Study on the Effect of Data Augmentation on Image Learning

Tae-hwan Choi

ABSTRACT

This paper aims to demonstrate the effective use of data in AI research by examining the impact of data augmentation on AI learning. This study utilized a subset of the AI Hub's Ocean Deposition Garbage Image dataset. The accuracy of image classification was measured using CNN's VGG16 and VGG19. After increasing the data volume through augmentation, the accuracy was re-measured and re-compared. The results of the study indicated that accuracy improved in most cases, leading to the conclusion that data augmentation positively affects image learning.
This paper is revision of the author's master degree thesis. and This research Ocean Deposition Garbage Image used datasets from 'The Open AI Dataset Project (AI-Hub, S. Korea)'. All data information can be accessed through 'AI-Hub (www.aihub.or.kr)'.

☞ keyword : AI, Data Augmentation, Image, CNN, VGG

1. Introduction

Artificial intelligence is considered one of the core technologies of the Fourth Industrial Revolution. AI has been described as a "machine equipped with cognitive, judgment, action, and learning functions."(Choung Hye-Uk, 2021).

AI technology is being applied and utilized in various fields such as education, art, industry, and counseling. So-yeon Park, Chang-yeop Lee, Chan-hyung Ahn(2020) stated in an article that "different institutions classify AI-related technologies in varying ways, leading to differences in both the number and scope of categories. Among the 14 institutions studied, commonly cited technological areas include visual intelligence, language intelligence, machine learning, speech intelligence, robotics, and expert systems. Among these, visual intelligence is described as a field of technology that recognizes image or video data to assess situations or processes data to generate new images or videos."

Among the technological fields, computer vision－an image recognition technology within visual intelligence－is widely utilized. In a study by Song Jae-min, Lee Sae-bom, and Park Ah-reum(2020), various cases are highlighted. For example, when a camera installed beneath a tractor captures soil images, AI analyzes the data to distinguish between weeds and crops, enabling the precise spraying of herbicide only on weeds and applying fertilizer to crops, thereby increasing crop yields. Other examples include tools that utilize images captured by drones to measure stockpile volumes and services that analyze satellite images with AI to identify the latest trends in diverse economic activities occurring worldwide.

Data is a critical element in the utilization of artificial intelligence. Mi-jeong Park(2019) stated, "Most AI applications require vast amounts of available data to learn and make intelligent decisions." Similarly, Hyung-il Koo(2018) observed that "deep learning tends to exhibit improved performance in proportion to the volume of data." Summarizing these insights, it becomes evident that a large quantity of data is essential for effectively utilizing AI technology. In particular, the application of computer vision technology necessitates extensive image data.

However, collecting and accumulating data requires substantial costs, and various challenges, such as issues with the quality of the data collected at great expense, are likely to arise.

Therefore, one approach to addressing these challenges is to use data augmentation techniques. Hee-chan Cho(2020) defined data augmentation as "a technique that increases the amount of data using various algorithms based on a small amount of initial data." By utilizing this technique, it becomes possible to train AI models with a relatively limited

quantity of data.

2. Literature Review

2.1 Data Augmentation

Soo-Young Cho(2020) defined data augmentation as “a technique that generally increases data by randomly transforming the information of images or videos into other forms.” Joo-yeong Song(2021) described it as “a method that uses subsets of the original data, or subsequences, as augmented data.” Jeong-hwa Lim(2023) defined it as “a data augmentation technique that synthesizes and generates data by maximizing the use of the various characteristics of the given data.” Sung-nam Kim(2023) stated that it involves “creating semantically identical new data by transforming the original data in various ways,” while Chan-il Park(2021) described it as “a method that randomly modifies information such as images, text, and signals to increase the data with similar yet distinct information.” Lastly, Jin-sang Lee(2023) referred to it as “a technique for increasing the amount of data by applying various methods in situations where data is scarce.”

In summary, data augmentation can be described as a method of increasing the amount of existing data by generating identical or similar data through the transformation of various types of data in specific ways. To utilize artificial intelligence effectively, it is essential to train models using collected data, which requires a large quantity of data with a high level of quality. However, in cases where data collection is challenging, data augmentation offers advantages by increasing the amount of data available for AI training.

There are various methods of data augmentation. Connor Shorten and Taghi M. Khoshgoftaar(2019) identified techniques such as geometric transformations, color space transformations, kernel filters, mixing images, random erasing, feature space augmentation, adversarial training, GAN-based augmentation, neural style transfer, and meta-learning schemes.

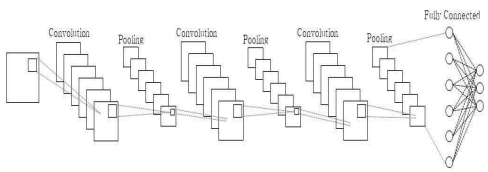
2.2 CNN(Convolution Neural Network)

2.2.1 Definition of CNN

Eun-Ju Lee(2017) defined CNN as “an optimal method for extracting high-level, abstracted features or processing texture information from images.” In-kyeong Hwang(2017) described it as “a type of deep learning that represents a feed-forward artificial neural network, which resembles the pattern connections between neurons that play a crucial role in processing visual information in animals.” Su-Bin Park(2019) defined CNN as “a type of artificial neural network particularly specialized for the field of computer vision, based on convolution operations to extract unique features of images and train these extracted features through the neural network.” Hyeon-Ho Lee(2020) described CNN as “a type of deep neural network(DNN) that uses convolution in one or more of its layers.” Kang-Hyun Park(2021) defined it as “a deep learning algorithm inspired by the human object recognition process, a learning model that adds two types of layers－convolutional layers and pooling layers－to traditional artificial neural networks.” Finally, Gyeong-Won Jang(2023) described it as “a type of DNN that mimics the structure of the human optic nerve.”

CNN is an advanced form of the Deep Neural Network(DNN). The DNN architecture consists of an input layer, multiple hidden layers, and an output layer. Due to the multiple hidden layers, the learning outcomes are enhanced compared to the Artificial Neural Network(ANN). However, in the case of large-scale image or video data, training can require a significant amount of time, and there is an issue of spatial information loss during the flattening process into a one-dimensional form.

The general structure of a CNN is shown in Figure 1 below.



(Figure 1) CNN Structure(Hyeon-Ho Lee, 2020)

CNN is “composed of three key structures: the

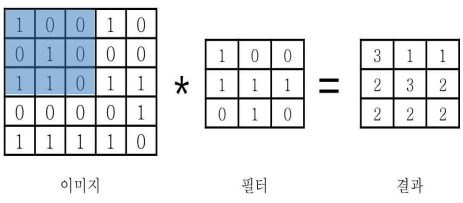
Convolution Layer, Pooling Layer, and Fully Connected Layer"(Ian Goodfellow, et al., 2016; quoted in In-kyeong Hwang, 2017). In CNNs, features are extracted in the Convolution and Pooling Layers, while classification occurs in the Fully Connected Layer.

2.2.2 Convolution Layer

In the Convolution Layer, convolution operations are performed to extract features using filters (also known as kernels). If the stride value is set to 1 during convolution, the filter moves one step at a time. For grayscale images, the channel value is 1, but for RGB images, the channel value is 3. The result of these calculations generates a feature map, with the number of feature maps depending on the number of filters used.

Kwang-Bok Hwang(2021) stated that "the position and interval settings for applying filters to input data, along with the filter size, affect the size of the output data."

The method for calculating convolution can be illustrated with an example: applying a 3×3 filter with a stride of 1 to a 5×5 original image matrix. The result of the convolution operation is shown in Figure 2 below.



(Figure 2) Example of Convolution Operation

The operation is calculated as shown in Figure 3 below, and by shifting one step at a time, the same calculation is repeated to obtain the result.

(1 x 1) + (0 x 0) + (0 x 0) + (0 x 1) + (1 x 1) + (0 x 1) + (1 x 0) + (1 x 1) + (0 x 0)

(Figure 3) Example of Convolution Operation Method

Min Ae-ri(2020) stated that "the Convolution Layer applies filters to input data to minimize the number of shared parameters and uses nonlinear activation functions (such as ReLU or Sigmoid) to identify features in the image."

2.2.3 Pooling Layer

The Pooling Layer takes the output of the Convolution Layer as input and reduces its size or emphasizes certain parts of the data. Kwang-Bok Hwang(2021) stated that the Pooling Layer "does not contain parameters like the convolution operation and performs independent operations for each channel, generating output data with the same number of channels as the input data." As the number of filters increases, the feature maps grow, resulting in a higher-dimensional parameter count, which can lead to overfitting and increased time requirements for model size and output. Therefore, the Pooling Layer is used to reduce dimensions.

Jong-Bum Kim and Sung-kwun Oh(2015) stated that "in the Pooling Layer stage, the size is reduced by an integer factor, primarily through Max Pooling or Average Pooling."

Hyun-Su Lee(2018) noted that Max Pooling is also known as subsampling. Max Pooling involves applying the maximum value within the filter size and then moving by the stride amount to perform the calculation.

Average Pooling applies the average value within the filter size and moves by the stride amount to perform the calculation. Another method, Min Pooling, applies the minimum value within the filter size and moves by the stride amount to complete the calculation.

2.2.4 Fully Connected Layer

In-kyeong Hwang(2017) stated that "the Fully Connected Layer is similar to a convolutional layer with a filter size of (1×1). It fully connects the features extracted from the previous layers, transforming all activation values into a one-dimensional vector to enable classification."

Gyeong-Won Jang(2023) stated that "as the final layer of CNN, this stage determines the

classification. Each layer is converted into a one-dimensional array, and the final output is determined through an activation function. The size of the final output corresponds to the number of classes of the input data to be classified."

Han-Nah Kim(2020) stated that "the Fully Connected Layer functions to connect all input and output neurons, and the softmax activation function used in this layer performs multi-class classification, outputting probability values for each class."

In other words, the Fully Connected Layer is the final classification stage in the CNN architecture, where classification is performed using the softmax function.

2.3 VGG(Visual Geometry Group)

2.3.1 VGGNet

The ILSVRC (ImageNet Large Scale Visual Recognition Challenge) is one of the major conferences in computer vision, held annually.

VGG is a model that achieved outstanding results in the ILSVRC 2014. In their paper, Karen Simonyan and Andrew Zisserman(2014) developed six different VGGNet architectures to compare their performance. Among these, the structure with 16 layers is known as VGG16, while the one with 19 layers is referred to as VGG19.

Ae-Ri Min(2020) stated that "VGG16 focuses on using convolution layers with 3×3 filters with a stride of 1, instead of relying on a large number of hyperparameters, and consistently uses same-padding and max-pooling layers with 2×2 filters and a stride of 2."

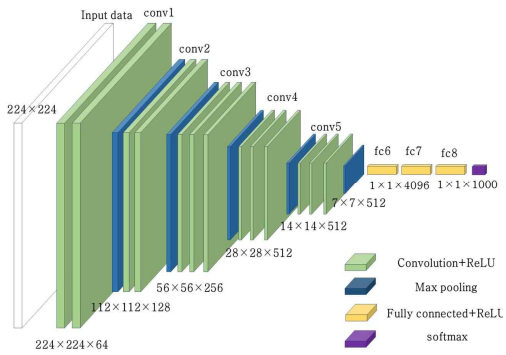
Karen Simonyan and Andrew Zisserman(2014) stated that the input image size for VGG is fixed at 224×224, and the preprocessing involves subtracting the mean RGB value from each image. In the Convolution Layers, 3×3 filters with a stride of 1 pixel are used, with padding applied to preserve resolution. The architecture includes a total of five max-pooling layers, each performed with 2×2 pixels and a stride of 2. The Fully Connected Layer consists of two layers with 4096 channels each, followed by a third layer for classification with 1000 channels, and finally a softmax layer. This structure incorporates ReLU nonlinearity in all hidden layers (Krizhevsky et al., 2012; quoted in Karen Simonyan & Andrew Zisserman, 2014).

2.3.2 VGG16

VGG16, a CNN model proposed by Karen Simonyan and Andrew Zisserman at Oxford University, is one of the renowned models submitted to ILSVRC-2014. While it has drawbacks such as requiring substantial training time and large network weights, it is relatively easy to implement (Liang Han Sheng, 2021).

Jun-Yeol Ryu, Tae-Wan Kim, Ja-Hoon Jeong(2022) stated that "despite its deep 16-layer structure, VGG16 uses 3×3 convolution filters, resulting in a relatively small number of training parameters. This provides greater nonlinearity, making it an efficient model in terms of time and accuracy."

Figure 4 illustrates the structure of VGG16, which is composed of a total of 16 layers: 13 Convolution Layers and 3 Fully Connected Layers.



(Figure 4) VGG16 Structure(Liang Han Sheng, 2021)

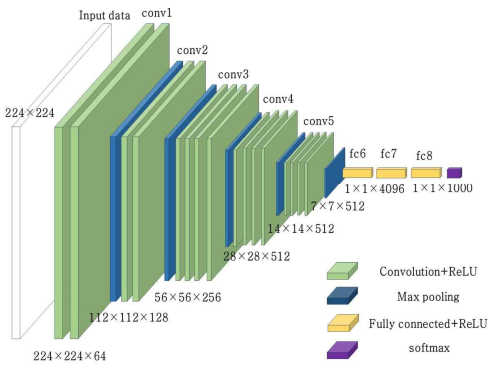
2.3.3 VGG19

(Figure 5) illustrates the structure of VGG19, which consists of a total of 19 layers: 16 Convolution Layers and 3 Fully Connected Layers.

In the study "Performance Comparison of CNN Models for Retinal Disease Diagnosis" by Hye-won Kim, Xuefeng Zhang, Yong-Soo Kim, Il-Hong

Jung(2022), experiments applying VGG16, VGG19, ResNet, and DenseNet showed that the models performed in the following order: DenseNet, ResNet, VGG19, and VGG16.

In Jeong-Hwan Lee's (2021) study, "Detection of Number and Character Areas of Car License Plates Using Deep Learning and Semantic Image Segmentation," ResNet18, ResNet50, VGG16, and VGG19 were used, with results showing performance in the order of ResNet50, ResNet18, VGG19, and VGG16. Thus, prior research indicates that VGG19 performs better than VGG16.



(Figure 5) VGG19 Structure

3. Methods

3.1 Methods

3.1.1 Research Procedure and Method

First, I explored the research topic. To create a robust model through AI learning, a large amount of data is essential. However, obtaining such extensive data proved challenging. Upon investigation, I found that data augmentation is a method that can increase the volume of existing data by utilizing the original data itself. This led me to focus on examining the effects of data augmentation on learning, aiming to provide a foundation for applying data augmentation in future research. Therefore, I decided to investigate whether data exists to support the impact of data augmentation on learning.

Next, I determined which data to use for data augmentation and found that socially relevant topics are of interest. Consequently, I identified the “Marine Submerged Waste Images” dataset available on AI Hub. This dataset includes sonar survey images and underwater photography images. The sonar survey images consist of categories such as tires, traps, nets, wood, and ropes, while the underwater images are categorized into ropes, wood, nets, tires, and traps.

The "Ocean Deposition Garbage Images" dataset from AI Hub was established with funding from the Ministry of Science and ICT and support from the National Information Society Agency of Korea. Having secured suitable data to carry out the explored research topic, I finalized the decision to conduct the research using this dataset.

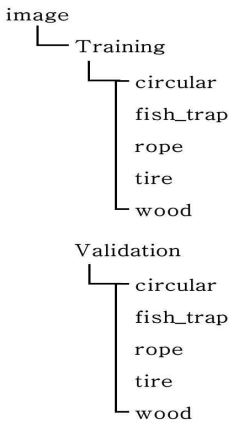
Next, I selected the research method. To examine the impact of data augmentation on images, I decided to measure the differences before and after applying data augmentation. Among deep learning methods, CNN is known for its effectiveness in image processing, so I chose to conduct the research using CNN. Specifically, I selected the VGG model due to its ease of implementation and efficiency. I planned to measure the performance of VGG16 and VGG19 three times each before and after data augmentation, setting the number of epochs—representing the training cycles of the data—to 10 for this study.

In this study, an experiment was conducted to examine the impact of data augmentation on images. The procedure involved applying CNN’s VGG16 and VGG19 models to the original image dataset obtained for the research and measuring accuracy. Next, data augmentation was applied to double the number of images, after which accuracy was measured again using the VGG16 and VGG19 models. Following the completion of implementation and execution, I planned to compare the accuracy before and after data augmentation to assess any performance differences. The data augmentation methods included image rotation, vertical and horizontal translation and transformation, brightness adjustment, flipping, zooming, and shifting.

After selecting the research method, I performed data classification to prepare the data for use. As

shown in Figure 6, images were categorized into folders by type. In this study, only the underwater photography image data were used. In the training data, I focused on underwater images of ropes, wood, nets, tires, and traps, excluding images where multiple objects appeared to be overlapping, due to concerns about quality degradation. I used a randomly selected subset of training data and validation data for the study.

Ultimately, the image data used in the study consisted of 4,618 training images: 1,010 circular, 1,331 fish_trap, 472 rope, 1,028 tire, and 777 wood images. The validation data included a total of 3,008 images: 368 circular, 978 fish_trap, 331 rope, 765 tire, and 566 wood images.



(Figure 6) Data Classification

After completing data classification, the VGG16 and VGG19 models were implemented, and the code was executed to measure the pre-selected performance metrics. Finally, the results were interpreted based on the metrics obtained from the study.

3.1.2 Research Environment

The environment in which this study was conducted is shown in Table 1 below.

NVIDIA's CUDA (Compute Unified Device Architecture) is a development environment tool for creating high-performance GPU-accelerated applications and enabling GPU acceleration.

Previously, to perform general parallel computing, programmers had to configure every detail of the GPU. To address this issue, NVIDIA enhanced the flexibility of the traditional Geforce series, transforming it from a graphics-specific architecture to one capable of supporting general-purpose programming(Ji-Hoon Jo, 2013).

(표 1) Research Environment

항 목	내 용
CPU	AMD Ryzen5 7600
Memory	DDR5-5600(2800 ㎒) 16G * 2EA
Graphics	Geforce RTX 3090 24G Driver Version : 546.33
OS	Windows 10 64bit
Python	Anaconda3 2023.09-0(64-bit) Python 3.8.18
Tensorflow	2.8.0
Tensorflow-gpu	2.4.1
cuda	CUDA 11.0
Library	cuDNN 8.0.4

cuDNN (CUDA Deep Neural Network) is a GPU-accelerated library for deep neural networks that facilitates the implementation of standardized operations such as convolution, attention, matrix multiplication, pooling, and normalization (NVIDIA Developer).

The supported CUDA version varies depending on the GPU in the development environment, and cuDNN must be installed to match the installed Python and CUDA versions, so careful attention to compatibility is required.

3.2 Evaluation Metrics

The metric defined to verify the effectiveness of this study is accuracy. Accuracy represents the number of successful predictions out of the total data, with higher accuracy indicating better predictive performance. Since accuracy is a key metric for model evaluation, it was selected as the primary measurement indicator.

Accuracy refers to the number of successful predictions out of the total data. It can be calculated using the following formula:

$$Accuravy = \frac{TP + TN}{TP + FN + FP + TN}$$

TP(True Positive) refers to cases where the actual value is true, and the predicted value is positive, meaning that the actual and predicted values match. TN(True Negative) occurs when the actual value is false, and the predicted value is negative, indicating that the actual and predicted values match. TP and TN represent successful predictions where the actual and predicted values align. FP(False Positive) is when the actual value is false, but the predicted value is positive, meaning the actual and predicted values differ. FN(False Negative) occurs when the actual value is true, but the predicted value is negative, also indicating a mismatch. FP and FN represent failed predictions where the actual and predicted values do not align.

Dividing the total data TP+FN+FP+TN by the value of TP+TN which represents correct predictions, yields the accuracy. A higher accuracy value indicates greater predictive accuracy. However, care should be taken when applying accuracy to datasets with imbalanced data, as it may not fully reflect the model's performance in such cases.

4. Results

4.1 Performance Measurement Before Applying Data Augmentation

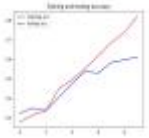
4.1.1 VGG-16

Table 2 shows the performance measurements of VGG16 before applying data augmentation. In this table, accuracy is divided into two categories: accuracy represents the accuracy on the training data, and val_accuracy represents the accuracy on the validation data. Similarly, loss denotes the loss value for the training data, and val_loss indicates the loss value for the validation data.

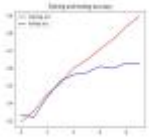
(Table 2) Performance Measurement Results of VGG16 Before Applying Data Augmentation

epoch	1		2		3	
	accuracy	val_	accuracy	val_	accuracy	val_
1	0.2735	0.3235	0.2921	0.3328	0.2739	0.3251
2	0.3224	0.3504	0.3234	0.3211	0.2880	0.3511
3	0.3344	0.3364	0.4428	0.4362	0.3031	0.3916
4	0.4169	0.4066	0.5004	0.5266	0.3961	0.3810
5	0.4904	0.4757	0.5990	0.5632	0.4542	0.4315
6	0.5493	0.5376	0.6292	0.5741	0.5379	0.4581
7	0.6177	0.5293	0.7068	0.6110	0.5645	0.5259
8	0.6743	0.5831	0.7684	0.5984	0.6385	0.6011
9	0.7491	0.5964	0.8355	0.6287	0.6955	0.6077
10	0.8201	0.6114	0.8996	0.6230	0.7753	0.6147

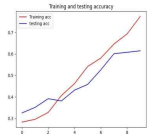
Figures 7, 8, and 9 graphically represent the results from Table 2. Observing the trends in the graphs, we can see that all values increase as epochs progress. This indicates that accuracy improves with the number of training iterations.



(Figure 7) First Measurement of VGG16 Accuracy Before Applying Data Augmentation



(Figure 8) Second Measurement of VGG16 Accuracy Before Applying Data Augmentation



(Figure 9) Third Measurement of VGG16 Accuracy Before Applying Data Augmentation

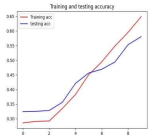
4.1.2 VGG-19

Table 3 shows the performance measurements of VGG19 before applying data augmentation, and Figures 10, 11, and 12 graphically represent these results. Observing the trends in the measured results, all three sets of results generally show an upward trend. Thus, accuracy increases with the number of training iterations.

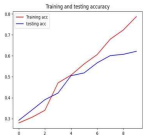
(Table 3) Performance Measurement Results of VGG19 Before Applying Data Augmentation

epoch	1		2		3	
	accuracy	val_ accuracy	accuracy	val_ accuracy	accuracy	val_ accuracy
1	0.2712	0.3245	0.2724	0.2916	0.2684	0.2543
2	0.2944	0.3251	0.2842	0.3404	0.2661	0.3251
3	0.2887	0.3281	0.3374	0.3903	0.2996	0.3022
4	0.3226	0.3561	0.4579	0.4219	0.3986	0.4146

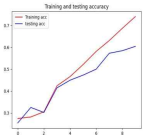
5	0.3772	0.4209	0.4896	0.5050	0.4739	0.4495
6	0.4458	0.4561	0.5462	0.5183	0.5216	0.4727
7	0.4947	0.4691	0.5969	0.5665	0.5797	0.5000
8	0.5403	0.4940	0.6789	0.6011	0.6208	0.5725
9	0.5787	0.5532	0.7189	0.6070	0.6871	0.5841
10	0.6566	0.5808	0.7941	0.6217	0.7252	0.6041



(Figure 10) First Measurement of VGG19 Accuracy Before Applying Data Augmentation



(Figure 11) Second Measurement of VGG19 Accuracy Before Applying Data Augmentation



(Figure 13) Third Measurement of VGG19 Accuracy Before Applying Data Augmentation

4.2 Performance Measurement After Applying Data Augmentation

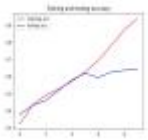
Data augmentation was applied to double the training data by adding transformations such as image rotation, vertical and horizontal shifts, and scaling. The VGG16 and VGG19 models were then applied to measure performance. For accurate assessment, performance was measured a total of three times.

4.2.1 VGG-16

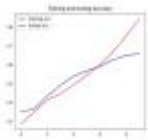
Table 4 shows the performance measurements of VGG16 after applying data augmentation, and Figures 13, 14, and 15 graphically represent these results. Observing the measurement results, all values generally increase as epochs progress, indicating that accuracy improves with the number of training iterations.

(Table 4) Performance Measurement Results of VGG16 After Applying Data Augmentation

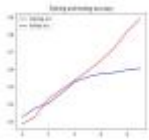
epoch	1		2		3	
	accuracy	val_ accuracy	accuracy	val_ accuracy	accuracy	val_ accuracy
1	0.2977	0.3767	0.2666	0.3494	0.2778	0.3251
2	0.4313	0.4318	0.3381	0.3660	0.2993	0.3763
3	0.4850	0.4601	0.4090	0.4325	0.4219	0.4043
4	0.5223	0.5203	0.4389	0.4897	0.4802	0.4634
5	0.5648	0.5795	0.4834	0.5379	0.5256	0.5276
6	0.6142	0.6184	0.5399	0.5665	0.5796	0.5562
7	0.6985	0.5934	0.5952	0.5901	0.6520	0.5745
8	0.7814	0.6253	0.6622	0.6253	0.7251	0.5808
9	0.8733	0.6360	0.7594	0.6476	0.8091	0.5964
10	0.9373	0.6443	0.8480	0.6582	0.8963	0.6054



(Figure 13) First Measurement of VGG16 Accuracy After Applying Data Augmentation



(Figure 14) Second Measurement of VGG16 Accuracy After Applying Data Augmentation



(Figure 15) Third Measurement of VGG16 Accuracy After Applying Data Augmentation

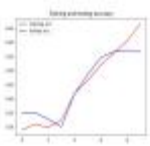
4.2.2 VGG-19

Table 5 shows the performance measurement results of VGG19 after applying data augmentation, and these results are represented graphically.

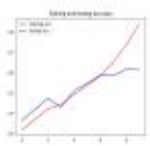
(Table 5) Performance Measurement Results of VGG19 After Applying Data Augmentation

epoch	1		2		3	
	accuracy	val_ accuracy	accuracy	val_ accuracy	accuracy	val_ accuracy
1	0.2739	0.3504	0.2908	0.3637	0.2741	0.3354
2	0.3119	0.3511	0.3672	0.4166	0.2983	0.3444
3	0.3041	0.3275	0.4200	0.4737	0.3114	0.3431
4	0.3134	0.3005	0.4469	0.4282	0.3612	0.4189
5	0.3971	0.4176	0.4824	0.5110	0.4354	0.4801
6	0.4512	0.4857	0.5325	0.5452	0.4844	0.5027
7	0.5129	0.5459	0.5838	0.5868	0.5090	0.5572
8	0.5619	0.5668	0.6511	0.5841	0.5805	0.5509
9	0.6021	0.5725	0.7373	0.6180	0.6311	0.5848
10	0.6697	0.5662	0.8475	0.6124	0.6848	0.6074

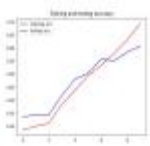
Figures 16, 17, and 18 graphically represent these results. Observing the trend in the graphs, accuracy increases with epochs for both the training and validation data, indicating that accuracy improves with the number of training iterations.



(Figure 16) First Measurement of VGG19 Accuracy After Applying Data Augmentation



(Figure 17) Second Measurement of VGG19 Accuracy After Applying Data Augmentation



(Figure 18) Third Measurement of VGG19 Accuracy After Applying Data Augmentation

5. Conclusion

5.1 Research Conclusion

This study aimed to determine the impact of data augmentation on image learning by comparing the results of CNN's VGG16 and VGG19 models before and after data augmentation.

The results, following the research sequence, are summarized as follows:

First, before applying data augmentation, accuracy increased with each epoch for both the VGG16 and VGG19 models. In the experiments with the VGG16 model, the highest accuracy at epoch 10 was 0.8996 for the training data and 0.6230 for the validation data. In the VGG19 model, the highest accuracy at epoch 10 was 0.7941 for the training data and 0.6217 for the validation data.

Second, after applying data augmentation, accuracy continued to increase with each epoch for both the VGG16 and VGG19 models. For the VGG16 model, the highest accuracy at epoch 10 was 0.9373 for the training data and 0.6582 for the validation data. For the VGG19 model, the highest accuracy at epoch 10 was 0.8475 for the training data and 0.6124 for the validation data.

(Table 6) Summary of Research Results Before and After Data Augmentation

	Before Data Augmentation	After Data Augmentation
VGG16 - Accuracy	0.8996	0.9373
VGG16 – val_Accuracy	0.6230	0.6582
VGG19 - Accuracy	0.7941	0.8475
VGG19 – val_Accuracy	0.6217	0.6124

The results show that VGG16's accuracy improved after applying data augmentation compared to before. For VGG19, the training data accuracy also increased post-augmentation, while the validation data accuracy showed a slight decrease, though this difference was minimal. Overall, most cases demonstrated an improvement in accuracy.

Consequently, it was concluded that applying data augmentation to image learning has a positive impact on model performance. Additionally, the generally lower accuracy of VGG19 compared to VGG16 appears to be related to the specific characteristics of the image data used in this study.

5.2 Limitations of Study

This study examines the impact of data augmentation on image learning, specifically using marine submerged waste images. The results may reflect the quality and characteristics of the image data applied in this context.

The study is also limited by the number of images used; thus, the results may vary depending on the amount of image data available for training. Additionally, duplicated instances of marine waste during the image classification process could have influenced the outcomes.

Since the CNN VGG model was applied to the image learning model, there are limitations in that other models, such as LeNet, AlexNet, ZFNet, GoogleNet, ResNet, or DenseNet, might yield different results.

Among the various data augmentation techniques, only specific methods were used, presenting a limitation in not applying a broader range of augmentation techniques.

Finally, image learning is heavily influenced by computing specifications, indicating potential limitations based on the research environment.

5.3 Further research

This study investigated the impact of data augmentation on image learning and concluded that

it has a positive influence. Based on these results, applying data augmentation can effectively expand limited datasets when utilizing image data in AI applications.

During this study, several areas for further in-depth research were identified, leading to the following recommendations:

First, additional research is needed to examine the impact of data augmentation on image learning using different types of image data than those employed in this study.

Second, further studies should explore the effects of applying other models, beyond those used in this research, on image learning.

Third, it would be beneficial to investigate whether data augmentation has an impact on other types of data, such as text and audio.

Fourth, conducting experiments in different research environments from those in this study is necessary to assess the extent to which results are influenced by the environment.

Finally, in this study, VGG19 was initially expected to achieve higher accuracy than VGG16. However, the actual results showed VGG19 had lower accuracy than VGG16. Therefore, further research is needed to determine which model is most suitable for specific contexts.

Reference

Hye-Uk Choung. (2021), Criminal Liability of Artificial Intelligence, *Chung-Ang Law Association,* 23. 4. 53-80.

So-yeon Park, Chang-yeop Lee, Chan-hyung Ahn. (2020), AI: Present and Future – Part 1. How is Artificial Intelligence Technology Classified?, 2econsulting, https://www.2e.co.kr/news/articleView.html?idxno=300957.

Jamin Song, Sae Bom Lee, Arum Park. (2020), A Study on the Industrial Application of Image Recognition Technology, *The Journal of the Korea Contents Association*, 20. 7. 86-96.

Mi Jeong Park. (2019), A Study on Artificial Intelligence and Data Ethics: Focusing on Health Data Used in Artificial Intelligence, *Korea J Med Ethics,* 22. 3. 255-273.

Hyung Il Koo. (2018), A Study on Artificial Intelligence and Data Ethics: Focusing on Health Data Used in Artificial Intelligence, *Korea J Med Ethics,* 22. 3. 255-273.

Hee-chan Cho. (2020), A layerd-wise data augmenting algorithm for small sampling data, Graduate School of Information Security Korea University.

Soo-Young Cho. (2020), Data augmentations to improve deep learning network model accuracy, Kwangwoon University.

Joo-Yeong Song. (2021), A Study on the Improvement of Sequential Recommender System through Data Augmentation, Graduate School of Convergence Science and Technology Seoul National University.

Jeong-Hwa Lim. (2023), A Study on ECG Augmentation to Improve Deep Learning Performance, The Graduate School Yonsei University.

Sung-Nam Kim. (2023), A study on Korean text data augmentation techniques through sentences summarization for document classfication and sentiment analysis, Kyunghee University.

Chan-Il Park. (2021), Data augmentation for sign language translation AI learning, The Graduate School Kwangwoon University.

Jin-Sang Lee. (2023), Research on Korean Sentiment Classification Model Using Data Augmentation, Korea University.

Connor Shorten., Taghi M. Khoshgoftaar. (2019), “A survey on Image Data Augmentation for Deep Learning”, *Journal of Big Data.*

Eun-Ju Lee. (2017), CNN과 RNN의 기초 및 응용 연구, *Broadcasting and media magazine,* 22. 1. 81-89.

In-Kyeong Hwang. (2017), Study on Hangul font characteristics using CNN, The Graduate School Seoul National University.

Ian Goodfellow., Yoshua Bengio., Aaron Courville. (2016), "Deep Learning", MIT Press, https://www.deeplearningbook.org/.

Su-Bin Park. (2019), Implementation of Multiple Object Detection and Recognition System using H-SSD based on VGG Network, Graduate School Don-A University.

Hyeon-Ho Lee. (2020), CNN Generalization Error Evaluation Method, The Graduate School Pusan National University.

Kang-Hyun Park. (2021), Displavement prediction of structures using Signal Image data Based CNN(SIBCNN), The Graduate School Yonsei University.

Gyeong-Won Jang. (2023), Exploring Optimization Techniques for CNN in Binary Image Classification, The Graduate Seoul National University of Science and Technology.

Kwang-Bok Hwang. (2021), A Study on the Design of Steering Controller for Autonomous Vehicle with Variable Speed Using CNN, Graduate School Gyeongnam National University of Science and Technology.

Ae-Ri Min. (2020), A Study on Transfer Learning for Image Classification of Convolutional Neural Network – Based on VGG16 Deep Convolutional Neural Network, The Graduate School Hansei University.

Jong-Bum Kim, Sung-Kwun Oh. (2015), Design of Convolutional RBFNNs Pattern Classifier for Two dimensional Face Recognition, *The Korean Institute of Electrical Engineers*, 1355-1356.

Hyun-Su Lee. (2018), A Structure of Convolutional Neural Networks for Image Contents Search, Graduate School of Chung-Ang University.

Han-Nah Kim. (2020), Sentiment Analysis and Classification using Convolutional Neural Networks, Graduate School of Soonchunhyang University.

Karen Simonyan., Andrew Zisserman. (2014), "Very Deep Convolutional Networs for Large-Scale Image Recognition", *arXiv:1409.1556*.

Alex Krizhevsky., Ilya Sutskever., Geoffrey E. Hinton. (2012), "ImageNet Classification with Deep Convolutional Neural Networks", *NIPS,* 1106-1114.

Liang Han Sheng. (2021), "VGGNet – Convolutional Network for clssification and Detection", https://medium.com/analytics-vidhya/vggnet-convolutional-network-for-classification-and-detection-3543aaf61699.

Jun-Yeol Ryu, Tae-Wan Kim, Ja-Hoon Jeong. (2022), An AI Model for classifying The State of Armored and Mechanized Weapon Systems Based on VGG16, *Journal of the Korea Academia-Industrial cooperation Society*, 23. 12. 741-748.

Hye-won Kim, Xuefeng Zhang, Yong-Soo Kim, Il-Hong Jung. (2022), Comparison of the Performance of CNN Models for Retinal Diseases Diagnosis, *Journal of Korean Institute of Intelligent Systems*, 32. 1. 51-60.

Jeong-Hwan Lee. (2021), Detection of Number and Character Area of License Plate Using Deep Learning and Semantic Image Segmentation, *Journal of the Korea Convergence Society*, 12. 1. 29-35.

Ji-Hoon Jo. (2013), An Implementation of a smart camera system using CUDA Based parallel CAM shift Algorithm, Hannam University.

◐ ABOUT THE AUTHOR ◑

Tae-hwan Choi

A Seoul School of Integrated Sciences & Technologies(aSSIST) M.S
Areas of interest : AI, Data Analysis, Cloud, Computer Vision, Natural Language Processing, Data Mining etc.
E-mail : suhosin@kakao.com