informatics

MDPI

Article

Performance Comparison of CNN Models Using Gradient Flow Analysis

Seol-Hyun Noh

Department of Statistical Data Science, ICT Convergence Engineering, Anyang University, Anyang 14028, Korea; shnoh@anyang.ac.kr

Abstract: Convolutional neural networks (CNNs) are widely used among the various deep learning techniques available because of their superior performance in the fields of computer vision and natural language processing. CNNs can effectively extract the locality and correlation of input data using structures in which convolutional layers are successively applied to the input data. In general, the performance of neural networks has improved as the depth of CNNs has increased. However, an increase in the depth of a CNN is not always accompanied by an increase in the accuracy of the neural network. This is because the gradient vanishing problem may arise, causing the weights of the weighted layers to fail to converge. Accordingly, the gradient flows of the VGGNet, ResNet, SENet, and DenseNet models were analyzed and compared in this study, and the reasons for the differences in the error rate performances of the models were derived.

Keywords: CNN; gradient vanishing problem; gradient flow; performance comparison; error rate

check for updates

Citation: Noh, S.-H. Performance Comparison of CNN Models Using Gradient Flow Analysis. *Informatics* **2021**, *8*, 53. https://doi.org/10.3390/informatics8030053

Academic Editor: Antony Bryant

Received: 25 June 2021
Accepted: 11 August 2021
Published: 13 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Convolutional neural networks (CNNs) are one of the most widely used deep learning techniques because of their superior performance in the fields of computer vision and natural language processing [1]. Moreover, CNNs have been successfully applied to different machine-learning-related tasks, such as object detection, recognition, classification, regression, and segmentation [2–4]. Recently, CNNs have been actively applied in the medical field, while CNN models that can be used as automated diagnostic tools to aid experts in the detection of hypertension, coronary artery disease, myocardial infarction, and congestive heart failure have been proposed [5–8]. However, in order to use deep CNNs in mobile and embedded systems, it is necessary to overcome challenges relating to the necessity of several computations and high memory usage [9,10]. Several studies have been conducted on the gating mechanism to overcome these limitations [11].

In a fully connected neural network (FCNN), spatial information is lost, and the features of adjacent images cannot be recognized in the process of learning and classifying by flattening 3D image data into a one-dimensional array. In contrast, a CNN has a translation invariance feature that effectively recognizes the features of adjacent images while maintaining the spatial information of the image. As a CNN uses a filter as a shared parameter for data from each image, this is an effective deep learning algorithm for learning and classifying images because extremely few learning parameters are required compared to an FCNN. Computer vision performance has significantly improved in recent years with the re-emergence of CNNs and deep learning techniques [12–15]. The performance of neural networks such as CNNs has improved as the depth of CNNs has increased by effectively extracting the locality and the correlation of input data using structures in which convolutional layers are successively applied to input data. From the beginning of the ImageNet Large Scale Visual Recognition Competition (ILSVRC), the depth of CNNs has increased to improve the accuracy of object recognition. The AlexNet model [16], comprising 8 weighted layers, lowered the top-5 error rate to 16% and won the ILSVRC in 2012; the VGGNet model [17], comprising 16 weighted layers, lowered the top-5 error rate

to 7.3% in 2014; and the ResNet model [18], comprising 152 weighted layers and shortcut identity connections, lowered the top-5 error rate to 3.6% and won the ILSVRC in 2015. The DenseNet model is a DCNN that was proposed by Huang et al. in 2017; the top-5 error rate of DenseNet-161, comprising 161 weighted layers, was found to be 5.30% [19].

However, the accuracy of neural networks does not necessarily increase with an increase in the depth of CNNs. This is because the weights of the weighted layers may not converge owing to the gradient vanishing problem [20,21]. The weights of the weighted layers are updated in the direction in which the loss function decreases with the backpropagation algorithm. The gradient at each node is calculated based on the chain rule in a backward manner as follows:

$$(local\ gradient) \times (gradient\ flowing\ from\ ahead)$$

Therefore, the gradient calculated at the data input node may vanish to an extremely small value close to 0 or diverge infinitely if the depth of the neural network is sufficiently deep. The ResNet and DenseNet models, which introduced skip connections, were developed to overcome this problem. In addition, SENets [22] were proposed to improve the representational power of CNN models. In the ResNet model, gradients do not vanish but are effectively transmitted when the weights are updated because there is a shortcut identity connection for each of the two weighted layers as in Figure 2, even in the case of deep neural networks. DenseNet can be considered a model that maximizes the idea of skip connections. In the DenseNet model, each layer is connected to all the other layers in a feed-forward manner, as in Figure 4. Therefore, each layer receives full output data from the previous layers. The DenseNet model effectively overcomes the gradient vanishing problem even when the depth of a neural network is deeper because of this feature. SENets model the interdependencies between the channels of convolutional features by introducing a squeeze-and-excitation (SE) block, which improves the representation power of CNNs by being plugged into various CNN models. In a performance comparison study of CNN models [18,22], the superiority was demonstrated in comparison to conventional models by calculating the accuracy of the trained models on several datasets (CIFAR-10 and ImageNet) and by comparing the number of parameters and the amount of computation (FLOPs) required for forwarding a single input image. He et al. (2016) measured the top-1 error rate performance of ResNets and plain networks composed of the same rules as the VGGNet model on the ImageNet validation set. He et al. (2016) have verified that ResNet reduces the top-1 error rate compared to a plain network on an extremely deep system, as shown in Table 1. He et al. (2016) also measured the training and testing error rates of plain networks and ResNets on the CIFAR-10 dataset and verified that ResNets have accuracy gains compared to plain networks when the depth increased, as shown in Figure 5. Hu et al. (2017) measured the single-crop top-1 and top-5 error rate performance of ResNets and SE-ResNets on the ImageNet validation set. Hu et al. (2017) have verified that SE blocks consistently improve the error rate performance across different depths with an extremely small increase in computational complexity, as shown in Table 2 and Figure 6.

Table 1. Top-1 error (%, 10-crop testing) of plain networks and ResNets in ImageNet validation (source: [18]).

Number of Weighted Layers	Plain Networks	ResNets
18 layers	27.94%	27.88%
34 layers	28.54%	25.03%

Table 2. Single-crop error rates (%) on the ImageNet validation set and complexity comparison of ResNets and SE-ResNets (source: [22]). GFLOPs refers to the amount of computation required for forwarding a single 224 × 224 pixel input image.

	ResNet (Reimplementation)			SE-ResNet		
	Top-1 err. (%)	**Top-5 err. (%)**	**GFLOPs**	**top-1 err. (%)**	**top-5 err. (%)**	**GFLOPs**
ResNet-50	24.80	7.48	3.86	23.29	6.62	3.87
ResNet-101	23.17	6.52	7.58	22.38	6.07	7.60
ResNet-152	22.42	6.34	11.30	21.57	5.73	11.32

The aim of this study was to present an analysis tool that can be used for the performance analysis of CNN models in the future by deriving the theoretical basis for the performance difference of the top-1 error rates of the four models using the analysis and comparison of gradient flows based on a single bottleneck layer for the VGGNet, ResNet, SE-ResNet, and DenseNet models. The proposed gradient flows analysis method based on a single bottleneck layer can also be applied to design a CNN model to enhance the learning ability of the model.

2. Materials and Research Method

In this study, we thought that the error rate performance of CNN models with various architectures could be predicted by analyzing how efficiently the gradient vanishing problem can be overcome. Therefore, the following research questions were established.

- Research question 1: How can we mathematically represent gradient flows based on a single bottleneck layer for CNN models with iterative bottleneck blocks?
- Research question 2: Do the analysis results for research question 1 coincide with the experimental results of error rate performance for various CNN models?

The analysis results for research question 1 are described in Section 3, while the analysis results for research question 2 are described in Section 4. By analyzing and comparing the gradient flow of the VGGNet, ResNet, SE-ResNet, and DenseNet models based on a single bottleneck layer, this study aimed to draw the theoretical basis for the differences in the error rate performance of CNN models.

Let $F(x)$ denote the output data that passed through the weighted layer in a bottleneck block of a CNN model. Let $H(x)$ denote the output data of a bottleneck block of a CNN model with skip connection. For the analysis, the gradient of loss function $L(x)$ with respect to the input data x of the bottleneck layer was expressed according to the chain rule as the product of the rate of change of the loss function with respect to $F(x)$(or $H(x)$) and the rate of change of $F(x)$(or $H(x)$) with respect to x. When $\frac{\partial F}{\partial x}$ (or $\frac{\partial H}{\partial x}$) converged to 0, the lower limit of $\frac{\partial L}{\partial x}$ in the VGGNet, ResNet, SE-ResNet, and DenseNet models was investigated to study the gradient vanishing problem, which occurs when $\frac{\partial F}{\partial x}$ (or $\frac{\partial H}{\partial x}$) converges to 0 at an increased number of weighted layers. Moreover, the findings from this investigation were compared with the error rate performance analysis results [18,19,22] to check for consistency. Through this method, this study aimed to provide a theoretical basis for the analysis results of the error rate performances [18,19,22].

If the number of filters in the convolutional layer is n, the width of the filter is k_W, and the height is k_H, the architecture of the VGGNet, ResNet, SE-ResNet, and DenseNet models when the filters of the convolutional layer are simply expressed as $n@k_W \times k_H$ is summarized in Table A1 in Appendix A. The symbol f_C used in the architecture of the SE-ResNet model in Table A1 indicates the output dimensions of the two fully connected layers of the SE block.

3. Research Results

In the VGGNet model, the size of all the filters of the convolution layer is 3×3, and the activation function ReLU is applied to the data that have passed through the convolution layer. A certain level of complexity is maintained for each layer by doubling the number of filters in the next convolutional layer when the size of the feature map is halved by a 2×2 max pooling layer. Three fully connected layers (FC layers) are located in the last stage of the network and serve as classifiers. The ResNet model is a CNN that overcomes the problem of degradation by introducing residual learning. If $H(x)$ denotes the underlying mapping of the stacked layer and x denotes the input data, the activation function ReLU is applied after training, $F(x) = H(x) - x$, and adding x to the output data of the stacked layer. He et al. (2016) used a bottleneck building block composed of $1 \times 1 - 3 \times 3 - 1 \times 1$ *conv* to present the structure of an iterative ResNet, as shown in Table A1. SE-ResNet is a model that improves feature discriminability by scaling the output data of the residual block by plugging an SE block into the bottleneck block of ResNet and extracting channel-wise multiplication factor s from the output data of the residual block. The DenseNet model has a structure in which the dense block of $BN - ReLU - 1 \times 1$ *conv* $- BN - ReLU - 3 \times 3$ *conv* is repeated, as shown in Table A1. In the DenseNet model, $x_l = H_l[x_0, x_1, \cdots, x_{l-1}]$, where x_l denotes the output data of the l-th dense block, $H_l[\cdot]$ denotes the nonlinear transformation of the l-th dense block, and $[x_0, x_1, \cdots, x_{l-1}]$ denotes the concatenation of feature maps created in previous layers.

A single bottleneck block in the VGGNet model is illustrated in Figure 1. The gradient of $L(x)$ is expressed as shown in Equation (1), where x denotes the input data of the bottleneck block of the VGGNet model, $F(x)$ denotes the output data that passed through the weighted layer, and $L(x)$ denotes the loss function.

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial F} \times \frac{\partial F}{\partial x} \tag{1}$$

Therefore, a gradient vanishing problem occurred when $\partial F / \partial x$ vanished to 0.

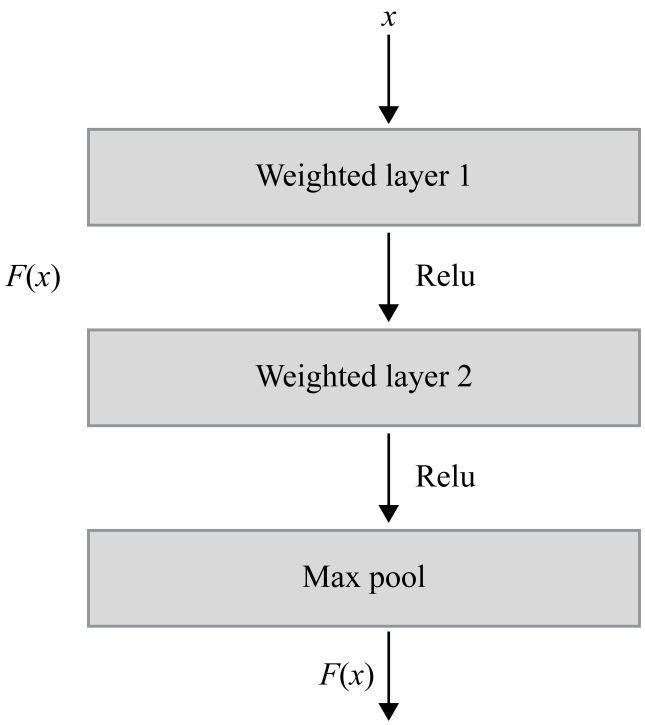

Figure 1. Bottleneck block in the VGGNet model.

A single bottleneck block in the ResNet model is illustrated in Figure 2. The output data of the bottleneck block are represented as $F(x) + x$, where x denotes the input data of the bottleneck block of the ResNet model and $F(x)$ denotes the output data that passed through the weighted layer. If $H(x) = F(x) + x$ and the loss function is $L(x)$,

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial H} \times \frac{\partial H}{\partial x} = \frac{\partial L}{\partial H} \times \left(\frac{\partial F}{\partial x} + 1 \right) = \frac{\partial L}{\partial H} \times \frac{\partial F}{\partial x} + \frac{\partial L}{\partial H} \tag{2}$$

Therefore, the gradient vanishing problem could be overcome more effectively than by using the VGGNet model because $\partial L/\partial H$ components remained even if $\partial F/\partial x$ vanished to 0.

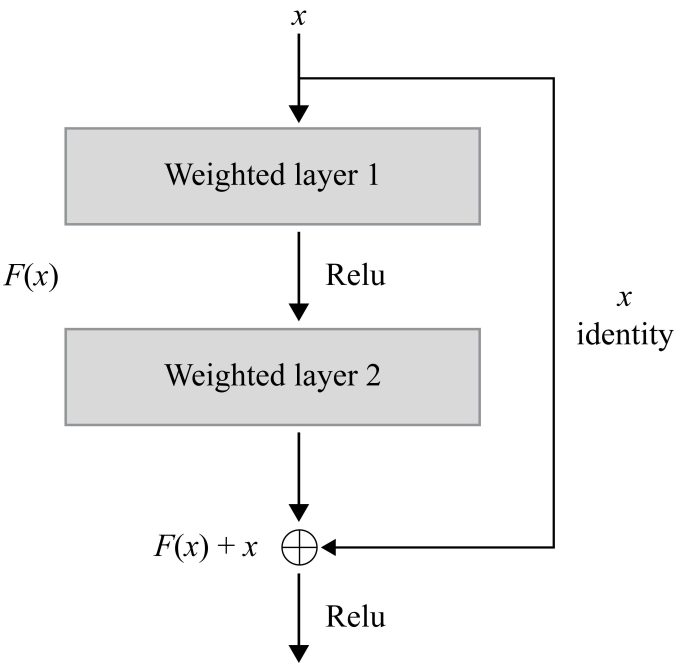

Figure 2. Bottleneck block in the ResNet model.

A single bottleneck block in the SE-ResNet model is demonstrated in Figure 3. Let x denote the input data of the bottleneck block of the SE-ResNet model and $F(x)$ denote the output data that have passed through the residual block. Then, $F(x) = (u_1, u_2, \cdots, u_C)$, assuming that $F(x)$ includes C feature maps. If the result of the squeeze step by applying global average pooling to $F(x)$ is $z = F_{sq}(F(x))$ and the result of the excitation step by applying $FC - ReLU - FC - Sigmoid$ to z is $s = (s_1, s_2, \cdots, s_C)$, s transforms $F(x)$ into $\widetilde{F}(x) = (s_1 u_1, s_2 u_2, \cdots, s_C u_C)$ as a channel-wise multiplication factor that acts as a scale factor to improve the feature discriminability. The output data of the SE-ResNet model bottleneck block become $H(x) = \widetilde{F}(x) + x$ as x is added to $\widetilde{F}(x)$ because of the shortcut identity connection. Therefore, if the loss function is $L(x)$, the following equation is established:

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial H} \times \frac{\partial H}{\partial x} = \frac{\partial L}{\partial H} \times \left(\frac{\partial \widetilde{F}}{\partial x} + 1 \right) .$$

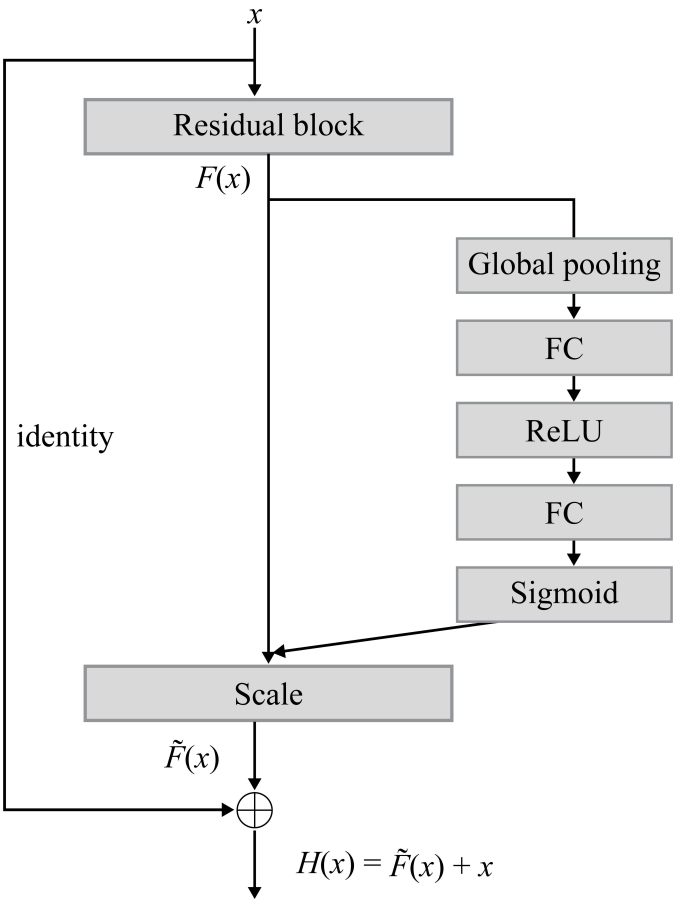

Figure 3. Bottleneck block in the SE-ResNet model.

As $\widetilde{F}(x)$ can be expressed as $\widetilde{F}(x) = s{\cdot}F$, which is the Hadamard product of s and $F(x)$, the following equation holds:

$$\frac{\partial \widetilde{F}}{\partial x} = \frac{\partial s}{\partial x}{\cdot}F + s{\cdot}\frac{\partial F}{\partial x}\,.$$

Therefore,

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial H} \times \frac{\partial H}{\partial x} = \frac{\partial L}{\partial H} \times \left(\frac{\partial \widetilde{F}}{\partial x} + 1\right) = \frac{\partial L}{\partial H} \times \left(\frac{\partial s}{\partial x}{\cdot}F + s{\cdot}\frac{\partial F}{\partial x} + 1\right). \tag{3}$$

The possibility of overcoming the gradient vanishing problem more effectively than can be achieved using the ResNet model increased because $\frac{\partial L}{\partial H} \times \left(\frac{\partial s}{\partial x}{\cdot}F + 1\right)$ components remained even if $\partial F/\partial x$ vanished to 0 in Equation (3).

A single dense block in the DenseNet model is demonstrated in Figure 4. If x_0 denotes the initial input data of DenseNet, x_{i-1} denotes the input data of the i-th dense block, and x_i denotes the output data, the following equations are established.

$$\begin{aligned} &x_1 = F_1(x_0),\\ &x_2 = F_2(x_1)||x_1||x_0,\\ &\cdots,\\ &x_{l-1} = F_{l-1}(x_{l-2})||x_{l-3}||\cdots||x_0,\\ &x_l = F_l(x_{l-1})||x_{l-1}||\cdots||x_0 \end{aligned} \tag{4}$$

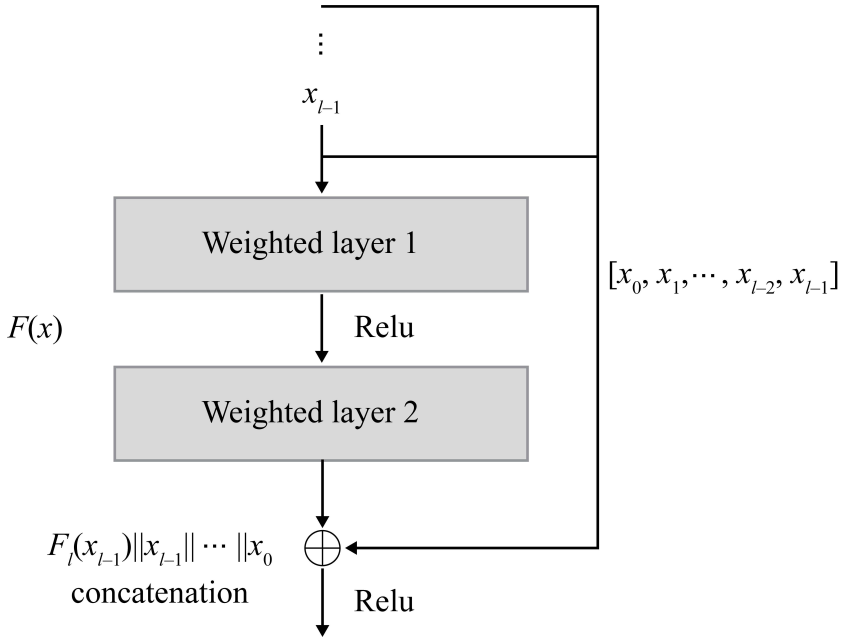

Figure 4. Dense block in the DenseNet model.

If $H([x_0, x_1, \cdots, x_{l-1}]) = F_l(x_{l-1})||x_{l-1}|| \cdots ||x_0$ and $x = (x_0, x_1, \cdots, x_{l-1})$ in Equation (4):

$$\begin{aligned} \frac{\partial L}{\partial x} &= \frac{\partial L}{\partial H} \times \frac{\partial H}{\partial x} \\ &= \frac{\partial L}{\partial H} \times \left(\frac{\partial H}{\partial x_0}, \frac{\partial H}{\partial x_1}, \cdots, \frac{\partial H}{\partial x_{l-1}} \right) \\ &= \frac{\partial L}{\partial H} \times \left(\frac{\partial F_l(x_{l-1})}{\partial x_0} \left|\left| \frac{\partial x_{l-1}}{\partial x_0} \right|\right| \cdots \left|\left| \frac{\partial x_0}{\partial x_0}, \cdots, \frac{\partial F_l(x_{l-1})}{\partial x_{l-1}} \right|\right| \frac{\partial x_{l-1}}{\partial x_{l-1}} \right) \\ &= \left(\frac{\partial L}{\partial H} \times \left(\frac{\partial F_l(x_{l-1})}{\partial x_0} \left|\left| \frac{\partial x_{l-1}}{\partial x_0} \right|\right| \cdots ||1 \right), \cdots, \frac{\partial L}{\partial H} \times \left(\frac{\partial F_l(x_{l-1})}{\partial x_{l-1}} \left|\left| 1 \right|\right| \right) \right) \end{aligned} \tag{5}$$

Therefore,

$$\begin{aligned} &\left|\left| \frac{\partial L}{\partial x} \right|\right| \\ &= \left| \frac{\partial L}{\partial x} \right| \times \sqrt{\left\{ \left(\frac{\partial F_l(x_{l-1})}{\partial x_0} \right)^2 + \left(\frac{\partial x_{l-1}}{\partial x_0} \right)^2 + \cdots + \left(\frac{\partial x_1}{\partial x_0} \right)^2 + 1^2 \right\} + \cdots + \left\{ \left(\frac{\partial F_l(x_{l-1})}{\partial x_{l-1}} \right)^2 + 1^2 \right\}} \\ &\geq \left| \frac{\partial L}{\partial H} \right| \times \sqrt{\left(\frac{\partial F_l(x_{l-1})}{\partial x_0} \right)^2 + \cdots + \left(\frac{\partial F_l(x_{l-1})}{\partial x_{l-1}} \right)^2 + l} \end{aligned} \tag{6}$$

Therefore, the gradient vanishing problem could be overcome more effectively than it can using the ResNet model because $\partial L/\partial H \times \sqrt{l}$ remained even if all $\partial F_l(x_{l-1})/\partial x_0$, $\partial F_l(x_{l-1})/\partial x_1$, $\cdots$, $\partial F_l(x_{l-1})/\partial x_{l-1}$ vanished to 0.

4. Discussion

The results of measuring the top-1 error rate performances for the CIFAR-10 dataset of ResNets and plain networks using the same rules as the VGGNet model while varying the number of weighted layers are presented in Figure 5. The results of measuring the top-1 error rate performances for the ImageNet dataset of plain networks and ResNets by varying the number of weighted layers are presented in Table 1.

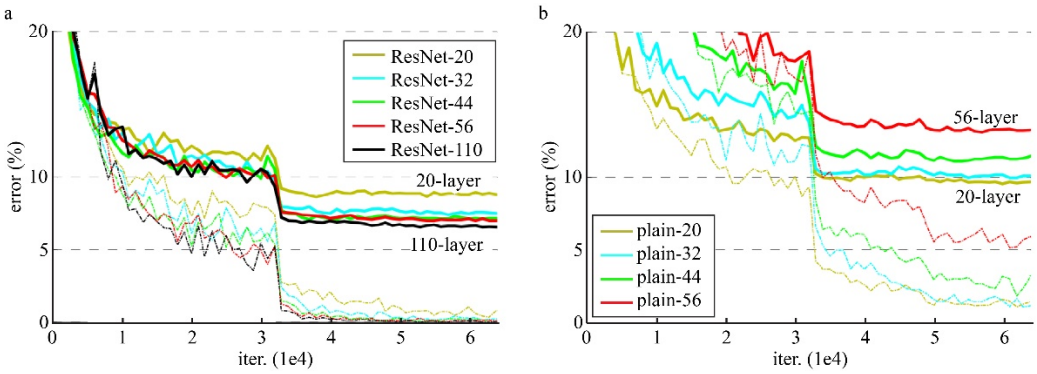

Figure 5. Training on CIFAR-10. Dashed lines denote training error and bold lines denote testing error. (**a**): ResNets. (**b**): plain networks (source: [18]).

The performance of plain networks decreased as the number of weighted layers increased, whereas there was a performance gain in which the accuracy of ResNets increased as the number of weighted layers increased by overcoming the gradient vanishing problem, as presented in Figure 5 and Table 1.

The performance in terms of the top-1 error rate, top-5 error rate, and amount of computation (GFLOPs) needed in the ResNet and SE-ResNet models for the ImageNet dataset when the number of weighted layers is 50, 101, and 152 is summarized in Table 2 [22]. A graph showing the change in the top-1 error rate based on the epochs of ResNet-50 and SE-ResNet-50 is depicted in Figure 6, which shows that the validation of the top-1 error rate of SE-ResNet-50 is lower than that of ResNet-50 [22]. Although the amount of computation needed for the SE-ResNet model with the SE block plugged into the ResNet model slightly increased, the top-1 and top-5 error rates were lower than those of the ResNet model, as presented in Table 2 and Figure 6. This was consistent with the result of the gradient flow analysis, in which it can be seen that the SE-ResNet model was more likely to effectively overcome the gradient vanishing problem than the ResNet model.

The differences in error rate performance in Tables 1 and 2 can be said to be statistically significant if we consider the results showing that the standard deviation of the accuracies of CNN for 15 datasets is less than 1% [23] and the results showing that the standard deviation of the layer response, which is the output of the 3×3 layer of ResNet, is less than 1 [18]. The experimental results shown in Table 1 were obtained using the following parameters: the learning rate starts from 0.1 and is divided by 10 when the error plateaus, and the models are trained for up to 60×10^4 iterations. The experimental results shown in Table 2 were obtained using the following parameters: the learning rate is set to 0.6 and decreased by a factor of 10 every 30 epochs, and the models are trained for 100 epochs from scratch.

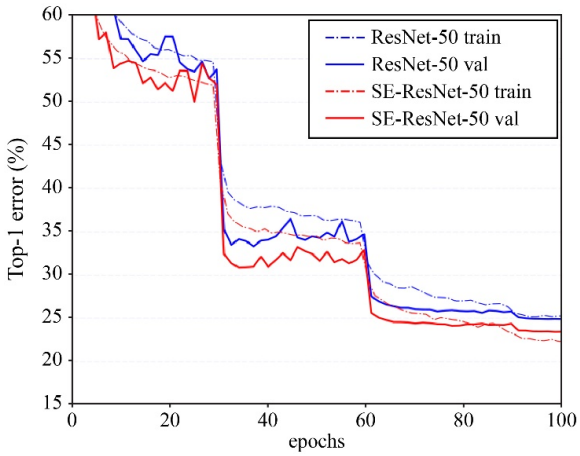

Figure 6. Training curves of ResNet-50 and SE-ResNet-50 on ImageNet (source: [22]).

Figure 7 depicts a graph showing the number of parameters of ResNet and the performance of the validation for the top-1 error rate for ImageNet when the number of weighted layers is 34, 50, 101, and 152. It also shows the number of parameters of DenseNet and the performance of the validation for the top-1 error rate for ImageNet when the number of weighted layers is 121, 169, 201, and 264. Although DenseNet-201 has fewer parameters and more weighted layers than ResNet-50, the validation top-1 error rate of DenseNet-201 for ImageNet was lower than that of ResNet-50 as it overcomes the gradient vanishing problem more effectively.

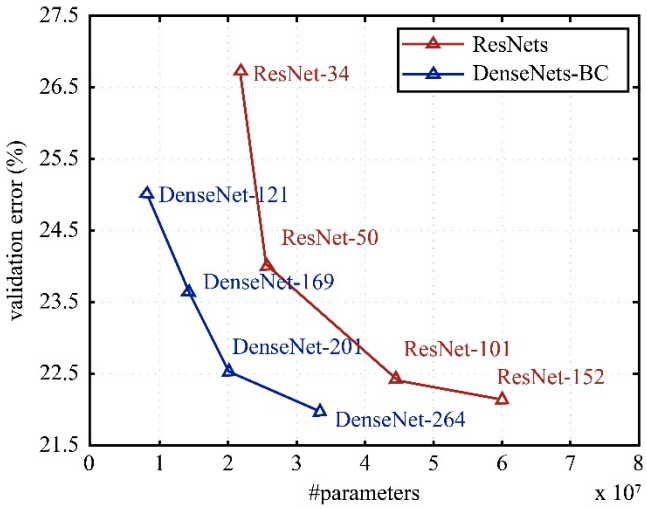

Figure 7. Comparison of the DenseNet and ResNet top-1 error rates (single-crop testing) on the ImageNet validation dataset as a function of the learned parameters (source: [19]).

The performance analysis results consistent with the gradient flow analysis results derived in this study are shown in Figures 5–7 and Tables 1 and 2.

Although the error rate of VGGNet might increase because of the gradient vanishing problem if the number of weighted layers is increased to improve the performance by effectively extracting the input data from CNNs, ResNet can overcome the disadvantages of VGGNet using a shortcut identity connection. SE-ResNet increases the possibility of overcoming the gradient vanishing problem more effectively than ResNet through its improved feature discriminability from plugging the SE block into ResNet. DenseNet overcomes the gradient vanishing problem more effectively than ResNet by maximizing the idea of skip connections in ResNet to connect each layer to all other layers in a feed-forward manner. However, the computational amount (FLOPs) also increases compared to ResNet when forwarding a single input image owing to the complexity of the model [19].

5. Conclusions

This study is meaningful because the basis for the difference in the performance of the four models was derived by analyzing and comparing the gradient flow based on a single bottleneck block for the VGGNet, ResNet, SE-ResNet, and DenseNet models, which are representative models of CNNs.

A gradient vanishing problem occurred when the gradient of $L(x)$ was calculated, as shown in Equation (1), where x denotes the input data of the bottleneck block of VGGNet, $F(x)$ denotes the output data that passed through the weighted layer, and $L(x)$ denotes the loss function. In the case of the ResNet model, the output data of the bottleneck block were $F(x) + x$. If $H(x) = F(x) + x$, the gradient vanishing problem could be overcome more effectively than it could in the VGGNet model because $\partial L/\partial H$ components remained even if $\partial F/\partial x$ vanished to 0 when the gradient of $L(x)$ was calculated, as shown in Equation (2). If the input data of the bottleneck block of SE-ResNet were expressed as x, the output data of the residual block were expressed as $F(x)$, the channel-wise multiplication factor obtained

by $F(x)$ through the SE block was expressed as s, and the loss function was expressed as $L(x)$, then the gradient of $L(x)$ was calculated as shown in Equation (3). Therefore, the possibility of overcoming the gradient vanishing problem more effectively than it could be by the ResNet model increased because $\frac{\partial L}{\partial H} \times \left(\frac{\partial s}{\partial x} \cdot F + 1\right)$ components remained even if $\partial F/\partial x$ vanished to 0. In the case of the DenseNet model, if the initial input data were expressed as x_0, the input data of the l-th dense block were expressed as x_{l-1}, the output data were expressed as x_l, and $H([x_0,\ x_1,\ \cdots,\ x_{l-1}\ ]) = F_l(x_{l-1})||x_{l-1}||\cdots||x_0$, then the gradient of $L(x)$ for $x = (x_0,\ x_1,\ \cdots,\ x_{l-1})$ was calculated as shown in Equation (5) with a lower limit as shown in Equation (6). Therefore, the gradient vanishing problem could be overcome more effectively than it could in the ResNet model because $\partial L/\partial H \times \sqrt{l}$ remained even if all $\partial F_l(x_{l-1})/\partial x_0,\ \partial F_l(x_{l-1})/\partial x_1,\ \cdots,\ \partial F_l(x_{l-1})/\partial x_{l-1}$ vanished to 0.

The performance of a plain network with the same rules as the VGGNet model decreased as the number of weighted layers increased, whereas there was a performance gain where the accuracy of the ResNet model increased as the number of weighted layers increased due to overcoming the gradient vanishing problem, as shown in the performance analysis results in Figure 5. The results in Table 1 show that ResNet-34 reduces the ImageNet validation top-1 error by 3.5% compared to a plain network with the same number of layers and parameters. Although the amount of computation slightly increased in the SE-ResNet model compared to in the ResNet model, the validation top-1 error rate for ImageNet was lower because it effectively overcame the gradient vanishing problem, as presented in Table 2 and Figure 6. The results shown in Table 2 demonstrate that SE-ResNet-50 reduces the ImageNet validation top-1 error by 1.51% compared to ResNet-50. The DenseNet model had a lower validation top-1 error rate for ImageNet than ResNet because it effectively overcame the gradient vanishing problem even when the number of weighted layers increased, as shown in the performance analysis results in Figure 7.

In the future, more related studies will be conducted quantitatively and qualitatively if the causes for these differences in performances are investigated by analyzing the gradient flow for other CNN models and the causes of the differences in the performances of the CNN models, other than the gradient flow, are identified.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Architectures of the VGG-19, ResNet-50, SE-ResNet-50, and DenseNet-121 models.

Baseline Model	Architecture	Number of Param.
VGG-19	conv : 64@3 × 3 conv : 64@3 × 3 2 × 2 max pool conv : 128@3 × 3 conv : 128@3 × 3 2 × 2 max pool conv : 256@3 × 3 conv : 256@3 × 3 conv : 256@3 × 3 conv : 256@3 × 3 2 × 2 max pool conv : 512@3 × 3 conv : 512@3 × 3	38.9 M

Table A1. *Cont.*

Baseline Model	Architecture		Number of Param.
	conv : 512@3 × 3		
	conv : 512@3 × 3		
	2 × 2 max pool		
	conv : 512@3 × 3		
	conv : 512@3 × 3		
	conv : 512@3 × 3		
	conv : 512@3 × 3		
	2 × 2 max pool		
	$FC - 4096$		
	$FC - 4096$		
	$FC - 10$		
	(In case of CIFAR-10)		
	softmax		
ResNet-50	conv : 64@7 × 7		23.6 M
	3 × 3 max pool (stride = 2)		
	Bottleneck block(1)		
	conv : 64@1 × 1	×3	
	conv : 64@3 × 3		
	conv : 256@1 × 1		
	Bottleneck block(2)		
	conv : 128@1 × 1	×4	
	conv : 128@3 × 3		
	conv : 512@1 × 1		
	Bottleneck block(3)		
	conv : 256@1 × 1	×6	
	conv : 256@3 × 3		
	conv : 1024@1 × 1		
	Bottleneck block(4)		
	conv : 512@1 × 1	×3	
	conv : 512@3 × 3		
	conv : 2048@1 × 1		
	average pool		
	FC-10		
	(In case of CIFAR-10)		
	softmax		
SE-ResNet-50	conv : 64@7 × 7		21.4 M
	3 × 3 max pool (stride = 2)		
	Bottleneck block(1)		
	conv : 64@1 × 1		
	conv : 64@3 × 3	×3	
	conv : 256@1 × 1		
	f_c : [16, 256]		
	Bottleneck block(2)		
	conv : 128@1 × 1		
	conv : 128@3 × 3	×4	
	conv : 512@1 × 1		
	f_c : [32, 512]		
	Bottleneck block(3)		
	conv : 256@1 × 1		
	conv : 256@3 × 3	×6	
	conv : 1024@1 × 1		
	$f_{c:}$[64, 1024]		
	Bottleneck block(4)		
	conv : 512@1 × 1		
	conv : 512@3 × 3	×3	
	conv : 2048@1 × 1		
	$f_{c:}$[128, 2048]		

Table A1. *Cont.*

Baseline Model	Architecture	Number of Param.
	average pool FC-10 (In case of CIFAR-10) Softmax	
DenseNet-121 ($k = 32$)	conv : $2k$@7×7 (stride = 2) 3×3 max pool (stride = 2) Dense block(1) $BN + ReLU + conv(4k@1 \times 1)$+Dropout $+BN + ReLU + conv(k@3 \times 3)$+Dropout $\times 6$ Transition layer(1) conv : $0.5 \times input\_ch$@1×1 $+2 \times 2$ average pool Dense block(2) $BN + ReLU + conv(4k@1 \times 1)$+Dropout $+BN + ReLU + conv(k@3 \times 3)$+Dropout $\times 12$ Transition layer(2) conv : $0.5 \times input\_ch$@1×1 $+2 \times 2$ average pool Dense block(3) $BN + ReLU + conv(4k@1 \times 1)$+Dropout $+BN + ReLU + conv(k@3 \times 3)$+Dropout $\times 24$ Transition layer(3) conv : $0.5 \times input\_ch$@1×1 $+2 \times 2$ average pool Dense block(4) $BN + ReLU + conv(4k@1 \times 1)$+Dropout $+BN + ReLU + conv(k@3 \times 3)$+Dropout $\times 16$ 7×7 average pool FC-10 (In case of CIFAR-10) Softmax	7 M

References

1. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Chen, T. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
2. Batmaz, Z.; Yurekli, A.; Bilge, A.; Kaleli, C. A review on deep learning for recommender systems: Challenges and remedies. *Artif. Intell. Rev.* **2019**, *52*, 1–37. [CrossRef]
3. Chouhan, N.; Khan, A. Network anomaly detection using channel boosted and residual learning based deep convolutional neural network. *Appl. Soft Comput.* **2019**, *83*, 105612. [CrossRef]
4. Wahab, N.; Khan, A.; Lee, Y.S. Transfer learning based deep CNN for segmentation and detection of mitoses in breast cancer histopathological images. *Microscopy* **2019**, *68*, 216–233. [CrossRef]
5. Jahmunah, V.; Ng, E.Y.K.; San, T.R.; Rajendra Acharya, U. Automated detection of coronary artery disease, myocardial infarction and congestive heart failure using GaborCNN model with ECG signals. *Comput. Biol. Med.* **2021**, *134*, 104457. [CrossRef] [PubMed]
6. Soh, D.C.K.; Ng, E.Y.K.; Jahmunah, V.; Oh, S.L.; Tan, R.S.; Rajendra Acharya, U. Automated diagnostic tool for hypertension using convolutional neural network. *Comput. Biol. Med.* **2020**, *126*, 103999. [CrossRef]
7. Zhou, W.; Chen, F.; Zong, Y.; Zhao, D.; JIE, B.; Wang, Z.; Huang, C.; Ng, E.Y.K. Automatical detection Approach for Bioresorbable Vascular Scaffolds Using U-shape Convolutional Neural Network. *IEEE Access* **2019**, *7*, 94424–94430. [CrossRef]
8. Li, Y.; Zhang, Y.; Zhao, L.; Zhang, Y.; Liu, C.; Zhang, L.; Zhang, L.; Li, Z.; Wang, B.; Ng, E.Y.K.; et al. Combining convolutional neural network and distance distribution matrix for identification of congestive heart failure. *IEEE Access* **2018**, *6*, 39734–39744. [CrossRef]
9. Canziani, A.; Paszke, A.; Culurciello, E. An analysis of deep neural network models for practical applications. *arXiv* **2016**, arXiv:1605.07678.
10. Lu, Z.; Rallapalli, S.; Chan, K.S.; La Porta, T. Modeling the resource requirements of convolutional neural networks on mobile devices. In Proceedings of the 25th ACM international conference on Multimedia, Mountain View, CA, USA, 23–27 October 2017; pp. 1663–1671.

11. Hua, W.; Zhou, Y.; De Sa, C.; Zhang, A.; Edward Suh, G. Boosting the performance of CNN accelerators with dynamic fine-grained channel gating. In Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture, Columbums, OH, USA, 12–16 October 2019; pp. 139–150.
12. Bengio, Y. Learning deep architectures for AI. *Found. Trends Mach. Learn.* **2009**, *2*, 1–55. [CrossRef]
13. Bengio, Y.; Courville, A.; Vincent, P. Representation Learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828. [CrossRef] [PubMed]
14. LeCun, Y.; Bengio, Y.; Hinton, G.E. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
15. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
16. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
17. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
18. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
19. Huang, G.; Liu, Z.; Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
20. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166. [CrossRef] [PubMed]
21. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. *Int. Conf. Artif. Intell. Stat.* **2010**, *9*, 249–256.
22. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
23. Gong, Z.; Zhong, P.; Hu, W. Statistical Loss and Analysis for Deep Learning in Hyperspectral Image Classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 322–333. [CrossRef] [PubMed]