the distribution.
- Through the above three factors, it was confirmed that problems and perform ances in GAN can be improved.

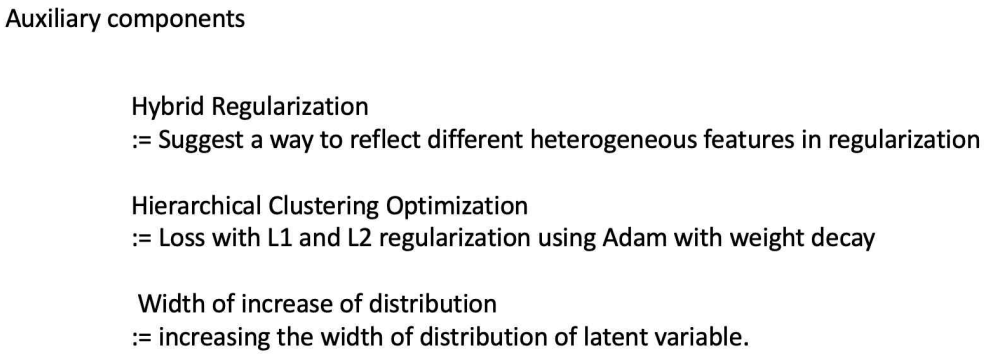


Figure 12. Our Auxiliary Components

Proposal Method: We propose hybrid regularization to reflect heterogeneous features. The hybrid regularization concept is shown below. We definition of hybrid regularization that can reflect the heterogeneous features proposed. We tested using two cases. i.e.) a combination of nonlinear and linear chrematistic, the combination of nonlinear static and nonlinear dynamic chrematistic. We are learning through the regularization of single characteristic proceeds with model optimization with limited scope for model optimization. We have defined the cause of the problem. Through this, we intend to improve the proposed problem through heterogeneous features. This may not be the best point for the model. Therefore, it is necessary to find the optimal point of the model by expanding the range that expresses by reflecting various characteristics. To verify this, we propose a novel hybrid regularization that can reflect characteristics. We intend to verify the reason for reflecting heterogeneous characteristics using three analyzes.

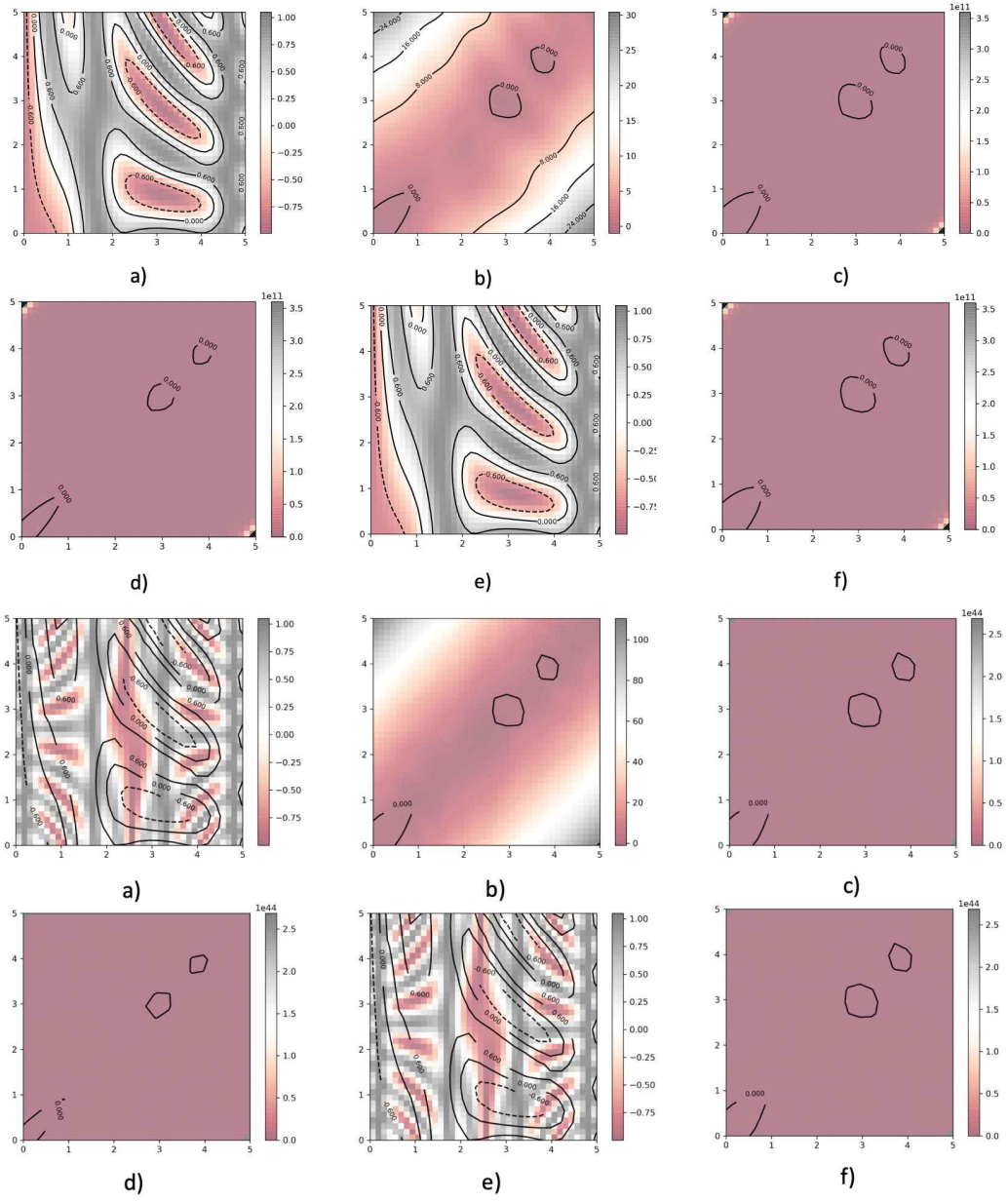


Figure 13. Visualization using contour map to analyze the impact of the propose d method

First, the analysis using contour lines is as follows. Figure 13 shows result that t he generated value is analyzed using the formula np.sin(x) ∗ ∗10 + np.cos(10 + y ∗ x) ∗ np.cos(x). First, in the case of generating an unbalanced value, x

generated 40 values up 0 to 5, and y was analyzed using 40 values up 0 to 5. The result represents the top six of the Figure 2. Second, in the case of generati ng a balanced value, x generated 40 values up -5 to 5, and y was analyzed usin g 40 values up 0 to 5. The results represent in the bottom six of Figure 13

Figure 13(a) expresses the frequency of occurrence of the value in the existing e xperimental equation as a contour line. Figure 13(b) expresses the frequency of o ccurrence of the value as a contour line when applied to the experi- mental form ula using the L1 and L2 regularization addition formulas. Figure 13(c) expresses the frequency of occurrence of the value as a contour line when Nonlinear expo nential regularization applies to the experimental formula. Figure 13(d) expresses the frequency of occurrence of the value as a contour line when Nonlinear expo nential regularization and linear combination L1 and L2 regularization applies to the experimental formula. Figure 13(e) expresses the frequency of occurrence of t he value as a contour line when Exponential average moving regularization is ap plied to the experimental formula. Figure 13(d) expresses the frequency of occurr ence of the value as a contour line when Exponential average moving regularizat ion and nonlinear exponential regularization are applied.

Figure 13, when the regularization is applied, we can see that a lot of values of 0 were generated compared to the existing one. In the case of the picture, Figur e 13(b), values other than 0 are gathered to the upper left and lower right. This seems to be a phenomenon that maximizes the margin of 0 as values other than 0 widen at both ends. Also, in the case of the picture Figure 13(c)(d)(f), these c ontour lines appear, which can be viewed as a local minimum in the hyperplane space of the neural network model. In a situation where a value of 0 occurs a l ot, expanding the range (the size of the contour line) where the local minimum can be seen or finding a range that can be optimized by the model increases the amount of information the model can learn, resulting in better optimization. It se ems that we can go to the branch.

In Figure 13, it was analyzed by generating balanced data and unbalanced data. I n the case of balanced data, it can be seen that the value near 0 is more cluster ed than the contour line. If we explain this in an analogy to the hyperplane in a neural network, it can also be seen that a lot of minimum local areas are genera ted. In addition, when generating positive and negative values, it can be confirme

d that the shape of the contour line is seen as a straight line. This confirmed th at the occurrence of the extreme value was generated as a stronger straight line t han creating a soft boundary line.

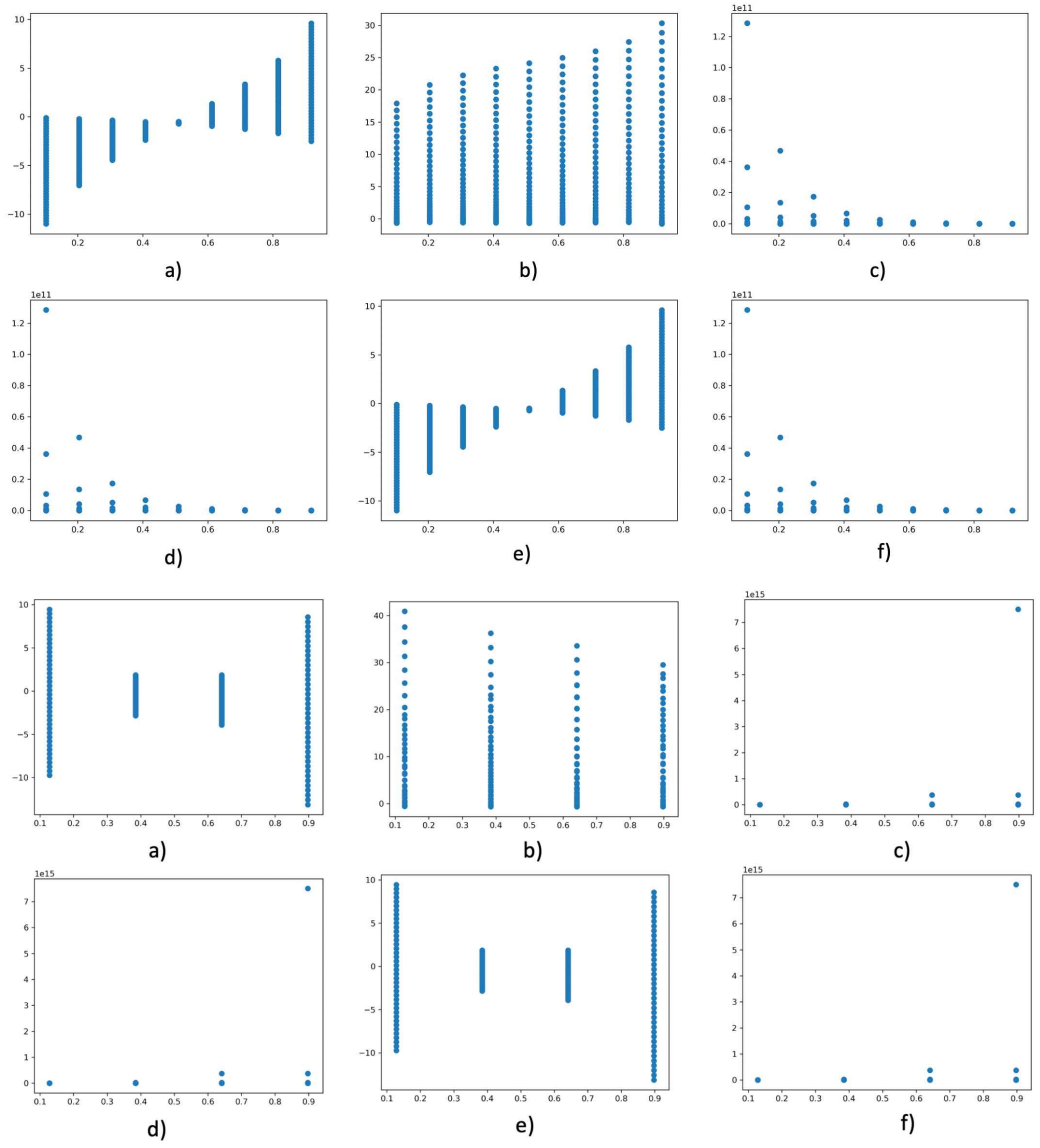


Figure 14. Visualization frequency using loss error to analyze the impact of the proposed method

Second, analysis using frequency of occurrence is as follows. In the Figure 14, e ach regularization method is applied to the cross-entropy. When an unbalanced v

alue generated as input, x generated 40 values up 0 to 5, and y analyzes using 40 values up 0 to 5. In the case of generating a balanced value, x generated 40 values up to -5 to 5, and y was analyzed using 40 values up to -5 to 5. The re sult value output using the above experiment method is as follows. In the case o f the picture in Figure 14, the upper part is the case of using unbalanced data. I t can be seen that the result value generated is between 0 and 1, but 9 specific values generate it. In addition, in the case of Figure 14(c),(d),(f), the frequency o f occurrence largely gather at the value of 0. In the case of Figure 14(a), (e), it was confirmed that the generated value is symmetrically generated based on 0.5 on the x-axis through the opposite sign. The bottom is the case that occurred usi ng balance data. The results show that four results generated on the x-axis. In th e case of Figure 14(a), (e), it appears that the generated value is symmetrical ab out the y-axis. In the case of Figure 14(c), (d), (f), it appears that the value is gathered to the bottom right.

Eventually, the results from the experimental formula can be checked by gatherin g specific result values. This can be confirmed by the phenomenon that the deep learning model constantly learns, and the uniform value constantly learns. Therefo re, it seems that research should be conducted in the future by optimizing genera lization by reflecting various values rather than constantly reflecting specific valu es.

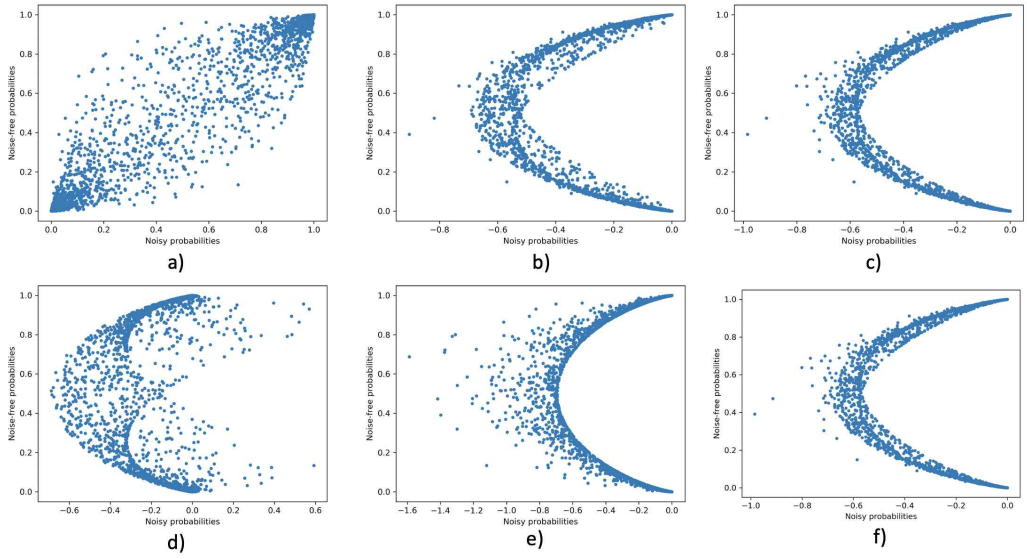


Figure 15. Visualization scatter plot using loss error map to analyze the impact o f the proposed method

Third, the analysis using the LogisticGroupLasso model is as follows. Figure 15 analyzes the results generated using the LogisticGroupLasso model. We tried to a nalyze the model and sparsity values generated through this. The generated x-axis is noisy probabilities, and the y-axis is noise-free probabilities. It was confirmed that the overall generated shape create as the shape of the convex function. It w as confirmed that the degree of clustering of the results generated by applying ea ch regularization method was different. Depending on the regularization method, a variety of non-clustered values generate, but the value of the frequency that is not clustered can be determined as an outlier in the process of learning the mod el, causing learning in the wrong direction. Therefore, it can be confirmed that it is important to design the regularization method efficiently so that sparse values are less likely to occur.

Algorithm 1 $Hybridregularizationcase1$

while $Epoch \neq 0$ **do**
  $Loss \leftarrow Crossentropyloss$
    $+Nonlienar\ regularization$
    $+ L1\ regularization + L2\ regularization$
  $Optimization\ model\ using\ loss\ error$
end while

Algorithm 2 $Hybridregularizationcase2$

while $Epoch \neq 0$ **do**
  $Loss \leftarrow Crossentropy$
    $+Exponential\ average\ moving$
    $+Nonlinear\ exponential\ regularization$
  $Optimization\ model\ using\ loss\ error$
end while

The pseudo-code that applies the hybrid regularization proposed to two cases is a s follows. Pseudo-code 1 and Pseudo-code 2 show how the proposed method is applied to a deep learning model training.

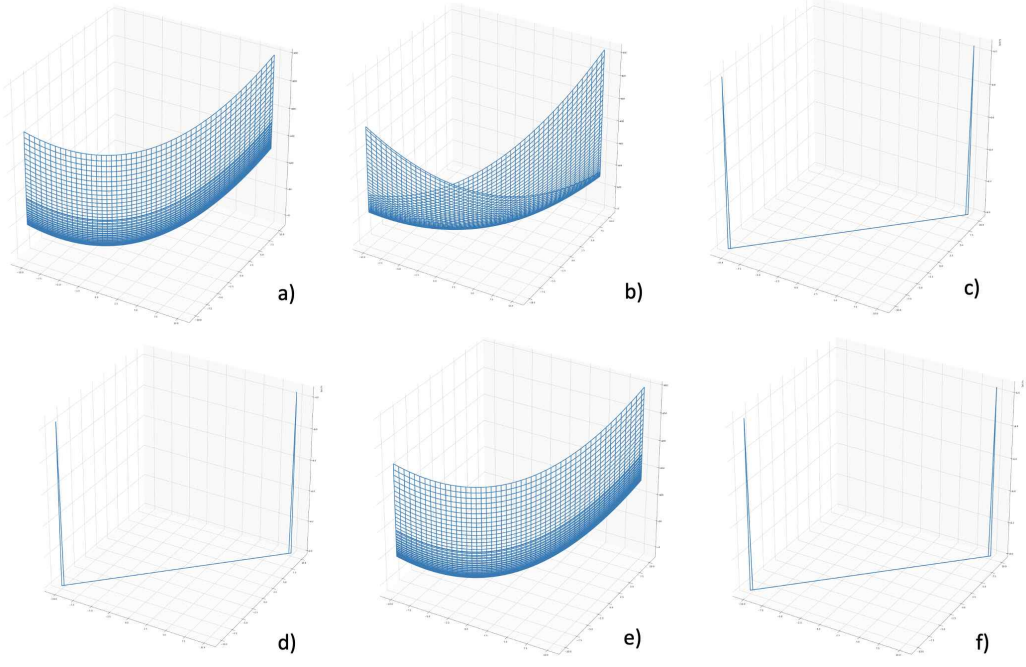


Figure 16. Comparison of the regularization methods tested using visualization

Qualitative analysis of our proposal: When the weight visualization of the nonline ar exponential regularization effect for each regularization is performed, a differen t decision boundary is expressed for each regularization. Depending on the shape of the decision boundary, the deep learning model helps to optimize it stably. W e showed the result of alleviating the optimization problem slightly in the case o f hybrid regularization, which reflects both fixed and dynamic features simultaneo usly. However, when using the hybrid feature, it was confirmed that this model has features that correlate with each other and that the model helps optimize. W hen the features of different features have a shape of features that did not help i n learning the model, the model was less optimized, and the performance deterio rated. In the end, it was confirmed through experiments with various regularizatio ns that it is important to find an adaptive regularization suitable for the model.
In Figure 16, the regularization methods tested are analyzed by using the formula of $x^2 + 4x + y^2 - 6y$. This can be thought of as the hyperplane shape of deep lear ning models. Figure 16(a) shows no normalization experiment, Figure 15(b) show s L1 and L2 regularization, Figure 16(c) shows Nonlinear exponential regularizati on, Figure 16(d) shows nonlinear exponential regularization. L1 and L2 when reg ularization is applied, Figure 16(e) is when Exponential average moving regulariz ation is applied, and Figure 16(f) is when exponential average moving regularizat ion and nonlinear regularization is applied. Through the above results, according to the regularization, the hyperplane space that the model can have a narrow vall ey shape. If it has such a narrow valley shape, it seems that the model can quic kly converge to learn to the optimal point.

Relation analysis: We use five functions to analyze the relationship between the l oss and the regularization function. e.g.) Sqrt, relu function, eLU, bi-relu [36], an d bi-eLU [36]. We experimented with four methods. e.g.) sqrt, relu, eLU, bi-relu, bi-eLU. We adopt five functions on four regularization methods. e.g.) Exponential moving average with linear coefficient, exponential moving average with convex coefficient, hybridv2 with linear coefficient, and hybridv2 with the convex coeffic ient. We tried to confirm that it is effective for deep learning models to reflect meaningful information through analysis of the relationship between the loss and

regularization.

Seed analysis: We analyzed the regularization performance using five seed values. We analysis the experimented using five seed values. e.g.) 1, 250, 500, 777, and 999. We tried to confirm by using a regularization that the quality expressed in the model is changed by the seed value.

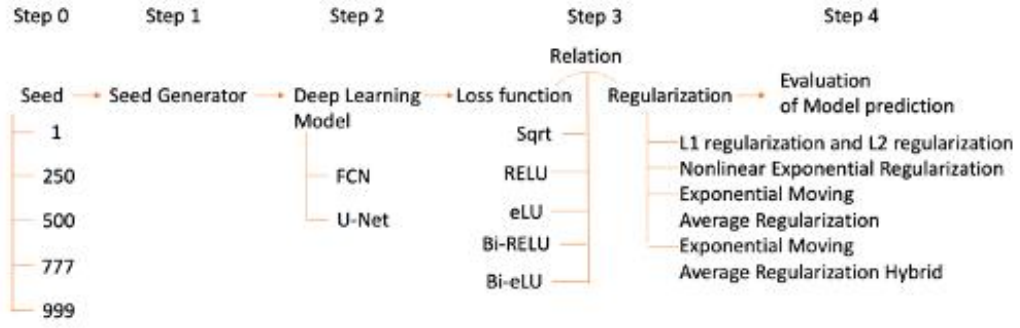


Figure 17. Visualization of experiment system configure

Hybrid regularization experiment system Figure 17 shows the system configuratio n tested. The experiment consists of five stages. Step 0 is an experimental metho d of impact analysis on seed values. Step 1 takes seed values and generates initi al values for the model. The second stage is the FCN and U-Net used in the ex periment. Step 3 is to analyze the association between loss and the regularizatio n. Step 4 is to evaluate the prediction results of the model. To improve the abo ve problems, we propose a hybrid regularization to reflect static and dynamic fea tures.

First, we analyze the relationship between the loss function and the regularizatio n. This is because reflecting

meaningful information on the regularization is a process of optimizing the mode l, and it confirms that it converges quickly through learning with the shortest pat h. It can also prevent going in the wrong direction.

Second, to analyze the relationship between the covariate of regularization, two a nalyzes were used: convex and linear coefficient. Deep learning models generally have hyperplanes with convex characteristics. At this time, the learning position v ibrates while the learned position vibrates during the optimization of the model.

This is to check whether the characteristics of the regularization should have con vex characteristics or whether the model is reflected, including the linear characte ristics, to help optimize the model.
Third, the analysis conduct by setting five seed values using regularization. Beca use the deep learning model train with the initial distribution of the model chang ed according to the initial seed value. Finally, we try to confirm whether the mo del is trained stably by analyzing the regularization according to the initial state of the model.

Hybrid regularization experiment setting : The experiment apply to U-Net and FC N of the semantic segmentation task. We experimented with our proposal with K eras in an Ubuntu environment. We recorded the average value through 5 experi ment iterations. Experiments apply to FCN and U-Net using VOC and ATR data sets in the image segmentation task. Cross entropy loss use for experiments. The applied results evaluate using DSC, F1 Score, IOU, Loss, Precision, and Recall. We analysis the experimented using five seed values. E.g.) 1, 250, 500, 777, and 999. The equation for our hybrid regularization method is as follows. A and B a re the covariates of the regularization.

combination L1 and L2 regularization
with convex coefficient := (1)
A∗L1+(1−A)∗L2

combination L1 and L2 regularization
with linear coefficient := (2)
A ∗ L1 + B ∗ L2

Nonlinear exponential regularization
with linear coefficient (static):= (3) A ∗ L1e(1−A)∗L2

Exponential moving average with linear coefficient (dynamic) :=
Step1:=A∗L1−(1−A)∗L1−B∗L2 (4) Step2 := A ∗ Step1 + B ∗ L2

Exponential moving average with convex coefficient (dynamic) := Step1:=A∗L1−(1−A)

∗L1−B∗L2 Step2 := A ∗ Step1 + ((1 − A) ∗ L2)
(5)

Hybridv2 regularization with convex coefficient
(static and dynamic) := Step1 := A ∗ L1 − (1 − A) ∗ L1 − B ∗ L2
□
Step2:= A∗Step1+((1−A)∗L2) +A ∗ L1e(1−A)∗L2

Hybridv2 regularization with linear coefficient
(static and dynamic) Step1 := A ∗ L1 − B ∗ L1 − B ∗ L2 Step2 := A ∗ Step1 + (B ∗ L2) +A ∗ L1eB∗L2

Hybridv2 regularization with linear coefficient adopted relu
(relation analysis using linear filtering) := Step1 := A ∗ L1 − B ∗ L1 − B ∗ L2 S tep2 := Relu(A ∗ Step1 + (B ∗ L2)) +A ∗ L1eB∗L2

Hybridv2regularization with linear coefficient adopted Bi−eLU
(relation analysis using bi-ploar filtering) := Step1 := A ∗ L1 − B ∗ L1 − B ∗ L2 Step2 := eLU(A ∗ Step1 + (B ∗ L2) −eLU(A ∗ Step1 + (B ∗ L2)) +A ∗ L1eB ∗L2

Formula of regularization method used in experiment: The equation used in the e xperiment is as follows. Equation 1 is the convex combination of L1 and L2 reg ularization. Equation 2 is a linear combination of L1 and L2 regularization. This is to confirm the effect of convex and linear on the relationship between L1 and L2 regularization in Equation 1 and 2. Equation 3 is nonlinear exponential regula rization. This is to confirm the static effect of the nonlinear exponential regulariz ation function. Equation 4 is an exponential moving average with the linear coeff icient. Equation 5 is an exponential moving average with the con- vex coefficien t. Equation 4 and 5 are nonlinear exponential regularization. This is to confirm t he dynamic effect of the convex and linear relationship on the nonlinear exponen tial regularization. Equation 6 is an exponential moving average and nonlinear ex ponential regularization with the convex coefficient. Equation 7 is an exponential moving average and nonlinear exponential regularization with the linear coefficien t. Equation 6 and 7 is an exponential moving average and nonlinear exponential

regularization. This is to confirm the static and dynamic effect of the convex an d linear relationship on the nonlinear exponential regularization. Equation 8 is an exponential moving average, and nonlinear exponential regularization with the line ar coefficient adopts relu. Equation 9 is Exponential moving average with linear coefficient hybrid adopt Bi-eLU. Equation 8 and 9 is the exponential moving ave rage. This is to confirm the static and dynamic characteristic effect of filtering a bout a single filter (6) and bi filter relationship on the nonlinear exponential regu larization. The results obtained from the experiments are as follows. A combinati on of exponential moving average and a combination of L1 and L2 regularizatio n is called hybrid regularization. Combination exponential moving average and no nlinear exponential regularization is called hybrid v2 regularization.

Table 10. Experiment index of Hybrid regularization

Name	Index A
None	Experiment 1
Combination L1 and L2 regularization with convex coefficient	Experiment 2
Combination L1 and L2 regularization with linear coefficient	Experiment 3
Nonlinear exponential regularization with convex coefficient	Experiment 4
Nonlinear exponential regularization with linear coefficient	Experiment 5
Exponential hybrid regularization with convex coefficient	Experiment 6
Exponential hybrid regularization with linear coefficient	Experiment 7
Exponential moving average regulariza tion with convex coefficient adopted no filter	Experiment 8
Exponential moving average regulariza tion with convex coefficient adopted relu	Experiment 9
Exponential moving average regulariza tion with convex coefficient	Experiment 10

adopted eLU	
Exponential moving average regularization with convex coefficient adopted bi-relu	Experiment 11
Exponential moving average regularization with convex coefficient adopted bi-eLU	Experiment 12
Exponential moving average regularization with linear coefficient adopted no filter	Experiment 13
Exponential moving average regularization with linear coefficient adopted relu	Experiment 14
Exponential moving average regularization with linear coefficient adopted eLU	Experiment 15
Exponential moving average regularization with linear coefficient adopted bi-relu	Experiment 16
Exponential moving average regularization with linear coefficient adopted bi-eLU	Experiment 17
Exponential hybridv2 regularization with convex coefficient adopted no filter	Experiment 18
Exponential hybridv2 regularization with convex coefficient adopted relu	Experiment 19
Exponential hybridv2 regularization with convex coefficient adopted elu	Experiment 20
Exponential hybridv2 regularization with convex coefficient adopted bi-relu	Experiment 21
Exponential hybridv2 regularization with convex coefficient adopted bi-elu	Experiment 22
Exponential hybridv2 regularization with convex coefficient adopted no filter	Experiment 23
Exponential hybridv2 regularization wi	Experiment 24

th convex coefficient adopted relu	
Exponential hybridv2 regularization with convex coefficient adopted eLU	Experiment 25
Exponential hybridv2 regularization with convex coefficient adopted bi-relu	Experiment 26
Exponential hybridv2 regularization with convex coefficient adopted bi-eLU	Experiment 27

Table 10 summarizes the index of the experiment used. We tested using 27 regularization equations.

Algorithm 3 *Hierarchical Concurrency Optimization*

while $Epoch \neq 0$ **do**
  $Loss \leftarrow loss$
    $+ L1\,regularization + L2\,regularization$
  $Optimization\,with\,weight\,decay$
end while

Hierarchical concurrency optimization Concurrency control has been applied to improve the performance of models. However, there was no hierarchical simultaneous optimization in the process of optimizing the model. Therefore, we propose a new hierarchical concurrency optimization for training deep learning model optimization. Hierarchical concurrency optimization is composed of two-component. The first component is loss with regularization. The second component is optimization with weight decay. The two-component are applied simultaneously to different model positions. The proposed method express in pseudo-code. It applied weight decay to optimization in the process of optimizing the model hierarchically with L1 regularization and L2 regularization. Algorithm 3 is the pseudo-code of hierarchical optimization proposed. The advantages of this hierarchical concurrency optimization method are as follows. Firstly, it can be reconfigured in a simpler, smaller step in the optimization phase. It is easy to understand deep learning models and to design and implement models. Second, it provides a standard interface that each method can work with. The independence of each method simplifies the method of the whole method. Third, when you need to correct or improve the functional errors of each method, you can complete it by replacing only that method without having to rewrite the entire deep learning model. There is an advant

age that the internal change of one method does not affect the operation of anot her method.

It is necessary to provide hierarchical concurrency optimization in a situation whe re multiple data accesses the model simultaneously while the deep learning model learns multiple data. This should make it possible to consistently reference the in formation inside the model in the process of updating the values in the model. Because, in the process of requiring precise calculation using deep learning mode l, two problems can occur. First, when different transactions perform update oper ations continuously in a situation where loss and optimization execute simultaneo usly, a phenomenon in which the previously executed loss operation is overwritte n may occur. Second, optimization may be executed while loss is being execute d, and information inside the model may be broken. Therefore, we propose a hie rarchical concurrency optimization that can alleviate these problems and verify it with performance evaluation for image generation. When the method proposed is applied, in the process of updating the weight of the model, it is possible to gen erate an image with high resolution through precise calculation by simultaneously controlling the updating statement inside the model. Also, it is possible to acquir e a phenomenon in which the learning error of the model is reduced compared t o the existing one. The proposed method is not only applicable to GAN, but we can be applied to all deep learning models.

Effect of model optimization We will divide and explain two cases of informatio n in the process of model optimization. e.g.) if they have the same characteristic s or if they have different characteristics.

First, the case has the same characteristics will be described. Here, the same cha racteristic means information that is semantically similar and can be viewed as th e same characteristic. At this time, the characteristics produced through a similar degree can be viewed as a group or as a single information. When information generated in the model is similar, the overlap may occur between the informatio n, and convergence may occur more quickly. In contrast, when the information g enerated in the model is not similar, the margin between the two pieces of infor mation is maximized to obtain a criterion for maximizing the information.

Second, the case of having different characteristics will be described. Here, the d

ifferent characteristics are cases where they have completely different characteristi cs, such as examples of fixed characteristics and dynamic characteristics. In the c ase of reflecting the fixed and dynamic characteristics of the example in the mod el at the same time, by reflecting the heterogeneous characteristics, the model ac quires robust generalization to various features.

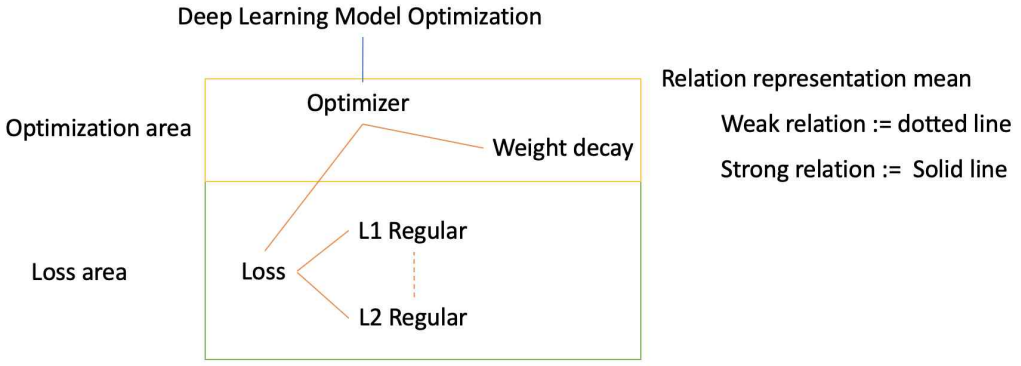


Figure 18. Relation analysis in optimization

We will explain the relationship between the components used by the deep learni ng model for optimization through two examples. Figure 18 is expressed by divi ding the components used to optimize the model into two regions. In the optimiz ation area, there are components of the optimizer and weigh decay method. In th e Loss area, there are components of loss and L1 and L2 regularization. For eac h component, the relationship was expressed in two ways. When there is a direct relationship, it is expressed as a solid line as a strong relationship. And if there is an indirect relationship, it is expressed as a dotted line as a weak relationship. The components used in the experiment were expressed in a form similar to a g raph network or tree. After confirming the influence on each component through visualizing the relationship between each component helps the actual model to op timize. However, if the model is not a component that does not help to optimiz e, you need to insert other components to fix the model's components so that th e model can optimize. It is important to extract the maximum performance throu gh the optimization of the model through the correlation analysis for each of the se components.

Effect of image generation The environment for generating images has been studi ed a lot in the field of existing graphics research. Many studies have been cond ucted to make images realistic in the process of generating images in the above research field. Also, in recent deep learning, the field of graphics research using deep learning has been actively researched to make the image similar to the real thing [46]. It argues that the creation of an image in real life should be applied to the generator to accurately determine the real image and real data in the Gene rative adversarial network. In the process of generating an image using deep lear ning, the hypothesis was that by removing the noise by analyzing the noise gene rating part in terms of temporal information reflected in the generator, the image could be generated as a real image with high resolution. In this study, we first t ry to explain this using the graph coloring example, which is well known in exi sting algorithms.

First, the image is composed of objects, cases of similar things grouped based o n low-level information. The information in the image can be visualized using th e form of a tree. If you visualize using the tree form, you can do an overall to pic about the image [47]. Through this, it is possible to understand the image. A lso, to create semantic images, associations are created through pre-defined infor mation. Therefore, the relationship between the image information and the tree or graph network is expressed. This is compared to the generator of the Generative Adversarial Network (GAN) experiment. This was visualized in the picture 8. Fir st, the hyper parameter setting consists of the potential variable space and the ba tch size. Second, the image generator consists of the generator of the GAN. Thir d, the image created from the constructor is displayed. Fourth, the generated ima ge is measured using PSNR. Three problems can occur in the process of creatin g an existing image. First, if particle noise or the like is not removed, blurring of the image may occur. Second, in the image rendering process, information nee ds to be created semantically and graphically. If it is not created correctly, strang e pictures may be created. Third, as the number of batch sizes and the informati on on the grid generated in the latent variable space increase, complexity increas es. This increases the complexity of the information and the amount of informati on that must be reflected from the viewpoint of the generator. Since the amount of information increases, if the amount of information is not optimized, a proble

m that the image generation performance of the model is low may occur.
In addition to the previous problems, to create a realistic image during the image generation process, a sophisticated generator was designed in the generator during the image generation process [48]. It is essential to increase the performance by making the generator complex, but we thought it was essential to select the opti mal performance of the model through simple ideas like the proposed method. T herefore, it was thought that problems generated in the model could be reduced even if the hyper parameter tunning is efficiently performed in the GAN using a generator. Therefore, the analysis of the impact on image creation through hyper parameter tunning of the model is as follows.
Effect of hyper parameter fine tuning In the initial learning process, the hyper pa rameter setting is essential for deep learning [49]. In the initial setting, in the pr ocess of model learning, the simulated annealing process of the model by the hy per parameter occurs gradually and precisely. When the information of the existin g learning data reflect in this way, the information generates in the model, and t he adaptation of the new data is more stably generated, thereby reducing the cat astrophic problem, which is an effect of forgetting the existing data [50]. Therefo re, it can learn more stably than the conventional learning method. As a disadva ntage, it seems that the learning process may take longer than the existing learni ng method at the optimal point of the model. However, for the deep learning m odel to go in the direction of the life long model, it seems to be important to a daptively modify the hyper parameter whenever an input event occurs so that the model can stably reflect new information. Through this, the process of fine- tuni ng the hyper parameters of the model is essential in the process of learning the pre-trained model that has already trained, but in the initial learning process, the deep-learning model fine-tuning the hyper parameters of the model to learn as a sustainable model. We can see the possibility that the process is essential.

Hierarchical concurrency optimization experimental setting: We conducted experim ents using MNIST and Fashion MNIST. The results of the model evaluate using two PSNR and MSE. We experimented with four losses (i.e., Cross entropy, Lea st-squares loss, Smooth loss, and Focal loss) using Vanilla GAN and LSGAN r esults using two generative adversarial networks. The average value of the experi

ment describes two times.

Figure 19 is the experimental configuration system. Step 0 is to set the number of batch sizes of learning data. Step 1 is the initial potential variable size setting of the Generative adversarial network. Step 2 is LSGAN and Vanilla GAN, whic h are the models used in the experiment. Step 3 is the loss setting, the rule for training the model. Here, origin loss means cross-entropy for Vanilla GAN and Least square loss for LSGAN. Step 4 is two evaluation indicators for evaluating the generated image and model error. PSNR is an index for evaluating the resolu tion by applying to the generated image. Also, MSE (Mean Square Error) is an evaluation index for measuring errors in the difference between the predicted val ue and the actual value of the model.

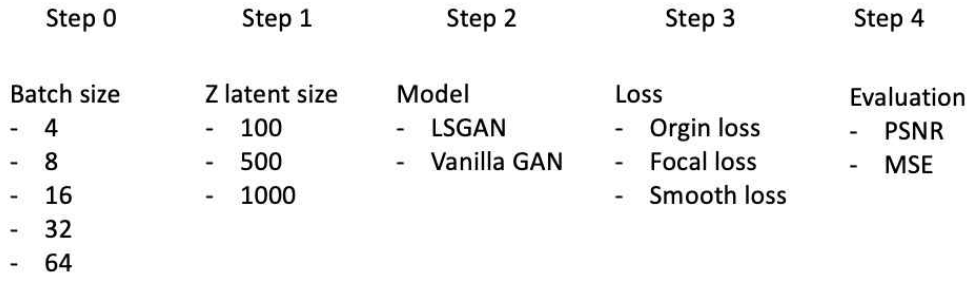


Figure 19. Experimental System

Width of distribution of latent variable : We observed that the effect of the distr ibution of latent variable on the GAN is related to three existing problems. Ther efore, To confirm the effect of the distribution of latent variable on the GAN thr ough the setting of distribution of a latent variable, the experimental method was first performed by increasing the width of the distribution of the latent variable. In the analysis method, according to the increasing width of the distribution. The refore, the experiment carries out by increasing the width of the distribution of t he latent variable. Since changing the distribution of the latent variable directly t o find the cause can identify the problem, it is essential to change the distributio n of the latent variable directly.

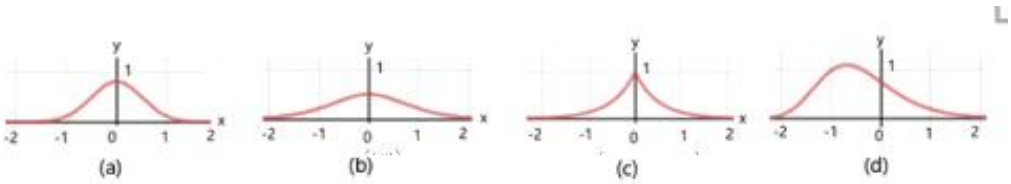


Figure 20 Four distribution visualizations : (a) Laplace distribution, (b) logistic di stribution, (c) Normal distribution, and (d) Gumbel distribution

Figure 20 shows the visualization of four distributions. Figure 20(a) shows the L aplace distribution, Figure 20(b) shows the logistic distribution, Figure 20(c) sho ws the normal distribution, and Figure 20(d) shows the Gumbel distribution.

In Table 11, x is the distribution input, m is the mean of the distribution input, s is the standard deviation, and $\frac{x-m}{B}$ in the Gumbel distribution of Table 11 (d), B is the scale parameter. The distribution shown in Figure 20 is the distribu tion used in the experiment with increasing the width of distribution of latent var iable to show three problems. It was confirmed through experiments with increasi ng distribution width that the effects of the setting of distribution of latent variab les were related to three existing problems. First, mode collapsing use to show t he effect of increasing the width of the distribution of the latent variable. To ver ify, we experimented with Unrolled GAN [51], which can visualize mode collaps ing.

Table 11. Comparison of effects on Unrolled GAN training by increasing width of initial latent variable

	Discriminator for real image	Discriminator loss for fake images	Loss of Generator
a) Laplace distribution	0.682	0.687	0.715
b) logistic distribution	0.682	0.694	0.711

c) Normal distribution	0.676	0.689	0.714
d) Gumbel distribution	0.669	0.697	0.741

The reason for using mode collapsing in the analysis is that it can intuitively check the effect of training on the setting of distribution of the latent variable while tracking the degree of reflection on the distribution. Also, the shape of the distribution of the latent variable can be interpreted.

The results of verifying the effect on the training of Unrolled GAN by increasing the width of the distribution of the latent variable shown in Figure 21.

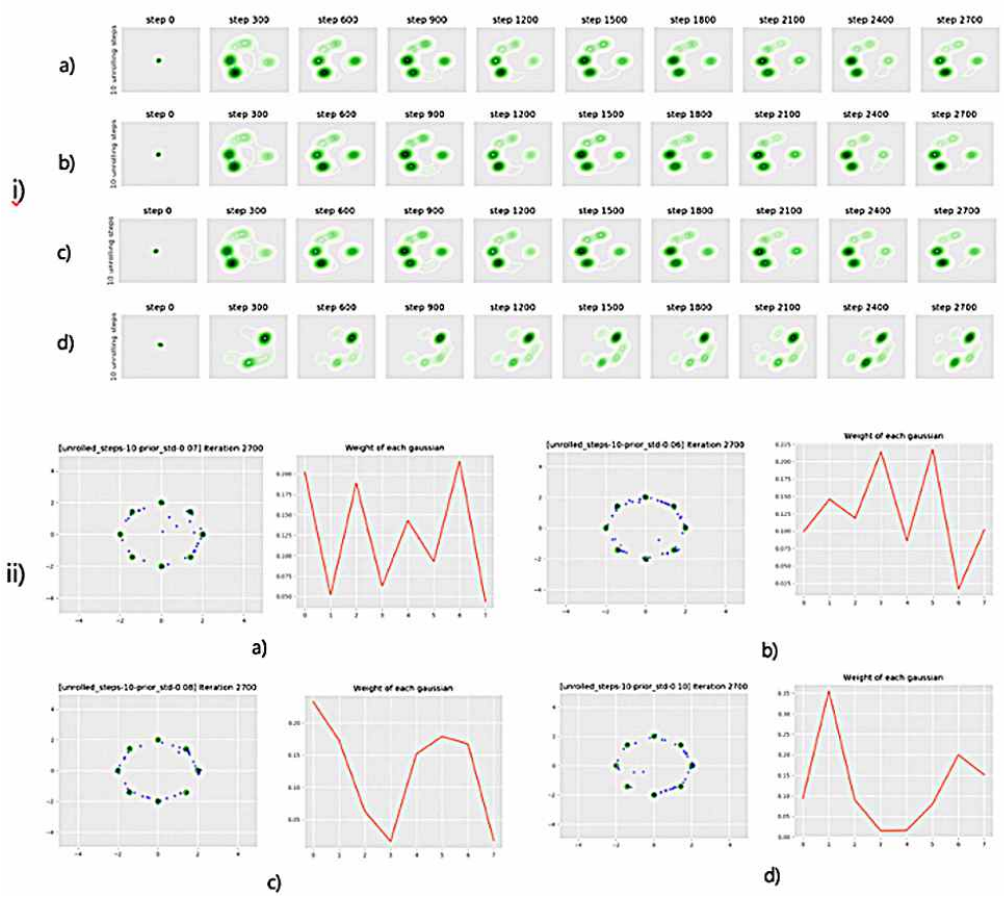


Figure 21. Mode collapse visualization of four distributions using Unrolled GAN, i) visualization of eight distributions during the training process, and ii) visualization of eight distributions after training

The discriminator loss for the real image is the loss value when the real data is judged. The discriminator loss for a fake image is the loss value when judging the data produced by the generator. The loss of the generator is the loss value when the generator creates a new image with a variable as input. As the width of distribution increases, the loss value of the generator increases, and the discriminator loss for the real image decreases. This means that as the width of the distribution increases, it becomes possible to distinguish between real and

fake images more accurately. The distributions in Figure 21(i)(a)-(c) show the concentration of the distribution in the lower-left corner. However, the distribution in Figure 21(i)(d) shows the density of distribution concentrated on the bottom right and top. This means that the training tendency changes when the width of the distribution exceeds a certain width. The red line in Figure 21(ii) is a visualization of the weight distribution of the Unrolled GAN. As the width of the distribution gets wide, the distribution of weights in the Unrolled GAN tends to be softer. The tendency to soften is that the gradient information reflected in the existing distribution is reflected by the change of the smooth gradient information. In Figure 21(ii)(d), we can see a lot of blue dots gathered in the lower right corner. This is because when the width of the distribution exceeds a certain width, the training tendency is changed to reflect the information of the inclination of the distribution as the information of the constant inclination. When the width of the distribution goes beyond a certain level, the information of the gradient is gathered into a certain space, so it is confirmed that the GAN is stable in training. The experiment of increasing the width of the distribution means that the existing training can be used as a way to go stably by changing the direction. We confirmed that the GAN model has an important influence on training according to the influence of the setting of the distribution of the latent variable.

In measuring the effect on the GAN according to the setting of distribution of a latent variable, it inspires that the interior of the model interprets in terms of generalization by inputting the distribution with generalization characteristics. Information that can interpret the model information is expressed differently according to the expressive power that interprets according to the number of parameters of the discriminator of the GAN model. This is because the

expression of the trained feature is trained in various ways according to the discriminator's performance. Therefore, the result of the image generated by reducing the influence on the distribution through the averaging and making the generalized distribution is to check the shape of the pattern according to the clustering degree of pixels. Analyze the shape of the pattern and analyze the effect of the pattern to show the possibility of being used as a way to interpret the model. Since the process of selecting the distribution is necessary, we tried to identify the generation pattern that responds to the generalization aspect by inputting the distribution having the generalization characteristic in the distribution.

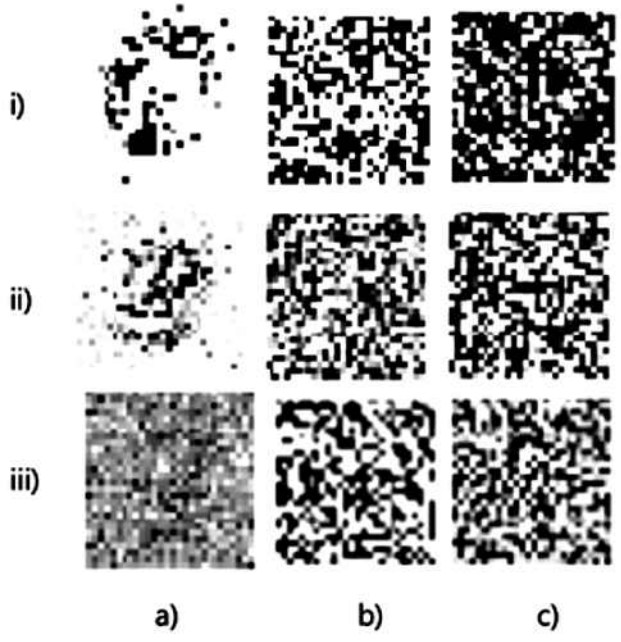


Figure 22. Comparison of GAN discriminator parameters

In Figure 22(i), we experimented with a discriminator consisting of dropouts on three layers 1024, 512, and 256 to the existing distribution. In Figure 22(ii),

we applied the log method to the existing distribution and the discriminator consisting of dropouts on three layers 512, 512, and 256. In Figure 22(iii), we experimented with the logarithmic and 0.1 scale multiplying method of the existing distribution and the discriminator consisting of three layers (512, 512, and 256). Figure 22(a)-(c) shows the experimental results of increasing the distribution width using GAN. The GAN model was used to identify the generation pattern. GAN not well trained. Therefore, the experiment conducts using GAN because the effect of distribution change clearly shows. The distributions used to increase the distribution width shows in Figure 22(a) Normal distribution, Figure 22(b) logistic distribution, and Figure 22(c) Gumbel distribution. The log applied for the experiment shows that when the input value is 0 1, the smaller the input value, the smaller the output value. It shows the effect of separating and discarding rarely generated values from the training data. Also, as the input value increases, the output value does not increase in proportion but increases slightly. This phenomenon has the effect of averaging the input data. By averaging and reducing the impact on the distribution of the latent variable and creating a generalized distribution, the results of the generated images attempted to confirm the pattern shape according to the degree of clustering. We also want to confirm that the characteristics of the main components reflect through the characteristics of the log. We experimented with the effect of maintaining the pixel color information and the clustering of the pixel distribution according to the distribution of latent variable change by reducing the number of parameters of the GAN discriminator. In Figure 22(i), the discriminator consists of three layers 1024, 512, and 512. Figure 22(i)(b) Logistics and Figure 22(i)(c) Gumbel distribution, the pattern produced by the experiment, does not have a human-understandable pattern. However, the pattern

is important to interpret from the standpoint of the model. The above results show that as the width of the distribution of the latent variable increases, a pattern is created through the combinatorial optimization of information that reflects the characteristics of the existing data. In addition, when the number of discriminator parameters is analyzed by reducing the weight, it is confirmed that the principal component characteristics of the generated image are represented and generated. Second, the effect of increasing the width of the distribution of the latent variable is shown by using a vanishing gradient. The loss of the trained model use to confirm the importance of the setting of the distribution of the latent variable.

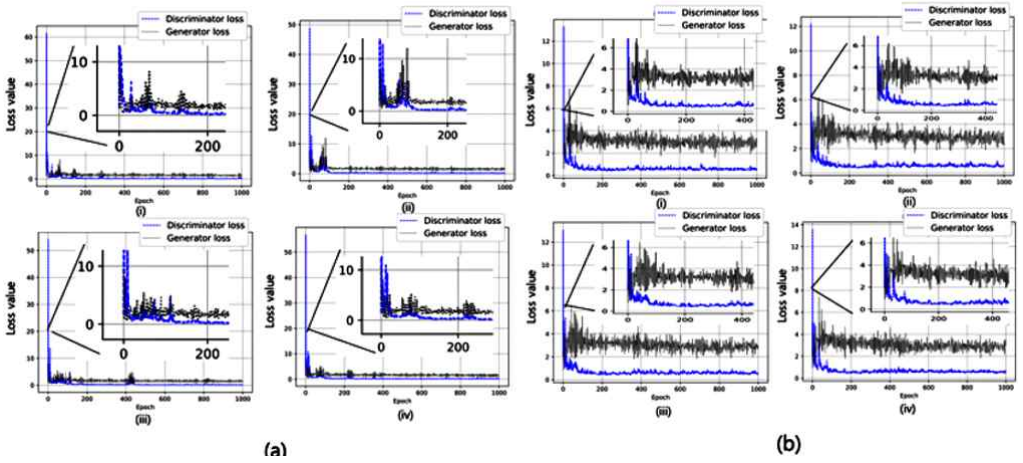


Figure 23. Variation of loss value analysis of vanishing gradient using four distributions in LSGAN, (a) MSE loss, (b) Hinge loss, (i) Normal distribution, (ii) logistic distribution, (iii) Laplace distribution, and (iv) Gumbel distribution

We are applying these four distributions to the LSGAN, the loss results of the model shown in Figure 23. Figure 23 shows the results of the loss change

analysis of vanishing gradient using four distributions in LSGAN. As shown in Figure 23(a), the loss variation in the Gumbel distribution is very similar to the Normal distribution. Also, the results in Figure 23(b) show that the Gumbel distribution has the smallest loss variation. The small fluctuations in the loss width of the generator are shown by maintaining a constant interval through the information of the gradient that does not disappear as the model is repeatedly trained.

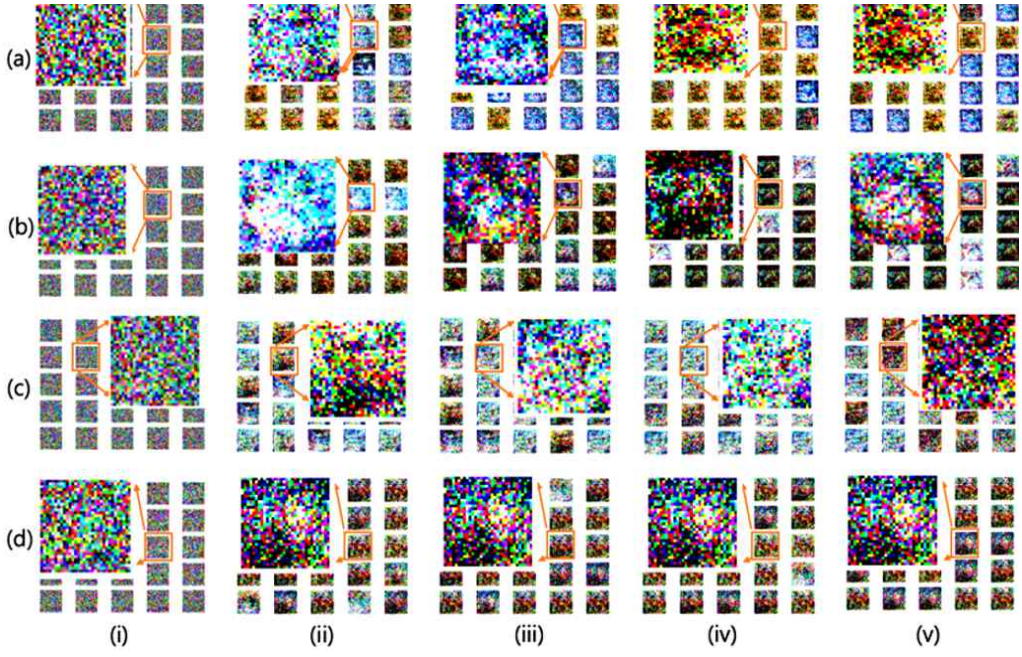


Figure 24. Stability analysis for pixel location and catastrophic forgetting where color information can be forgotten in test images according to random training data selection and training for GAN using size 8, a) Normal distribution, b) logistic distribution, c) Laplace distribution, d) Gumbel distribution, i) 0 epoch, ii) 250 epoch, iii) 500 epoch, iv) 750 epoch, and v) 1000 epoch

Third, the effect of increasing the width of the distribution of the latent variable shown by catastrophic forgetting. The importance of the distribution is

verified by sampling the generated results of the trained model at equal intervals as the width of the distribution of latent variable increases. Figure 24 shows an equally spaced image of a training procedure using randomly selected eight batch sizes from GAN with hinge loss. As the width of the distribution of the latent variable increases, it confirms that better performance maintains without forgetting the color information trained in the same parameter space. In particular, it sees that the result of the Gumbel distribution is well maintained without forgetting the color information acquired in the same parameter space. Figure 24 shows an equally space an equally spaced image of a training procedure using randomly selected 8 batch sizes from GAN with hinge loss. As the width of distribution increases, it is confirmed that better performance is maintained without forgetting the color information trained in the same parameter space. In particular, it can be seen that the result of the Gumbel distribution is well maintained without forgetting the color information trained in the same parameter space. This confirms that catastrophic forgetting can be reduced by distribution selection.

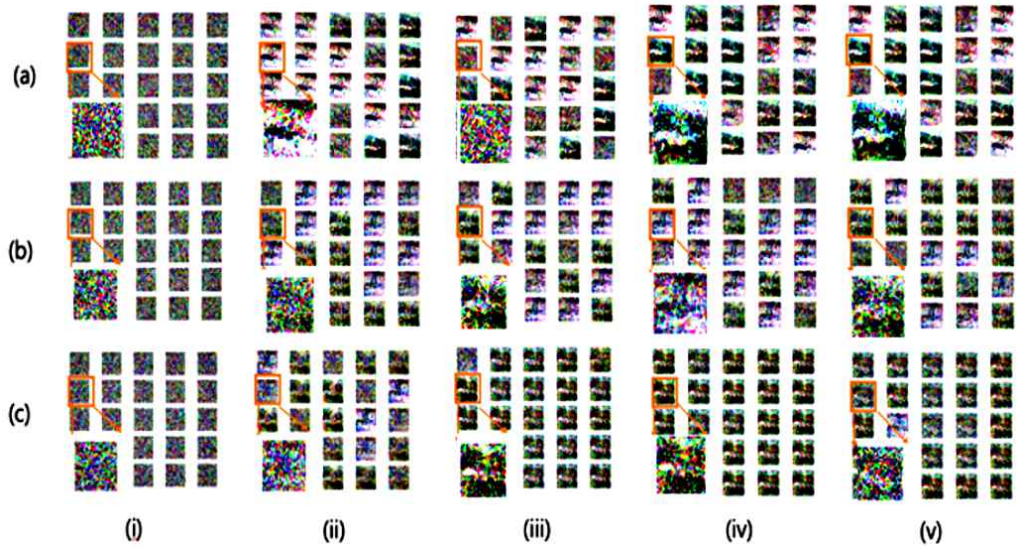


Figure 25. Spatio-temporal analysis of color space mapping of latent variable based on the relationship between batch size and distribution size for the performance of generated images during hinge loss on GAN training, i) static selection batch size 2, ii) static selection batch size 8, a) 100 latent variable sizes, b) 500 latent variable sizes, and c) 1000 latent variable sizes

Figure 25 shows the spatio temporal analysis of color information mapping in the GAN. Spatio-temporal analysis of color information in each distribution was performed with cifar10 data for each batch size and potential variable size. Gradually decreasing the size of the latent variable tends to gather and train important information from the data. Given the change over time, the general characteristics of the latent variable of the input tend to remain large. However, in the case of color information, it can be seen that the smaller the latent variable, the smaller the change in color information. As a result of spatial analysis over time, it has been found that general features are more resistant to size changes. However, as the batch size changes, the color information feature changes more often. In other words, catastrophic forgetting of existing deep training models often occurs when the distribution space is small.

Experimental Setting: We examined the importance of the distribution using three problems of experiments with increasing the width of the distribution. By confirming and observing the importance of the distribution, we propose a distribution in which the Gumbel distribution helps the GAN image generation performance. To verify this, we experiment with four distribution and five-generative adversarial networks: LSGAN, GAN, cGAN [52], ACGAN [53], and semiGAN [54]. The CIFAR-10 and CIFAR-100 data sets were also used to test for loss of MSE and loss of hinge.

Experiment Result: First, the experimental results of the first component, hybrid regularization, are as follows. The order of the results of the bar graph listed in the order of index in Table 10. Also, the standard deviation was drawn on a bar graph to express the degree of dispersion of each method. The loss showed the most stable error performance when convex coefficients were used in hybrid v2, reflecting nonlinear fixed and nonlinear dynamic information. And in terms of performance, hybrid v1 combining exponential moving average and combination L1 and L2 regularization showed the best performance. Through this, it confirmed that the deep learning model was stably trained or showed the high performance to reflect the hybrid characteristics.

It was confirmed that the deep learning model could learn most stably by reflecting the non-linear dynamic and fixed features. We confirmed that reflecting the non-linear dynamics and linear fixed features is higher than other regularization methods in terms of performance.

Table 12. Experiment result of regularization using U-Net on ATR dataset with Seed 250

Regularization	DSC	F1 Score	IOU	Loss	Precision	Recall
Experiment1	0.724	0.724	0.768	1.072	0.724	0.724
Experiment2	0.723	0.723	0.767	1.525	0.723	0.723
Experiment3	0.724	0.724	0.768	1.398	0.724	0.724
Experiment4	0.724	0.724	0.768	1.364	0.724	0.724
Experiment5	0.724	0.724	0.767	1.287	0.724	0.724
Experiment6	0.723	0.723	0.767	1.222	0.723	0.723
Experiment7	0.724	0.724	0.768	1.278	0.724	0.724
Experimen8	0.724	0.724	0.768	1.255	0.724	0.724
Experiment9	0.724	0.724	0.768	1.228	0.724	0.724
Experiment10	0.724	0.724	0.768	1.237	0.724	0.724
Experiment11	0.724	0.724	0.768	1.195	0.724	0.724
Experiment12	0.724	0.724	0.768	1.164	0.724	0.724
Experiment13	0.724	0.724	0.768	1.136	0.724	0.724

Experiment14	0.724	0.724	0.768	1.112	0.724	0.724
Experiment15	0.724	0.724	0.768	1.115	0.724	0.724
Experiment16	0.724	0.724	0.768	1.095	0.724	0.724
Experiment17	0.724	0.724	0.768	1.08	0.724	0.724
Experiment18	0.725	0.724	0.768	1.075	0.724	0.724
Experiment19	0.725	0.725	0.768	1.09	0.725	0.725
Experiment20	0.725	0.725	0.768	1.077	0.725	0.725
Experiment21	0.725	0.725	0.768	1.069	0.725	0.725
Experiment22	0.725	0.725	0.768	1.07	0.725	0.725
Experiment23	0.725	0.725	0.768	1.077	0.725	0.725
Experiment24	0.725	0.725	0.768	1.07	0.725	0.725
Experiment25	0.725	0.725	0.768	1.052	0.725	0.725
Experiment26	0.725	0.725	0.768	1.044	0.725	0.725
Experiment27	0.725	0.725	0.768	1.042	0.725	0.725

Table 12 shows the performance of the experiments to verify the proposed method tested. As a result of Table 11, it was confirmed that the hybrid regularization proposed improved 0.1 % in terms of segmentation performance. In terms of loss, the best experiment, compared to the previous one, showed an error reduction of 3 % or more. At this time, the hybrid regularization v2 showed the lowest performance through non-linear fixed features and dynamic features through bi-directional non-linear filtering. This seems to be that the non-linear features adequately reflected in the model learning process in the same non-linear environment.

As a result of analyzing the relationship between the loss and the regularization function in Table 12, it was confirmed that the loss value is slightly higher when simple addition performed. What reflected through addition is that the complexity of information is somewhat improved through the superposition of information, so the loss value seems to be rather high. However, as a result of using RELU, eLU, Bi-relu, and Bi-eLU to reflect the degree of normalization, filtering using RELU reflects the linear characteristics, so the loss

is reduced compared to the conventional addition. Also, as a result of filtering using eLU, it can be seen that non-linear features reflected more stably than linear features. And bi-activation was applied to learn by extracting essential features by reflecting the information of bi-polar. The bi-relu linearly filtered effect shows a learning loss value similar to a single non-linear filtering result. Also, it confirmed that bi-eLU shows a simple linear effect when the non-linearly filtered effect is not right in both directions but shows a lower error value when finding an essential feature through non-linear filtering.

Additionally, the regularization performance, according to the change of seed value, was analyzed. The results of the regularization analysis through the change of the seed value were included in the supplementary. As a result of analyzing the performance of regularization by changing the seed value, it shows the effect that the performance of regularization changes somewhat depending on the initial seed occurrence. Because the learning of the deep learning model changes with probability by the initial state of the deep learning model, it shows a non-uniform trend. Through this, it can be confirmed that it is essential for the deep learning model to create an initial state that can stably reflect information. Also, it seems necessary to study the robust regularization even in this initial state.

Hybrid test results show better results when the features that dynamically reflected, and the features that dynamically reflected are effectively mixed and reflected. But in all cases, it does not improve. To further develop the hybrid regularization, it seems necessary to make an optimal combination of static and dynamic features. In the case of Bi-eLU, It confirms that the deletion of shared information that can be reflected in the loss and non-linear sparse information reflected in the model optimization. Hybrid regularization finds a lower error and

slightly improved performance as a result of the seed value experiment. Seed test results show that the performance varies greatly depending on the initial state of the model. This confirms the need for a study of robust regularization in the initial model state.

The experimental result of the second component, Hybrid concurrency optimization, is as follows. The results of the experiment for hierarchical concurrency optimization described the average values of Vanilla GAN and LSGAN. We show the result when weight decay and L1 and L2 regularization are not applied.

We show the result when only weight decay is applied. When only weight decay is applied, it can be seen that the overall image generation performance is slightly improved. This is because Adam optimization learns as the optimization shift reduces by zooming in on the weight decay value, so it seems that the performance improves by optimizing the generator during the image generation process.

This was compared not only by using image generation performance, but also by using error values. This was a comparative analysis of discriminator errors. We show the result when weight decay and L1 and L2 regularization are not applied. When this applies to weight decay, the performance is similar to the previous result. When weight decay and L1 and L2 regularization are applied simultaneously, the error value reduces. However, if the coefficients of L1 and L2 regularization strongly enter according to the potential variable space size, data type, and batch size number. It confirms that the error performance slightly improved. Therefore, it was confirmed that it is important to find the regularization coefficient adaptively according to the hyperparameters. Also, to check the influence of weight decay when there is regularization, weight decay,

and no weight decay were tested and compared in the case of L1 0.25 and L2 0.25 regularization. When comparing all the results of the experiment, there was no change. This seems to have obtained a similar error value because it has been reached through regularization on a similar optimization point. Therefore, the hierarchical concurrency optimization method seems to have a difference in performance depending on the characteristics of the regularization coefficient.

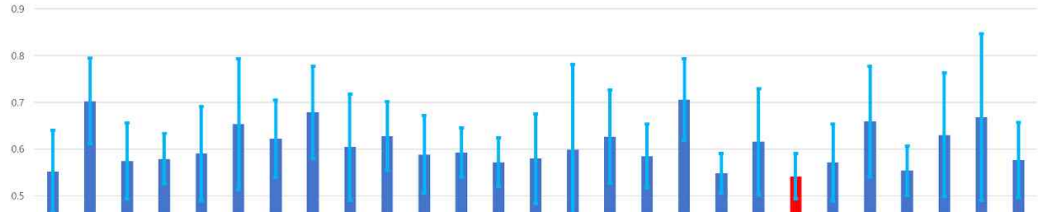


Figure 26. Comparison of the regularization methods tested using loss

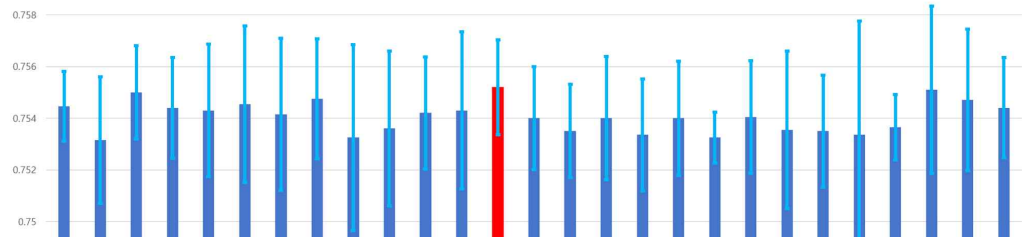
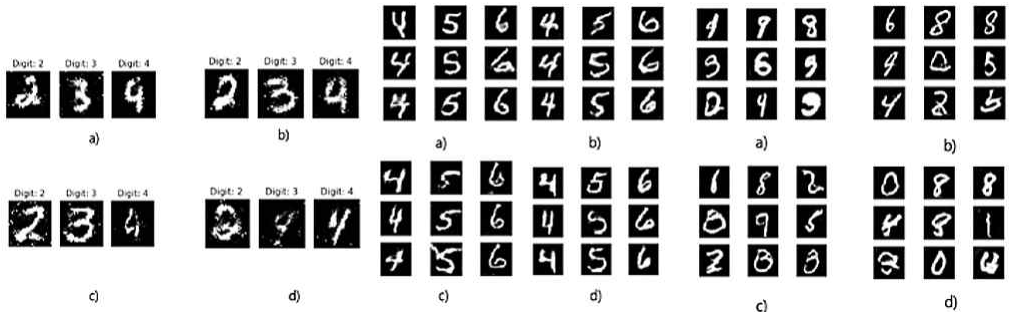


Figure 27. Comparison of the regularization methods tested using IOU

Third, the experimental results of the third component, the width of the distribution of a latent variable, are as follows. We tested the mean values of PSNR (Peak Signal-to-Noise Ratio) and MSE (Mean Square Error) of five experiments for LSGAN and GAN using cross-entropy and hinge loss based on four batch sizes with CIFAR-10 and 100 datasets. (Batch size 256, batch size 128, batch size 64, and batch size 32). The other results of the other three

distributions are similar to the Gumbel distribution. The difference of the reference distribution values was obtained from the experimental distribution values when calculating the improvement values. We then divided by the difference and used the improvement using the product of 100. The reference distribution used as a reference is the Normal distribution. Experimental distribution values are Laplace, logistic, and Gumbel. For the quantitative analysis of the generated image, we use the mean pixel value of the generated image and the standard deviation of the pixels. The PSNR and MSE of the Normal distribution are μ= 47.923, σ = 0.7442 and μ = 103.4947, σ = 14.152, respectively. The PSNR and MSE of the logistic distribution are μ = 47.938, σ= 0.760 and μ = 103.152, σ = 14.460, respectively. The PSNR and MSE of the Laplace distribution are μ = 47.923, σ = 0.716 and μ = 103.412, σ = 13.755, respectively. The PSNR and MSE of the Gumbel distribution are μ= 47.959, σ = 0.750 and μ = 102.651, σ = 14.108, respectively. Gumbel distribution performs better than others. The distribution does not have a significant impact on performance, but this performance difference indicates that the Gumbel distribution certainly helps to ensure stable training. We also visualized the mean and standard deviation to confirm the variance of each method and found similar variances in all four distributions. In our experiments, we often experience catastrophic forgetting in discriminator parameters when the width of the distribution is small. Also, the results of images obtained from cGAN, ACGAN, and semiGAN without qualitative influences of the discriminator parameters proposed to provide qualitative analysis of the four distributions in Figure 28.



Figures 28. Qualitative analysis of images generated for stable training in generator of GAN using a series of conditional GAN. i) cGAN, ii) ACGAN, iii) semi GAN, a) Normal distribution, b) Laplace distribution, c) logistic distribution, and d) Gumbel distribution

We can see that the image generated as a result of the Gumbel distribution in Figure 28 creates a smooth shape. This is because the curve shape of the Gumbel distribution has a smooth elliptic curve shape than other distributions, so the curve shape shows a smoother image. Figure 24-28. As can be seen, most performance or generated images will improve performance depending on the trend of Normal distribution ≥ logistic distribution or Laplace distribution ≥ Gumbel distribution. The influence of the distribution width causes this phenomenon. The density of the distribution effect depends on the size of the bell-shaped region of the distribution. It can see that both tails of the distribution become thicker to better reflect the information. The cumulative density function for the fastest cumulative speed at which the Gumbel distribution rises to the upper left. The Gumbel distribution improves performance in most experiments. However, the Gumbel distribution showed the best performance, but the generalization performance tends to decrease somewhat. Therefore, for generalized power generation performance and stable training correlation, it is efficient to

find the distribution width adaptively for the optimal point distribution of the two relations about generalized and stable training.

Conclusion: We proposed new auxiliary components to help optimize deep learning. First, we proposed a novel hybrid regularization to reflect static and dynamic characteristics. For further analysis, we analyzed the relationship with loss. Relation analysis conducted based on seed values. As a result, it confirmed that the adequate reflection of static and dynamic characteristics in the regularization through hybrid helps the deep learning model to converge and optimize stably. In the process of analyzing the relationship between loss and regularization, it confirms that necessary to filter and pass only meaningful information. And when the analysis using the seed value, it confirms that the result obtained has a slightly larger deviation. However, hybrid regularization is somewhat robust to change due to the smaller standard deviation than the existing method.

Second, we propose a hierarchical concurrency optimization method for training deep learning model optimization. As a result of experimenting with the proposed method , we can confirm that the proposed method is stable learning. It seems that information applied from the hierarchical and simultaneous learning in the model space adequately reflects in the process of learning the model, which derives from the relationship between the information reflected in regularization and information reflected in optimization.

Third, We identified the importance of establishing the distribution using three problems. We performed distribution width increase analysis for stable training to use distribution variables in the generative adversarial network. According to the increase of the width of distribution has the advantage of information reliably

reflects in the model. To verify, we experimented with five models and two-loss functions. This study experimentally validates in the generative adversarial network but may be used as a distribution of latent variables for other image classifier models or reinforcement learning. To quickly and reliably reflect large amounts of information in a deep learning model, a broad density function distribution is useful as the distribution of the deep learning model. This shows that when the density continues to accumulate, the cumulative distribution function can reflect faster than other distributions. A sensitivity analysis of the distribution suggests that an optimal distribution can propose. We also gained a generalized distribution through selective removal of partial distributions and new inputs of partial distributions. After all, in terms of NP-hard distribution, the GAN can generate in real-time by approximating and reflecting the training data information in polynomial time effectively.

In future work, the energy functions of the loss and regularization define to visualize the effects of the model. It seems that research through visualization method to find optimal points by visualizing the possible relationship of the model is needed. Also, it seems to be necessary to optimize the direction of learning only meaningful information by visualizing the complexity between information from loss area and optimization area.

3.5 Ensemble Normalization for Stable Training [56]

Introduction : Deep learning model learns repeatedly to find the best point. In order to find the optimal point, the weights of the model are repeatedly changed in the deep learning model training process to move to the optimal point. However, if a large number of internal covariate shifts occur, it may represent a

distribution of weights that cannot be derived from the training data. This means that the distribution of untrained weights moves in different directions and converges in the wrong direction. Therefore, the study of normalization to correct and learn the internal covariate shift phenomenon is necessary. However, if learning by the existing single normalization can be learned in an unoptimized direction. To induce different features to be learned in the right direction through learning, we propose an ensemble normalization that learns different features through two normalizations.

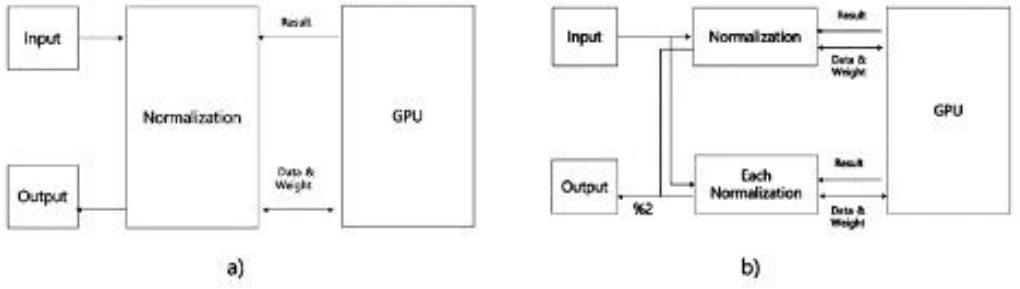


Figure 29. Process of normalization method, a) Existing normalization method, b) Ensemble normalization method

Proposed Method : We propose a new normalization method that is ensemble normalization method. Ensemble normalization calculates the addition of batch normalization and existing normalization. Then apply division 2 to the preceding result. The above calculation sequence is shown in Figure 29. Figure 28shows the process that is calculated in normalization using the existing method and the proposed method. To have a clear understanding of the existing methods, We conducted the ablation study on normalization to conduct an impact analysis on each existing normalization method before verifying the proposed method. We analyzed each normalization in the learning process through Focal loss [55],

Hinge loss, and Cross entropy. Experimental results show average results from each normalization's loss. The experimental method uses VOC and ATR datasets for FCN and U-Net and uses the average value of three losses and uses the average value through five iterative experiments. Experimental evaluation used 7 evaluation indicators. Tables 15 and 16 indicate in bold when performance improves compared to single normalization on Table 13 and 14. We comparative analysis of group normalization application using feature maps. it is confirmed that the normalization affects the density of the picture rather than the change in the appearance of the object.

Table 13. Quantitative comparison of three normalization in two models using V OC dataset.

Model	Model	DSC	F1 score	IOU	Loss	Mean Acc	Precision	Recall
FCN	None	0.733	0.733	0.776	0.498	0.975	0.733	0.733
	IN	0.735	0.735	0.777	0.277	0.975	0.735	0.735
	BN	0.704	0.704	0.759	0.333	0.972	0.705	0.705
	GN	0.732	0.732	0.775	0.309	0.974	0.732	0.732
U-Net	None	0.742	0.742	0.782	0.701	0.975	0.742	0.742
	IN	0.738	0.738	0.78	0.546	0.975	0.738	0.738
	BN	0.581	0.581	0.696	1.266	0.96	0.582	0.582
	GN	0.744	0.744	0.783	0.561	0.976	0.744	0.744

Table 14. Quantitative comparison of three normalization models using ATR data set.

Model	Model	DSC	F1 score	IOU	Loss	Mean Acc	Precision	Recall
FCN	None	0.718	0.718	0.764	0.489	0.969	0.718	0.718
	IN	0.727	0.727	0.769	0.287	0.970	0.727	0.727
	BN	0.712	0.712	0.760	0.350	0.968	0.712	0.712
	GN	0.723	0.723	0.767	0.332	0.969	0.723	0.723
U-Net	None	0.725	0.725	0.768	0.401	0.969	0.725	0.725
	IN	0.725	0.725	0.769	0.346	0.970	0.725	0.725

	BN	0.499	0.499	0.642	1.823	0.945	0.501	0.501
	GN	0.723	0.723	0.767	0.335	0.969	0.723	0.723

Tables 13 and 14 show the results of each normalization using the VOC and ATR datasets in the FCN and U-Net models. In the case of group and instance normalization, the result is improved when the normalization is not applied. Summarize the experiments from the previous ablation study. This phenomenon is because group normalization efficiently clusters the pixel distribution in the process of adjusting and learning the distribution of the pixels. In the case of instance normalization, each normalization is performed to obtain more precison pixel clustering results.

Experimental Results :

Table 15. Quantitative comparison of each normalization combination in the ense mble method using VOC dataset.

Model	Model	DSC	F1 score	IOU	Loss	Mean Acc	Precision	Recall
FCN	IN	0.735	0.735	0.777	0.320	0.975	0.735	0.735
	BN	0.736	0.736	0.778	0.356	0.975	0.736	0.736
	GN	0.733	0.733	0.776	0.296	0.975	0.733	0.733
U-Net	IN	0.721	0.721	0.769	0.348	0.973	0.721	0.721
	BN	0.637	0.637	0.718	1.464	0.965	0.637	0.637
	GN	0.711	0.711	0.763	0.317	0.972	0.711	0.711

Table 16. Quantitative comparison of each normalization combination in the ense mble method using ATR dataset.

Model	Model	DSC	F1 score	IOU	Loss	Mean Acc	Precision	Recall
FCN	IN	0.735	0.735	0.775	0.260	0.971	0.735	0.735
	BN	0.722	0.722	0.766	0.331	0.969	0.722	0.722
	GN	0.725	0.725	0.769	0.318	0.969	0.725	0.725

U-Net	IN	0.731	0.731	0.772	0.269	0.97	0.731	0.731
	BN	0.628	0.628	0.708	1.203	0.959	0.628	0.628
	GN	0.737	0.737	0.777	0.247	0.971	0.737	0.737

Table 15 and 16 show experimental results using two datasets for the proposed method and overall performance is improved. It is because they learn complementary to each other by catching the characteristics that cannot be captured by existing normalization methods through other normalization methods. With the addition of instance normalization, batch normalization, and group normalization, the loss each changes were +2.3%, +0.4%, and -4.2% for the FCN model and in the U-Net model, there were -42.9%, -20.3%, and -54.2%, respectively. However, there is a disadvantage that the performance changes depending on the type of normalization additionally applied. As a result, it can be seen that a study is needed to generate a feature generated by the combination optimization in a direction having a feature that helps in changing the internal weights of the model.

Conclusion : We propose a new normalization which is an ensemble normalization. First, the impact of existing normalization was normalization was analyzed through ablation studies to determine the impact of the existing method. We also verified the ensemble normalization method. As a result, we confirmed that the proposed method is effective for stable training on semantic segmentation. However, this study has a limitation that experiment based on batch normalization. Therefore, we necessary about analysis using various normalization methods. In future research, we will verify the proposal method in generating images and classifying images.

3.6 Similarity Analysis of Actual Fake Fingerprints and Generated Fake Fingerprint by DCGAN [57]

Recently, biometrics technology with the activation of PinTech is attracting

attention as the authentication technology. Biometrics is a method for recognizing human biological characteristics, such as signature, iris, and fingerprint recognition. Although these biometric technologies are widely used in electronic financial transactions, financial damages are also increasing due to fake biometric information. In order to solve this problem, various methods for discriminating fake biometric information have been recently developed [59–66]. Especially, as the successful applications of recent deep learning are increasing, some methods for discriminating fake biometric information using deep learning are being studied [59, 63–66]. Convolution neural networks (CNN), which are major methods of image information processing, are mainly used for fake fingerprint discrimination methods using deep learning [61–64]. In these methods, about 5 to 7 convolution layers are used for high fake discrimination performances and they require thousands to tens of thousands of training images. However, it takes a lot of time and cost to acquire real fingerprints and fake fingerprints. In addition, each time a fingerprint sensor is changed, a large amount of new data must be acquired. This situation occurs in most cases of applying to deep learning. In certain applications, it is very difficult to obtain data even at a lot of time and cost. To solve the above problem, some methods have been devised to acquire augmenting data using the acquired training data. One of such methods is to rotate, move, or scale up/down the acquired learning data. However, this method is a simple modification not to make a lot of additional training data, so it is difficult to improve the performances [63–65]. We propose a similarity verification method for augmenting training data between generated fake fingerprints by deep convolution generative adversarial networks (DCGAN) and actual fake fingerprints. To make augmenting data, we use the DCGAN, which has been applied to various fields recently. After training actual fake fingerprints

in DCGAN, we generate fake fingerprints using the generator of DCGAN. In order to use the generated fake fingerprints to augment training data, generated fake fingerprints made by the DCGAN must maintain the characteristics of the actual fake fingerprints. Otherwise, the performances of the fake fingerprint discriminator using generated fake fingerprints made by the DCGAN may be lowered. Therefore, we show through various experiments how similar the fake fingerprints made by DCGAN are to the actual fake fingerprints. We compare the distribution of the mean and standard deviation of the fake fingerprints generated by the DCGAN with those of the actual fake fingerprints as a first way to verify. In the second method, the mean Hamming distance (MHD), which is one method of evaluating the similarity of images, is used for measuring the similarity between the generated fake fingerprints and the actual fake fingerprints. The third method is to obtain the histograms of the generated fake fingerprints and the actual fake fingerprints and measure the similarity by calculating Pearson correlation of the two group of histograms. The fourth method is to calculate intersection of union (IOU) between the generated fake fingerprints and actual fake fingerprints. IOU is a method of evaluating the shape similarity of images. To evaluate the above methods, we trained DCGAN using actual fake fingerprints and generated fake fingerprints using the generator of trained DCGAN. For experiments, four data settings are provided with a combination of generated fake fingerprints and actual fake fingerprints that are not trained to DCGAN. We tested similarity between generated fake fingerprints and actual fake fingerprints on these four data settings with four similarity measures. Experimental results showed that the generated fake fingerprints made by DCGAN are similar to the actual fake fingerprints in most verification methods. This means that the generated fake fingerprints could be used to augment fake

fingerprint data for training deep learning.

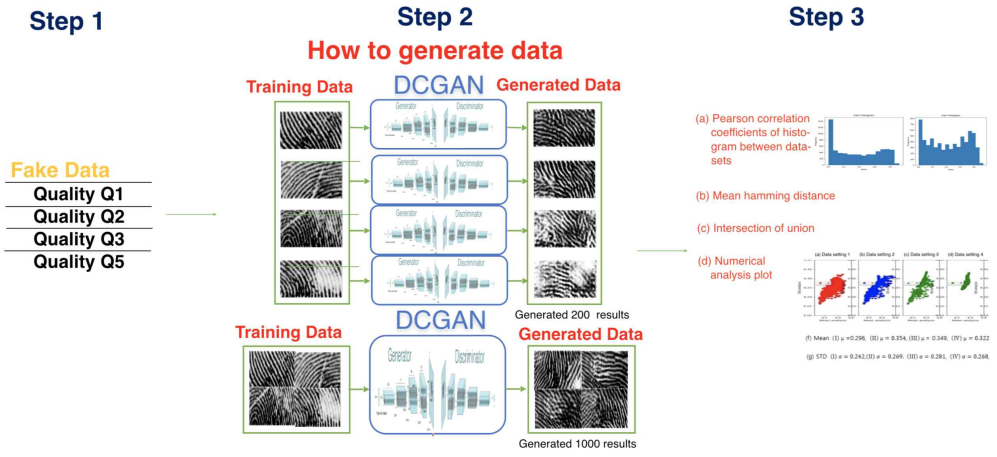


Figure 30. Overall process of proposed method.

Proposal Method : We propose four similarity measures to verity that the generated data by DCGAN can be used as augmented data. Figure 30 shows the overall structure of the fake fingerprint generation and evaluation methods. In step 1 of Figure 30, fingerprint data is classified into four qualities for training by each quality and by all quality. Step 2 of Figure 30 shows the generation method of fake fingerprints data by DCGAN. In order to investigate whether there is a difference between two training methods, DCGAN is trained by quality and by all quality. Step 3 is to measure the degree of similarity between fake fingerprints made by DCGAN and actual fake fingerprints. It is necessary to verify whether the fake fingerprint data made by DCGAN is similar to the actual fake fingerprint data. In order to evaluate the similarity of the generated fake fingerprints to the actual fake fingerprints, we proposed four similarity measures. Firstly, the mean and standard deviation of images are calculated and compared between the generated fake fingerprint images and actual fake fingerprint images.

Comparing the similarity with the mean and the standard deviation makes it possible to check the overall distribution of the images but not the detailed comparison of the images. Secondly, similarity was measured using the MHD developed as a method of measuring the similarity of two images. Thirdly, for comparison at the image histogram level, histograms of generated fingerprints and actual fake fingerprints were obtained and Pearson correlation between the histograms of the two data were obtained. Finally, shape similarity was measured using the IOU developed as a method of measuring the similarity of two images. The method of generating fake fingerprints using DCGAN is as follows. We first obtain actual fake fingerprint images from the material of fake fingerprint generation such as silicon or clay. At this time, the quality of actual fake fingerprints may vary due to the state of the material or the shaking of the hand when the fingerprint image is acquired. That is, fake fingerprint images with bad quality are obtained if the material is too stiff to properly contact the measurement sensor. We divide the actual fake fingerprint data with four qualities to see if there is a difference according to the quality of fake fingerprints. The quality of the actual fake fingerprint is classified into four levels. Q1 is the best quality, clean overall. Q2 is fake fingerprints that the outline or part of those is whitened. Q3 is worse than Q2, and Q5 is the case where only a part of the fingerprint is acquired. In order to generate fake fingerprint images, DCGAN is trained by quality or by all quality together. After training the DCGAN, the generator of DCGAN generates new fake fingerprints images by applying random latent z to the generator of DCGAN. The DCGAN structure used in our experiments is the same as that proposed by Radford *et al*. [65].

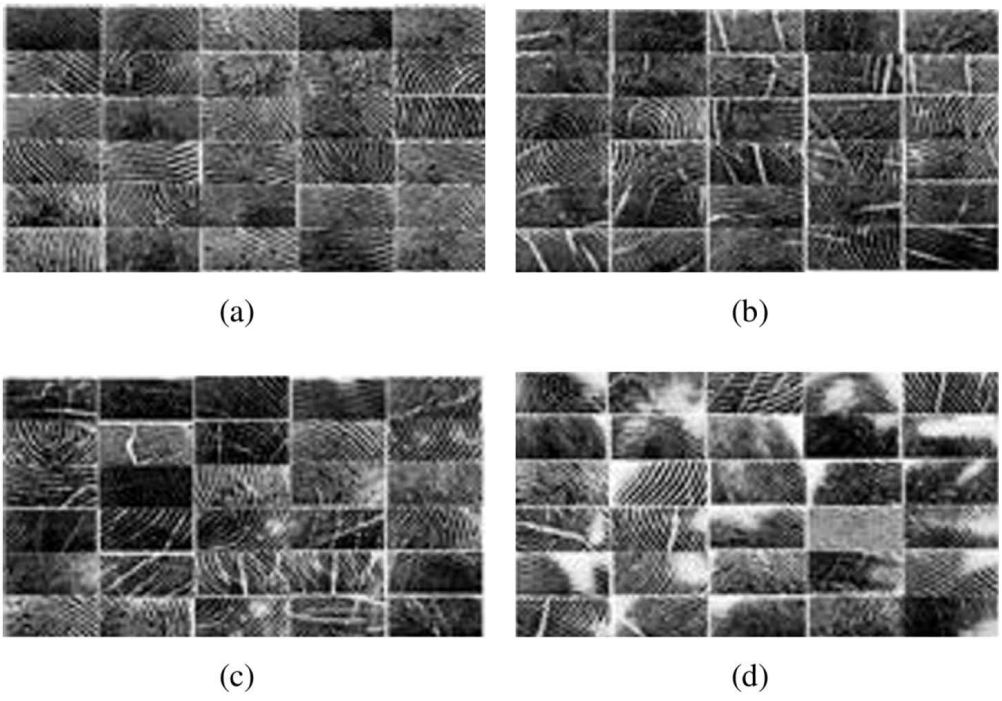


Figure 31. DCGAN training data by quality: (a) Q1, (b) Q2, (c) Q3, (d) Q5.

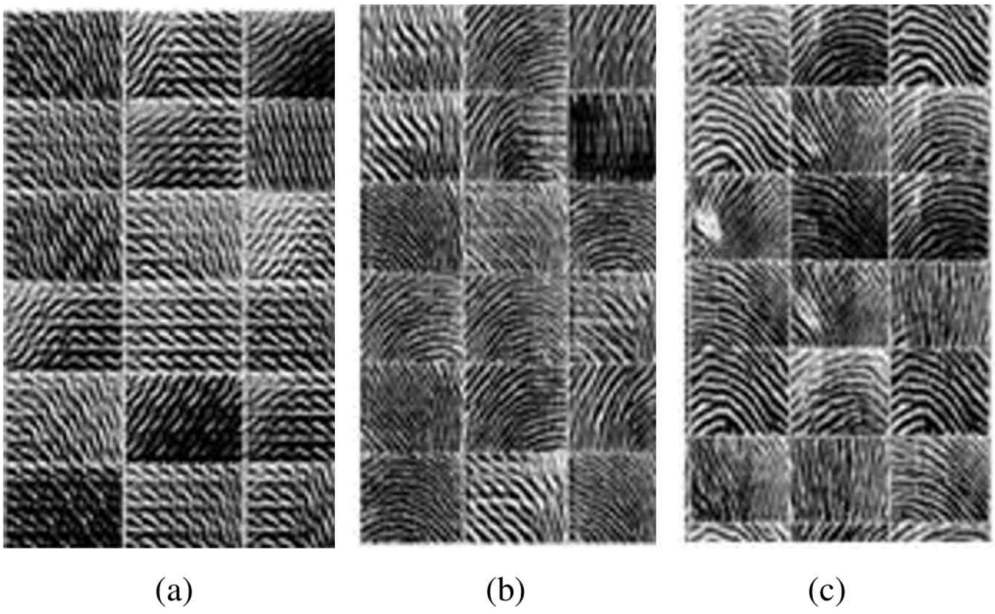


Figure 32. DCGAN training process. At the beginning of training (a), during

the middle of training (b), and the end of training.

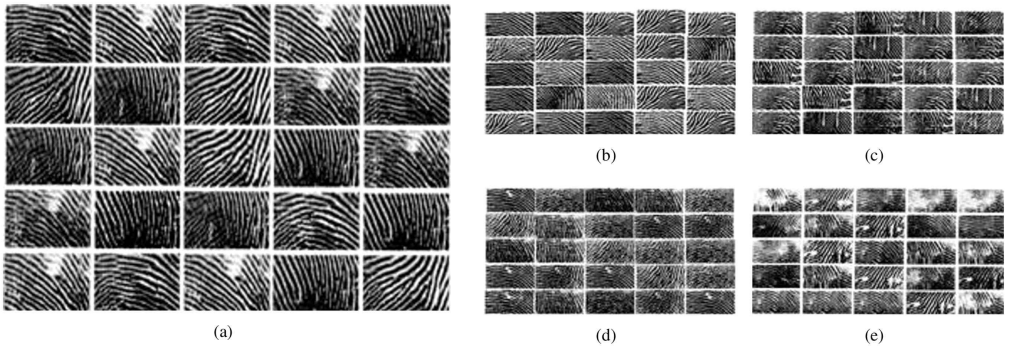


Figure 33. Generated Fake Fingerprint Data by DCGAN

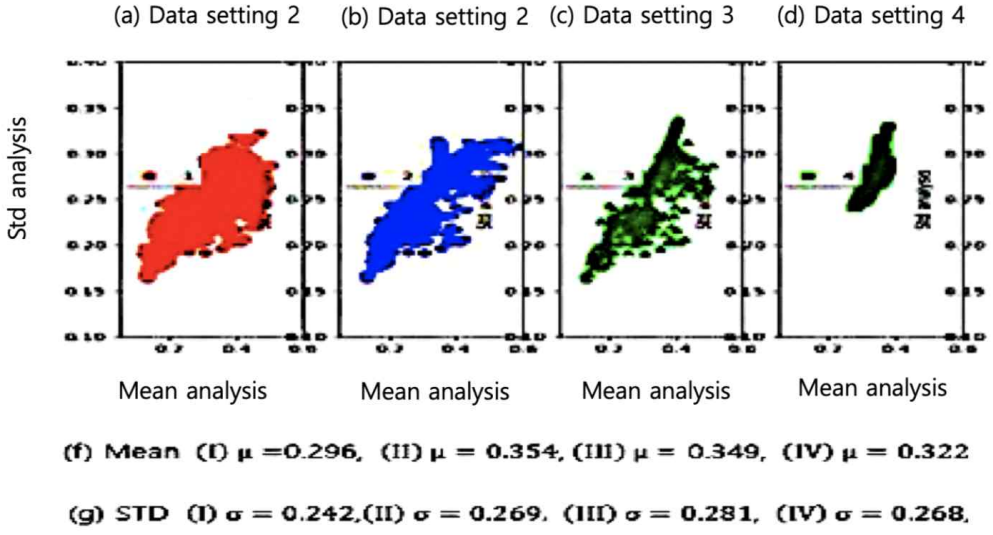


Figure 34. Plot of mean and standard deviation of four data sets.

Experimental Results : The DCGAN for generating fake fingerprints was implemented using TensorFlow developed by Google. Figure 31 shows the actual fake fingerprint samples for each of the four qualities Fake fingerprints in Figure 31(a) are very accurate because there are no cracks in the fingerprints and no

problem in acquiring the actual fake fingerprints. Figure 31(b) shows actual fake fingerprints with cracks. Fake fingerprints in Figure 31(c) are partially whitening fingerprints with cracks due to poor pressure on fingerprint acquisition. Figure 31(d) shows that the fingerprint acquisition did not work properly, resulting in a lot of white areas.

The DCGAN generator generates fingerprints that are similar to trained fake fingerprints as training progresses using the characteristics of fake fingerprints. Figure 32 shows the fake fingerprints generated by the generator during the training process. Figure 32(a), 32(b), and 32(c) are generated images by the generator at the beginning of training, during the middle of training, and at the end of training, respectively. As you can see in Figure 32, the more the training progresses, the more and more similar fingerprints to the characteristics of the training fake fingerprints are created.

If DCGAN is trained enough, it will generate fingerprints that are quite similar to the actual fake fingerprints. Figure 33 shows the fake fingerprints generated by the DCGAN generator after training. As shown in the Figure 33, it can be seen that they are similar enough to be indistinguishable from the actual fake fingerprints. As shown in Figure 33(c) and 33(e), it can be seen that cracks and white parts occur similarly to the characteristics of the training fake fingerprints. As a result, the generator of DCGAN fully reflects the training data and generates fake fingerprints.

Table 17. Data settings for verification of fake fingerprints

Data setting	DCGAN		Actual	Total
	Trained	Generated		
I (Original Quality)	0	0	1000	1000
II (Each Quality)	600*4	200*4	200	1000

III (All Quality)	2400	800	200	1000
IV (All Quality)	2400	1000	0	1000

We should verify the similarity of generated fake fingerprints to actual fake fingerprints with four measures. When the similarity is verified with four measures, the generated fake fingerprints by DCGAN can be used as training data for augmenting training data. In the verification method with four measures, we use four data settings to see if the similarity depends on the quality of the fake fingerprints. Table 17 shows four data settings for verification of the similarity according to generated data by DCGAN and actual data. In data setting I, there are no generated fake fingerprints and 1,000 actual fake fingerprints. In data setting II, 600 actual fake fingerprints are trained for each quality in DCGAN, then 200 fake fingerprints are generated by DCGAN for each quality, and 200 actual fake fingerprints are combined to make 1,000 total fake fingerprints. In data setting III, DCGAN learns 2,400 actual fake fingerprints without distinguishing quality and generates 800 fake fingerprints, and combined to 200 actual fake fingerprints. The data setting IV is the same as the third one, but it generates 1,000 fake fingerprints. Therefore, data setting I and IV are composed of only actual fake fingerprints and only generated fake fingerprints, respectively. In these data settings, all data sets were used after normalization process.

Similarity Analysis : To analyze whether the fake fingerprints generated by DCGAN are similar to the actual fake fingerprints, our analyzes were performed. In this analysis, we compared them according to the four training data sets as shown in Table 17. This is because that it would be more meaningful to compare the generated fake fingerprints according to the methods of using them as training data. First, the pixel value of the image was used to calculate and

compare the mean and standard deviation of the image. In other words, the averages and standard deviations of images of four data sets are shown on the two-dimensional coordinates in order to compare the similarity of four data sets. Figure 34(f) and 34(g) shows the results of the mean and standard deviation of four data sets. First, the data set II, III, and IV, which use fake fingerprints generated by DCGAN, are slightly larger than the data set I in average, which use only actual fake fingerprints. This is probably because DCGAN generates a crack or a white part in the actual fake fingerprints with low quality.

In the data set II and III, since many generated fake fingerprints are included, we can see that the result is almost similar. It can be seen that there is almost no difference in terms of mean and standard deviation in the case of generating by quality and by all quality together. As you can see in Figure 34, data set IV composed of only generated fake fingerprints are overlapped to the data set I composed of only actual fake fingerprints. This indicates that the generated fake fingerprints by DCGAN are similar to the actual fake fingerprints in terms of the distribution of mean and standard deviations.

As a result, from the viewpoint of average and standard deviation analysis, they can be used to augment fake fingerprints. However, the mean and standard deviation represent the overall characteristics of the image and can not be used to measure the similarity of the image in detail. Therefore, three additional analyzes were performed to analyze the detailed characteristics. The MHD used in Sixt *et al*. [66] was used for the analysis of four data sets. The Hamming distance is increased when the reference pixel value is different from the comparison pixel value. All the extracted Hamming distances are added and divided by the number of pixels, then the total average Hamming distance is obtained.

Table 18. Analysis of various similarity methods.

	Dataset			
	I-I	I-II	I-III	I-IV
Mean hamming distance	5955.69	6463.39	6446.29	6550.22
Pearson correlation of histogram	0.682 ± 0.219	0.151 ± 0.272	0.272 ± 0.245	0.180 ± 0.111
Intersection of union	0.50	0.45	0.55	0.53

Table 18 shows the MHD results between two data sets based on data set I. Hamming distance is obtained over all data pairs in two data sets and averaged. The smaller the MHD is, the more similar the two sets of images are. MHD of data set I-I is the smallest because they calculate themselves. The next best thing is data set I-III made by combining all qualities and then data set I-II made by quality. However, the difference between the two sets is very small, so there is no big difference. The worst case is comparison data set I-IV, where all the fake fingerprints were generated by the DCGAN. Although DCGAN produces something that is very similar to an actual fake fingerprint, it shows that it produces something slightly different from the actual fake fingerprint. Finally, to analyze the similarity of the distribution of brightness values of images, each image is represented by histogram and analyzed by Pearson correlation of histogram for each data set. In other words, all pairs are generated from two sets of images and the Pearson correlation is obtained from each pair. As shown in Table 18, all results show positive correlation results. That is, the histogram of the generated fake fingerprints are similar to the histogram of the actual fake

fingerprints. Experimental results show similar tendency to MHD measurement results. These results show that DCGAN can produce similar results to the brightness distribution of training data. Table 18 shows the IOU results. IOU, a quantitative representation of overlapping parts of object detection, has been used as an evaluation of segmentation [67]. We used the IOU as a similarity measure of two data sets through the full combination of the data of each data set based on the data setting I. For examples, I-II column in Table 18 shows the similarity of data set I and II, and shows the lowest value of similarity analysis in comparison data set I-II. Since data set II is made by each quality, the similarity of generated fake fingerprints is not totally followed much of fake fingerprints. In the case of comparison data set I-III and I-IV, it can be confirmed that the methods generated by whole quality are more similar to each other than those made by each quality. The difference between comparison data set I-III and I-IV is 0.02, which appears to be a difference in the experimental method depending on the presence or absence of 200 actual fake fingerprints. Through the four similarity measurement methods, it was shown that the fake fingerprint generated by DCGAN is similar to the actual fake fingerprint and they can be used for augmenting training data.

Conclusion : We analyzed similarity of actual fake fingerprints and generated fake fingerprints by DCGAN with four similarity measures for augmenting training data. Experimental results showed that fake fingerprints generated by DCGAN are made by combining the features of training actual fake fingerprints and confirmed that the characteristics of the generated fake fingerprint are substantially similar to those of the actual fake fingerprints. From these results, we could conclude that the generated fake fingerprints by DCGAN were used for

augmenting training data. These results are useful in the case where training data is costly and time-consuming to acquire, especially in areas where acquisition of training data is very difficult. As a further work, we will directly test the performance improvements of CNN and DNN using the augmenting training data by DCGAN.

3.7 Multi Way Decoder Scheme with Error Reduction Embedding on one-hot bi-directional Seq2Seq with adaptive regularization for Music Composition[73]

In the existing research on music composition using deep learning, most of the composition does not reflect details such as pitch and duration [68-72]. This makes it impossible to compose realistic music like composers. In addition, since the patterns are created using various combinations using notes, beats, and melodies, the complexity of the combinations expressed from each element is very high. Finding meaningful patterns from high complexity is a difficult problem. In previous research, most of the deep learning does not reflect the details of the training data. Currently, the field of study is compressing information in order of signal, information, and knowledge in comparison with the flow of information. However, the current structure has been studied to change within the scope of the state of information if the current state is the information state. It is essential to research through improvement within the same perspective as the research has been conducted so far, but we thought that the direction of research should be seen through the new perspective from the scheme perspective by expanding the structure. This makes it impossible to compose realistic, like composers.

To overcome this problem, we tried to propose a structure from a scheme point of view. In this way, the structure of deep learning is essential to understand the structure, explainable method, interpretable method, and structural components from the transparent structure perspective. Also, looking at the encoding structure of a structure at the schema level of the structure using scheme has the advantage of being able to encode by explicitly changing the internal steps. It is also essential to building on a structure because the structure can be made procedural.

- We propose a multi-way decoder module from the scheme point of view for the first time in determining deep learning structure internal information with heterogeneous normalization.
- We proposed a one-hot bi-directional Seq2Seq with the adaptive l2 regularization.
- We propose an information error minimization for use as an input.

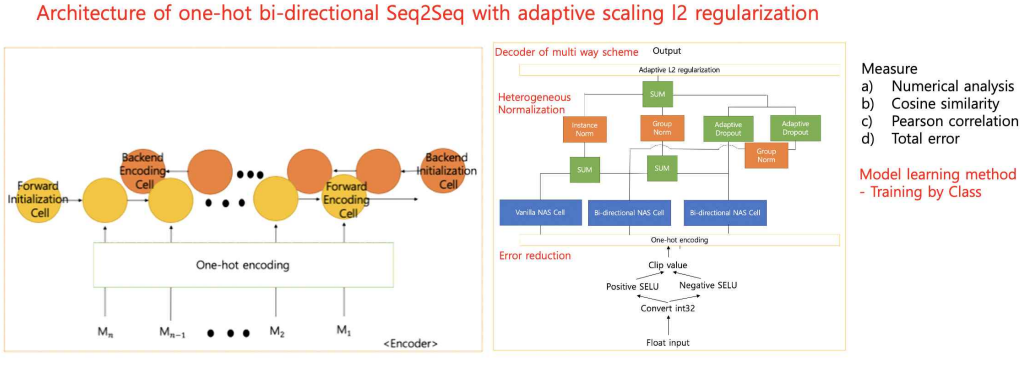


Figure 35. Our Proposal System about Multi-way Decoder Scheme

Our Proposal : Solving problems with new perspectives is challenging and stressful. Also, it is a difficult problem to efficiently optimize the process of

embedding in space and determining information at the same time. We will try to improve these difficult problems through the new method. We propose a new method for the embedding part and decoding part of the information in the process of determining the information through one decision in Figure 35. The embedding part proposes a three-step information processing method for encoding the information of the input data into one codeword (One-hot encoding) in the encoding process. First, integer using convert int32 reduces bit errors. Second, the size of the information corrects by scaling the bi-directional information. Third, clipping is to simplify the information. Through the above process, co-adaptive information of input information extract. In the decoding part, we propose an optimal decision- making method by optimizing the combination of multi-way information based on the scheme structure. The proposed structure has a new structure of information processing from three perspectives. This is because performance improved through the synergistic effect of each information through optimization of the combination of information generated in the process of transmitting the information generated from various components to a single target. However, it believes that this method has limited performance that judges according to the degree of optimization between parts.

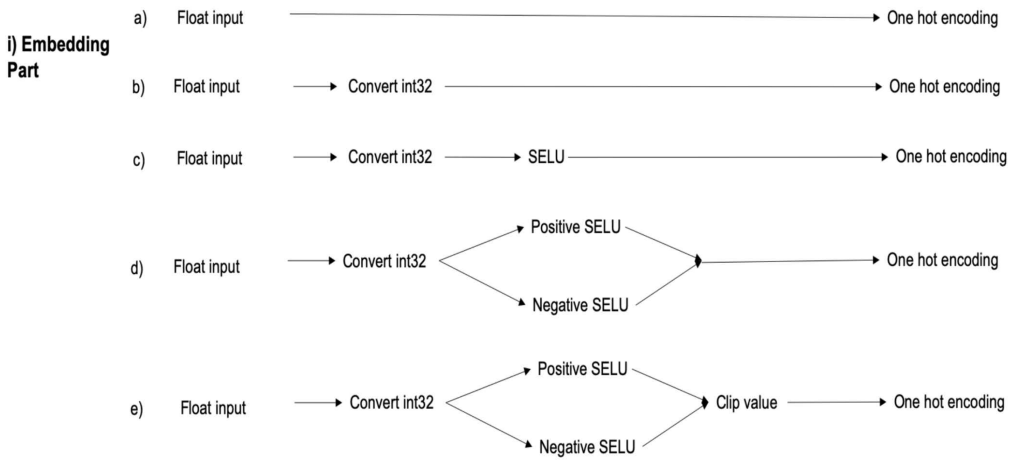


Figure 36. Analysis of our proposal. i) Embedding part

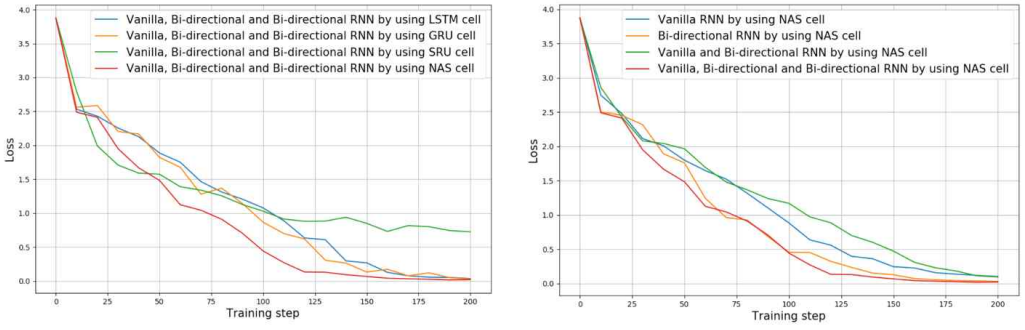


Figure 37. Comparison of decoder part experiments with Seq2Seq model

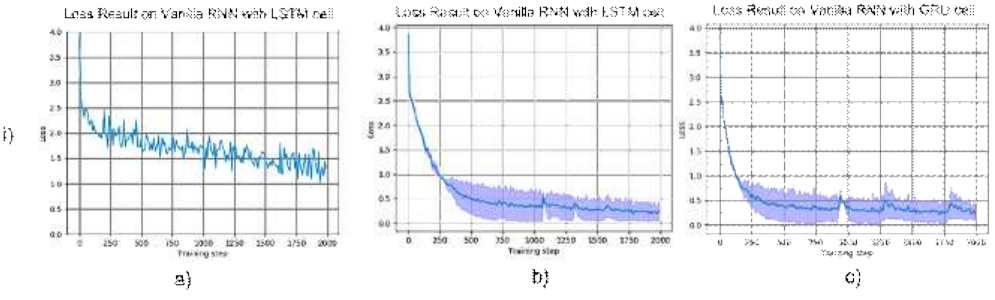


Figure 38. Comparison of the impact on learning loss

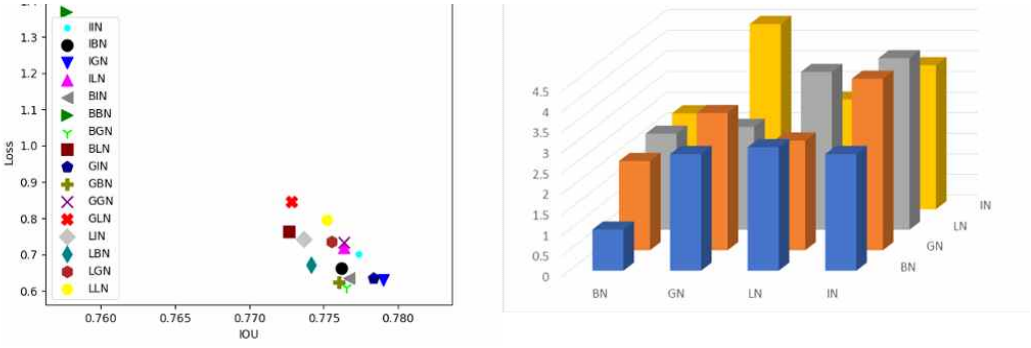


Figure 39. Evaluation of normalization performance used in experiments using scatter plot and histogram

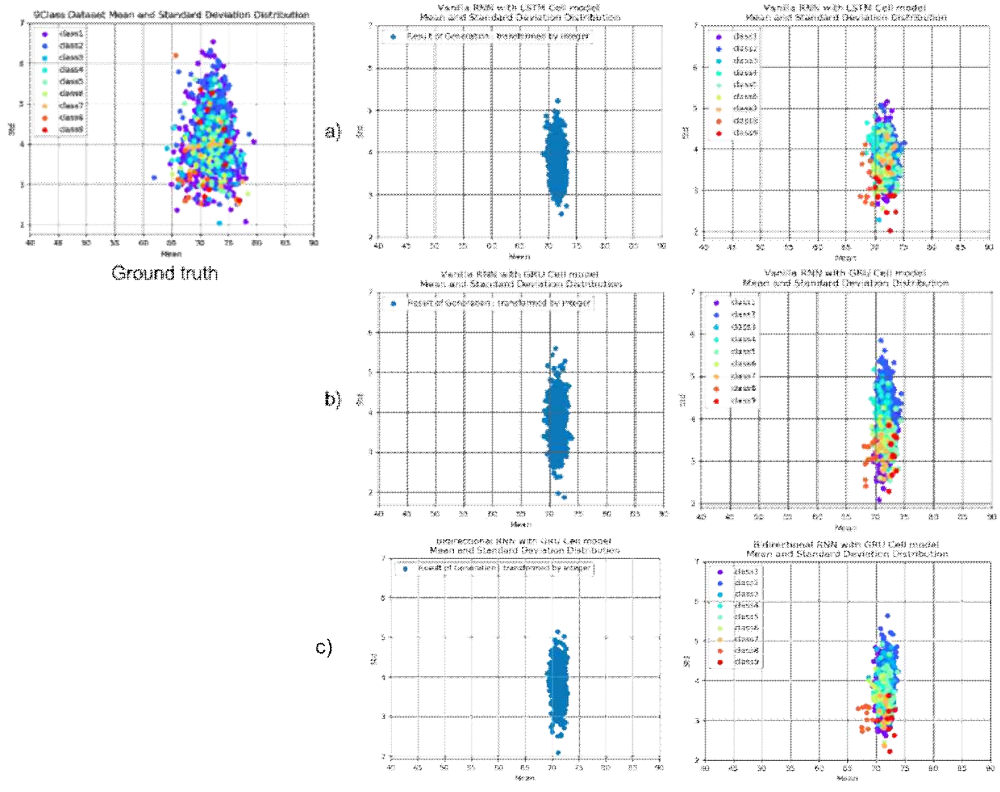


Figure 40. Comparison result using numerical analysis of training in each class

Figure 41. Analysis of our proposal

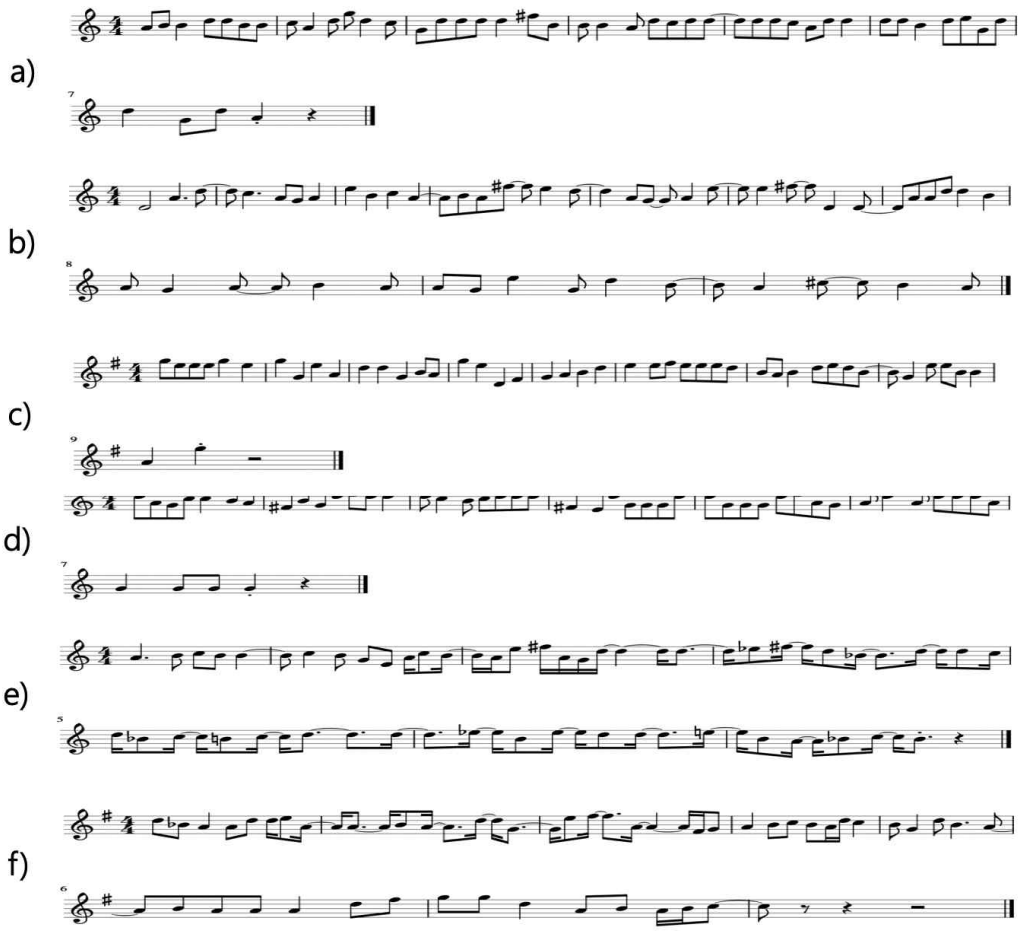


Figure 42. Visualization of generated music about one-hot bi-directional Seq2Seq

Table 19. Comparison of one-hot encoding and whole encoding analysis

	Duration	Pitch
Convert int32	0.013	0.060
Scaling SELU to Convert int32	0.050	0.083
Scaling Bi-directional dot product SELU to Convert int32	0.026	0.076
Scaling Bi-directional dot product SELU to Clip value to Convert int32	0.010	0.083

Table 20. Comparison of one-hot and whole encoding analysis using total error

Test	Bi-directional RNN with whole input	Bi-directional RNN with whole one-hot input
Duration	240	168
Pitch	10.0	9.5
Test	Bi-directional Seq2Seq with whole input	Bi-directional Seq2Seq with one-hot input
Duration	224	190
Pitch	14.5	11.5

Table 21. One-hot encoding on experimental models, a) bi-directional RNN, b) bi-directional seq2seq, c) bi-directional seq2seq using adaptive l2 0.5 regularization

Train for duration	Loss of 0	Loss of 20	Loss of 60	Loss of 90
a	3.893	0.511	0.427	0.083
b	3.898	0.586	0.424	0.117
c	3.893	0.591	0.495	0.164
Test for duration	Total error	Cosine similarity	Pearson correlation	Numerical analysis
a	168	0.998	0.007	μ : 72.0 σ: 2.965
b	190	0.998	-0.017	μ : 73.163 σ : 1.976
c	140	0.999	0.455	μ : 72.326, σ : 2.717
Train for pitch	Loss of 0	Loss of 20	Loss of 60	Loss of 90
a	3.874	2.330	1.571	0.588
b	3.864	2.432	1.576	0.627
c	3.863	2.432	1.220	0.307
Train for pitch	Total error	Cosine similarity	Pearson correlation	Numerical analysis
a	9.5	0.851	0.067	μ : 0.628, σ : 0.266
b	11.5	0.825	-0.239	μ :0.581

				σ : 0.214
c	7.5	0.901	0.255	μ : 0.501, σ : 0.185

Table 22. Various efficient on one-hot bi-directional seq2seq2 with l2 regularization using 0.5, 0.4, 0.3, 0.25, 0.2, 0.1 and none l2 regularization.

Train of duration	Loss of 0	Loss of 20	Loss of 60	Loss of 90
None	3.898	0.586	0.424	0.117
l2 0.1	3.898	0.661	0.325	0.082
l2 0.2	3.898	0.633	0.364	0.147
l2 0.25	3.893	0.591	0.495	0.164
l2 0.3	3.899	0.651	0.363	0.092
l2 0.4	3.898	0.612	0.371	0.093
l2 0.5	3.897	0.562	0.499	0.126
Test of duration	Total error	Cosine similarity	Pearson correlation	Numerical analysis
None	190	0.998	-0.017	μ : 73.163 , σ : 1.976
l2 0.1	151	0.998	-0.094	μ : 71.558 , σ : 2.038
l2 0.2	214	0.997	-0.479	μ :72.233, σ : 2.752
l2 0.25	140	0.999	0.455	μ:72.326, σ : 2.717
l2 0.3	181	0.998	0.47	μ :72.907, σ : 2.568
l2 0.4	172	0.998	-0.059	μ: 71.860, σ : 2.707
l2 0.5	171	0.998	-0.073	μ : 71.6, σ :2.596
Train of pitch	Loss of 0	Loss of 20	Loss of 60	Loss of 90
None	3.864	2.432	1.576	0.627
l2 0.1	3.827	2.493	1.220	0.573
l2 0.2	3.868	2.476	1.365	0.350
l2 0.25	3.863	2.432	1.220	0.307

l2 0.3	3.864	2.488	1.835	0.648
l2 0.4	3.873	2.306	1.09	0.241
l2 0.5	3.874	2.436	1.140	0.222
Test of pitch	Total error	Cosine similarity	Pearson correlation	Numerical analysis
None	11.5	0.825	-0.239	μ :0.581, σ : 0.214
l2 0.1	14.0	0.805	-0.097	μ :0.709, σ : 0.345
l2 0.2	5.0	0.910	0.314	μ : 0.570, σ : 0.173
l2 0.25	7.5	0.901	0.255	μ : 0.501, σ : 0.185
l2 0.3	11.0	0.834	-0.108	μ : 0.570, σ : 0.224
l2 0.4	9.0	0.885	-0.139	μ : 0.523, σ :0.105
l2 0.5	12.25	0.828	-0.001	μ : 0.657, σ :0.305

Table 23. Comparison of our proposal based on one-hot bi-seq2seq with adaptive regularization using music dataset using top 1 error

Train	Loss of Duration	Loss of Pitch
Bi-directional RNN using LSTM Cell	0.040	0.077
Bi-directional RNN using SRU Cell with Double A LL Group Normalization	0.052	0.124
Bi-directional RNN using SRU Cell with Double A LL Group Normalization with 0.7 Dropout	0.056	0.128
Change NAS Cell	0.045	0.108
Change 0.75 Dropout	0.038	0.087
Both Bi-directional RNN	0.020	0.102

and Vanilla RNN using GRU Cell with Double All Group Normalization with 0.7 Dropout		
Change NAS Cell	0.024	0.067

Table 24. Comparison of multi-way decoding scheme analysis

	Duration	Pitch
Our scheme + Our embedding	0.015	0.054
Only adopted encoding module that composed of Our scheme + Our embedding	0.006	0.063
Only adopted encoding and decode module that composed of Our scheme + Our embedding	0.027	0.080
Only adopted one module that composed of Our scheme + Our embedding	0.020	0.058

We analyzed a new method for embedding method and decoder module. Figure 36 shows an analysis of the proposed method for the embedding part. Figure 36 consists of five parts. Figure 36(a) shows how to apply one-hot encoding on existing float data. Figure 36(b) shows how to apply one-hot encoding after changing integer 32 from existing float data. Figure 36(c) shows how to apply a one-hot encoding after applying to scale after changing integer 32 from existing Float data. Figure 36(d) is a method of applying one-hot encoding after changing to integer 32 from existing Float data, dividing it into Bi-direction, applying to scale. Figure 36(e) shows how to apply one-hot encoding after clipping to reflect

only important information after changing the existing float data to integer 32 and dividing it in Bi-direction. In the process of clipping, we set various ranges and experimented to find the optimal point and clipped it using the value of the category. During the experiment to find the clipping coefficient, a large variation in performance occurred depending on the degree of clipping. Therefore, it confirms that it is important to find a range of meaningful information.

In Table 19, in the first step, results show a decreased error in music duration and pitch. This is because the method applies after changing to an integer type to minimize bit error during the embedding process. In the second step, the result is increased compared to the first step. It seems that the error is improved because of the result of correcting the information by applying scaling before changing to an integer type, not cluster. In the third step, The Top 1 error is reduced compared to the ninth method. In the scaling process, information scaling is performed by dividing into bidirectional information and applying it to scale. In the fourth step, the Top 1 error shows the smallest error value in the pitch information. It seems that the lowest error obtains by extracting only the common information from the extracted information. The positive SELU and the negative SELU use despite the increased error in the process of inserting new elements. It intends to use only information generated in the process of merging through bidirectional information. Sparse information generates. Such sparse information is an important key point in judging an image. Using these key points to judge the information of the model was judged efficiently and considered to be the correct judgment. On the contrary, common information is not a significant difference in actual judgment, so confusion arises in the process of model judgment.

We compare of decoder part experiments with the Seq2Seq Model in Figure 37.

Firstly, we compared the cases where Vanilla RNN, Bi-directional RNN, Vanilla RNN, and Bi-directional RNN, Vanilla RNN, and Bi-directional RNN, and Bi-directional RNN use when using NAS Cell. This was to check whether it is more efficient in the decoding process to reflect various features in the decoding process. Through this experiment, it confirms that reflecting various features showed faster convergence speed and lower error. This confirms that learning by using various ways rather than using one way converges to the shortest point of the model. Secondly, we tried to find a robust internal cell inside the RNN used in this experiment using LSTM Cell, GRU Cell, SRU Cell, and NAS Cell. It also shows a faster convergence rate than other cells. Cell, which shows the lowest error value when using NAS It confirms that the structure to find the internal cell through the search method could acquire the lowest error of the model. We analyzed different features in the decoding process. It intends to use bidirectional and single features simultaneously. It was important to reconstruct it by reflecting on various features. Therefore, to reflect different features, the experiment was conducted by applying an RNN to extract a single characteristic and a bi-directional RNN to extract a bidirectional characteristic to the decoder model part. Reflecting by using different features is considered to show a better synergistic effect than reflecting a single characteristic when synergistic effects occur.

We show the results of experimenting on how to learn all the classes and how to learn each class to see the impact of learning in each class. Figure 38 shows a comparison of the impact on learning loss for each class learning. In the case of learning in each class, nine loss values were expressed as the centerline of averaged values, and the change amount was expressed using the standard deviation. Figure 38(a) shows how to learn with the whole class. Figure 38(b)

shows how to learn with each class. The learning of each class results in lower loss errors than the learning of the entire data, which results in a lower error because the overall complexity of the data set is lower. In the case of learning by each class, we try to compare the effect of learning by each class by changing the cell of Vanilla RNN. Figure 38(c) shows the results of experiments with GRU cells. In the case of learning each class as a GRU cells, it confirms that the spark spacing in the loss is wider and higher in size than the LSTM Cell. These results seem to be caused by the lack of one memory gate in GRU Cell than LSTM.

In the process of training heterogeneous features in the process of training different features, the training may conduct in the right direction or the wrong direction. Therefore, there is a need for a combination optimization that can stably training heterogeneous features. We are analysis of interactive associate feature calibration at ensemble normalization using association relation analysis. We are an experiment with four normalization methods (i.e., batch normalization, instance normalization, group normalization, and layer normalization). Through observation, we could confirm that group normalization and instance normalization showed the best performance by combining normalization experimented in Figure 39. Also, when the performance improved compared to the existing one. This visualizes using a bar graph. We use instance normalization and group normalization in an ensemble normalization. It confirms that the combination of group normalization and instance normalization has improved through more stable acquisition.

For more realistic music composition, we propose a new music composition using a one-hot bi-directional Seq2Seq structure with a new adaptive l2 regularization to reflect the detailed characteristics of music in the new music

composition in Figure 35. We adopted a method of class training. Training as a class reflects the information in the data more efficiently than reflecting the entire data. Applying one-hot encoding allows us to extract and learn only important features, so we can evaluate performance through the principal components of data. Also, we adopted the bi-directional Seq2Seq. Because we can learn more about the training data through bi-directional training, but bi-directional training can also increase the trained information on trained music. We propose a new adaptive l2 regularization during training. Adaptive l2 regularization helps the Seq2Seq to train more detailed information by clustering the training music. Also, we use one-hot encoding for pitch and bits in both encoding and decoding models, because one-hot encoding allows us to learn trained songs more carefully and accurately. To check the composition results, we train the bi-directional Seq2Seq model with 420 songs. When learning different genres of music, we found it difficult to learn more because of the mix of musical characteristics of each genre. So, we experimented with only one genre to compose a more realistic song. We showed that a model trained with only one genre song produced a more realistic song.

The composition of the experiment environment is as follows. The music dataset is consisting of nine genres that include 420 songs. The experiment was performed by configuring the output with 48 melodies and 48 melody inputs in the experimental model. The music also consists of two pieces of information about duration and pitch. We apply the proposed method to duration and pitch. We were experimented with Adam optimizer, epoch 100, a learning rate of 0.01, and Seed is 77. We experimented with the effect of music composition on the proposed method through the average of the duration and pitch results for quantitative analysis. The improved values are shown in bold.

We first show the results of experimenting on how to learn all the classes and how to learn each class to see the impact of learning in each class. We compare the impact on learning loss for each class learning. In the case of learning in each class, nine loss values express as the centerline of averaged values, and the change amount expresses using the standard deviation. The learning of each class results in lower loss errors than the learning of the entire data, which results in a lower error because the overall complexity of the dataset is low. In the case of learning by each class, we tried to compare the effect of learning by each class by changing the cell of Vanilla RNN. In the case of learning each class as a GRU cell, it can confirm that the spark spacing in the loss is wider and higher in size than the LSTM Cell. These results seem to be caused by the lack of one memory gate in GRU Cell than LSTM.

Figure 40 shows a comparison of the effects of music generation on the ground truth for each class training. Figure 40(a) is a case of training using the LSTM cell, Figure 40(b) is a case of learning using GRU cell, and Figure 40(i) is a case of training using whole ground truth. In this case, the ground truth represented in Figure 40 is the result of visualizing the ground truth using the mean and standard deviation. Also, the generated music data visualize by calculating the mean and standard deviation. As a result of numerical analysis in Figure 40, we found that the case of using GRU Cell is more similar to the actual data distribution than the case of using LSTM Cell. The LSTM cell has one memory gate larger than that of the GRU cell. Training in each class confirms that the model is trained reliably and produces a better representation of the actual data. We compared and analyzed the case of training in one direction and both directions using Vanilla RNN and bi-directional RNN in Figure 40. Figure 40 shows a comparison result using numerical analysis of training with

bi-directional in each class. Figure 40(b) is the result of training all classes. Figure 40(c) is the result of training by each class. Figure 40(i) is the result of training with Vanilla RNN using GRU Cell. Figure 40(ii) is the result of training with Bi-directional RNN using GRU Cell. In the case of learning by Bi-directional RNN in the learning method of each class, information from bi-directional learn. It can be seen that the result generated above is similar to the shape of the existing important data distribution of the data than the results learned with the existing Vanilla RNN. These results seem to generate by reflecting the main component characteristics, which are important characteristics of existing data characteristics.

From the above results, we can see that bi-directional training generates distribution similar to real data in single class learning. Therefore, this study compared and analyzed Bi-directional RNN and Bi-directional Seq2Seq. Also, we apply the adaptive l2 regularization proposed in Table 21. Table 21 analyze of duration total error and pitch total error using bi-directional RNN and bi-directional Seq2Seq for one-hot encoding and whole encoding input. In the case of the bi- directional Seq2Seq with whole one-hot encoding, the total error is reduced by 72, and the pitch error is reduced by 0.5. Bi-directional Seq2Seq reduced duration total error 34, and pitch error decreased to 3.0. The above results show that most of the methods reduce when one-hot encoding is applied. These results seem to be because the model can learn as it reflects discrete information. Reflecting discrete information is similar to learning only important features of the data. After all, reflecting only important input features shows that deep learning models can train efficiently.

The errors in loss of 90 during train for music duration in Table 22. Table 22 shows that the error values are slightly higher when comparing bi-directional

seq2seq and bi-directional Seq2Seq with adaptive l2 0.25 regularization using bi-directional RNN. However, the bi-directional Seq2Seq with an adaptive l2 0.25 regularization shows a 28 error reduction in Table 22. Pearson's correlation showed an improvement of 0.448 in Table 22. We improved cosine similarity 0.050 and improvement of Pearson correlation 0.188 in Table 22. Table 22 show that the overall performance of adaptive l2 regularization improves by composition. It is explained in terms of Duration and Pitch. Pitch is the degree of highness or lowness of a tone. Duration is the length of time a note lasts for. Because the model used in the experiment is robust to time continuity, it shows a better similarity than the pitch in terms of duration measurement. However, in terms of pitch, it seems to generate a rather high error in hierarchical sound generation.

The results of train and test for music duration in Table 24 show that the performance of l2 regularization is generally improved compared to l2 regularization. Table 24 shows that regulation has improved the performance by looking for generalization features. However, in the case of strong regulations, the performance fell somewhat. Therefore, it is important to find generalized functions through adaptive regulation. Finally, to verify the effectiveness of the adaptive l2 regularization, quantitative analyses are performed for the duration in the pitch in Table 24. We also compare the duration and pitch improvements to the mean values for averaging the adaptive l2 regularization. In the result of train and test for music pitch Table 24 can be seen that the result of applying l2 regularization is mostly improved compared with the result before application. Also, Table 24 shows the performance improvement/reduction according to the scale value of l2 regularization. These results confirm the need for adaptive l2 regularization to apply regularization. In the regularization experiment, similarly,

it shows better similarity than the pitch in relation to the duration measurement. However, in terms of pitch, it seems to generate a rather high error in hierarchical sound generation.

Figure 39 shows an analysis of the decoding method. The analysis was conducted to confirm the impact on each component of the proposed method. Figure 39(a) shows how to experiment with LSTM cell on the existing bi-directional RNN. Figure 39(b) shows how to experiment with SRU cell on the existing bi-directional RNN. Figure 39(c) shows how to increase existing bi-directional RNNs to two, apply Group Normalization, and merge them. Figure 39(d) extends the existing bi-directional RNN to two, applies the same Group Normalization, and extracts the features through Drop before applying normalization to one path, and two information extracted through Drop after applying normalization to one path. Use as one information and combine it with another. Figure 39(e) is based on Figure 39(d) to experiment using NAS Cell instead of SRU Cell. Figure 39(f) experiments after changing the probability value from 0.7 to 0.75 in Figure 39(e). Figure 39(g) creates one path using Bi-directional RNN and other Vanilla RNN based on GRU Cell, applies Instance normalization, extracts the feature, and applies it to the last part of Figure 39(f). Figure 39(h) experiment based on Figure 39(g) based on NAS Cell.

Table 23 shows the results of the Top 1 loss error learned through the method proposed. The performance was improved through experimental verification through the introduction of new methods one by one. We proposed a new decoding module based on the proposed encoding and decoding scheme. Therefore, we experimented based on the encoding and decoding structure to generate music. The proposed structure consists of one-hot bi-directional Seq2Seq with adaptive L2 regularization. Table 23 summarizes the tendency to loss. Loss

results tend to increase and then decrease. It was motivated to improve performance over the existing method by reflecting the synergy effect by reflecting the characteristics of heterogeneous models. Therefore, to create different features, we tried to apply the method of randomly removing one feature and using the feature that not apply to the other. Through this, we tried to extract different features. However, if you design as above, it seems that the model is optimized, but a higher error occurs because it did not go to a better optimization point than before. Since this is a less-optimized state, we tried to find the optimal state by using a NAS cell inside to find the optimized state. Through this, optimized results obtain through NAS Cell. Nevertheless, since it shows a higher error than the previous one, the optimal extraction was found through the adaptive coefficients in the probability extraction. To improve than the previous one, reflecting various features can be reflected in a more optimized state, so we experimented with a method of combining and applying various RNNs to reflect various features to find a better optimization model. This seems to be improved as it reflects the various features to help optimize further.

Experimental Results about Music Composition: numerical analysis, cosine similarity, Pearson correlation, and total error [57] to compare the similarity between training music and generated music.

Figure 42 shows the results of the qualitative analysis on the generation score of the one-hot bi-directional Seq2Seq. Figure 42 is composed of six experimental results. Figure 42(a) is the result of whole input RNN, Figure 42(b) is the result of whole input bi-directional RNN, Figure 42(c) is the result of whole input bi-directional Seq2Seq, Figure 42(d) is the result of one-hot bi-directional RNN, Figure 42(e) is the result of one-hot bi-directional Seq2Seq, Figure 42(f) is the result of one-hot bi-directional Seq2Seq with adaptive l2 regularization. Figure

42(a) whole input RNN shows that it produces a monotonous pattern. Figure 42(b) whole bi-directional RNNs generate more diverse scores by reflecting more and more order information in the encoding and decoding process than whole input RNNs. Figure 42(c) whole input bi-directional Seq2Seq creates more diverse flow configurations and reflects information trends about the previous state. Figure 42(d) It is a one-hot bi-directional RNN, showing that it generates clearer content information than whole bi-directional RNN. These results seem to be because the model learns its input through one-hot encoding. Figure 42(e) It is a one-hot bi-directional Seq2Seq, which generates by reflecting only clearer information about the previous state. Also, two-way reflects the two-way flow, creating more diverse music. Figure 42(f) one-hot bi-directional Seq2Seq with an adaptive l2 regularization shows not only clear information but also diversity through forward and reverse information. Also, the results obtained through adaptive l2 regularization show that some variation occurs for bits. This adjusts the regulation of existing information as much as possible, but it can confirm that various music is generated by adaptively applying it to the space that needs to be adjusted. We can also upload the generated music to the homepage and listen to the samples at the author's blog.

In Table 24, In the first step, the proposed scheme is applied to the encoder of the encoder-decoder structure. After applying the experimental method, the experiment performs to confirm the use of the existing model for the decoder part. The Top 1 error is reduced by 0.025 in duration and 0.013 in pitch compared to the first method. This seems to be due to minimizing bit errors in embedding and optimizing the information extracted through various information in decoding. In the second step, we applied the existing model to the encoder of the encoder-decoder structure, applied the proposed scheme structure to the

decoder part, and then experimented with applying the experimental method. The Top 1 error is reduced by 0.034 in duration and 0.014 in pitch compared to the first method. This can confirm that it is essential to extract information about the data efficiently. In the third step, the proposed method is applied and verified to the encoder-decoder structure. The Top 1 error is reduced compared to the first method, but the error is increased compared to the application of encoding and decoding, respectively. This seems to increase the top 1 error as the complexity increases during encoding and decoding. In the fourth step, the experiment was performed after applying the experimental method to verify using only the method proposed without using the encoder-decoder structure. We show that the Top 1 error reduces the fourth step. In the process of judging the information, it finds that having a model structure that is not too complicated can help to make an efficient decision.

Conclusion: First, we propose a new multi-way decoder module using the scheme perspective in determining information. Second, we propose a one-hot bi-directional Seq2Seq with adaptive l2 regularization. Third, we propose to use instance and group normalization in an ensemble normalization. Fourth, we propose a new embedding method that minimizes information during embedding to use as an input. We confirmed that the performance improves by suggesting a module with a scheme point of view that minimizes information errors in the process of embedding information and learns by reflecting mutual features in the process of determining information. We have room to increase performance by the combination optimization that each component place efficiently.

Chapter 4. Application Technique

4.1 Study on the Importance of Adaptive Seed Value Exploration [74]

Introduction: Initial seed values are set during training, relearning, and deployment of the deep learning model [75-81]. In the existing deep learning model, and initial random probability value is generated according to the initial seed value. The generated probabilities have a great influence on the initial position of the data preprocessing process or the learning process of deep learning. Therefore, it is necessary to train the deep learning model through seed values that are not significantly changed from the initial seed values. Because the seed value is set, it has an important initial influence on the training of the deep learning model, which affects the overall learning of the model. Therefore, setting the initial seed value is important for setting the direction of learning deep learning.

We propose a study to train the deep learning model by setting the seed value by searching for the adaptive seed value.

Proposal Method : We propose that it is important to search the seed value (key) adaptively to help the training of a deep learning model. The reason why setting the key (seed) is important is explained in three ways. In addition, the limitations of adaptively searching the seed value (key) are also described.

Firstly, we explain why it is important for search the seed value (key) adaptively to help the training of deep learning model. The seed (key) value is used to learn, relearn, or deploy. The seed value is the key that is open to the developer. In the process of distributing a model to a server using a key and inferring using new data, the key can be inferred from a key that has not been

trained at all. Through this, the result of the misclassified model can be obtained. Therefore, the process of deploying and inferring the model on the server by setting the same key is necessary. Also, it is necessary to encrypt the key so that only the developer can see it and reuse it.

Secondly, We explain why it is important to search the seed value (key) adaptively to help the training of a deep learning model. Generating seed values occurs using a random function. However, what happens by using a random function is that reverse engineering using a lot of supercomputing can search the random value that is generated. That is, generating a seed value using a Qubit is much more secure in terms of security than generating a seed by a real value, so a study of generating a seed value using a random quantization function is necessary.

Thirdly, the limitations of this study are described. The first thing that a developer can see by creating a seed value is to see only the results of the model. It is difficult to search the rules of the pattern of values generated by randomness. To search a pattern from such randomness, we need to experiment with various seed values and random number generator based on a supercomputer. As a result, it is necessary to search formula to search the seed (key) value that can be adaptively used by searching the characteristics of the pattern through statistical inference.

However, in this study, two experiments were used to confirm through a simple toy sample. First, we compare each CNN model using three seed values. Second, we compare the performance after training two models according to the data size of the four inputs. The experiment described the average result value through five experiments. This experimented with four datasets. The model used for the experiment was conducted using three CNN models with different CNN

layers.

Table 25. Evaluation of experimental performance using three seeds (1, 500, and 999) on a small CNN model

(a)	MNIST		Fashion MNIST		Cifar10		Cifar100	
Train	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
RELU	2.303	0.110	2.304	0.101	2.304	0.103	4.611	0.01
eLU	2.826	0.138	2.820	0.139	2.841	0.136	6.918	0.035
Test	Acc		Acc		Acc		Acc	
RELU	0.125		0.1875		0.1875		0	
eLU	0.0625		0.125		0		0	
(b)	MNIST		Fashion MNIST		Cifar10		Cifar100	
Train	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
RELU	2.303	0.111	2.304	0.102	2.304	0.102	4.610	0.009
eLU	2.834	0.139	2.839	0.139	2.84	0.137	6.902	0.315
Test	Acc		Acc		Acc		Acc	
RELU	0.125		0.125		0.1875		0	
eLU	0.0625		0.0625		0.0625		0	
0(c)	MNIST		Fashion MNIST		Cifar10		Cifar100	
Train	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
RELU	0.142	0.973	0.41	0.863	2.304	0.102	4.61	0.010
eLU	2.843	0.139	2.834	0.14	2.84	0.137	6.923	0.035
Test	Acc		Acc		Acc		Acc	
RELU	0.875		0.875		0.1875		0	
eLU	0.0625		0.125		0		0	

Table 26. Performance evaluation based on four input data using three CNN models at seed 1 (batch sizes 128, 86, 64, and 32)

Batch 128	MNIST		Fashion MNIST		Cifar10		Cifar100	
Train	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
RELU	1.412	0.437	0.316	0.95	2.303	0.0626	4.610	0
eLU	2.82	0.125	2.82	0.125	2.82	0.0625	6.93	0
Test	Acc		Acc		Acc		Acc	
RELU	0.125		0.125		0.0625		0	
eLU	0.125		0.125		0.0625		0	
Batch 86	MNIST		Fashion MNIST		Cifar10		Cifar100	

Train	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
RELU	2.302	0.125	0.735	0.725	2.303	0.1875	4.610	0
eLU	2.854	0.0625	2.831	0.125	2.837	0.125	6.912	0
Test	Acc		Acc		Acc		Acc	
RELU	0.125		0.775		0.1875		0	
eLU	0.0625		0.125		0.125		0	
Batch 64	MNIST		Fashion MNIST		Cifar10		Cifar100	
Train	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
RELU	2.30	0.125	0.272	0.9625	2.303	0.1875	4.610	0
eLU	2.837	0.125	2.699	0.212	2.844	0.187	6.914	0
Test	Acc		Acc		Acc		Acc	
RELU	0.125		0.9625		0.1875		0	
eLU	0.125		0.212		0.187		0	
Batch 32	MNIST		Fashion MNIST		Cifar10		Cifar100	
Train	Loss	Acc	Loss	Acc	Loss	Acc	Loss	Acc
RELU	1.424	0.437	0.322	0.925	2.304	0	4.610	0
eLU	2.816	0.125	2.305	0.55	2.847	0.125	6.897	0.125
Test	Acc		Acc		Acc		Acc	
RELU	0.437		0.925		0		0	
eLU	0.125		0.55		0.125		0.125	

Experimental Result: As shown in the results of Table 25, the results show that the variation of the final training performance occurs somewhat according to the initial seed setting. Table 26 analyzes the effect of the input data size with the same seed value. A similar performance was observed compared to the amount of change in the size of the input data. This confirms that the change caused by the change in the size of the input data according to the seed value is robust.

Conclusion: In training the deep learning model, it is important to set the initial seed value efficiently. In order to confirm the importance, this study explained for three reasons. Two experiments have shown that setting initial seed values is an important issue. Through this, it is necessary to study the security

of initial seed value and key generation using a quantum random number generator.

4.2 Comparison module about image captioning [82]

Introduction: Image captioning [83], such as sports commentary [85], video storytelling [87], and video captioning [86] is a method of training the model using image and caption data describing the images [83-87]. Image captioning is a relatively difficult problem because it needs multi-modal processing with two different data types, natural language processing for caption data and computer vision for efficient information extraction from images [84]. It is important to analyze the impact of each module in image captioning. However, none of the existing researches have dealt with the comparative analysis of each module in image captioning. Moreover, most of the existing researches does not help to analyze which module of image captioning can improve the whole performance [85-87]. From observation, we think that it is inevitable to analyze the influence of each module using quantitative and qualitative analysis. Here, we analyze the influence of five modules, sequential module, word embedding module, initial seed module, attention module, and search module, through quantitative and qualitative analysis on the modules.

The components of each module are as follows. The sequential module consists of three components, feature extraction to create feature vectors of the input image, model structure of the sequential module, and internal cell types of the sequential module. We took Resnet50 [89] and Vgg16 [88] for feature extraction, Vanilla-RNN [90] and Bi-directional RNN [91] as the model structure, and GRU [93] and LSTM [92] as internal cell types of the sequential module. To see the effect of attention [94], we analyzed the effects of using and not using attention

in the attention module. Embedding module has two components, Keras embedding and Glove [97]. In the search module, we used a beam and greedy search as components. In general, weights initialization of RNN is known to have a significant performance impact. Thus we analyzed the effects of the three weights initialization methods with normal and uniform distribution in the seed module.

We performed the comparative analysis by a component of each module using the Flicker 8K dataset and analyzed which components had an effect on image captioning with five measures. Comparative analysis by a component of each module can provide a basis for image captioning research by an understanding of the effects of each module.

Proposed Methods: Image captioning has been proposed and studied as a method of generating text by inputting visual information through a combination of CNN and RNN [83]. In image captioning, it is necessary to accurately recognize semantic relationships between image objects and the properties of objects and to generate semantically accurate text.

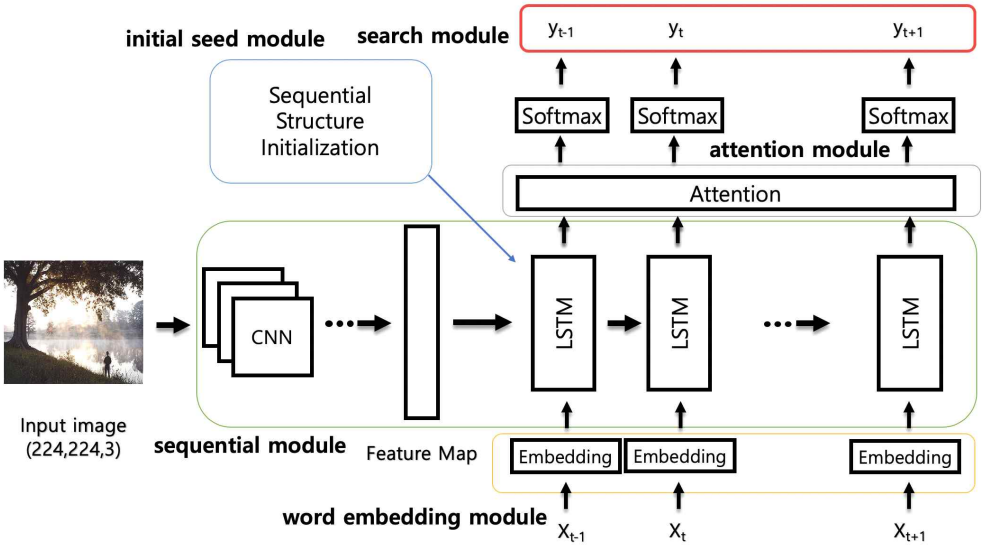


Figure 43. Comparative analysis by a component of each module of image captioning

Figure 43 shows the location of applying the comparative analysis by a component of each module. The green, yellow, gray, red, and blue boxes in Figure 42 indicate the sequential module, word embedding module, attention module, search module, and initial seed module, respectively. Image captioning combines CNN for encoding features of image and LSTM for generating caption of the input image in the sequential module.

We took Vgg16 and ResNet50 for feature extraction and Vanilla-RNN and Bi-directional RNN in the sequential module. Bi-directional RNN is reflecting two pieces of information by extracting forward and backward information in receiving the input information. To more reflect input information in sequence module, LSTM or GRU cells were used instead of basic RNN cells in our experiments.

The attention mechanism in image captioning has been studied in various ways. The advantage of attention can be seen as focusing on the input. However, the performance of image captioning may be lower if attention is incorrectly focused on image captioning. After attention, a search module is necessary for the associated related analysis of the generated caption.

Evaluation of generated sentences relationships through parsing from the generated caption is used in the field of natural language processing. We can use the beam [95] and a greedy search for the search module. Beam search improves efficiency by limiting the number of nodes to be remembered based on the best-first search technique. Greedy search [96] calculates the highest priority using the tree structure. The advantage of the search algorithm is to find the

relationship between the consecutive inputs and analyze the relationship between words.

The word embedding module is used to convert textual information into vector values so that the textual information can be reflected in the model and computed by the computer. Glove [97] is the dot product of two embedded vectors that are the probabilities of the simultaneous appearance of the whole corpus.

The initial seed module has been studied to compare and analyze the initial seed value of the sequential model of image captioning [98]. This is because the learning of the sequential model from a good starting position can predict convergence reliably.

Based on the above description of components of each module, comparative analysis by a component of each module was described as follows. We analyze the sequential module by three methods. In the first method, we tested whether two models, Vanilla-RNN with ResNet50 and Bi-directional RNN with ResNet50, can stably train the sequential information in the viewpoint of effects of RNN. We analyzed the effects of feature extraction of two models, Vanilla-RNN with ResNet50 and Vanilla-RNN with Vgg16 in the second method. In the third method, we tested the effects of the long time dependency of LSTM and GRU cells of the sequence model.

We analyze the embedding module for measuring embedding performance of textual information. We tested on the embedding through Glove and Keras embedding. To reliable convergence on the training of the sequential model, we analyzed the initial seed module of the sequential module. The initial seed is tested using normal, uniform, he, and lecun initialization methods. We attempt to analyze the attention module on the sequential module. A comparative analysis

was conducted with and without attention. To analyze the evaluation of each relation of the generated sentence, we tested on generated caption using the search module through the beam search and greedy search.

Experiments were carried out using Keras and python3 in Ubuntu 18.04. The experimental results of the impact of each module were obtained on epoch 5, and the evaluation measures are BLEU-1, 2, 3, 4, and accuracy.

Table 27. Comparison of sequential modules a) LSTM and b) GRU, i) Vanilla-RNN, and ii) Bi-directional RNN

	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Acc
a	i	0.5872	0.3626	0.1939	0.1041	0.8109
b	ii	0.6343	0.4029	0.2277	0.1082	0.8129
a	i	0.6461	0.4096	0.2349	0.1176	0.8163
b	ii	0.2921	0.17	0.0839	0.0369	0.791

Experimental Results: Table 27 shows the experimental results of the sequential module. Table 27 shows that Vanilla-RNN increased the BLEU mean score of 10.1% when using GRU cells than when using LSTM cells. On the contrary, the Bi-directional RNN dropped about 58.6% when using the GRU cell than when using the LSTM cell. In addition, when using the LSTM cell, Bi-directional RNN showed a better performance of about 12.8% than Vanilla-RNN. In contrast, Vanilla-RNN showed about 57.5% higher performance when using GRU cells. From the above results, GRU with fewer memory gates shows better performance for Vanilla-RNN than LSTM. And for Bi-directional RNNs, LSTMs with larger memory gates perform better than GRU. We think that GRU is advantageous when processing in a single direction like Vanilla-RNN. This is because GRU extracts relatively important information than LSTM. However,

when processing in both directions like Bi-directional RNN, LSTM shows better results because more detailed information is extracted.

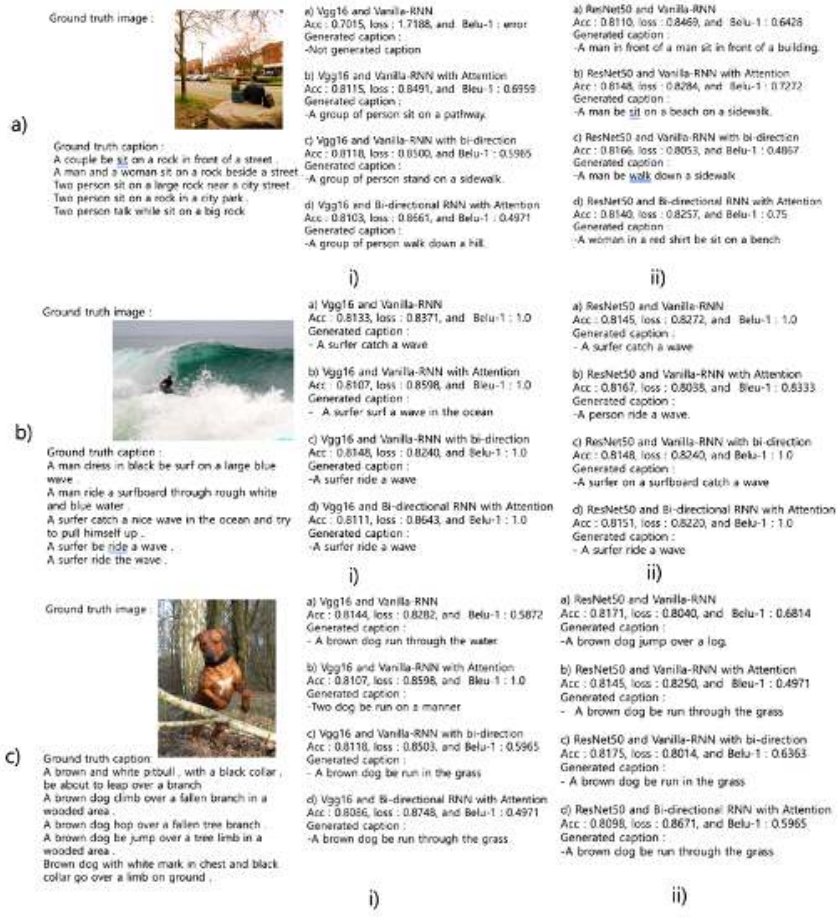


Figure 44. Comparative analysis according to the feature extraction, i) Vgg 16 and ii) ResNet50

Figure 44 shows experimental results for the qualitative analysis of the caption generated. As shown in Figure 44, ResNet50 shows better performance than Vgg16. This is because ResNet50 obtains features more efficiently using batch

normalization. In both cases of Vanilla-RNN and Bi-directional RNN, using attention is almost the same as improving performance and deteriorating performance. When using bi-direction in Vanilla-RNN, performance is improved except in one case of c)-ii). Bi-directional RNN with attention has both better and worse performance compared to Vanilla-RNN. Vanilla-RNN with bi-direction has better performance than Bi-directional RNN except for one case of b)-ii). From these results, we can conclude that the combination of Vanilla-RNN with bi-direction and ResNet50 is the best for the sequential module.

Table 28. Comparative analysis according to embedding module, a) embedding, b) Glove, i) Vanilla-RNN, and ii) Bi-directional RNN

	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Acc
a	i	0.5872	0.3626	0.1939	0.1041	0.8109
b	ii	0.2621	0.0991	0.0105	0	0.7447
a	i	0.6461	0.4096	0.2349	0.1176	0.8163
b	ii	0.1263	0.0651	0.0263	0.0081	0.7395

In Table 28, Keras embedding is superior to the pre-trained Glove. Keras embedding has 70.2% and 83.9% increase in performance over the Glove at Vanilla-RNN and Bi-directional RNN, respectively. This demonstrates that training embedding is better than pre-trained embedding for large data sets. It seems that the information learned in the pre-trained embedding causes confusion, resulting in a rather low performance. However, when the caption data is similar to the data actually learned in pre-trained Glove, it can be confirmed that the model can be trained quickly.

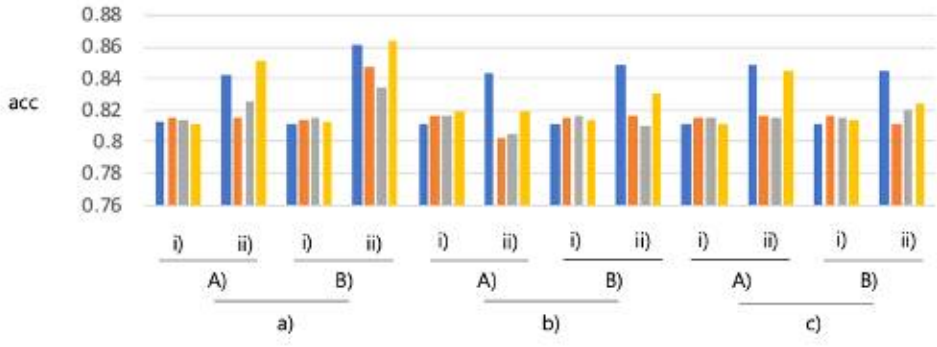


Figure 45. Comparison of components in seed module by the value of MSE a nd loss error. a) Random, b) he, c) lecun, A) normal, B) Uniform, i) Vgg16, ii) ResNet50, blue bar) CNN with Vanilla-RNN, orange bar) CNN with Bi-dir ectional RNN, gray bar) CNN with attention Vanilla-RNN, and yellow bar) C NN with attention Bi-directional RNN

Figure 45 shows the effects according to the components of initial seed modu le, which are composed of three methods, Random, he, and lecun with normal a nd uniform distribution. That is, Random normal, Random uniform, he normal, h e uniform, lecun normal, and lecun uniform. The analysis was conducted using t wo models, Vgg16 and ResNet50. The graph of Figure 45 showed the accuracy of six methods on four models, i.e., CNN with Vanilla-RNN, CNN with Bi-direc tional RNN, CNN with attention Vanilla-RNN, and CNN with attention Bi-directi onal RNN. As shown in Figure 45, it is con- firmed that the performance is the best when the uniform and random are applied at the same time.

When using Vgg16, there is no significant difference in performance dependin g on the seed method. On the other hand, it can be seen that the performance d ifference is large according to the seed method in ResNet50. In particular, CNN with Vanilla-RNN and CNN with attention Bi-directional RNN using a seed meth od of Random and uniform show excellent performance. Also, it is better to use Random rather than he or lecun generally.

Table 29. Comparison of attention modules, a) non-attention and b) attention, a) non-attention, b) attention, i) Vanilla-RNN, and ii) Bi-directional RNN

	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Acc
a	i	0.5872	0.3626	0.1939	0.1041	0.8109
b	ii	0.6318	0.4007	0.2172	0.106	0.8143
a	i	0.6461	0.4096	0.2349	0.1176	0.8163
b	ii	0.6203	0.3911	0.2162	0.1153	0.8145

We show the performance of the attention module in Table 29 by averaging results of Vgg16 and ResNet50. The attention Vanilla-RNN improved by about 8.6% compared to the non-attention Vanilla-RNN. The Bi-directional RNN fell ab out 4.6% compared to Vanilla-RNN. As you can see from the experimental resul ts, the effect of attention is unclear because it improves or deteriorates depending on the method.

Table 30. Comparison of search methods for correlation analysis of generated cap tions. a) greedy search, b) beam search, i) Vanilla-RNN, ii) Vanilla-RNN with at tention, iii) Bi-directional RNN, and iv) Bi-directional RNN with attention.

	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
a	i	0.6123	0.3774	0.1884	0.0911
b	ii	0.6305	0.393	0.2017	0.0935
a	i	0.5872	0.3626	0.1939	0.1041
b	ii	0.6318	0.4007	0.2172	0.106
a	iii	0.64	0.3999	0.2151	0.1054
b	iii	0.6461	0.4096	0.2349	0.1176
a	iv	0.6016	0.3684	0.1919	0.0977
b	iv	0.6203	0.3911	0.2162	0.1153

Table 30 shows the comparison of search methods in the search module. On ave rage, the beam search shows slightly better results than the greedy search, but it is not large. This seems to be because the beam search reflects the overall trend. However, when reflecting on some features, the greedy search may be better.
The quantitative comparative analysis of each module was conducted. First, most LSTM with bi-direction is improved performance in the sequential module. The Bi-directional RNN with attention is improved when compared to non-attention.

ResNet50 in feature extraction shows higher performances than Vgg16 in most cases. Secondly, at the seed module, the Bi-directional RNN with Random uniform shows the best performance. Third, Keras embedding in the embedding module greatly improves the performance of Bi-directional RNN when compared to Glove. Fourth, attention with Vanilla-RNN in attention module improved the performance than that with Bi-directional RNN. Fifth, bi-direction in beam search and greedy search improved the performance in the search module. Through this, it seems important to find an efficient method at each location. And optimal structure through the combination of the good component in each module seems to be important. From our analysis, a new model can be created by adjusting the combination of good components to create the optimal structure. However, the performance analysis of the new model must be conducted through work to obtain generalized performance because the combination does not guarantee the overall performance of the model.

Conclusion: We analyzed the effect of the modules of image captioning. Analysis of the effects on the sequential model showed that the Bi-directional RNN was slightly better than the Vanilla-RNN. This is because the interactive reflection of subtitle information is well trained in context. Impact analysis of the attention shows that the attention Vanilla-RNN is beneficial for performance because it focuses on the part of the input word relative to the word to be predicted. It is a search module for evaluating the correlation between the generated results, the beam search module performs better than the greedy search module. In analyzing the impact on embedding, Keras embedding showed better performance than pre-trained Glove. The comparative analysis of the feature extraction showed that the ResNet50 is higher than the Vgg16 in terms of image captioning and features. In the case of the seed methods of the sequential model, it can be seen that the seed value of the Random uniform efficiently reflects the sequential information. From this analysis, we can design more effective models for image captioning.

4.3 Visualization about anomaly data [99]

Introduction: It is essential to detect outlier detection inside the deep learning model. When the deep learning model detects outlier data, much variation occurs in

the model's parameters because the model's parameters tend not to have or contai n information about the outlier data. This transformation of the model with nume rous parameter information results in many misclassifications as the model infers new data. As a result, it is essential to visualize and classify these outlier data i n advance so that the model's parameters do not vary much. We studied a new visualization method to detect outliers efficiently. Contributions are as follows. W e propose a visualization technique that combined LBP LLE with Smote for outli er data detection. Second, we propose a confusion visualization method using sim ilarity of pixel density distribution. Third, we propose a histogram visualization u sing the frequency of position of pixel distribution in EDA (Exploratory Data An alysis).

Proposed Method : We show three visualization techniques with existing method s. First, we propose a combined LBP [101] LLE with SMOTE [102]. Second, w e propose the pixel similarity visualization technique. Third, we propose a pixel f requency visualization technique. Figure 46(a) shows the UMAP [100] method, w hich is the existing visualization method. Figure 46(b) shows the combined LBP LLE with Smote sampling proposed. The existing UMAP method is a visualizati on method that is not suitable for outlier detection. The reason for this is that th e clustering and visualization of outlier data are uniformly represented in one clu ster. However, in case of Figure 46(b) proposed, the information having similar data clustering degree is tilted, so the data that having similar characteristics has the advantage of being expressed outside clustering. Through this, we can see tha t the proposed method is efficient in determining outlier data. Figure 46(c) shows the proposed method for visualizing outlier detection data using the density of th e value distribution. The difference in pixel value density compares outlier data with that of existing data, allowing for more precise comparisons. Figure 46(d) s hows that histogram visualization using frequency of position of pixel distribution in EDA compares the similarity of the location of pixel distribution. Each techni que has been described in detail. Figure 47 shows four methods for experimental verification of the proposed method. Figure 47(a) is the existing method. Figure 47(b) is applied to LLE and SMOTE [102]. This is to observe the effect of corr ection when the amount of information in the feature map is unbalanced through

linear sampling. Figure 47(c) is the method of applying Sampling and Linear Co mbination after applying LBP (Local Binary Pattern) [101]. This is to see the eff ect of linearly correcting the imbalance of the sampling result when converting t o a binary feature before sampling.

Firstly, combined LBP LLE SMOTE is a method to efficiently show the characte ristics of the relations between data through information imbalance and binary pr ocessing.

Secondly, we propose a method of visualizing the characteristics of data using th e frequency of pixels. The number of pixels of a pixel held by r, g, and b in t he existing image is available. The frequency of these pixels is more robust to t he two types mentioned above because the frequency of occurrence of the pixel is checked regardless of the change in value or the change of position.

Finally, Figure 48 is a visual analysis of the similarity analysis of pixel density. Figure 48 is to check the existing values of the image and the accuracy of the c orrect rendering. This method is a visualization method using a confusion matrix generated by representing each pixel position according to the position value of t he pixel and expressing the similarity as a confusion matrix. The similarity was calculated using Pearson's correlation coefficient. This has the advantage of expre ssing the accuracy according to the position value of the pixel better. The heat map expressed above is the result of fitting the x and y axes to the size of the pixel data.

We compare existing MNIST data with images generated in GAN for a new vis ualization method. The generated image was generated using three GAN models of Vanilla GAN, DRAGAN [103], and EBGAN [104]. Fig 51 shows the existing image and the generated image. Figure 51(a) shows some samples of existing M NIST. Figure 51(b) is the result of Vanilla GAN. Figure 51(c) shows the result from DRAGAN. Visualization analysis was performed using the generated image and actual data as above.

We first compared the experiments with combination LBP LLE SMOTE on actua l data. Fig 52 shows the results of each experiment described.

To verify the effects of the first method proposed, we use 5 steps. Fig 52(a) sh ows the actual data, Fig 52(b) shows Smote applied to LLE after the LBP appli cation. Figure 52(c) shows the case of circular convolution [106] in convolution

when Smote applied to LBP. Figure 52(d) shows the initial case when Smote is applied to LBP and Uniform [105] when Figure 52(e) consists of the initial case where Uniform is applied when Smote not applied to LBP. In Figure 52(b), the result of Smote was
corrected in sparse areas compared to Figure 52(a) and Figure 52(c) includes the circular convolution, so that the more accurate corrected result confirmed by redu cing the error. Figure 52(d) makes uniform zeros by applying uniform to zero. F igure 52(e) shows the effect of reducing the complexity of the visualization distri bution representation of the data as Smote is applied compared to Figure 52(d).
Fig 53. compares and analyzes the results for the generated images, which is diff erent from the method shown. above. When look at the generated visualization i mage, we can see that the distribution tendency of the generated image overlaid near the zero value. That is because the generated image is generated based on t he normal distribution between 0 and 1. In the previous experimental analysis res ults of Figure 52., similar results are shown as each method is applied. However, in the case of Figure 53., it is the result of the generated data. The process of visualizing and determining the fake image shown than the actual data. We can see that it is a visualization method.
Analysis of existing smote methods about spatial information data distribution. Fi g 8. analyzes the impact of visual representations on changes in K values. The l arger the K value, the more compressed the image to be expressed can be seen. Such compressed images may not adequately show the representation of the data. We found that finding an appropriate K value for visualization can represent an efficient feature. Therefore, we found that finding an appropriate K value is an i mportant issue.
Conclusion : We proposed three visualization techniques. Combined LBP LLE S MOTE the advantages of generalized data visualization by correcting unbalanced data characteristics through sparse data correction. Through this, we can check th at it is efficient in fake image discrimination. Also, the Visualization of pixel de nsity similarity has an advantage of efficiently detecting when pixel position infor mation is wrongly generated through similarity analysis of pixel positions and sh owing correlation information of pixel information. Visualization of pixel density frequency shows the advantage of extracting fake pixels through the frequency of

pixel values generated through density distribution analysis. The proposed metho d can be used as a method for the detection of real-world images. Through this, we can confirm that the visualization method that analyzes outlier data should be studied to discriminate data similar to actual data.

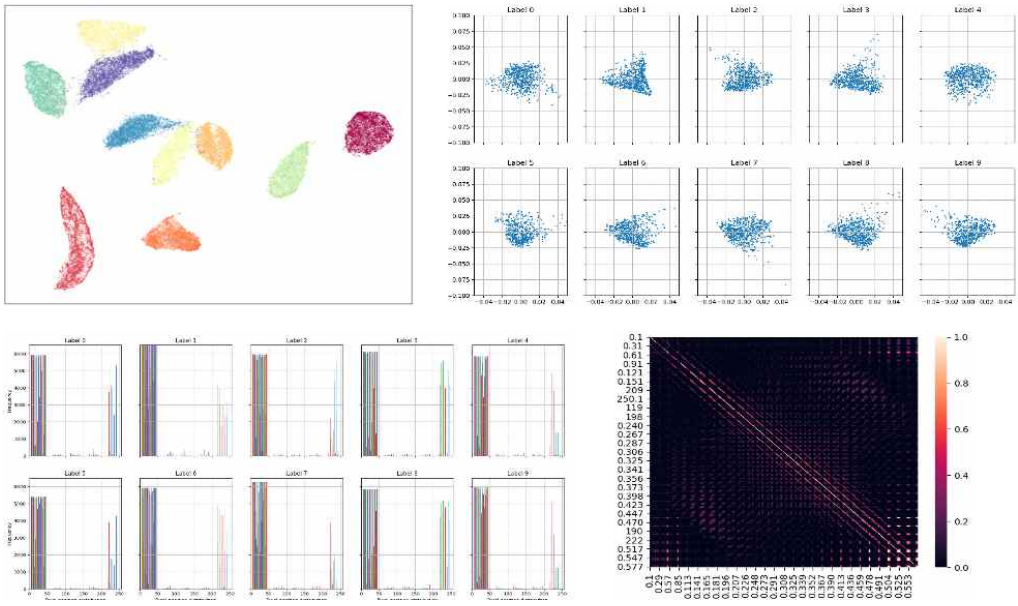


Figure 46. Existing visualization technique and proposed visualization techniques. a) UMAP [100], b) Combination LBP LLE SMOTE, c) Pixel similarity visualization technique using pixel density distribution. Pixel frequency visualization technique using position of pixel density distribution in EDA.

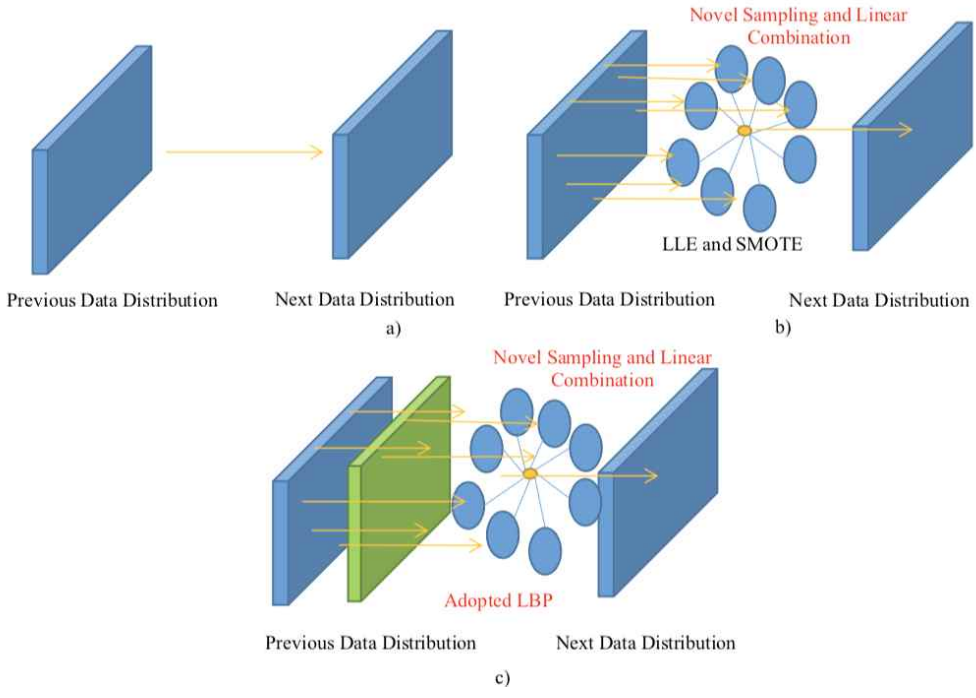


Figure 47 Experimental verification of combination LBP LLE SMOTE. a) Existing method, b) Existing method with SMOTE sampling, c) Apply LBP method before applying the existing method with SMOTE sampling, d) Apply LBP method after applying the existing method with SMOTE sampling

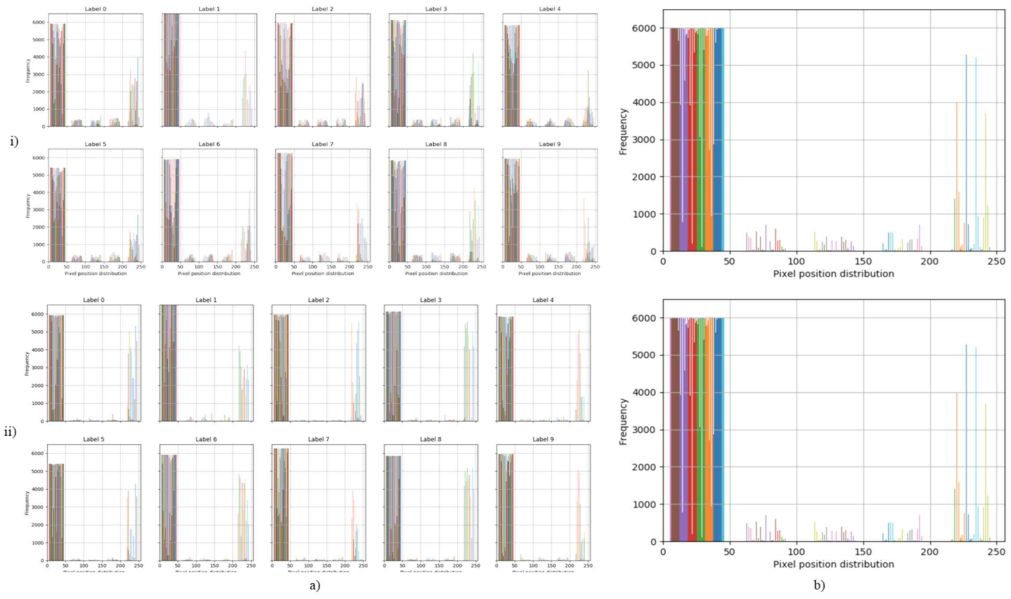


Figure 48 Visualize pixel density frequency of data, a) Accurate analysis for each class, b) Accurate analysis for whole class.

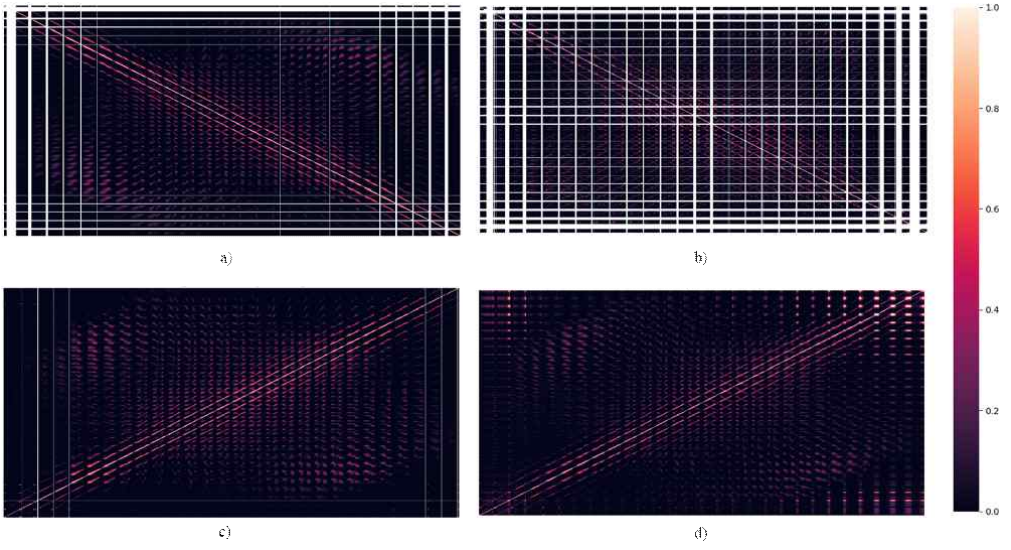


Figure 49 Visualize pixel density similarity of existing and generated data, a) Ground truth, b) Vanilla GAN, c) DRAGAN, d) EBGAN

Figure 50 Comparative analysis of existing and generated data. a) Ground truth MNIST, b) MNIST generated from Vanilla GAN, c) MNIST generated from DRAGAN.

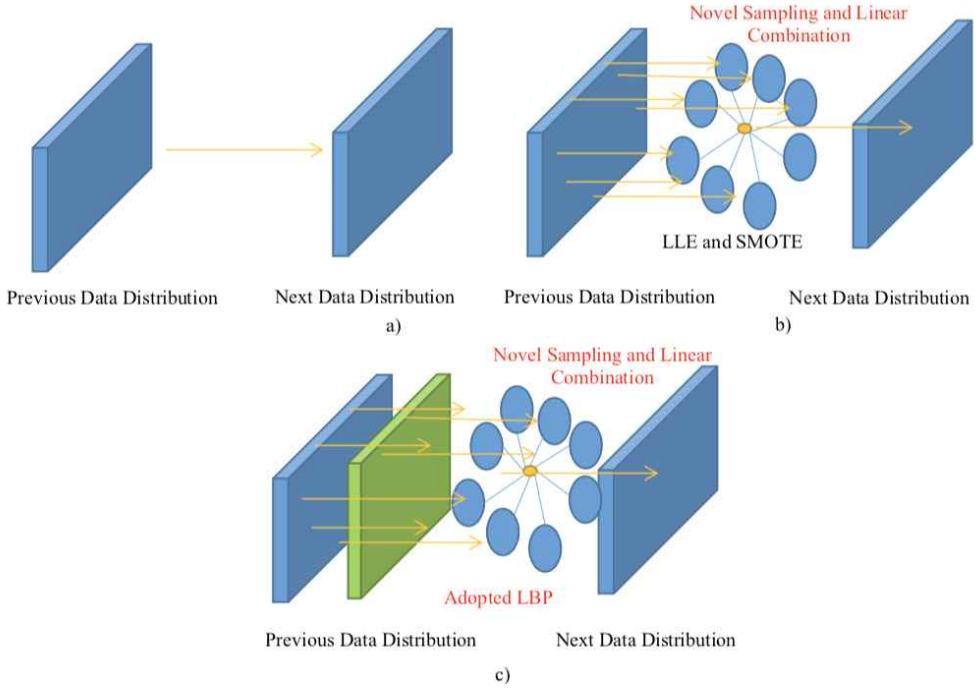


Figure 51 Experimental results to verify the combination LBP LLE SMOTE from a) Ground truth data. b) Ground truth LBP smote, c) Ground truth CLBP mote, d) Ground truth UCLBP smote, e) Ground truth UCLBP no smote.

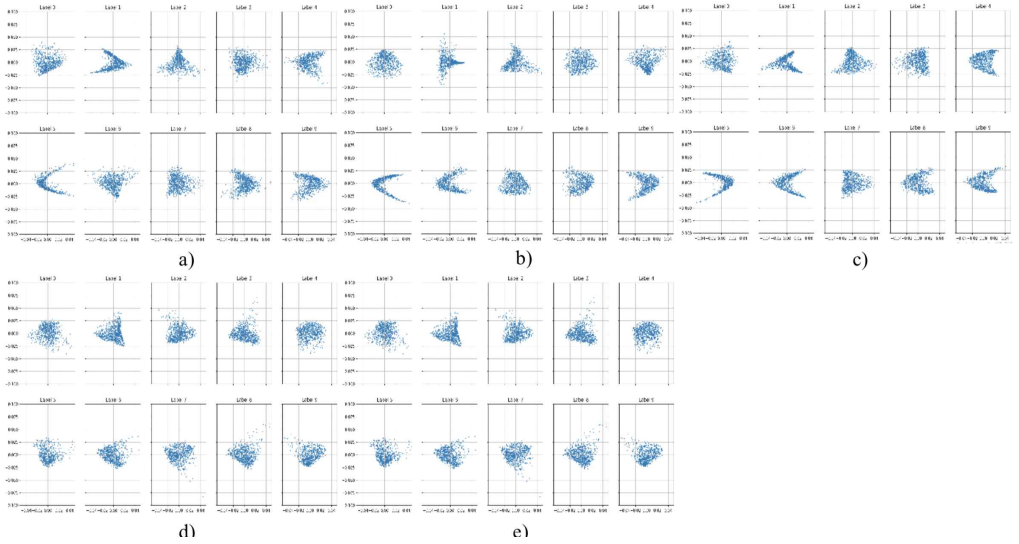


Figure 52 Comparison of changes according to the techniques applied in the first proposed method a) *Generated LBP smote, b) Generated CLBP mote, c) Generated UCLBP smote, d) Generated UCLBP no smote*

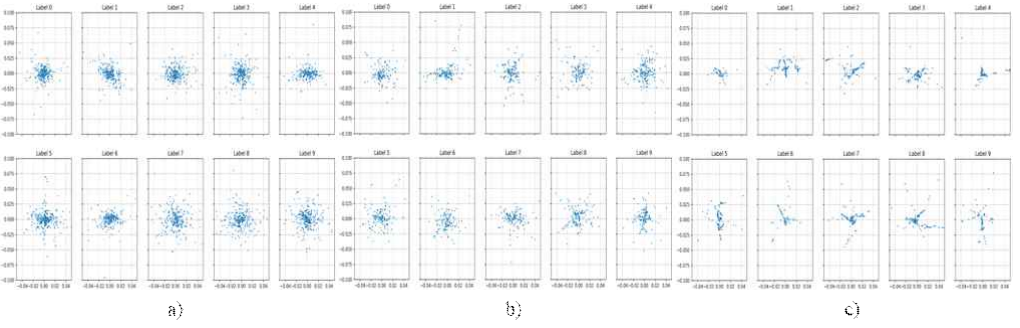


Figure 53 Comparative experiment according to K value in Smote technique. a) K=4, b) K=*16, c) K=41.*

4.4 Stable Acquisition of Fine-Grained Segments using Batch Normalization and Focal Loss with L1 regularization in U-Net Structure [107]

Introduction : Most images consist of various sizes, shapes, and textures. Therefo

re, it is necessary for specific image processing to operate appropriately under im ages with various characteristics. Since the textures of clothing images are very f ine-grained and diverse, acquisition of fine-grained segments is particularly import ant. Finally, obtaining accurate semantic segments of images is an important goal to achieve in semantic segmentation researches [108].

However, there are not many studies that explore the acquisition of fine-grained segments of semantic segmentation in images. Most existing researches on image semantic segmentation has adopted fully convolution network (FCN) [109]. Howe ver, their segmentation results were not good at finding fine-grained segments, se mantic segments because the convolution of FCN didn't maintain spatial informat ion. Even though there were related works on fully convolution network-condition al random field (FCN-CRF) [109], FCN using atrous convolution, and FCN using the skip diagram in the FCN up sampling process to improve the performances o f FCN, most of FCN showed poor results. In the methods, obtaining various size s of segments are not reflected in model training. Therefore, obtaining various si zes of fine-grained segments is a crucial factor in processing segmentation for th e acquisition of fine-grained segments in images.

To overcome the problem, we adopt two additional components on a U-Net bas ed structure for acquisition of fine- grained segments of semantic segmentation. T he first additional component is the use of normalization at all layers in the train ing process. Normalization alleviates the variation of multi-scale information, espe cially in the multi-scale processing of U-Net. We took the well-known batch nor malization (BN) in all U-Net layers. As second additional component, we take th e model prediction correction using focal loss with L1 regularization in training. Existing loss of cross-entropy does not reflect the fine-grained segments because it does not balance the model prediction space. From observation, we consider th e model prediction correction for fine-grained segments and propose a novel loss composed of focal loss with L1 regularization. As a result, we introduce a novel structure based on U-Net that trained with BN and a devised novel loss.

To measure performances of our method and previous meth- ods, we experimente d with three models such as U-Net, Attention U-Net [1100], and U-Net BN incl uding the existing FCN models on the ATR dataset [111]. U-Net BN refers to a model that applies BN to all layers of the existing U-Net. We also tested our m

ethod with various combinations of loss and provided their results with some me asures such as intersection over union (IOU), precision, and recall. From extensiv e experiments, we find that our method with two additional components have ge nerated correct fine-grained segments, especially in small and complex textures su ch as hands, feet, and glasses. Also, the overall performances of our method wer e considerably better than those of the existing methods based on FCN in terms of accuracy using IOU, precision, and recall measures.

Proposed Method : We propose a U-Net based semantic segmentation method for acquisition of multi-scale fine-grained segments in semantic segmentation with tw o additional components. U-Net is a well-known model that can find fine-grained information through efficient reflection of multi-scale information. However, sema ntic segmentation using only U-Net does not find fine-grained segments, especiall y for the images composed of various shapes and textures. To overcome the limi tation, we adopt two additional components, BN to efficiently learn stabilized mu lti-scale fine-grained segments and model prediction correction using focal loss wi th L1 regularization. We found from our extensive experiments that BN was a cr ucial process in finding multi-scale fine-grained segments because the normalizati on of data was more important to precisely calculate fine-grained segments for m ulti-scale data in all layers. Also, we use a combining method of focal loss and L1 regularization to balance the model prediction correction that enables the seg mentation to stabilize acquired fine-grained segments in multi-scale data. We conf irmed the results through extensive experimentation in Section 4

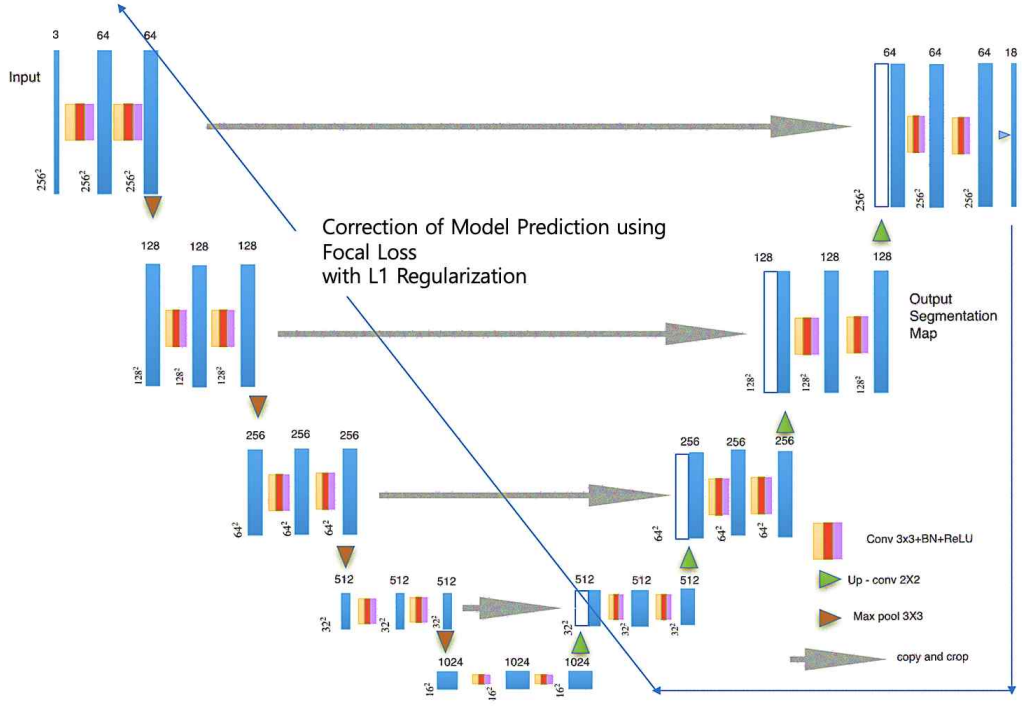


Figure 54 Proposed U-Net structure.

Figure 54 shows the structure of our U-Net based semantic segmentation. The ov erall structure is nearly the same as the original U-Net except for the BN of all layers, depicted as yel- low arrows and model prediction correction using focal l oss with L1 regularization. s2 , where s = 16, 64, 128, and 256 represents the width x height of filter maps. The number written above the blue block is the n umber of filter maps for the block. The gray arrow represents a skip diagram, in which the information of an existing block translated and transformed into a whit e block in the blue frame. The green, orange, and yellow arrows represent the u p-sampling convolution, the 3x3 max pooling, and the 3x3 convolution, respective ly. The proposed focal loss with L1 regularization is applied to the outputs of U -Net, and the loss is used to train our sturcture. As mentioned before, the BN in all layers and the focal loss with L1 regularization enable our structure to acquir e multi-scale fine-grained segments. Acquisition of fine-grained segments through BN on all layers can be calculated more precisely because the same size can be calculated in all spaces. To show the performances of our method with combinati

on of regularization coefficient parame- ters, we extensively experimented and co mpared the results in Section 4

Experiment Environment : The experimental environments are as follows. We co nducted our experiments with a learning rate of 0.001 and a batch size of 4. We chose an ATR dataset of clothing images because they are composed of various sizes, shapes, and textures and are therefore adequate to get fine-grained segment s. Background images excluded during the evaluation. In order to evaluate the pe rformances with various measures, we used recall, precision, F1 score, and IOU. In all experiments, we used zero-padding to make all the im- ages the same size as the long side of the image. We compared our method with the existing metho d of FCN and experimented on three U-Net models: U-Net without BN and foca l loss with L1 regularization, U-Net BN and focal loss with L1 regulariza- tion, and attention U-Net using focal loss with L1 regularization. Attention U-Net has an U-Net structure with an attention gate. To verify focal loss with L1 regulariza tion for optimization using regularization, we compared the performance of existi ng cross-entropy and proposed method, that is, focal loss with L1 regularization. In experiments, we tested proposed focal loss with L1 regularization, focal loss with L2 regularization, and focal loss with L1 and L2 to compare the regularizat ion. L1 and L2 regularization coefficient are added at a ratio of 0.5.

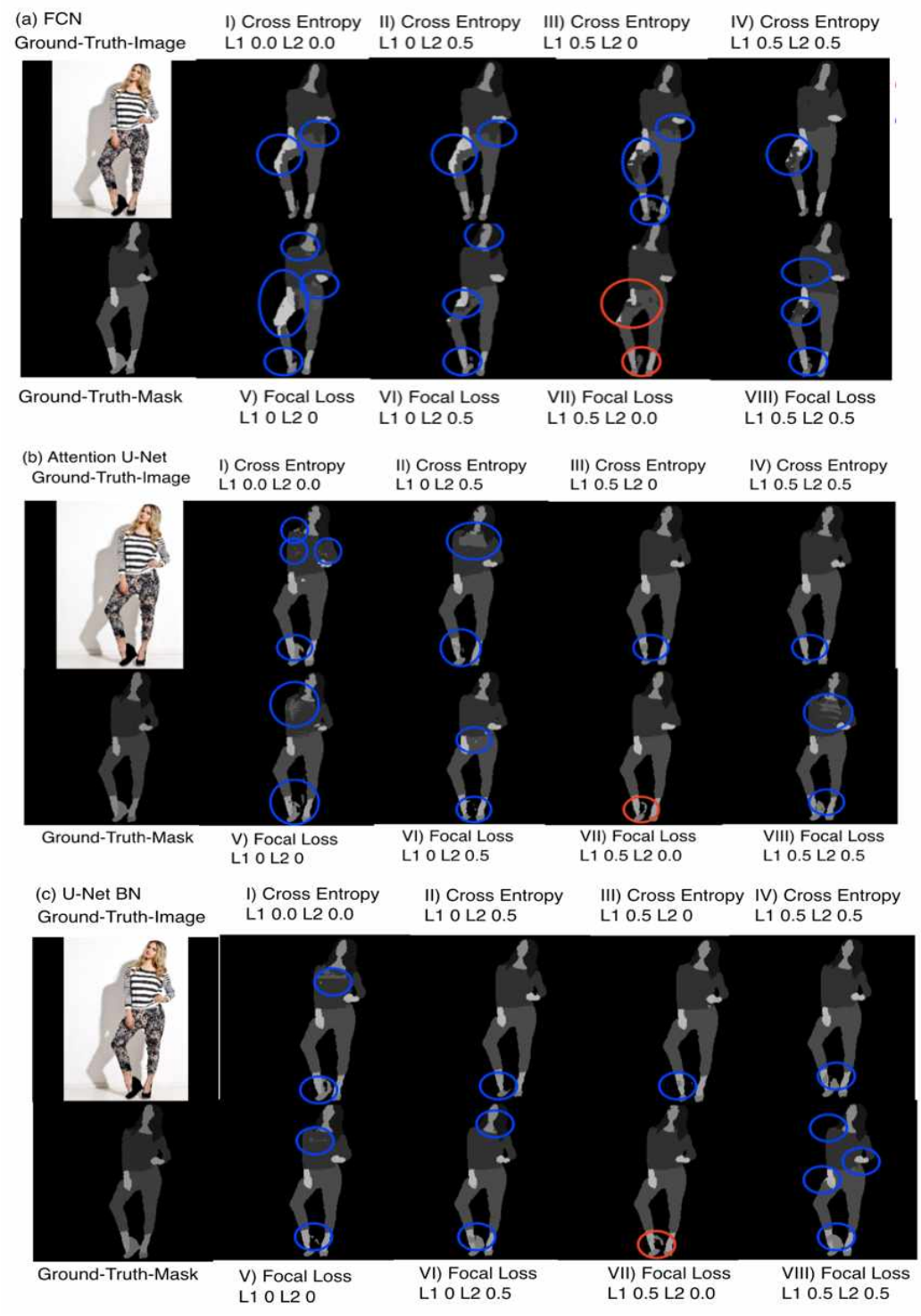


Figure 55. Result of focal loss regularization model. a) FCN, b) Attention U-Net,

c) U-Net BN, I) Cross-entropy with L1 0.0 and L2 0.0 regularization coefficient, II) Cross-entropy with L1 0.0 and L2 0.5 regularization coefficient, III) Cross-ent ropy with L1 0.5 and L2 0.0 regularization coefficient, IV) Cross-entropy with L 1 0.5 and L2 0.5 regularization coefficient, V) Focal loss with L1 0.0 and L2 0. 0 regularization coefficient, VI) Focal loss with L1 0.0 and L2 0.5 regularization coefficient, VII) Focal loss with L1 0.5 and L2 0.0 regularization coefficient, VI I) Focal loss with L1 0.5 and L2 0.5 regularization coefficient

Experimental Results and Discussion: We tested the performances of acquisition o f a fine-grained segment of semantic segmentation using previous cross-entropy l oss and proposed focal loss with L1 regularization on three semantic segmentatio n models such as FCN, Attention U-Net, and U-Net BN on the clothing images. Figure 54 shows the experimental results of the three models on four combinatio ns of regularization coefficient parameters. As mentioned before, we choose the t he regularization coefficient parameters combining 0 and 0, 0 and 0.5, 0.5 and 0, 0.5 and 0.5. In Figure 54, the blue and red circles are incorrect predictions, and red circles are the best among incorrect predictions. As you can see, the most re sults using focal loss with L1 regularization are better than those using the cross -entropy loss, especially on fine-grained segments shapes such as hands, foods, s hoes, and neck.

The best result of all experiments is shown in panel (VII) of Figure 55, which u ses focal loss with L1 regularization. That is, the L2 regularization is not helpful for the acquisition of fine-grained segments of semantic segmentation. L1 regulari zation was more robust than L2 regularization concerning the outlier. Many fine-grained segments in the images with various sizes, shapes, and textures are outlie rs. Therefore, L1 regularization is better than L2 regularization for the acquisition of fine-grained segments of semantic segmentation.

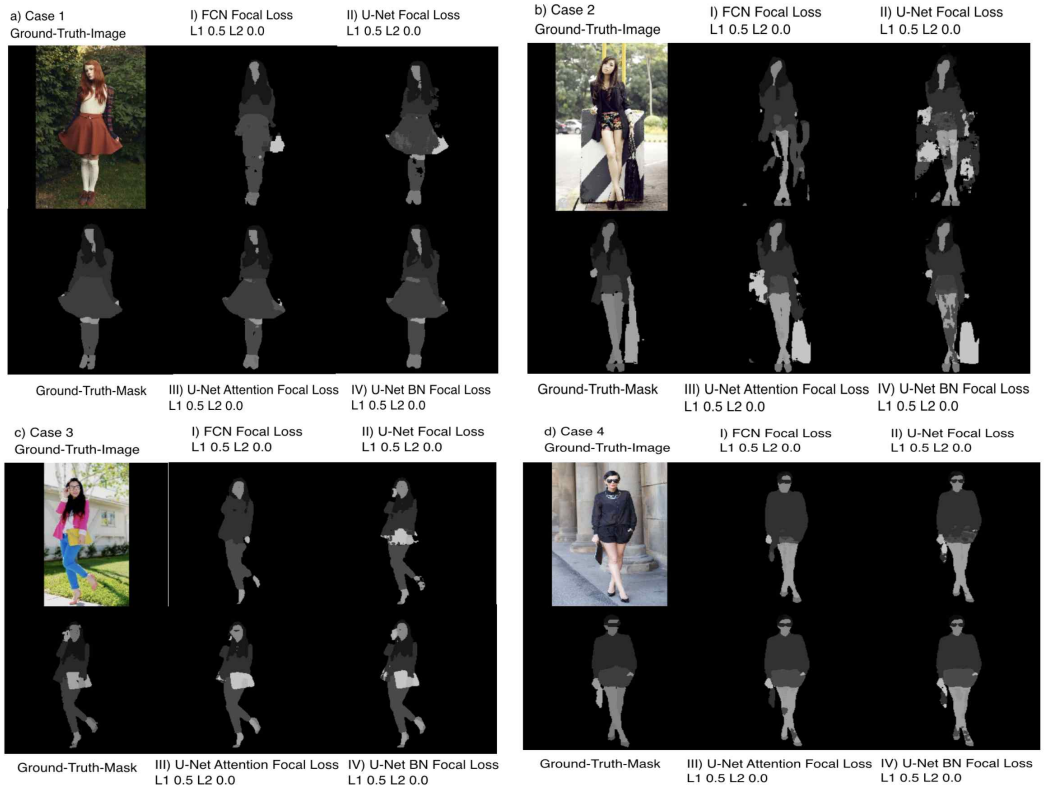


Figure 56. The proposed method is focal loss L1 0.5 regularization coefficient. T he proposed method is shown using four cases of figure 56.a, figure 56.b, figure 56.c, and figure 56.d. I) FCN, II) U-Net, III) Attention U-Net, IV) U-Net BN on focal loss with L1 0.5 regularization coefficient for each case.

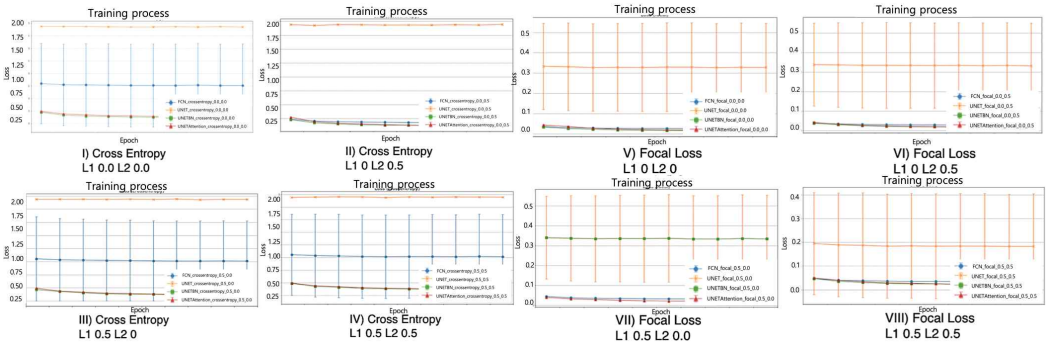


Figure 57. Comparison with or without BN about two loss function types and va rious regularization coefficients in loss function within training time I) Cross-entr opy with L1 0.0 and L2 0.0 regularization coefficient, II) Cross-entropy with L1

0.0 and L2 0.5 regularization coefficient, III) Cross-entropy with L1 0.5 and L2 0.0 regularization coefficient, IV) Cross-entropy with L1 0.5 and L2 0.5 regulariz ation coefficient, V) Focal loss with L1 0.0 and L2 0.0 regularization coefficient, VI) Focal loss with L1 0.0 and L2 0.5 regularization coefficient, VII) Focal loss with L1 0.5 and L2 0.0 regularization coefficient, VII) Focal loss with L1 0.5 a nd L2 0.5 regularization coefficient

The attention U-Net and U-Net BN are better than the FCN. Because the U-Net stably acquires multi-scale fine-grained segments. Of the three methods, the U-Ne t BN shows the best results of all experiments because the normalization of all l ayers helps find fine-grained segments. From these results, we can ascertain that the focal loss with L1 regularization and the normalization is useful for the acqu isition of fine-grained segmentation. In our experiments, we tested the intrinsic U -Net structure with cross-entropy loss, but the training loss does not decrease, as shown in the result Figures 56 and 57. As a result, the U-Net BN using focal l oss with L1 regularization shows excellent results.

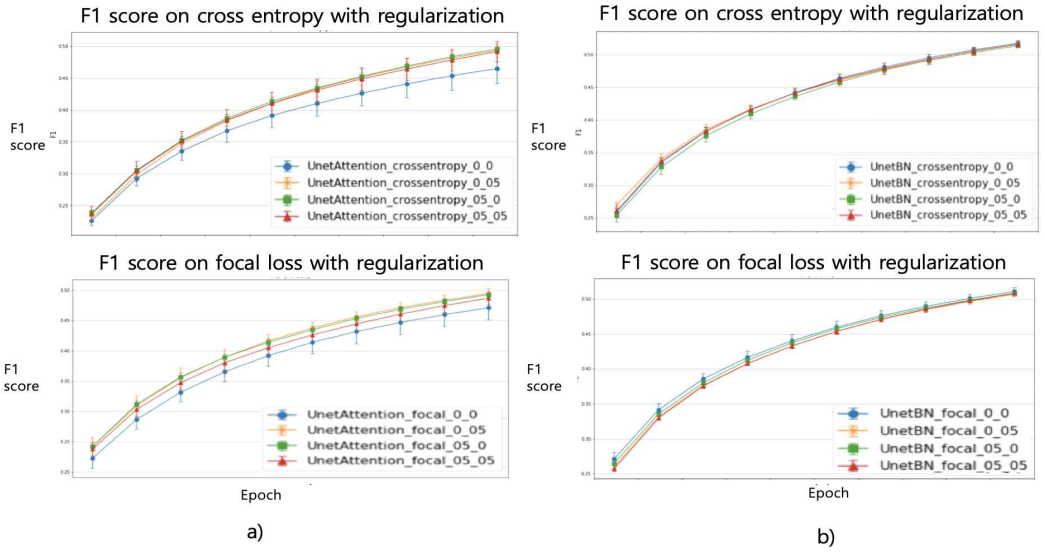


Figure 58. Comparison of F1 score according to addition of attention gate on se gmentation model within training time, a) In the attention gate included, b) In th e attention gate non included

To ensure whether the U-Net BN shows the best result for various clothes image s, we tested four methods on four clothing images, as shown in Figure 58. In th ese experiments, we added the results of U-Net using focal loss with L1 0.5 reg ularization. Figure 58 shows the experimental results of four methods on four im ages. As shown in Figure 58, U-Net BN using focal loss with L1 0.5 and L2 0. 0 regularization shows the best results for the leg, skirt, and neck. The results of Figure 58(a) are the worst because the colors in the background are similar to th e colors wrong by the women. Even the U-Net BN using focal loss with L1 0.5 and L2 0.0 generates the best quality because the localization effect of normaliza tion makes it possible for the method to better distinguish the colors in the back ground from those of the women. In Figure 58(c), the U-Net BN using focal los s with L1 0.5 and L2 0.0 regularization show the best results, especially on the small parts of the shoes, when compared to the other methods. The U-Net BN u sing focal loss with L1 0.5 and L2 0.0 regularization in Figure 58(d) generates more precise fine-grained segments in the area of the sunglasses than the other methods and even better than the ground-truth mask.

To more specifically analyze the effects of BN with two loss and two loss with some regularization coefficients on the ATR dataset, we showed the loss for the four models in Figure 56. Orange line indicates loss of U-Net structure. As you can see, the loss of the U-Net structure showed inferior results and nearly did n ot decrease. The intrinsic U-Net with cross- entropy and without normalization d oes not work well for fine- grained segments of semantic segmentation. The vari ation of loss of intrinsic U-Net is not very large in the cases of cross- entropy l oss but is quite large in the focal loss because the model prediction correction o f focal loss is sometimes successful in training the intrinsic U-Net.

FCN showed poor performances in the case of L1 0.5 regularization. In the proc ess of obtaining the segments in the FCN, the calculated difference between the predicted value and the correct value is not reflected in the process of reflecting the difference value. The variation of loss of U-Net BN becomes too large when L1 0.5 and L2 regularization. The L1 regularization reflects the differences that h ave diverse values according to experiments. Therefore, it is crucial to reflect the difference value efficiently. Overall, the methods with focal loss showed better re sults than those with cross-entropy loss. As shown in panel (VII) of Figure 56, t

he methods using both L1 and L2 regularization showed the worst performances. Because too much regularization provides a reverse effect.

We analyzed the influence of the attention gate in terms of an F1 score. Figure 57 showed the results of experiments: Figure 57(a) is with attention gate and Fig ure 57(b) is without attention gate. The result of U-Net without the attention gat e was higher than U-Net with the attention gate as shown in Figure 57. Because the attention gate in the initial steps of training tries to find the strongest charact eristics in the images, but it find erroneous characteristics.

Table 31. Comparison of both focal loss about U-Net models, a) U-Net, b) Atten tion U-Net, c) U-Net BN, (i) L1 0.0 and L2 0.0 regularization coefficient, (ii) L 1 0.0 and L2 0.5 regularization coefficient, (iii) L1 0.5 and L2 0.0 regularization coefficient, (iv) L1 0.5 and L2 0.5 regularization coefficient

		IOU	Precision	Recall
a)	i	0.496 ± 0.0	0.001 ± 0.0	0.0 ± 0.0
	ii	0.496 ± 0.0	0.001 ± 0.001	0.0 ± 0.0
	iii	0.632 ± 0.011	0.344 ± 0.029	0.333 ± 0.004
	iv	0.496 ± 0.0	0.003 ± 0.002	0.0 ± 0.0
b)	i	0.715 ± 0.008	0.536 ± 0.002	0.420 ± 0.002
	ii	0.724 ± 0.005	0.555 ± 0.005	0.444 ± 0.013
	iii	0.723 ± 0.002	0.554 ± 0.006	0.441 ± 0.004
	iv	0.723 ± 0.005	0.548 ± 0.012	0.437 ± 0.013
c)	i	0.731 ± 0.002	0.565 ± 0.007	0.464 ± 0.005
	ii	0.728 ± 0.004	0.564 ± 0.004	0.458 ± 0.004
	iii	0.729 ± 0.003	0.567 ± 0.003	0.459 ± 0.007
	iv	0.730 ± 0.001	0.564 ± 0.006	0.461 ± 0.001

Table 31 shows the impact of four regularization methods along with IOU, preci sion, and recall measurements for three models: U-Net, attention U-Net, and U-N et BN. The values shown in Table 31 are the mean and standard deviation of th e results obtained three times on the same model. As shown in Table 31, the ov erall performance of U-Net is worse than those of the other models. However, th e U-Net with L1 0.5 and L2 0.0 regularization Table 31, which showed about 2 3% improvement over U-Net with other regularization methods. We think that th e regularization with L1 0.5 and L2 0.0 coefficient is effective in the attention o

f U-Net. From the results, the attention is not helpful, and the performance of U -Net BN is improved by about 22% over those of U-Net. Therefore we can be deduced that attention gate in the process of learning induces learning in the wro ng direction. From these results, it is confirmed that a focal loss with L1 0.5 an d L2 0.0 regularization proves useful for the acquisition of fine-grained segments in semantic segmentation.

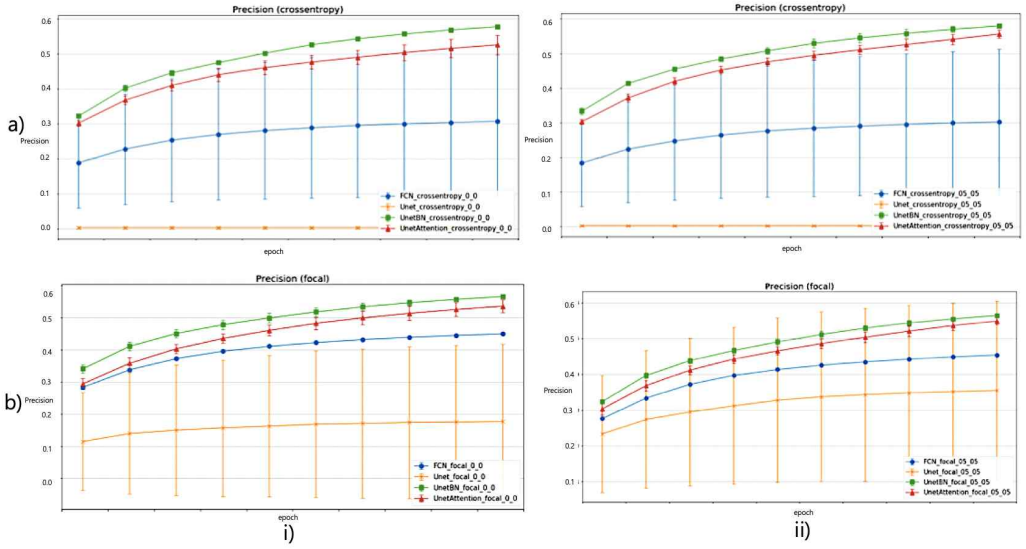


Figure 59. Comparison of regularization effect using non regularization and regul arization with L1 and L2 regularization. a) Cross-entropy loss, b) Focal loss, i) L1 0.0 and L2 0.0 regularization coefficient, ii) L1 0.5 and L2 0.5 regularization coefficient

Figure 59 shows the regularization effects of cross-entropy and focal loss using n on-regularization and regularization with L1 and L2. This confirms that regulariza tion does not significantly affect on cross-entropy results of U-Net. However, the performance of the acquisition of fine-grained segments of semantic segmentation is improved when the regularization is combined with focal loss. The focal loss has a term that reflects the reverse of the probability from the model prediction. That means that model correction prediction is more affected by the focal loss, u nlike the existing cross-entropy.

Conclusion: We proposed a method based on the U-Net structure with two additi onal components to the acquisition of fine-grained segments. For the acquisition of fine-grained segments, we added normalization to all layers of the U-Net struc ture and proposed a combined component composed of focal loss with L1 regula rization. We experimented with proposed methods on an ATR dataset and analyz ed their results. Experimental results showed that the proposed methods were bett er than the previous FCN and intrinsic U-Net. These results allowed us to know that the U-Net was a structure for semantic segmentation, adopted normalization about all layers on the U-Net, was beneficial for semantic segmentation, and the model prediction correction using focal loss with L1 regularization was good at a cquiring the fine-grained segments in semantic segmentation. In the future, we wi ll proceed with the semantic segmentation structure using the generative model to obtain a more robust acquisition of fine-grained segment of semantic segmentatio n.

Chapter 5. Conclusion

We studied new technology and the application of deep learning to improve t he performance of deep learning. There are 7 types of research—first, the Bi-acti vation function: an enhanced version of an Activation function in Convolution N eural Networks. Second, Scale calibration cascade the smooth loss of generative a dversarial networks with online continual task learning. Third, Nonlinear Exponen tial Regularization: An Improved Version of Regularization for Deep Learning M odel. Fourth, Novel Auxiliary Components to Help Optimize Deep Learning Mod el. Fifth, Ensemble Normalization for Stable Training. Sixth, Similarity Analysis of Actual Fake Fingerprints and Generated Fake Fingerprint by DCGAN.
Seventh, it is a Multi-Way Decoder Scheme with Error Reduction Embedding on one-hot bi-directional Seq2Seq with Adaptive Regularization for Music Compositi on.

The contents of research to apply deep learning in real life are composed of fo ur types—first, Study on the importance of adaptive seed value exploration. Seco nd, a comparison module about image captioning, Third, visualization about anom aly data. Fourth, stable acquisition of fine-grained segments using batch normaliza tion and focal loss with l1 regularization in the U-Net structure.
Through the above, new technologies and application fields studied in deep learni ng were studied.

In the future, we would like to study the following as a future study of deep learning. First, deep learning research using system biology, second deep learning research using parameter control, third deep learning research using complexity th eory, fourth, we would like to proceed with a study on the understanding of dee p learning.

Bibliography

[1] H.-M. Huang, W.-K. Chen, C.-H. Liu, and S. D. You, "Singing voice detection based on convolutional neural networks," *in 2018 7th International Symposium on Next Generation Electronics*, pp. 1-4, 2018.

[2] J. Lee and J. Nam, "Multi-level and multi-scale feature aggregation using pre-trained convolutional neural networks for music auto-tagging," *arXiv:1703.01793*, 2017

[3] N. Mauthes, "Vgm-rnn: Recurrent neural networks for video game music generation," *Master's thesis*, San Jose State Universit, 2018.

[4] A. Huang and R. Wu, "Deep learning for music," *arXiv:1606.04930,* 2016.

[5] R. Vohra, K. Goel, and J. K. Sahoo, "Modeling temporal dependencies in data using a dbn-lstm," *in 2015 IEEE International Conference on Data Science and Advanced Analytics,* pp. 1-4, 2015.

[6] Q. Lyu, Z. Wu, J. Zhu, and H. Meng, "Modeling high dimensional sequences with lstm-rtrbm: application to polyphonic music generation," *in Proceeding IJCAI'15 Proceedings of the 24th International Conference on Artificial Intelligence*, 2015.

[7] D. Spitael, "Style transfer of polyphonic music using sequence-to-sequence neural networks," *Master's thesis*, 2017.

[8] T. Hori, K. Nakamura, and S. Sagayama, "Music chord recognition from audio data using bidirectional encoder-decoder lstms," *in 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pp. 1312-1315, 2017.

[9] A. T. Cemgil, B. Kappen, and D. Barber, "A generative model for music transcription," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 14, no. 2, pp. 679-694, 2006.

[10] M. Kaliakatsos-Papakostas, M. G. Epitropakis, and M. Vrahatis, "Weighted markov chain model for musical composer identification," *European Conference on the Applications of Evolutionary Computation*, pp. 334-343, 2011.

[11] A. V. D. Merwe and W. Schulze, "Music generation with markov models," *IEEE MultiMedia*, vol. 18, no. 3, pp. 78- 85, 2011..

[12] M. Kaliakatsos-Papakostas and E. Cambouropoulos, "Probabilistic harmonization with fixed intermediate chord constraints," *in Joint 40th International Computer Music Conference and 11th Sound and Music Computing (SMC) Conference*, 2014.

[13] A. Konwer, A. K. Bhunia, A. Bhowmick, A. K. Bhunia, P. Banerjee, P. P. Roy, and U. Pal, "Staff line removal using generative adversarial networks," *arXiv:1801.07141,* 2018.

[14] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," *arXiv:1609.05473,* 2016.

[15] S. Lee, U. Hwang, S. Min, and S. Yoon, "A seqgan for polyphonic music generation," *arXiv:1710.11418,* 2017.

[16] H. Liu and Y. Yang, "Lead sheet generation and arrangement by conditional generative adversarial network," *arXiv:1807.11161*, 2018.

[17] R. Manzelli, V. Thakkar, A. Siahkamari, and B. Kulis, "Conditioning deep generative raw audio models for structured automatic music," *arXiv:1806.09905,* 2018.

[18] Y. Wang, Y.Huang, T. Lin, S. Su, and Y.Chen,"Modeling melodic feature dependency with modularized variational auto-encoder,"

arXiv1811.00162, 2018.

[19] Y. Hung, Y. Chen, and Y. Yang, "Learning disentangled representations for timber and pitch in music audio," *arXiv:1811.03271*, 2018.

[20] G. Brunner, Y. Wang, R. Wattenhofer, and S. Zhao, "Symbolic music genre transfer with cyclegan," *arXiv:1809.07575,* 2018.

[21] R. Lu, K. Wu, Z. Duan, and C. Zhang, "Deep ranking: Triplet matchnet for music metric learning," *in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 121-125, 2017.

[22] N. Kotecha, "Bach2bach: Generating music using A deep reinforcement learning approach," *arXiv:1812.01060*, 2018.

[23] S. Fukayama, K. Nakatsuma, S. Sako, T. Nishimoto, and S. Sagayama, "Automatic song composition from the lyrics exploiting prosody of japanese language," *in Proceedings of the 7th Sound and Music Computing Conference*, 2010.

[24] C. Moruzzi, "Creative ai: Music composition programs as an extension of the composer's mind," *Springer*, vol. 44, 2018.

[25] I. Hwang, H. Lee, and S. Choi, "Real-time dual-band haptic music player for mobile devices," *IEEE Transactions on Haptics,* vol. 6, no. 3, pp. 340-351, 2013.

[26] J. Wang, E. Chng, C. Xu, H. Lu, and Q. Tian, "Generation of personalized music sports video using multimodal cues," *IEEE Transactions on Multimedia,* vol. 9, no. 3, pp. 576-588, 2007.

[27] D. Coster and B. Mathieu, "Polyphonic music generation with style transitions using recurrent neural networks," *Master's thesis,* 2017.

[28] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language

understanding," *arXiv:1810.04805,* 2018.

[29] N. Kodali, J. D. Abernethy, J. Hays, and Z. Kira, "How to train your DRAGAN," *arXiv:1705.07215*, 2017.

[30] J. Guo, S. Lu, H. Cai, W. Zhang, Y. Yu, and J. Wang, "Long text generation via adversarial training with leaked information," *arXiv:1709.08624,* 2017.

[31] B. Xu *et al.*, Empirical Evaluation of Rectified Activations in Convolution Network, *arXiv:1505.00853v2,* 2015.

[32] A. Ng, Feature selection, L1 vs. L2 regularization, and rotational invariance, *ICML*, 2004.

[33] S. Loofa, and C. Szegedy, Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, *arXiv:1502.03167,* 2015.

[34] L. v. d. Maaten, Visualizing Data using t-SNE, *Journal of Machine Learning Research 9*, pp. 2579-2605, 2008.

[35] O. Vinyals *et al*,. Show and Tell: A Neural Image Caption Generator, *arXiv:1411.4555,* 2014.

[36] S.-H. Choi and Y. Kim, The Bi-activation function : an Enhanced Version of an Activation function in Convolutional Neural Network, *Published at KICS Winter Conferences,* 2020.

[37] S.-H. Choi, "Scale Calibration Cascade Smooth Loss of Generative Adversarial Networks with Online Continual Task Learning," *Summitted ECCV,* 2020.

[38] R. Aljundi, L. Caccia, E. Belilovsky, M. Caccia, M. Lin, L. Charlin, T., Tuytelaars, Online continual learning with maximally interfered retrieval, *arXiv:1908.04742,* 2019.

[39] D. Lopez-Paz, M. Ranzato, Gradient epsiodic memory for continual learning 711, *arXiv:1706.08840,* 2017.

[40] S.-H. Choi, "Nonlinear Exponential Regularization : An Improved

Version of Regularization for Deep Learning Model", *Published at KICS Winter Conferences*, 2020.

[41] X. Mao, *et al.*, Least Squares Generative Adversarial Networks, *arXiv:1611.04076,* 2016.

[42] J. Long, *et al,*. Fully Convolutional Networks for Semantic Segmentation, arXiv:1411.4038, 2014.

[43] O. Ronneberger *et al.,* "U-Net: Convolutional Networks for Biomedical Image Segmentation," *arXiv:15005.04597*, 2015.

[44] L.-C. Chen, *et al.,* Encoder-Decoder with Atrous Separable convolution for semantic image segmentation, *arXiv:1802.02611v3,* 2018.

[45] S.-H. Choi, "Novel Auxiliary Components to Help Optimize Deep Learning Model," *Submitted,* 2020

[46] J.-Y. Zhu, Z. Zhang, C. Zhang, J. Wu, A. Torralba, J. Tenenbaum, and W. Freeman, Visual object neworks: Image generation with disentangled 3d representation, *NIPS,* 2018.

[47] J. Johnson, A. Gupta, and L. Fei-Fei, Image generation from scene graphs, *arXiv:1804.01622*, 2018.

[48] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, and D. Huang, X.and Metaxas, Stackgan++: Realistic image synthesis with stacked generative adversarial networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, pp. 1947-1962, 2019.

[49] P. Probst, A.-L. Bouleseix, and B. Bischl, Tunability: Importance of hyperparameters of machine learning algorithms, *Journal of Machine Learning Research,* vol. 20, pp. 1-132, 2019.

[50] R. Aljundi, Continual learning in neural networks, *arXiv:1910.02718v2,* 2019.

[51] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, Unrolled generative adversarial networks, *arXiv:1611.02163*, 2016.

[52] M. Mirza and S. Osindero, Conditional generative adversarial nets, *arXiv:1411.1784*, 2014.

[53] A. Odena, C. Olah, and J. Shelns, Conditional image synthesis with auxiliary classifier gans, *arXiv:1610.095585*, 2016.

[54] A. Odena, Semi-supervised learning with generative adversarial networks, *arXiv:1606.01583*, 2016.

[55] T.-Y. Lin *et al*,. Focal loss for Dense object detection, *arXiv:1708.02002,* 2017.

[56] S.-H. Choi and J. Choi, Ensemble Normalization for Stable Training, *Published at KICS Conferences,* 2020.

[57] S.-H. Choi and S. H. Jung, Similarity Analysis of Actual Fake Fingerprints and Generated Fake Fingerprints by DCGAN, *Published at IJFIS,* vol. 19, no. 1, pp. 40-47, 2020.

[58] S.-H. Choi and S. H. Jung, Performance improvement of fake discrimination using time information in CNN-based signature recognition, *Journal of Digital Contents Society*, vol. 19, no. 1, pp. 205-212, 2018.

[59] Dhriti and M. Kaur, K-nearest neighbor classification approach for face and fingerprint at feature level fusion, *International Journal of Computer Applications*, vol. 60, no. 14, pp. 13-17, 2012.

[60] E. Marasco, P. Wild, and B. Cukic, Robust and interoperable fingerprint spoof discrimination via convolutional neural networks, *Proceedings of IEEE Symposium on Technologies for Homeland Security, Waltham*, pp. 1-6. 2016.

[61] S. H. Choi and S. H. Jung, Analysis of the effect of space-time information on CNN-based fake fingerprint discrimination, *Proceedings of 2017 the Korea Software Congress,* 2017, pp. 1968-1970.

[62] E. Park, W. Kim, Q. Li, and H. Kim, Fingerprint live-ness

detection using patch-based convolutional neural networks, *Journal of the Korea Institute of Information Security and Cryptology*, vol. 27, no. 1, pp. 39-47, 2017.

[63] E. Park, W. Kim, Q. Li, H. Kim, and J. Kim, Fingerprint liveness detection using CNN features of random sample patches, *Proceedings of 2016 International Conference of the Biometrics Special Interest Group, Darmstadt, Germany,* 2016, pp. 1-4. 2016.

[64] S.-H.Choi and S. H. Jung, Generation for fake fingerprint data through GAN for detecting fake fingerprint by deep learning methods, *Proceedings of 2017 the Institute of Electronics and Information Engineers (IEIE) Conference*, pp. 965-966, 2017.

[65] A. Radford, L. Metz, and S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial network, *arXiv:1511.06434,* 2016.

[66] L. Sixt, B. Wild, and T. Landgraf, RenderGAN: generating realistic labeled data, *arXiv:1611.01331,* 2017

[67] K. Regmi and A. Borji, Cross-view image synthesis using conditional GANs, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 3501-351, 2018.

[68] G. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. Dai, M. Hoffman, M. Dinculescu, and D. Eck, Music transformer: Generating music with long-term structure, *arXiv:1809.04281v3,* 2018.

[69] M. Schuster and K. K. Paliwal, Bi-directional recurrent neural networks, *IEEE Transactions on Singal Processing,* vol. 45, no. 11, pp. 2673-2681, 1997.

[70] I. Sutskever, O. Vinyals, and Q. Le, Sequence to sequence learning with neural networks, *NIPS,* 2014.

[71] L. Yu, W. Zhang, J. Wang, and Y. Yu, Seqgan: Sequence generative adversarial nets with policy gradient, *arXiv:1609.05473,* 2016.

[72] Q. Lyu, Z. Wu, J. Zhu, and H. Meng, Modeling high dimensional sequences with lstm-rtrbm: application to polyphonic music generation, *proceeding IJCAI'15 Proceedings of the 24th International Conference on Artificial Intelligence*, 2015.

[73] S.-H. Choi, Multi Way Decoder Scheme with Error Reduction Embedding on one-hot bi-directional Seq2Seq with adaptive regularization for Music Composition, *Submitted,* 2020

[74] S.-H. Choi, Study on the Importance of Adaptive Seed Value Exploration, *Published at KICS Conferences*, 2020.

[75] F. Dietrich, Track Seed Classification with Deep Neural Networks, *arXiv:1910.06779v1*, 2019.

[76] J. Bird, A. Ekart, and D. Faria, On the effects of pseudorandom and quantum-random number generators in soft computing, *Soft Computing*, 2019.

[77] P. Madhyastha and R. Jain, On Model Stability as a Function of Random Seed, *arXiv:1909.10447v1*, 2019.

[78] F. Fan and G. Wang, Learning from Pseudo-Randomness with an Artificial Neural Network-Does God Play Pseudo-Dice?, *arXiv:1801.01117,* 2018.

[79] C. Colas, O. Sigaud, and P.-Y. Oudeyer, How Many Random Seeds? Statistical Power Analysis in Deep Reinforcement Learning Experiments, *arXIv:1806.08295v2*, 2018.

[80] C. Zoufal A. Lucchi, and S. Woerner, Quantum Generative Adversarial Networks for learning and loading random distributions, *Quantum Information*, 2019.

[81] G. Song, H. Myeong, K. M. Lee, SeedNet : Automatic Seed Generation with Deep Reinforcement Learning for Robust

Interactive Segmentation, *CVPR*, 2018.

[82] S.-H. Choi, S. Joo, S. H. Jung, Component based comparative analysis of each module in image captioning, *Submitted*, 2020

[83] O. Vinyals, A. Toshev, S. Bengio and D. Erhan, Show and Tell: A Neural Image Caption Generator, *CVPR*, 2015.

[84] J. Yu, J. Li, Z. Yu, and Q. Huang, Multimodal Transfer with Multi-View Visual Representation for Image Captioning, *Journal of LATEX Class Files,* vol. 15, no. 8, 2015.

[85] H. Yu, S. Cheng, B. Ni, M. Wang, J. Zhang and X. Yang, Fine-grained Video Captioning for Sports Narrative, *CVPR*, 2018.

[86] L. Zhou, Y. Kalantidls, X. Chen, J. Coarso and M. Rohrbach, Grouned Video Description, *CVPR*, 2019.

[87] J. Li, Y. Wong, Q. Zhano and M. Kankanhalli, Video Stroytelling, *arXiv:/1807.09418v2*, 2019.

[88] K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, *arXiv:/1409.1556v6*, 2015.

[89] K. He, X. Zhang, S. Ren and J. Sun, Deep Residual Learning for Image Recognition, *arXiv:/1512.03385v1*, 2015.

[90] H. Sak, Andrew and F. Beaufays, Long Short-Term Memory Recurrent Neural Networks Architectures for Large Scale Acoustic Modeling, *Interspeech*, 2014.

[91] M. Schuster and K. Paliwal, Bidirectional recurrent neural networks, *IEEE Transactions on Signal Processing*, no. 45, vol. 11 pp. 2673-2681, 1997.

[92] S. Hochreiter, and J. Schmidhuber, Long short-term memory, *Neural computation*, no. 9, vol. 8, pp. 1735-1780, 1997.

[93] J. Chung, C. Culcehre, K. Cho, and Y. Bengio, Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, *arXiv:/1412.3555*, 2014.

[94] T. Kim, M.-O. Heo, S. Son, K.-W. Park and B.-T. Zhang, GLAC Net: GLocal Attention Cascading Networks for Multi-image Cued Story Generation, *arXiv:/1805.10973v3*, 2019.

[95] S. Wiseman and A. Rush, Sequence-to-Sequence Learning as Beam Search Optimization, *arXiv:/1606.02960v2*, 2016.

[96] S. Yan, Y. Xie, F. Wu and J. Smith, Image Captioning via a Hierarchical Attention Mechanism and Policy Gradient Optimization, *arXiv:/1811.05253v2*, 2019.

[97] J. Pennington, R. Socher and C. Manning, Glove: Global vectors for word representation, *EMNLP*, 2014.

[98] Y. LeCun, L. Bottou, G. Orr and K. Muller, Efficient Backprop, Neural Networks: *Tricks of the trade*. Springer, 1998b.

[99] S.-H. Choi, Y. Oh, and D. Koo, Visualization Techniques for Outlier Data, *2020 International Conference on Artificial Intelligence in Information and Communication* (ICAIIC), pp. 346-351, 2020.

[100] L. Mclnnes, J. Healy, and J. Melville, UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, *arXiv:1802.03426*, 2018.

[101] [Available at] https://en.wikipedia.org/wiki/Local_binary_patterns

[102] N. Chawla, K. Bowyer, L. Hall, W. Philip Kegelmeyer, SMOTE: Sy nthetic Minority Over- sampling Technique, *Journal of Artificial Intelligence Research*, no. 16, pp. 321-357, 2002.

[103] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, On Convergence and Stabiility of GANs, *arXiv:1705.07215*, 2017.

[104] J. Zhao, M. Mathieu, and Y. LeCun, Energy- based Generative Adversarial Network, *arXiv:1609.03126*, 2016.

[105] [Available at] https://en.wikipedia.org/wiki/Uniform_distribution_(co ntinuous)

[106] [Availableat] https://en.wikipedia.org/wiki/Circular_convolution

[107] S.-H. Choi and S. H. Jung, Stable Acquisition of Fine-Grained Segments Using Batch Normalization and Focal Loss with L1 Regularization in U-Net Structure, *Published at IJFIS*, vol. 20, no. 1, pp. 59-68, 2020.

[108] L. Huang, J. Peng, R. Zhang, G. Li, and L. Lin, "Learning deep representations for semantic image parsing: a comprehensive overview," *Frontiers of Computer Science*, vol. 12, no. 5, pp. 840-857, 2018.

[109] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," 2015.

[110] O. Oktay *et al.*, "Attention U-net: learning where to look for the pancreas," *arXiv:1805.03999,* 2018,

[111] X. Liang, S. Liu, X. Shen, J. Yang, L. Liu, J. Dong, L. Lin, and S. Yan, "Deep human parsing with active template regression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 12, pp. 2402-2414, 2015.

Appendices

Part I. Supplementary Materials of Scale Calibration Cascade Smooth Loss of Generative Adversarial Networks with Online Continual Task Learning

Visualization of the proposed method (Section 1)
Analysis result using the flow chart of the proposed method (Section 2)
Visualization of the proposed and existing method (Section 3)
Experimental setting about proposal method (Section 4)
Proposal analysis after applying weight decay to optimization method (Section 5)
Single loss analysis (Section 6)
Z latent size analysis about correction loss (Section 7)
Optimization analysis about correction loss (Section 8)
Optimization analysis of our cascade component (Section 9)
Optimization analysis about our scale calibration component (Section 10)
Analysis of single on bias component (Section 11)
Analysis experiment initial distribution about our proposal (Section 12)
Stable acquisition information analysis at smooth correction (Section 13)
Stable acquisition information analysis using parameter analysis at smooth correction (Section 14)
Problem analysis at smooth correction (Section 15)
Parameter analysis using L1 and L2 regularization at smooth correction (Section 16)
Regularization characteristic analysis using L1 and L2 regularization at smooth correction (Section 17)
Nonlinear regularization characteristic analysis using parameter changing at smooth correction (Section 18)
Analysis of qualitative proposal method (Section 19)

1) Visualization of the proposed method

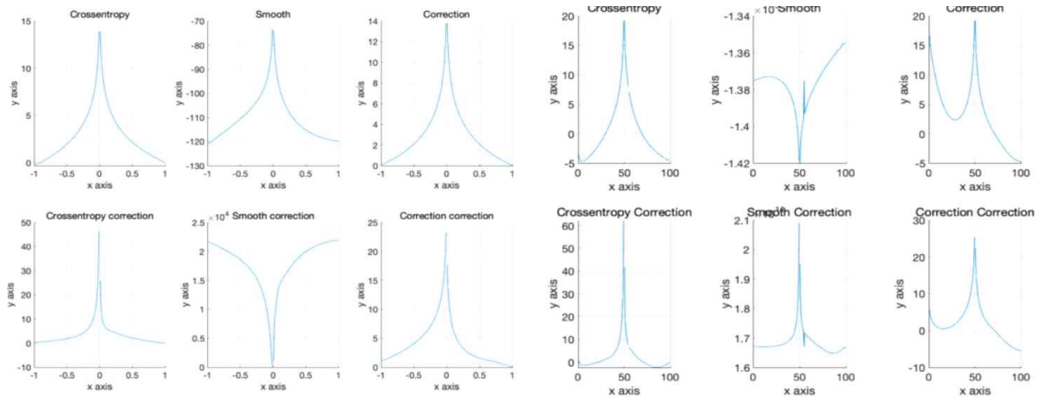


Figure 60 Visualization about experiment loss

The visualization result of the proposed loss shown in Figure 60. Figure 60(a) is the result of using the same value. Figure 60(i) is the result of using the x and y axes. Figure 60(ii) is the result of using the x, y, and z axes. To analyze the influence of the formula of the proposed method, we tried to explain the loss through a brief model with a distribution from −1 to 1. Also, we analyze the results of the analysis using the same size value and the equal size value. In Figure 60, the proposed loss is the Smooth calibration loss, which has the following effects when learning values with unequal size. It is composed of thicker both and values than the boundary due to the other loss, and a smaller value representing the boundary value.

2) Analysis results using a flow chart of the proposed method

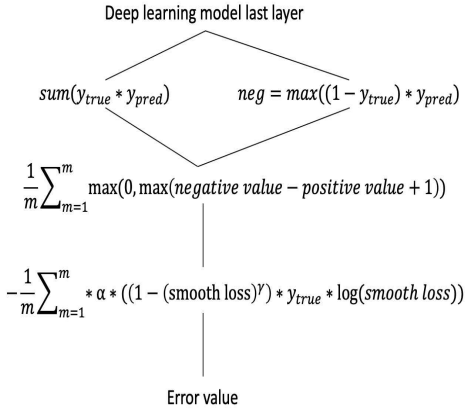


Figure 61 Flow chart about our proposal loss

Figure 61 is the method proposed. The proposed method is first divided into positive and negative to reflect two pieces of information. The second step is to smoothly reflect negative and positive information simultaneously. And thirdly, apply the cascade component and scale calibration component. Thirdly, the error value is output after applying.

3) Experimental system setting about proposal method

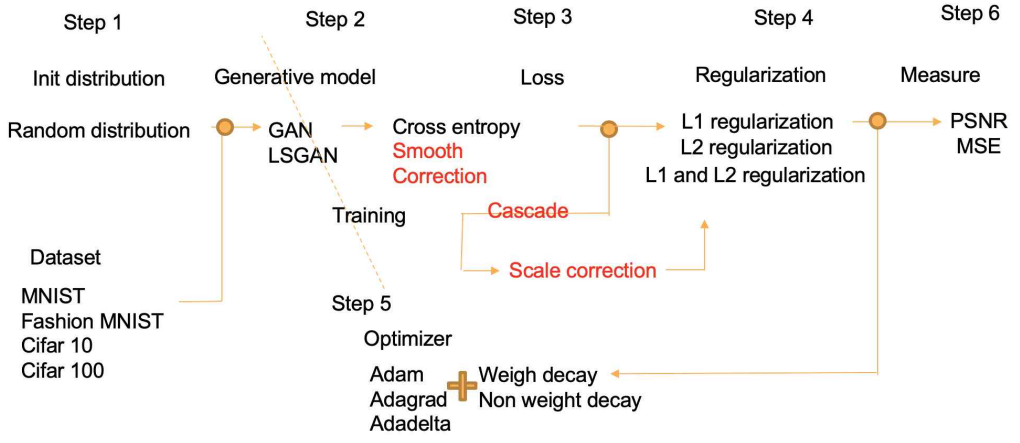


Figure 62 Visualization of experiment system configure.

Figure 62 is a visualization of the experiment system configures. The red letter in Figure 62 is the contribution. And the orange circle means the intersection point. And the orange plus sign implies a combination of three optimizer learning and Weight decay. The Z latent space is the initial distribution of the generation model in Step 1. Step 2 is composed of the generation model used in the experiment model. Step 3 describes the new loss of the cascade scale calibration. Step 4 and 5 are three optimizer learning and regularization for verification of proposal losses – finally, Step 6 measure by the performance of image generation.

4) Proposal analysis after applying weight decay to the optimization method

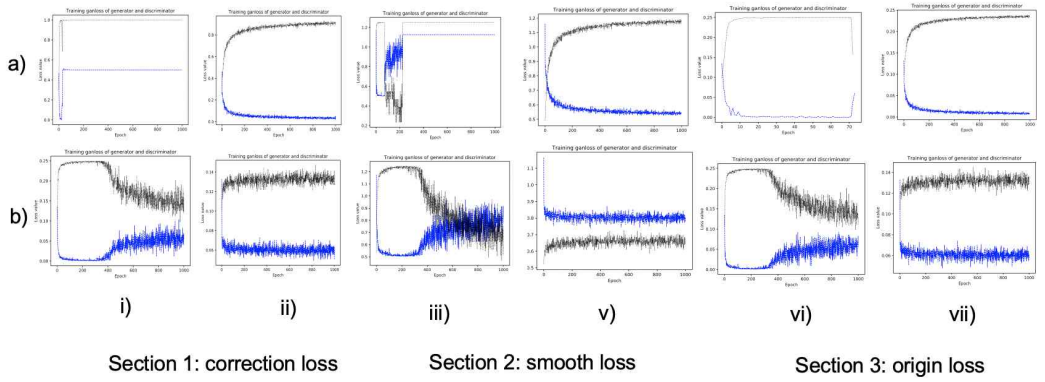


Figure 63 Compare of loss on weight decay in optimization process in Valina-GAN using MNIST dataset on Learning rate 0.0007, 16 batch size, L1 0.25, L2 0.0, and 500 Z latent space size, a) Adam b) Adagrad i) Nondecay ii) Weight decay

First, we tried to confirm the effect of Weight decay. And secondly, we verify to check the impact on Batch size. Finally, we analyze the influence of the Z latent space size. In Figure 63, We evaluated the performance evaluation and loss results with and without Weight decay through optimization methods Adagrad Adam, Adam delta, and Adam. As shown in Figure 63.ii, if the Weight decay is included, it can be seen that the interval between the generator loss and the discriminator loss widens as the epoch repeats. This shows that the Weight decay is learned by learning the continuous task information in the correct convergence direction while Learning with the interval between generator loss and discriminator loss. However, when the Weight decay is performed, the Learning is changed from the unstable state to the stable state. However, the result of measuring the actual generation performance is as follows.

Table 32 shows the top 1 and top 5 average PSNR when Weight decay is not applied. Table 32 shows the top 1 and top 5 average generation image performance when Weight decay is applied. We show that the performance of image generation is improved when Weight decay is not applied by when the performance comparison is performed with or without weight decay. This shows that Learning is performed stably when the Weight decay is applied. However, when the actual performance improvement is discriminated, it is seen that the performance image generation of improvement is reduced when the

generated image is discriminated more accurately by Learning stably. The smooth loss and the focal loss tend to be better than the existing loss when the measured performance is measured by the single loss state.

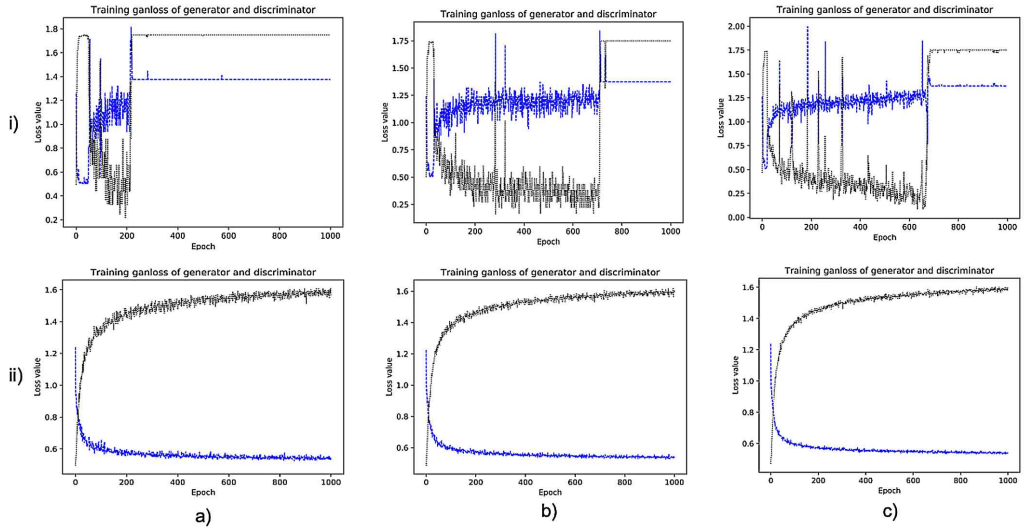


Figure 64 Compare of Batch size on smooth loss in optimization process using MNIST dataset on Learning rate 0.0007, L1 0.0, L2 0.75, and 1000 Z latent space size in Adam decay. a) 16 batch size, b) 32 batch size, c) 64 batch size, i) Non Weight decay, ii) Weight decay

The influence of Batch size on the single smooth loss was analyzed using Weight decay. As shown in Figure 64(a), we can see that the variation of the loss reduces by applying the weight decay, and the discriminator loss and the generator loss maximize.

Table 32 Performance evaluation of Z latent space size with L1 0.0, L2 0.0, Adam optimizer, no weight decay using GAN

	Accuracy methods data type	Fashion MNIST /MNIST				
	Origin loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size

100 Z latent size	Top 1 score PSNR	49.168/47.255	48.521 / 47.144	48.287/47.060	48.122 / 46.997	47.966 / 46.947
	Top 5 average PSNR	47.213/46.370	47.520/46.599	47.697/46.774	47.641/46.608	47.666/46.818
	Focal loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.168/47.255	48.521/47.144	48.287/47.060	48.122/46.997	47.966/46.947
	Top 5 average PSNR	47.213/46.370	47.520/46.599	47.697/46.774	47.641/46.808	47.666/46.818
	Smooth loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.245/29.516	48.842/50.370	48.386/50.921	48.346/49.547	47.753/49.864
	Top 5 average PSNR	47.185/46.354	47.508/46.591	47.678/46.742	47.619/46.789	47.648/46.783
500 Z latent size	Origin loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	48.888/47.230	48.768/47.147	48.286/47.062	48.066/46.987	47.967/46.941
	Top 5 average PSNR	46.918/46.193	47.570/46.631	47.625/46.738	47.689/46.828	47.663/46.828
	Focal loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	48.888/47.230	48.768/47.147	48.286/47.062	48.066/46.987	47.967/46.941

	Top 5 average PSNR	49.918/ 46.193	47.570/ 46.631	47.625/ 46.738	47.689/ 46.828	47.663/ 46.828
	Smooth loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.365/ 48.597	49.092/ 51.323	48.510/ 50.186	47.681/ 46.825	47.827/ 47.823
	Top 5 average PSNR	46.912/ 16.190	47.566/ 46.623	47.618/ 46.719	47.681/ 46.825	47.656/ 46.806
1000 Z latent size	Origin loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	48.888/ 47.230	48.495/ 47.108	48.171/ 47.055	48.066/ 47.023	47.968/ 46.944
	Top 5 average PSNR	46.918/ 46.913	47.582/ 46.664	47.539/ 46.733	47.724/ 46.832	47.672/ 46.823
	Focal loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	48.985/ 47.318	48.495/ 47.108	48.171/ 47.055	48.066/ 47.023	47.968/ 46.944
	Top 5 average PSNR	47.090/ 46.285	47.582/ 46.664	47.539/ 46.733	47.724/ 26.832	47.672/ 46.823
	Smooth loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.344/ 50.863	48.678/ 49.819	48.571/ 51.271	48.064/ 50.411	47.953/ 49.551
	Top 5 average PSNR	47.076/ 46.261	47.577/ 46.663	47.525/ 46.724	47.719/ 46.828	47.622/ 46.810

Table 33 Performance evaluation of Z latent size with L1 0.0, L2 0.0 regularization, Adam optimizer, weight decay using GAN

	Accuracy methods data type	Fashion MNIST /MNIST				
	Origin loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
100 Z latent size	Top 1 score PSNR	49.168/ 47.255	48.521/ 46.599	48.287/ 47.060	48.131/ 46.997	48.122/ 46.947
	Top 5 average PSNR	48.744/ 46.370	48.409/ 46.599	48.178/ 46.774	48.075/ 46.808	48.064/ 46.818
	Focal loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.168/ 47.255	48.521/ 46.599	48.287/ 47.060	48.122/ 46.997	47.966/ 46.947
	Top 5 average PSNR	48.775/ 46.370	48.409/ 46.599	48.178/ 46.774	48.064/ 46.808	47.950/ 46.818
	Smooth loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.162/ 47.286	48.522/ 46599	48.287/ 47.094	48.131/ 47010	47.974./ 46..984
	Top 5 average PSNR	48.782/ 46.370	48.415/ 47.177	48.194/ 46.772	48.075/ 46.808	47.952/ 46.817
500 Z latent size	Origin loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.168/ 47.230	48.521/ 47.147	48.287/ 47.062	48.131/ 46.987	48.122/ 46.941

	Top 5 average PSNR	48.744/46.193	47.201/46.631	48.178/46.738	48.075/46.828	48.064/46.828
	Focal loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.168/47.230	48.521/47.147	48.287/47.062	48.122/46.987	47.956/46.941
	Top 5 average PSNR	48.775/46.193	48.409/46.631	48.178/46.737	48.064/46.828	47.950/46.828
	Smooth loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.162/47.252	48.522/47.177	48.287/47.078	48.131/47.024	47.974/46.986
	Top 5 average PSNR	48.782/46.193	48.415/46.631	48.194/46.739	48.075/46.828	47.952/46.828
1000 Z latent size	Origin loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.168/47.138	48.521/47.108	48.287/47.055	48.131/47.023	48.122/46.944
	Top 5 average PSNR	48.744/46.285	47.201/46.664	48.178/46.733	48.075/46.832	48.054/46.823
	Focal loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.168/47.318	48.521/47.108	48.287/47.055	48.122/47.023	47.966/46.944
	Top 5 average PSNR	48.775/46.285	48.409/46.664	48.178/46.733	48.064/46.832	47.950/46.823

	Smooth loss	4 batch size	8 batch size	16 batch size	32 batch size	64 batch size
	Top 1 score PSNR	49.162/ 47.320	48.522/ 47.132	48.287/ 47.078	48.131/ 47.052	47.974/ 46.969
	Top 5 average PSNR	48.782/ 46.283	48.415/ 46.665	48.194/ 46.731	48.075/ 46.832	47.952/ 46.822

In Figure 64, We evaluated the performance evaluation and loss results with and without Weight decay through optimization methods Adagrad Adam, Adam delta, and Adam. The result of measuring the actual generation performance is as follows. Table 32 shows the top 1 and top 5 average PSNR when Weight decay is not applied. Table 33 shows the top 1 and high five average generation image performance when Weight decay used. We show that the performance of image generation is improved when Weight decay is not used by when the performance comparison performed with or without Weight decay that indicates that learning is performed stably when the Weight decay is applied. However, when the actual performance improvement discriminated against, it is seen that the performance image generation of development reduces when the generated image is discriminated more accurately by Learning stably. The smooth loss and the focal loss tend to be better than the existing loss when the single loss state measures the measured performance.

5) Single loss analysis

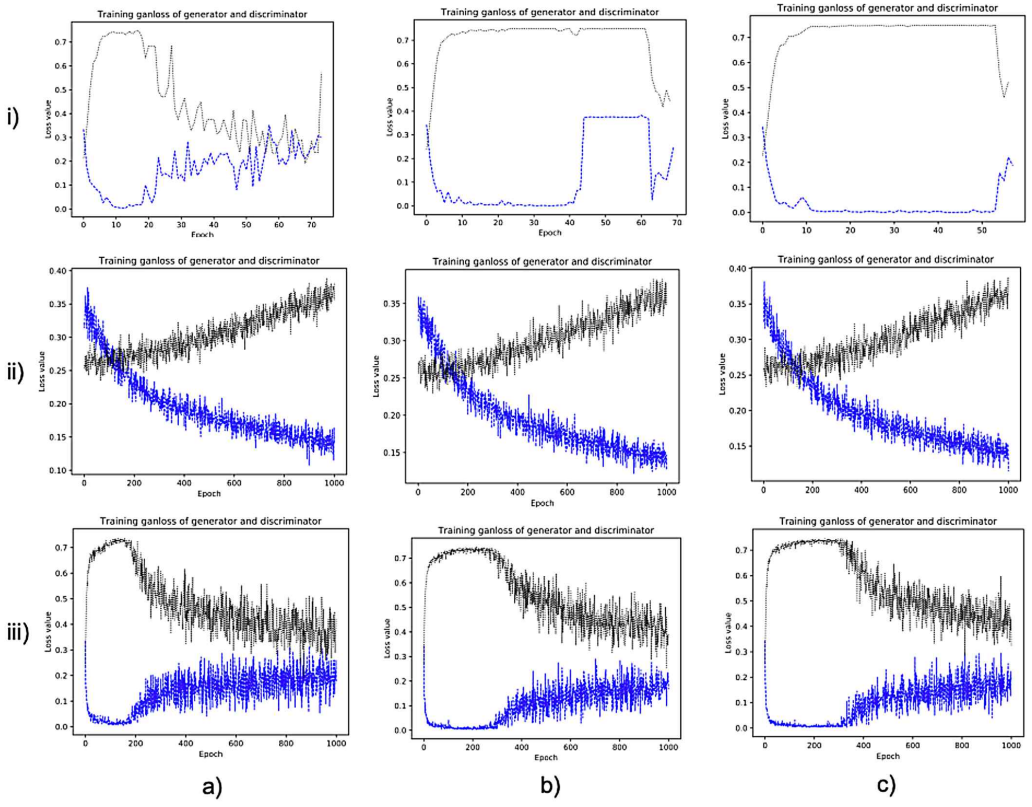


Figure 65 Comparison of 4 batch size, L1 0.0, L2 0.5, 100 Z Latent size, and in Valina-GAN model using Fashion-MNIST dataset. i) Adadelta, ii) Adagrad, iii) Adam, a) Origin loss, b) Smooth loss, c) Correction loss, i) Adadelta, ii) Adagrad, iii) Adam

If there is no Weight decay, we can see that the Batch size converges later depending on the larger. In Figure 65, Adam optimizer is well used and well used, but we analyzed the influence of three optimization methods to verify the proposed method. In Figure 65(i), the result of the adaDelta shows that the generator loss learns the original method and focal loss with similar loss values. In the case of the smooth loss, it can seem that the loss value is somewhat higher than the two methods in obtaining the continuous task information stably.

6) Z latent size analysis about correction loss

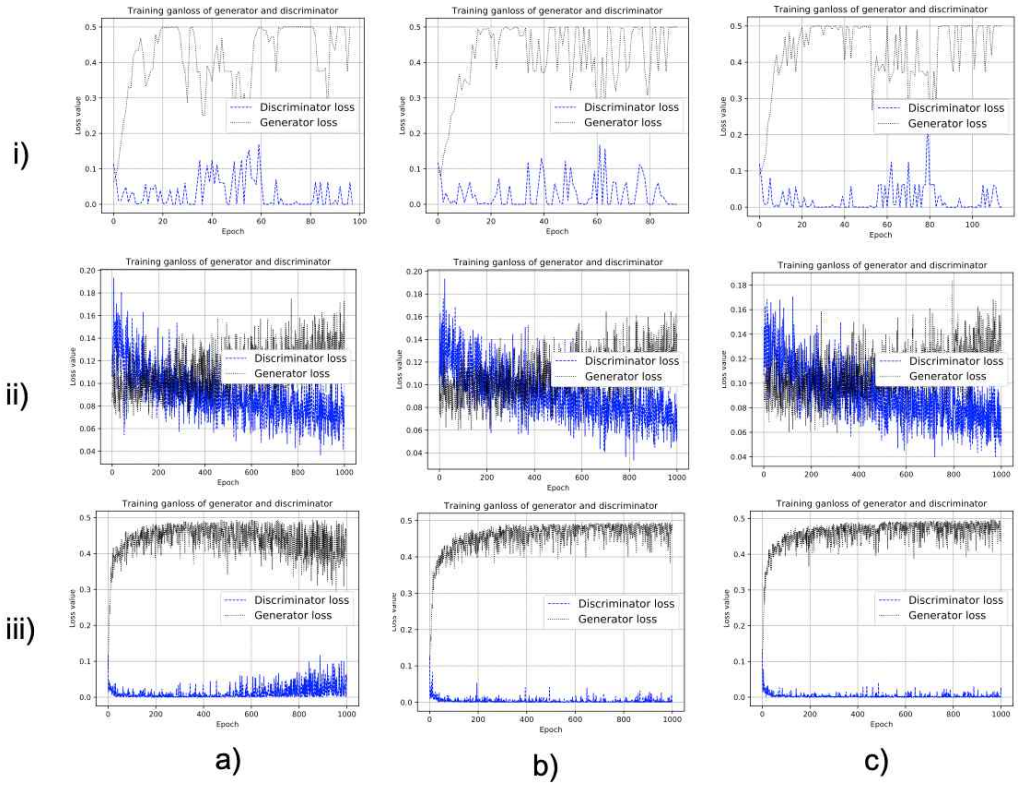


Figure 66 Z latent space size on correction loss by 16 batch size, L1 0, L2 0.5, in Valina-GAN using MNIST dataset. a) 100 Z latent space size , b) 500 Z latent space size, c) 1000 Z latent space size, i) Adam , ii) Adadelta iii) Adagrad

Smooth and focal loss analyzed according to three optimization methods and potential variable spaces that is to confirm the influence of generation performance according to the impact of the Z latent space size according to the input on the generation model in Figure 65, . As the latent variable space increases, the convergence phenomenon appears later when the influence of the Z latent space influenced by the size of the potential variable area.

7) Optimization analysis about our cascade component

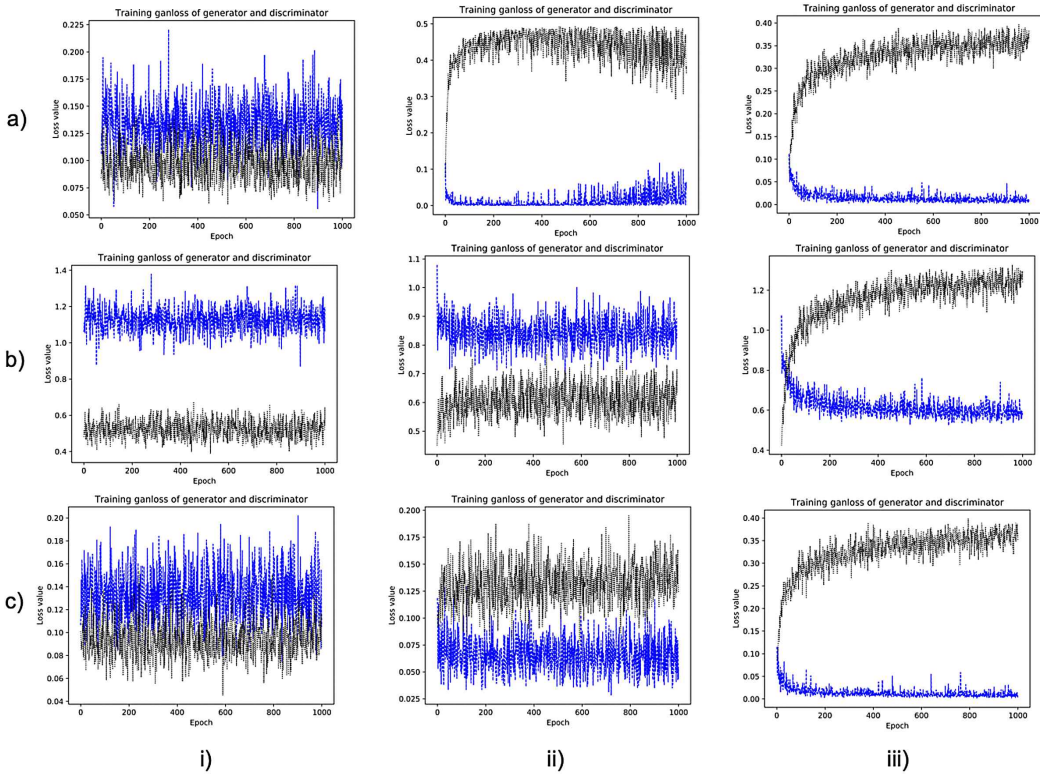


Figure 67 Optimization methods of Proposal Cascade Loss that composed of 4 batch size, L1 0.0, L2 0.5, 100 Z latent space size, and in Valina-GAN model using grayscale Fashion-MNIST dataset. i) Adadelta, ii) Adagrad, iii) Adam, a) Origin loss, b) Smooth loss, c) Correction loss, i) Adadelta, ii) Adagrad, iii) Adam

The proposed method in Figure 67 shows the best results when using Adam Optimizer as a result of experimenting with three optimization methods. In particular, the proposed method shows that the initial loss is better than the existing loss when the correction loss is corrected. In the case of correction loss and origin loss, a sharp convergence phenomenon is observed, which confirms that the slope from the correction and origin loss converges in the correct direction and is quickly learned. In the smooth loss, the derivative with the slope has a smooth slope, so it gradually converges.

8) Optimization analysis about our scale calibration component

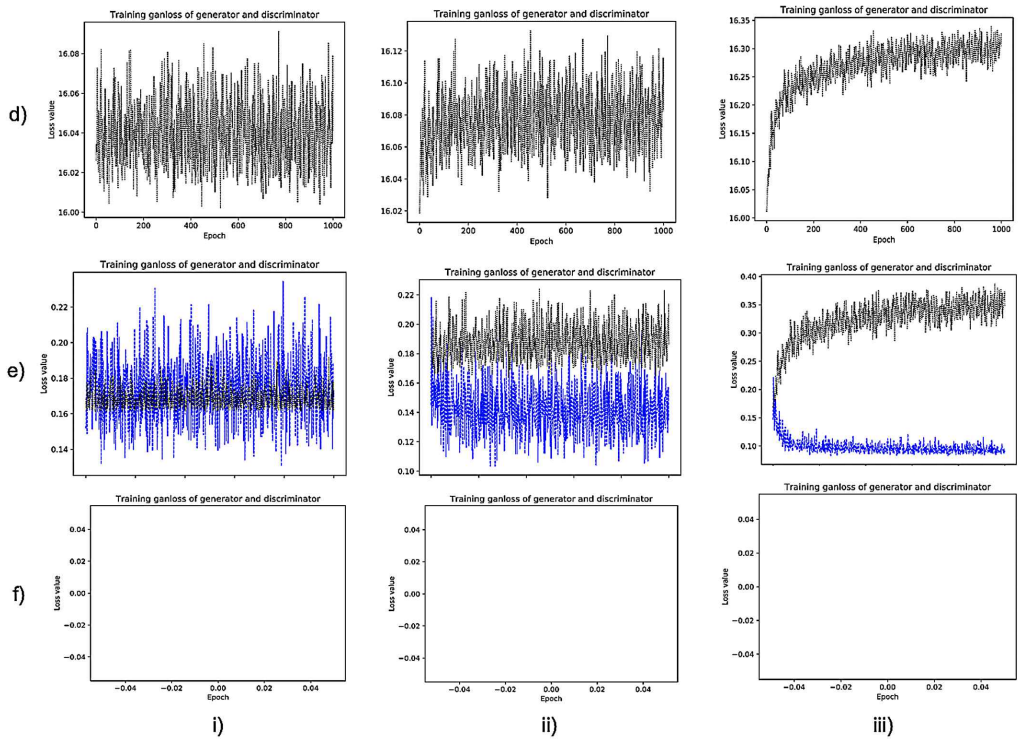


Figure 68 Compare of optimization methods on verified of Proposal Cascade Loss that composed of 4 batch size, L1 0.0, L2 0.5, Z latent space size 100, and in Valina-GAN model using grayscale Fashion-MNIST dataset. i) Adadelta, ii) Adagrad, iii) Adam, d) Origin correction loss, e) Smooth correction loss, f) Correction correction loss, i) Adadelta, ii) Adagrad, iii) Adam

The proposed method in Figure 68 is the result of an experiment comparing the optimization when applying scale calibration. If the scale calibration is correctly calibrated, learn to maximize the loss distance between the generator and the discriminator. However, if it is incorrectly corrected, one piece of information may collapse, and only one piece of information may appear. In the case of Correction Correction, it is a case of correcting the model again. At this time, the strong correction is entered rather than shows a phenomenon that can not learn.

9) Analysis of single on bias component

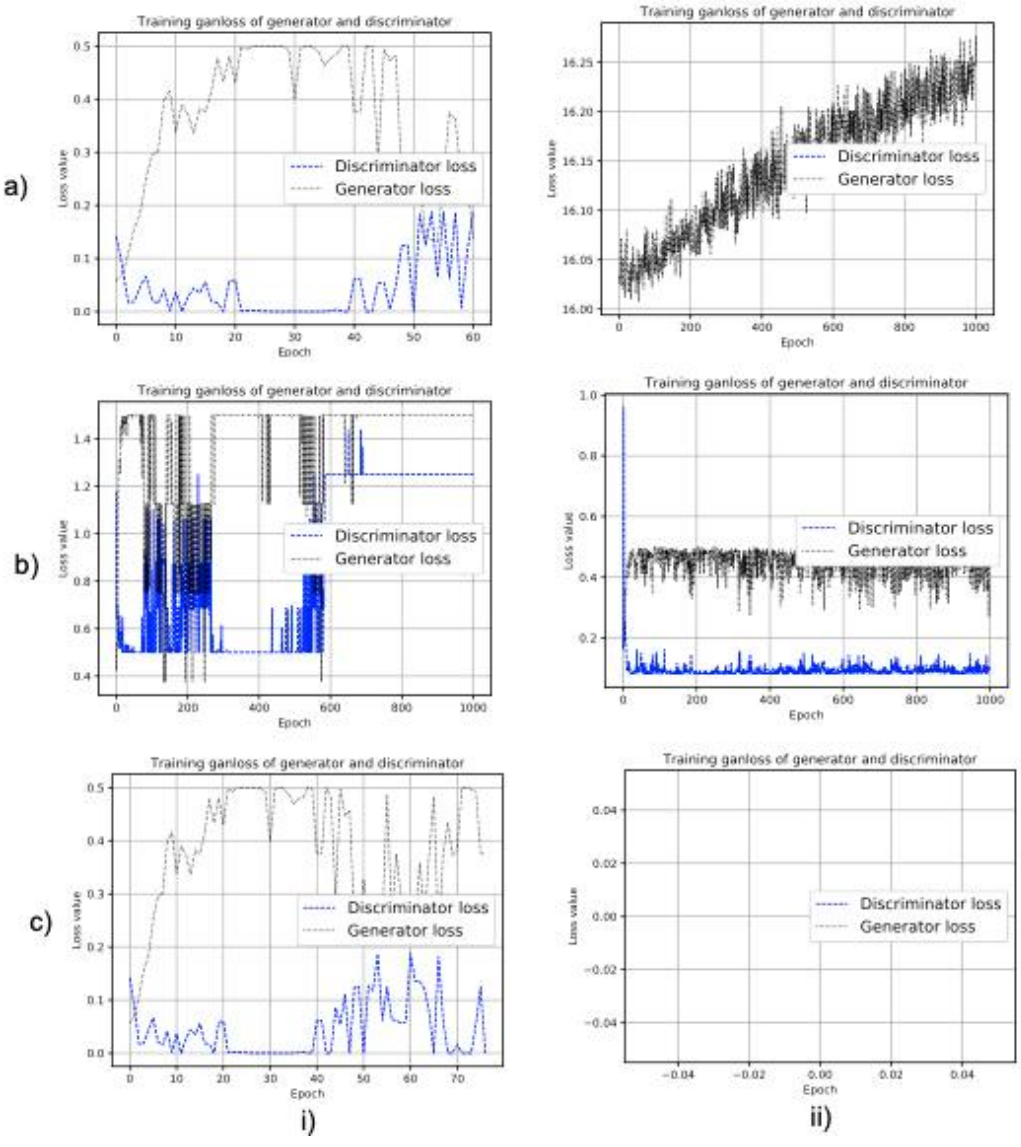


Figure 69 Compare of detail description by model loss on Scale correction term a) Origin loss, b) Smooth loss, c) Correction loss, i) Noncorrection, ii) Correction.

To verify the proposed Cascade method in Figure 69, we experimented with a single loss and Cascade method when learning with the Fashion MNIST dataset. The Cascade method proposed is in the case of a single loss of Figure 69(a) to Figure 69(c) Figure 69(d) to Figure 69(f) is the result of applying the Cascade method proposed. Figure 69(d) and Figure 69(f) are the results of applying the Scale calibration method,

Figure 69(f) is a strong influence of Scale calibration, and showed only one learning or no learning at the time of confrontational learning. However, the results of Figure 69(e) show that it is possible to improve the existing learning by calibration through Scale calibration. As shown in Figure 69(iii), Scale calibration shows better learning results.

10) Analysis experiment initial distribution about our proposal

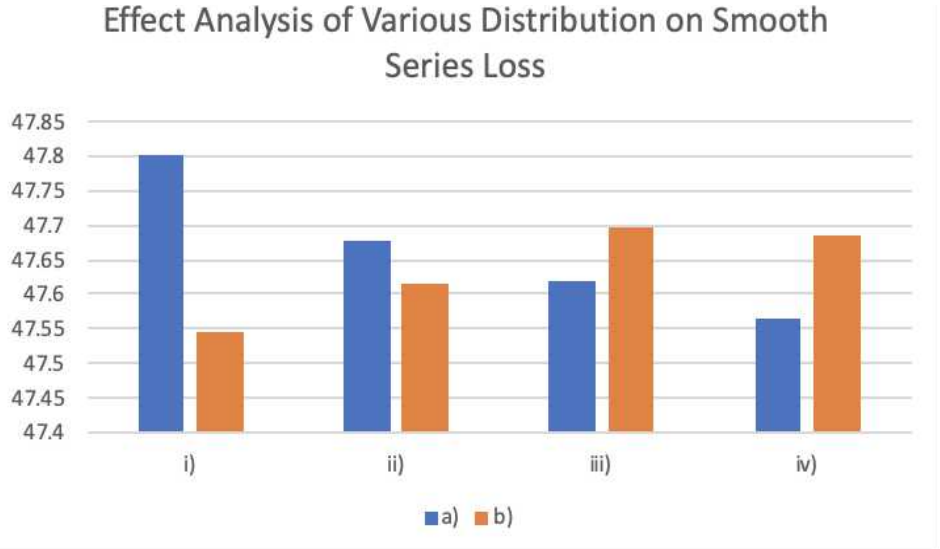


Figure 70 Performance of dependency analysis at color image generation measure PSNR of various distribution on 8 batch size. a) Smooth method b) Smooth correction, i) Random distribution, ii) Laplace distribution, iii) Logistic distribution iv) Gumbel distribution

Figure 70 has been carried out through various distributions for the two methods of the Smooth method and the smooth correction method proposed. The experimental results show that the average PSNR decreases with the Random distribution and Laplace distribution for the Smooth Correction method proposed. If the Random distribution constructs the Random distribution, the continuous task information will remain for a long time.

11) Stable acquisition information analysis at smooth correction

Figure 71 Influence analysis based on various batch sizes according to initial distribution for the acquisition of continuous task information reflection by smooth correction, Learning rate 0.0007, 4 batch size, L1 0.0, L2 0.0, 100 Z latent space size, Adagrad optimizer, LSGAN, and MNIST dataset. a) The normal distribution, b) Laplace distribution, c) logistic distribution, d) Gumbel distribution, i) 4 batch size, ii) 16 batch size, iii) 32 batch size

Figure 71 is a Scale correction method based on various Batch sizes according to initial distribution. In the case of Batch sizes 4 and 16, the training process Gumbel distribution shows that the average PSNR performance is more stable than the other distributions. 32 batch size and Batch size 64 indicate that Logistic distribution reflects information reliably. When the Batch size is small, and the vibration of the model is large in the parameter space, we show that the proposed Gumbel distribution is efficient that shows that the width of Gumbel distribution and the values of the tail of both stably reflect the continuous task information with thicker than the other distributions. However, when the Batch size is larger than a certain level, the influence of many changes in the parameter space of the model shows that the width of the distribution and the sharpness of the tail is more sharpened to reflect the continuous task information stably that shows that it is necessary to reflect more clear data from the data as the number of data increases. We show a concrete result of applying Gumbel distribution to the initial distribution to stably obtain the continuous task information when the parameter space of the model fluctuates much. Generating an image

through stably reflecting the continues task information is strongly influenced by the initial distribution of the latent space. As a result, it confirms that continuous task information could be obtained stably according to the initial distribution. Therefore, the analysis is carried out through various distributions to further analyze the proposed distribution as the initial distribution. To confirm of the Z latent space is robust to the generation performance through the generalized distribution and the production through the analytical distribution, the initial distribution is a) Random distribution, b) Gumbel distribution, c) Laplace distribution d) Logistic distribution.

12) Stable acquisition information analysis using parameter analysis at smooth correction

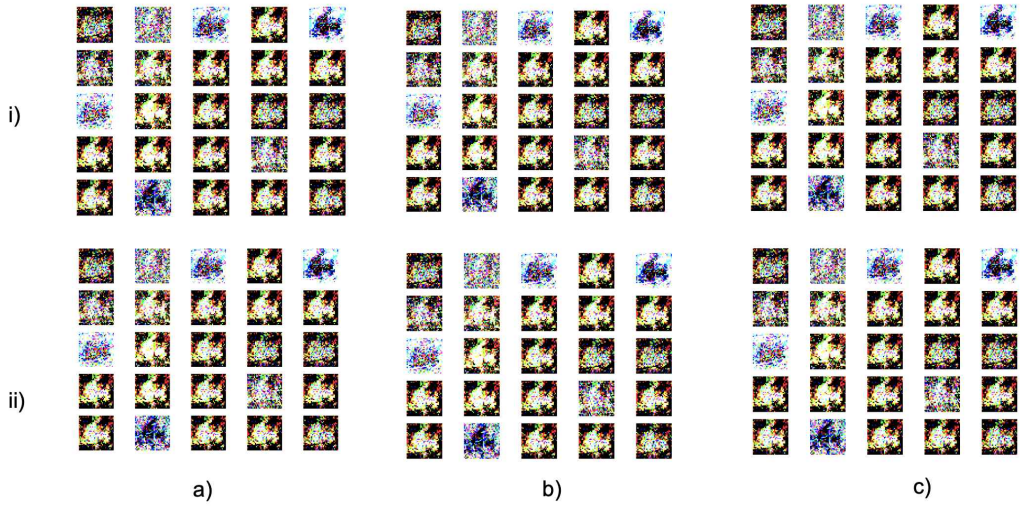


Figure 72 Compare of the effect of test smooth calibration method on alpha and gamma coefficients at Learning rate 0.0007, Batch size 4, L1 0.25, L2 0.25, Z latent space size 100, Adam optimizer, Gumbel distribution in Valina-GAN i.a) Alpha 0.25 i.b) Alpha 0.5 i.c) Alpha 0.75 ii.a) Beta 2.0 ii.b) Beta 3.0 ii.c) Beta 4.0

In Figure 72, when the smooth calibration method, alpha was changed from 0.25 to 0.75 through 0.25 increments, and beta was changed from 1 to 4 through 1 increase. Experimental results show that the influence of attention on the impact of alpha and Gamma does not affect the generation of the generated image. Scale term, the effect of the parameter in the Scale, is less influential in creating the image in the

Valina-GAN. These results show that, when the scale term enters, it is corrected for the acquisition of the continuous task information so that results obtained. However, it confirmed that the influence on the steady of the continuous task information acquisition is independent of the impact on Gamma and Alpha. It can be confirmed that continuous task information is obtained stably through the scale term.

13) Problem analysis at smooth correction

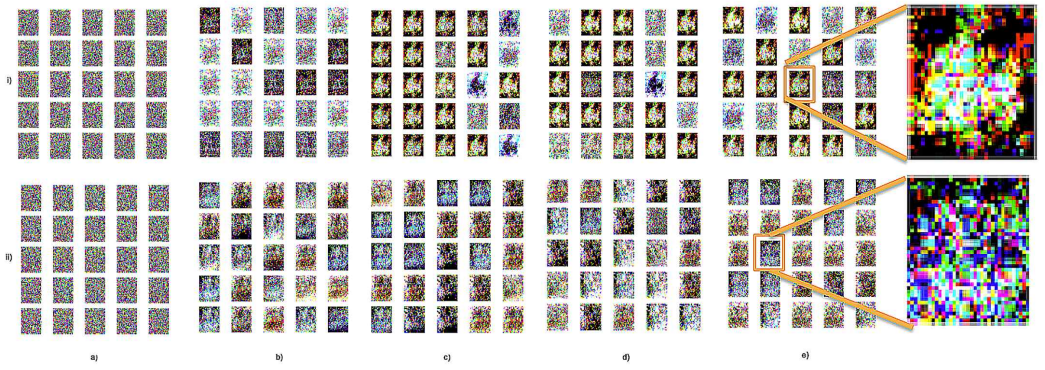


Figure 73 Smooth correction method training process about smooth correction at Learning rate 0.0007,4 batch size, L1 0.25, L2 0.25, 100 Z latent space size, Adam optimizer, Scale correction at scale coefficient of Alpha 0.75, scale coefficient of Gamma 4.0 compare about i) Gumbel distribution and ii) Random distribution a) 0 Epoch, b) 250 Epoch, c) 500 Epoch, d) 750 Epoch, e) 1000 Epoch.

In Figure 73., we show the effect of acquiring images generated by epoch 250. Figure 73(c) and Figure 73(d), which are similar to the two segments where the fluctuation occurs now, show that the images generated vary greatly compared to the previous step.
In other words, the state showing the variation in the loss expects to reflect the new value, so that the new image learn and the variation occurs. A more precise definition of the variety of the loss graph can help to interpret the loss trend of the model by comparing the loss shapes defined by spike neurons.

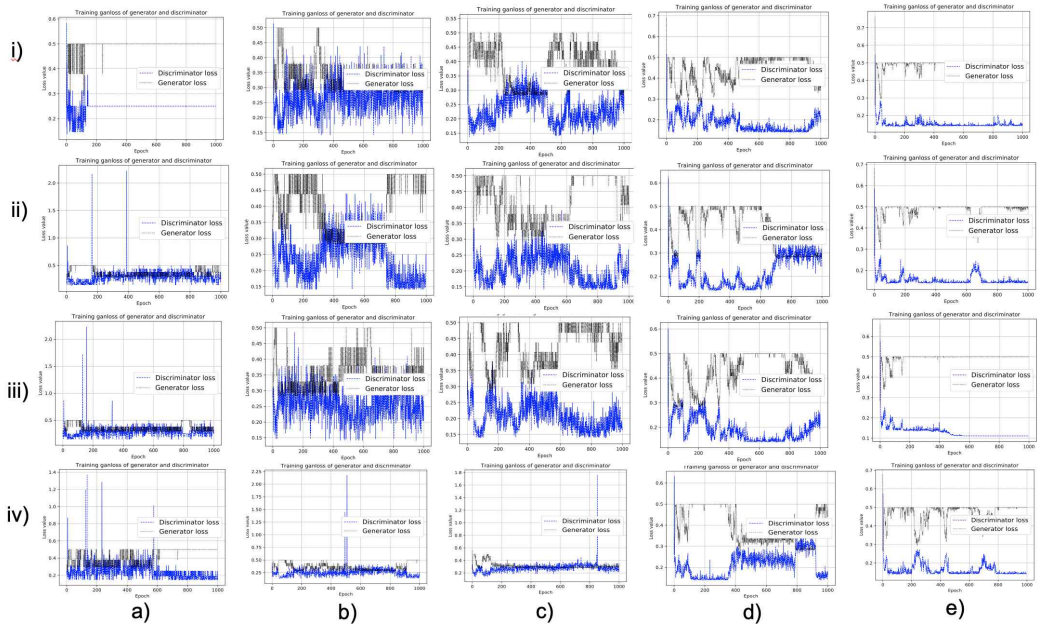


Figure 74 An indirect comparison of the initial distribution of the catastrophic problem every time new data is learned during model training. Smooth correction at Learning rate 0.0007, Batch size 16, L1 0.25, L2 0.25, Z latent space size 100, Adam optimizer, Alpha 0.75, Gamma 4.0 a) Batch size 4 , 8, 16, 32 and 64 in Cifar10, i) Random distribution, ii) Laplace distribution, iii) Logistic distribution, and iv) Gumbel distribution

This was verified by batch to verify the effect of continuous task information acquisition according to the number of data inputs in distribution.

Figure 74 shows the results of the model learning run when the Batch size differs according to each distribution. It seems that the convergence positions of epochs are different from each other, depending on the type of each distribution.

The convergence of the converged position to the fast covariance seems to be because the distribution reflects the convergence speed of the model increases as the continuous task information of the model.

Also, most of the loss graphs seem to be fluctuating between epoch 400 to 600 and 600 to 800.

As shown in Figure 75, each 5by5 image is a Random different sample. However, all 25 generated shapes show similar shapes. It knows that the Valina-GAN model has a problem with mode collapsing. The mode

collapsing phenomenon is a phenomenon that memorizes only one image in the course of learning. Therefore, it assumes that this cause by the mode collapsing occurring in the GAN series, not by the effect caused by the loss.

And the distribution of Gumbel shows a tendency to be slightly wider in white that allows the Laplace distribution to help interpret the effect of the generated image. Image acquisition from such a Laplace distribution does not have a tail width and distribution range that can reliably reflect the tilt and thus tends to fail to acquire tilt. However, we can verify that the Gumbel distribution to be used as an initial distribution has a thicker tail than other distributions and has an advantage in generating an image by acquiring the continuous task information of the distribution width stably.

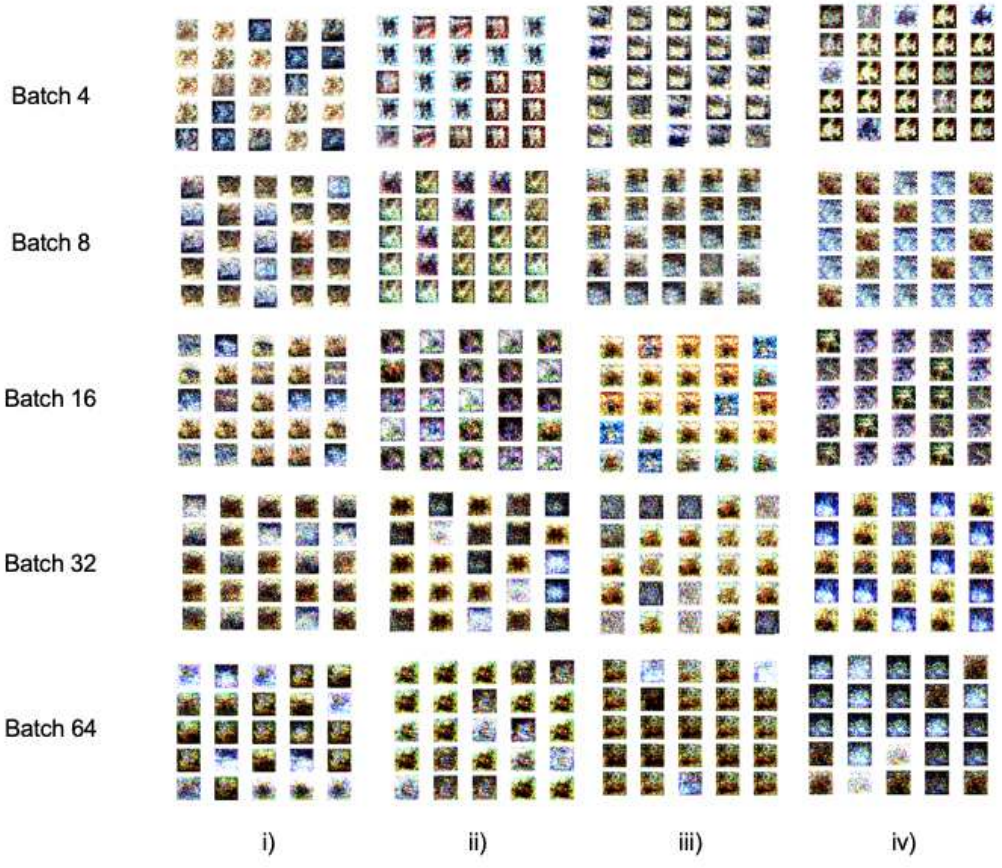


Figure 75 Continuous task information reflection analysis in the color image according to various Batch size and initial distribution. Smooth correction at Alpha 0.75, Gamma 4.0, Learning rate 0.0007, Batch size 4, L1 0.25, L2 0.25, Z latent space size 100, Adam optimizer, i) Random distribution, ii) Laplace distribution, iii) Logistic distribution, and iv) Gumbel distribution

When you look at the picture shown in Figure 75, you can see that it cluster with similar color values. This clustered distribution shows that the width varies. It confirms that the generated image is affected by the width of the existing distribution. 16 Batch, the Laplace distribution, and Gumbel distribution show that the white part of the Laplace sharper.

14) Parameter analysis using L1 and L2 regularization at smooth correction

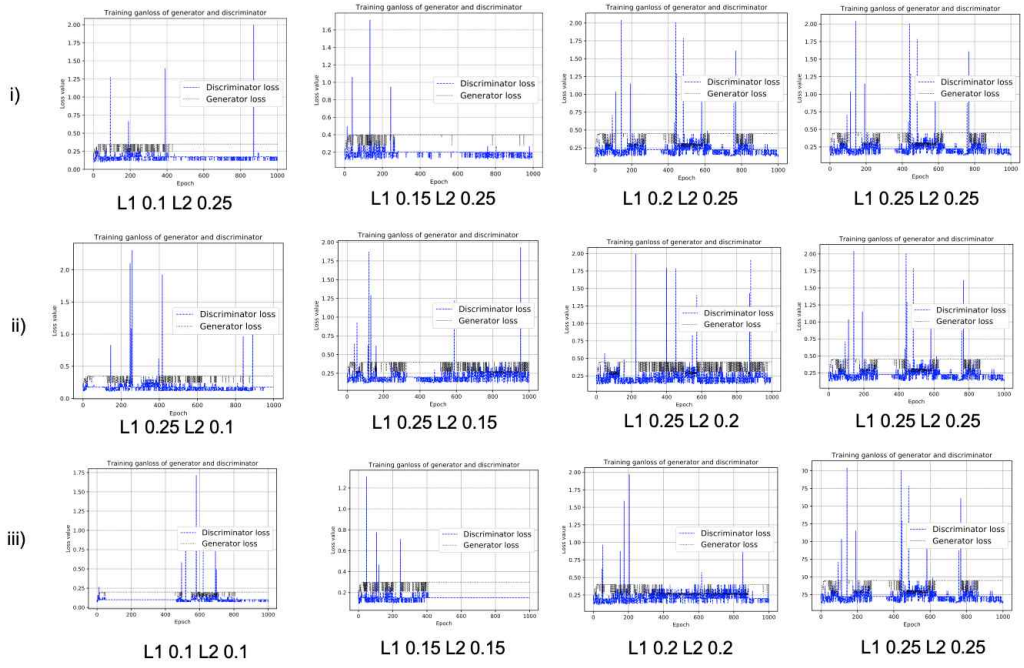


Figure 76 Analysis of smooth correction from the cost function perspective cost function about the effect of various regularization methods by Batch size 4, Z latent space 100, Scale correction coefficient at Alpha 2.0, Gamma 0.25, Valina-GAN, Cifar100 dataset, i) L1 & L2 about the L1 increase, ii) L1 & L2 about L2 increase, iii) L1 & L2 about L1 & L2 increase.

Figure 76 From the viewpoint of is the cost function, various regularization methods analyze according to the smooth correction cost function. Figure 76(i) shows the result as the L1 coefficient value increases. As L1 increases, it sees that vibration frequently occurs as Epoch increases. However, Figure 76 (ii) shows the result according to the increase of the L2 coefficient value. In the case of L2, the vibration tends to increase with increasing epoch. Also, it shows that the cost function of the concatenated part of the initial, middle part maximizes each other and learns. Figure 76(iii) shows the tendency of cost function when L1 and L2 increase at the same time. At this time, the magnitude of the vibration width increases as the value increases. It also shows that the number of oscillations increases and then decreases in Figure 76(ii)

From these results, it seems that the normalization effect produced by the linear combination of L1 and L2 is dependent on the size and the particular methods.

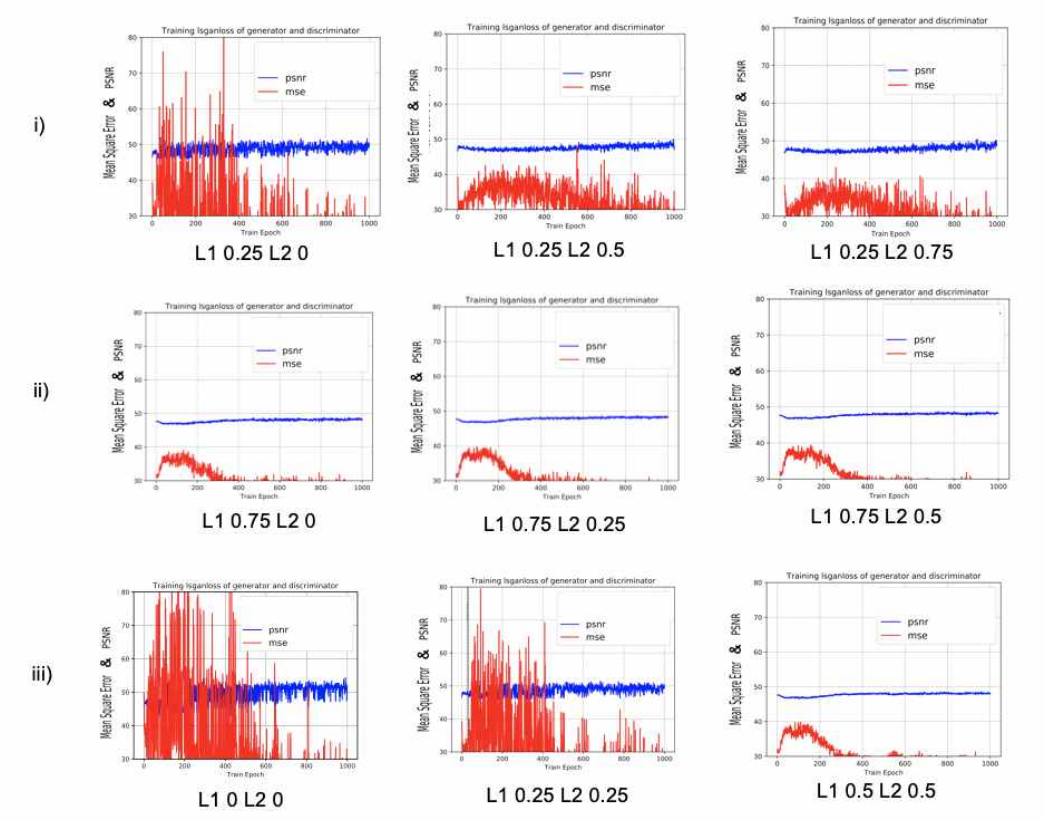


Figure 77 Smooth correction analysis from the viewpoint of PSNR and

MSE by Batch size 4, Z latent size 100, Adam optimizer, Fashion MNIST dataset, Valina-GAN, i) L2 increase at L1 0.25, ii) L2 increase at L1 0.75, and iii) L1 & L2 increase.

To analyze the influence of size value, we compare the performance of PSNR and MSE in Figure 77. First, we measure performance according to the increase of L2 value when L1 fixe to 0.25, When the value of 0.75 fixes, the influence of the increase of the L2 value and the third of the increase of the L1 and L2 values were analyzed. The first result shows that the amplitude of the MSE decreases as the L2 value increases. The second result shows that the variation of the MSE value is not as large as the L2 value increases. As the final result increases, the result of MSE decreases. Based on these results, if the L1 value is too large, the variation of L2 has an irrelevant effect, and the MSE value is shown to be reduced stably when the linear combination of L1 and L2 is equal. As a result, it seems that the performance varies depending on the conditions of the linear combination of L1 and L2 regularization.

15) Regularization analysis using parameter changing at smooth correction

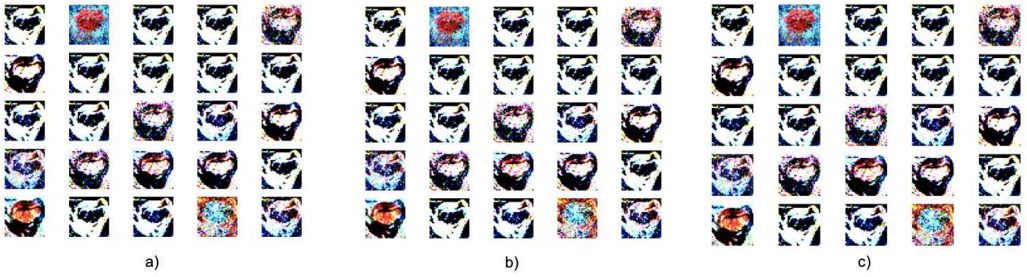


Figure 78 Exponential regularization for acquisition continuous task information analysis according to the variation of alpha in the scale term by Learning rate 0.0007, Batch size 2, L1 0.0, L2 0.15, Z latent space size 500, Adam optimizer, Gumbel distribution, Scale correction coefficient at Gamma 0.25, a) Alpha 2.0, b) Alpha 3.0, c) Alpha 4.0.

Figure 78 is the result of various Scale correction coefficients on the generated image by Valina-GAN. Thus, the Exponential regularization method show by acquiring the continuous task information independently of the cost function.

국 문 초 록

딥 러닝의 성능 향상 기법 및 응용에 관한 연구

한성대학교 대학원

전자정보공학과

전자정보공학전공

최 승 호

딥러닝은 현재 많은 실생활에서 높은 성능을 보여 다양한 환경에 적용되어 연구가 진행되어 오고 있다. 그러나 딥러닝은 블랙박스 모델이라 해석하기가 어렵고 또한 왜 좋아지는지 이해하기가 어렵다. 따라서 본 논문에서는 기존 딥러닝의 성능 향상을 위해서 딥러닝 성능 향상 기술 및 딥러닝을 이용한 응용 분야에 대해서 연구된 내용을 살펴본다. 딥러닝의 성능을 향상시키기위해서는 어떠한 부분에 문제점이 있는 지를 파악해서 개선한 딥러닝 기술 향상 관련 내용에 대해서 소개한다.

딥러닝 성능 향상에서는 7가지의 기술에 대해서 살펴본다. 첫번째로 양방향 활성화 기능 : 컨볼루션 신경망에서 향상된 활성화 기능이다. 두번째로 온라인 지속적인 작업 학습을 통해 규모적 보정 캐스케이드 연속 발생 적대 네트워크의 손실이다. 세번째로 비선형 지수 정규화 : 딥러닝 모델을 위한 개선된 정규화 버전이다. 네번째로 딥러닝 모델 최적화를 위한 새로운 보조 구성 요소이다. 다섯번째로 안정적인 학습을 위한 앙상블 노말라이제이션이다. 여섯번째로 실제 가짜 지문과 DCGAN에 의해 생성된 가짜 지문의 유사성 분석이다. 일곱번째로 음악 작곡을 위한 적응 적 정규화를 갖는 원-핫 양방향 Seq2Seq에 에러 감소 임베딩을 갖는 다중 경로 디코더 스킴이다.

그리고 높은성능 표현하는 딥러닝을 실제 생활에 응용한 기술에 대해서 소개한다. 딥러닝을 응용한 기술에서는 4가지의 기술에 대해서 살펴본다. 첫번째로 적응적 시드의 중요성 연구이다. 두번째로 이미지 캡셔닝에서 모듈 비

교연구이다. 세번째로, 이상치데이터에 대해서 시각화이다. 네번째로 U-Net 구조에서 배치 노말라이제이션 그리고 포컬 로스 와 L1 규제화을 이용한 안정적인 미세정밀한 세그멘트 획득이다.

이를 통해서 딥러닝을 이용한 새로운 딥러닝 이론을 만들어 딥러닝 모델의 성능 향상 및 새로운 연구 분야에 대해서 미래 연구로 진행하고자 한다.