A Resnet Based Image Classification Method on CIFAR10

Zheng Feng

*University of Sydney.*

*Sydney, NSW, Australia*

*Email: zfen9009@uni.sydney.edu.au*

**Abstract**

*Nowadays, training a well performed deep neural network on CIFAR10 dataset is not difficult problem anymore, but it’s still a difficult problem for neural network models to converge their best training result within a limited time. Current state-of-art deep learning models usually have hundreds of convolutional layers, training epochs and a low learning rate to achieve the best performance which around 99.5% accuracy. Even though those methods can good result, researchers still need to spend long time to training those models on expensive GPU which is uneconomic. Therefore, how to balance the training time and best accuracy is always a good deep learn topic to be researched. In this paper, I describe a new deep learning model based on Resnet* *neural network to classify CIFAR10 dataset. This deep learning model has 34 layers including 8 convolutional layers, 8 Batchnorm layers,10 Relu activation layer, 4 pooling layers, 3 linear layers and 1 dropout layer. Within 10 minutes, my neural network model achieved 87.6% accuracy which is better than VGG16, Resnet101 and my assignment2 models and it has potential to achieve higher accuracy with longer training period.*

**Keywords**

CIFAR10, ResNet *neural network*.

# 1. Introduction

The development of artificial intelligence technology started after the world 2. In the late 90s, prof. Yann LeCun [1] proposed the first end to end convolution neural network model LeNet-5 to solve the ImageNet classification challenge, but the model can only get 25.8% accuracy. Due to the limitation of computation power, the theory of intelligent system can only apply on simple theoretical problems instead of applying on complex image classification problems. After several decades’ technology development, the revolution of computer architecture and more integrated silicon chips has broken the previous limitation of AI technology. However, the deeper network not always bring better result, therefore there is always a question about how to balance the performance and cost of computational resources. In this paper, I researched several famous deep learning models on the past decade and proposed a neural network model based on ResNet to solve this question. The result shows proposed model can converge within 10 minutes and achieve 87.5% which is higher than those famous models.

# 2. Related works

In 2012, AlexNet [2] first proposed a deep learning model combining the convolution layers with the response normalization layers to normalize all values in a particular range and the rectified linear unit (ReLU) as activation layers. The revolution in deep learning model architecture helped AlexNet achieve 90-95% accuracy in the ImagNet challenge and accuracy of AlexNet is far more than traditional machine learning algorithms with handcraft features. After this time point, the industry and academia start to notice the potential of deep learning model.

Moving to 2014, the research of deep neural networks has become much more popular than before. Most technology companies start to research and propose their own neural network model to compete with academia’s models. Google proposed their own model GoogleLeNet to improve neural networks’ efficiency and accuracy, because they found the performance of original AlexNet model reached its limit which means more parameters and deeper network won’t improve the accuracy. On the contrary, the redundant convolution layers used more computation resources and longer training period for the same result. Therefore, Google introduced the inception module which is the local convolutional layers with different kernels in parallel. The inception module has 12 times less parameters than AlexNet and 28 times lesser parameters than VGG network for similar accuracy. However, GoogleLeNet also brings new problems that the more complex and deeper artificial neural network architecture makes the backpropagation harder to update the weights in the early layers, because the gradient of sigmoid activation function is less than 0.25 when the chain rule applied [3]. The deeper network means that the gradient of backpropagation will eventually close to 0 value. This phenomenon is also called gradient vanishing problem.

To reduce the effect of gradient vanishing phenomenon, the researchers from Microsoft brought out their solution. Kaiming He et.al [4] first designed an elegant, but powerful new architecture called ResNet which integrated residual module. Each residual module has two connected 3x3 convolution layers and a shortcut path to send the input of the first convolutional layer to the output of the second layer and add them together. The revolution structure losing through the bottleneck of deep learning model’s development and first lead the architecture of deep learning model up to 152 layers without loose the efficiency of GoogleLeNet model.

# 3. Methodology

The modified method is inspired by the previous work down by Kaiming He et.al [4] which is ResNet architecture. If we consider H(x) represent the mathematical process of a few stacked convolutional layers, the x is the original input value of these convolutional layers and Y represent the output value of neural network, the objective of this network is to use the neural network to extract valuable features from the original input and generate an output value based on the extracting features. As a famous hypothesis said: non-linear neural network layers can asymptotically approximate complicate functions [5]. In mathematical process, the function of convolutional neural network is similar to solving a high-degree multivariate system of equations. To reduce the difference between the input and output value, my method needs the loss function to implement the backpropagation and update the weight parameters in my network. The loss function is shown as below:

Y = H(x)

Loss = H(x) - x (1)

The main reason for using ResNet architecture as the baseline model instead of VGG architecture is that the ResNet architecture solved the degradation problem to preserve more valuable information from input. As the VGG architecture becomes deeper, every epoch training will let the input data pass the activation layers and convolutional layers to extract features, but part of valuable data also lost during training. Therefore, ResNet architecture uses residual modules to solve this issue. In math, it can be represented as equation 2 below:

Y = F (x, {Wi}) + x (2)

In this equation, x, Y represent the input and output of residual module and the F (x, {Wi}) represents the residual mapping to be learned.

## 3.1. Experiment Setup

The dataset used in this paper comes from the CIFAR-10 dataset which consists of 60000 32x32 color images in 10 classes, with 6000 images per class [6]. The dataset is pre-split into 6 batches which contains 5 training batches and 1 testing batch. To improve the networks’ performance, I applied normalization on all images and data argumentation on training set images to help the proposed model achieve high performance within 10 minutes’ training period. Here are 3 data augmentation techniques I used:

* Random Crop: All training images randomly crop into a size of 32x32 pixels with padding of 4 pixels. This augmentation technique helps introduce variations in the training data by selecting different regions of the image.
* Random Horizontal Flip: Randomly flips the input image horizontally with a certain probability. This augmentation technique increases the diversity of the training data by creating mirror images.
* Normalize: The technique normalizes images by subtracting the mean and dividing them by the standard deviation. This step helps in standardizing the input data and making it suitable for the model. The provided mean and standard deviation values correspond to the pre-computed values for the CIFAR-10 dataset.

## 3.2. Network Architectures

### 3.2.1. Baseline model

My baseline is inspired by the method of VGG16 nets. All the convolutional layers have 3 x 3 filters and follows with a ReLu and Batchnorm layer. I call this type of sub-module as one convolutional block. The baseline structure used 6 convolutional blocks, 4 max pooling layers and 2 dropout layers with 50% dropout rate and one full connected layer as the classifier. Within 10 minutes training, the accuracy of this model is 84.5%. The detailed network structure diagram is shown in the appendix.

### 3.2.2. Modified model.

Current model is inspired by the ResNet architecture, and I modified it base on the model from my assignment 2. The performance of the new model can achieve 88.52% accuracy within 10 minutes which is 4.02% improvement than my assignment 2 model. The new model consists of 8 convolutional blocks, 3 max pooling layers and 1 dropout layers with 50% dropout rate, 1 average pooling layer and one full connected layer as the classifier. As you can see, the basic structure of new model is similar to the previous model, but it is deeper and has 1 residual module. In the residual module, the output of sequential network 1 will directly add on the output of the sequential network 3. After the input data pass through the sequential network 4, the input data will go through the average pooling process instead of the max pooling process. This is because max pooling represents the maximum activation in each region of the input feature map which is better for extracting important features (extreme values). Average Pooling represents the average activation of each region in the input feature map which is better for capturing localized features and to preserve spatial details. Therefore, the structure of 3 max pooling and 1 average pooling layers is better than 4 max pooling layers.

All input data pass through convolution layers or activation layers will lose some information. If the dataset is large and training time is long, researchers don’t need to pay attention on those information loss. However, in this paper, the model must converge in limited time and achieve acceptable performance, so the amount of lost information must be small. Therefore, based on the model of assignment 2, I decreased the dropout layer from 2 to 1 and decreased the dropout rate to 0.4 to against the overfitting problem. Another technique to against information loses is the residual module. By adding the output of sequential network 1 on the output of the sequential network 3, the extreme values in feature map will be easier to extract and this process also preserved the lost information for later convolutional network to learn.

The convolutional layers predominantly utilize 3×3 filters and adhere to two straightforward design principles: (i) they employ the same number of filters for output feature maps of the same size, and (ii) when the feature map size is halved, the number of filters is doubled to maintain consistent time complexity per layer. The down sampling is achieved through convolutional layers with a stride of 2. The network concludes with a global average pooling layer and a fully connected layer of 4096 nodes with the ReLU activation function. The structure of the neural network is shown in Figure 1 as below.

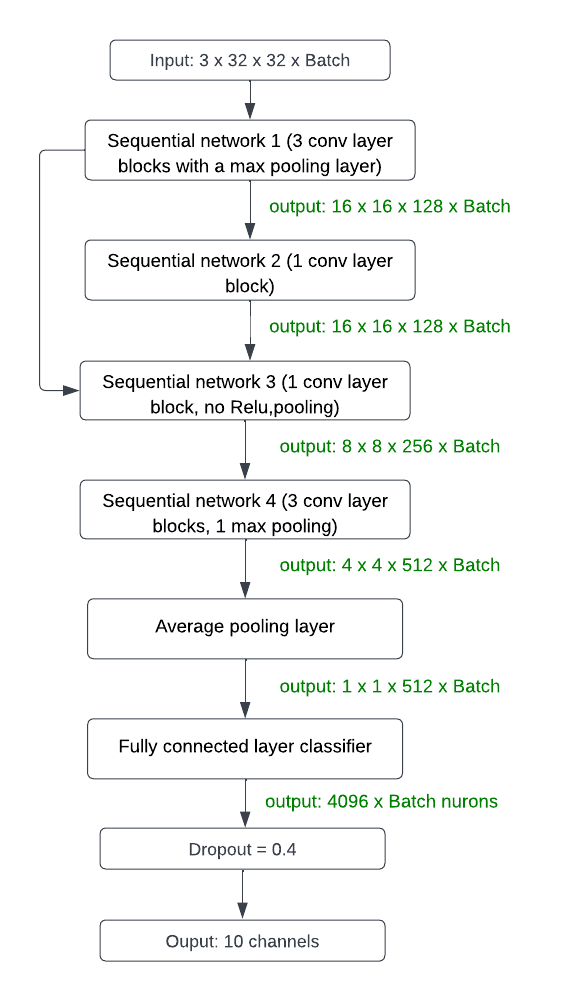


Figure 1. Network structure

# 4. Experiment

I evaluate the new model on the CIFAR10 dataset which contains 60000 images with 10 classes. The model is trained on the 50000 images training dataset with 3 data augmentation techniques including random crop, random horizontal flip and normalization. Those techniques can help models learn more features from the training dataset. The test dataset contains 10000 images with normalization techniques. Because the model aims to classify 10000 images into 10 classes, I evaluate the performance of models by comparing the prediction labels with the true labels. The value of accuracy is calculated by using the number of correct predictions dividing the total number of predictions.

## 4.1. Experiment on various model

### 4.1.1. Experiment on VGG16 model

I first evaluated the performance of VGG16 model on the CIFAR10 dataset. The result surprised me that VGG16 model can only get 64.1% accuracy on CIFAR10 model within 10 minutes. The possible reason is list as follow:

* Model complexity: VGG16 is a deep and complex model with many parameters designed for a large dataset like ImageNet. The CIFAR10 dataset is relatively small compared to ImageNet. The model's capacity may need to be lowered for the dataset, leading to overfitting or difficulty in capturing the dataset's specific patterns.
* Training time: The training period is limited to 10 minutes, which is not sufficient for VGG16 model, which will lead to lower performance of models.
* Hyperparameter Tuning: Deep learning model’s performance has large relationship with various hyperparameters such as learning rate, weight decay, batch size, and optimizer choice. I used the default hypermeters to test VGG16 model’s performance which is not the well-optimized hyperparameter set for CIFAR10.

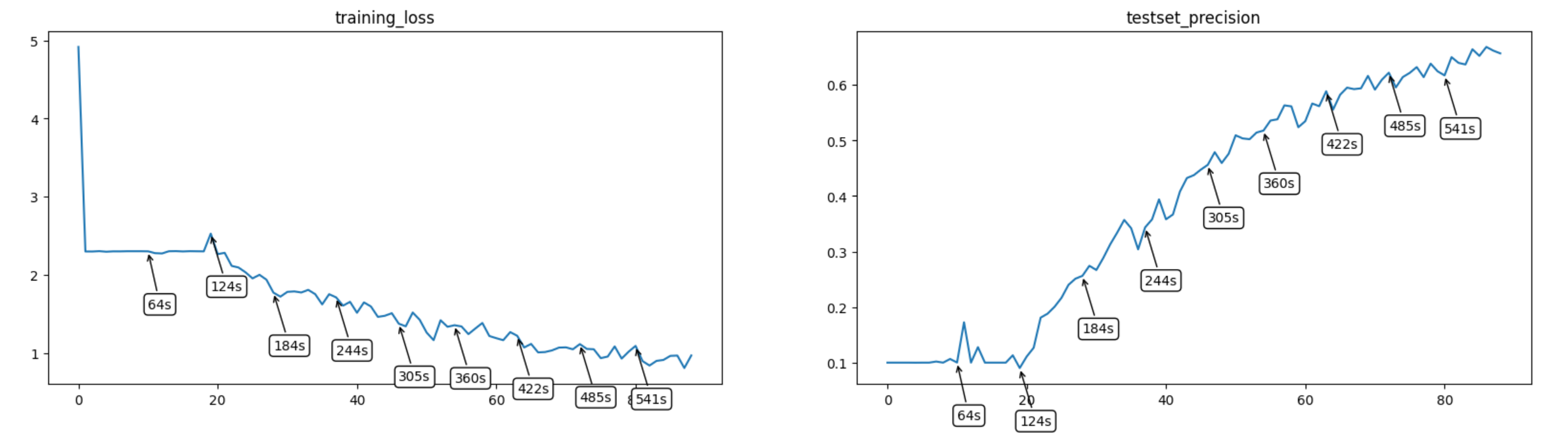


Figure 2. VGG16 model’s performance

### 4.1.2. Experiment on ResNet50 & ResNet18 model

The performance of the ResNet50 and ResNet18 is better than the VGG16 model, and their accuracy is 71.12% and 78.18%, respectively. The possible reason for such low performance should be similar to the VGG16 model, but the performance is increased for ResNet architecture, and the shallow ResNet model (ResNet18) is performed better than ResNet50. This experiment result inspired me to modify the baseline model to the ResNet architecture with a shallower structure.

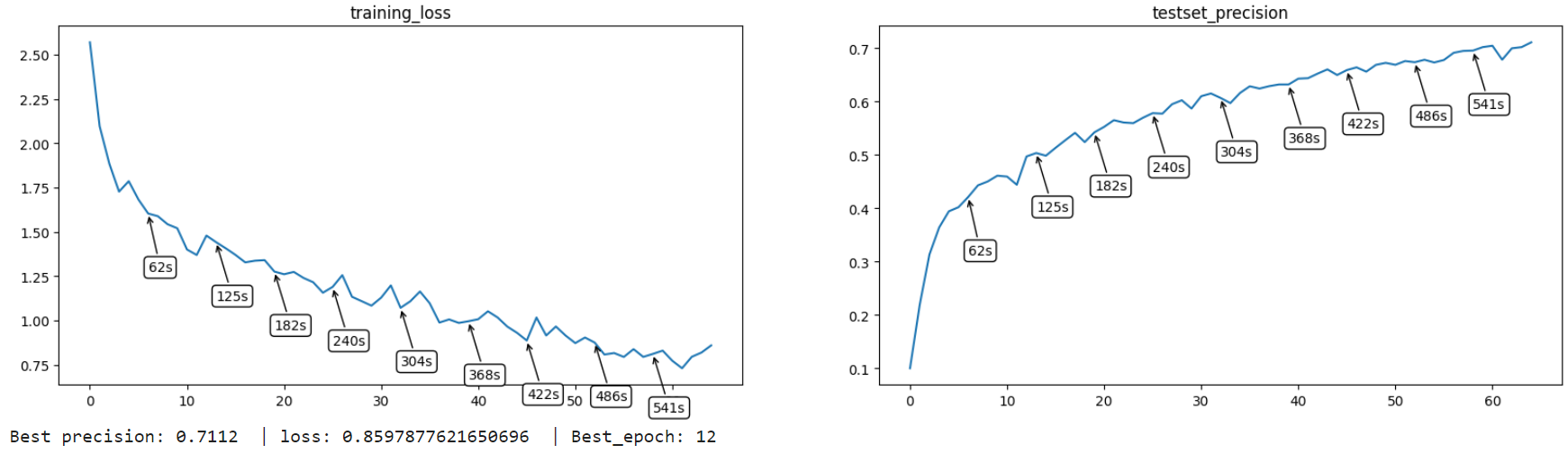


Figure 3. ResNet50 model’s performance

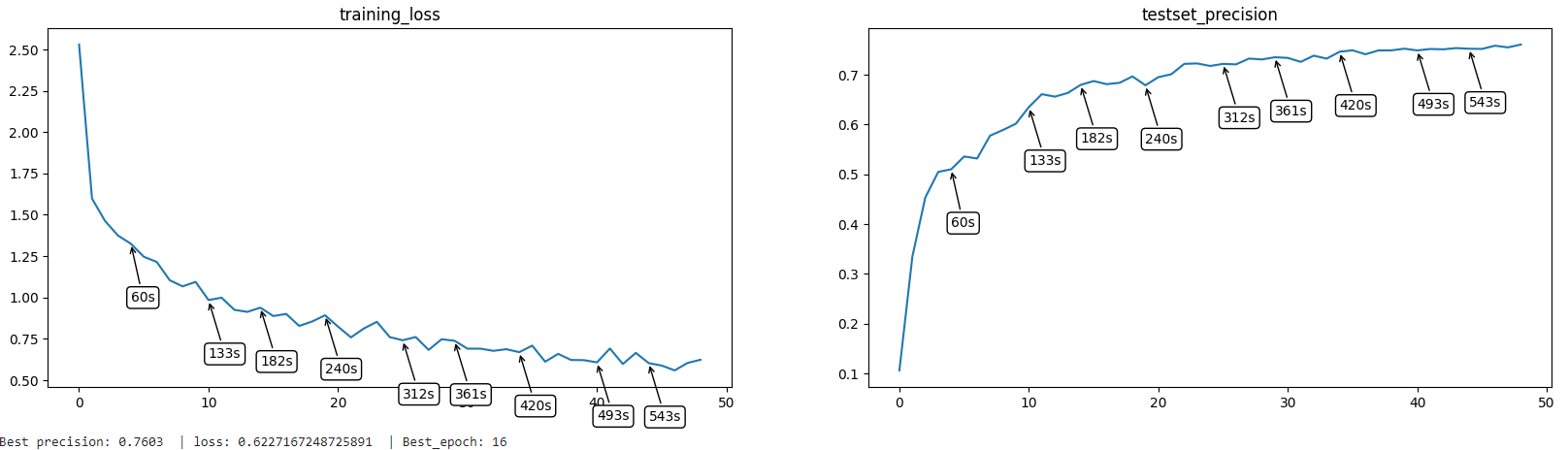


Figure 4. ResNet18 model’s performance

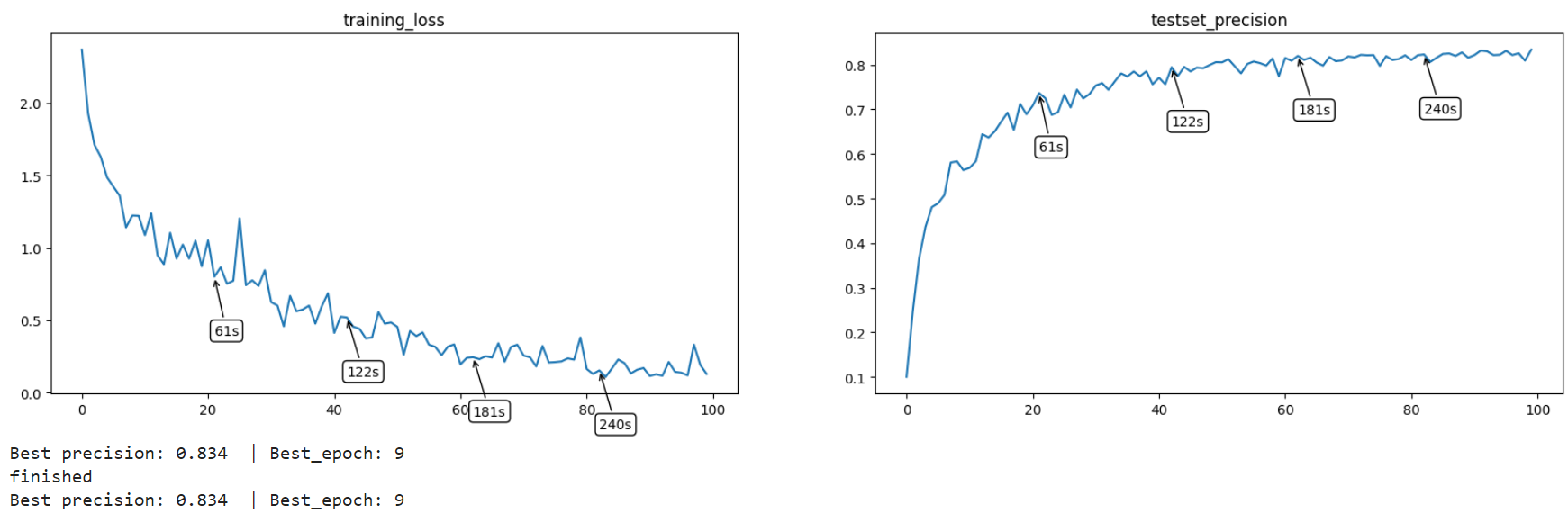


Figure 5. Assignment 2 model’s performance

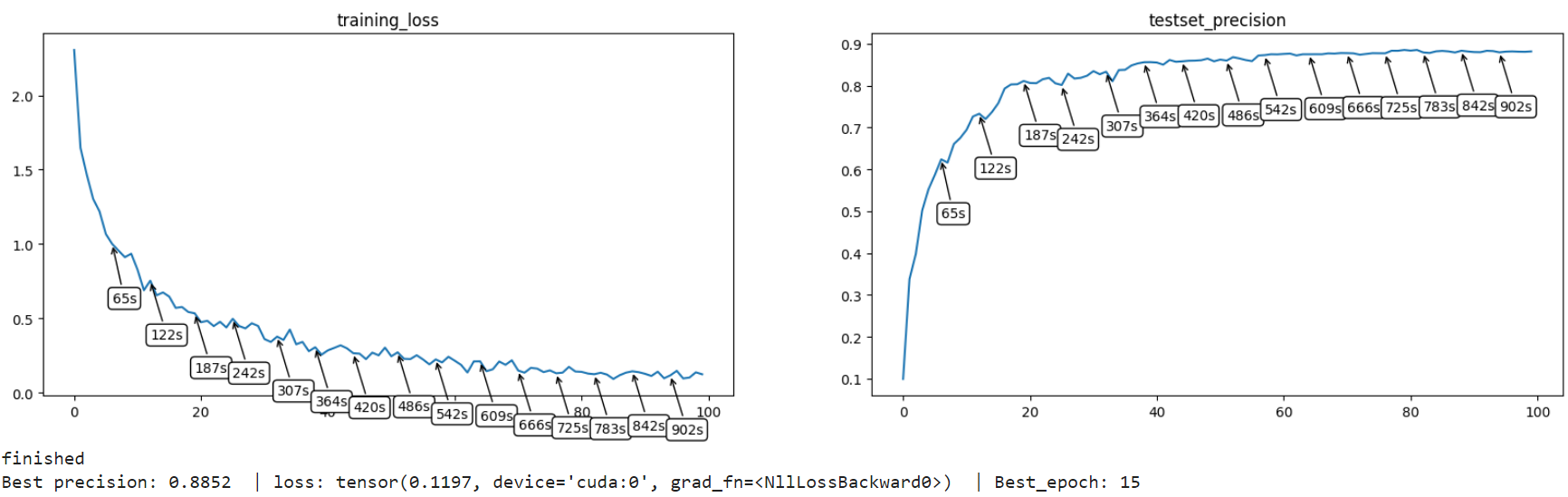


Figure 6. Assignment 3 model’s performance

### 4.1.3. Experiment on assignment 2 model

For the model of assignment 2, it has a straightforward structure inspired by VGG16 model, and it achieved 83.5%

Accuracy, which is higher than VGG16, ResNet18, ResNet50. The possible reason is that the assignment 2 model has simpler structure and less learnable parameters than those classic models, therefore the model can be trained with more epochs. The detailed network structure is shown in the appendix and the performance diagram is shown in Figure 5.

### 4.1.4. Experiment on assignment 3 model

The assignment 3 model is my final version model. I optimized the structure of baseline model to improve the classification accuracy. The modified model achieves 88.52% which is the highest across 5 models. The detail of network architecture is mentioned in network architecture section and the performance diagram is shown in Figure 6.

# 5. Results

The Figure 7 below shows the accuracy comparison of all 5 types of models. The lowest accuracy is obtained by VGG16 which is 64.1% and the highest accuracy is obtained by assignment 3 model which is 88.52%. From the diagram below, it’s clearly that the ResNet model has higher accuracy than the VGG model which also represented that the ResNet architecture is more advanced than VGG architecture. One interesting finding is that the ResNet50 is perform worse than ResNet18. Therefore, the complex network structure not always bring well performance. A well-designed deep learning model structure must fit the characteristic of dataset.

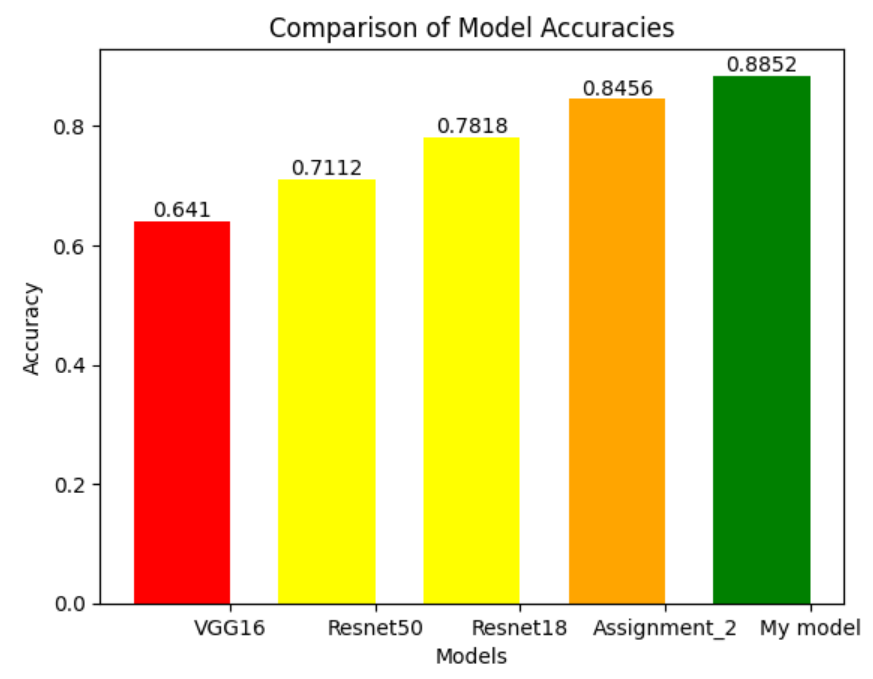


Figure 7. Model performance comparison

# 6. Discussion

The final version of proposed model achieves the best accuracy which proved it is most suitable model for categorizing images in CIFAR10 dataset within 10 minutes. The top performance is due to the appropriate network structure and hyperparameters.

## 6.1. The optimization process for the network

### 6.1.1. Fine tuning techniques

For deep learning models, suitable hyperparameters can help models achieve higher performance. The most important hyperparameter is the learning rate, which is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated [7]. It’s a challenge problem to choose a appropriate learning rate value, because a large learning rate might lead to the unstable training process, but a too small learning rate will result a long straining period which is not realistic for the proposed method. To optimize the model’s performance, I tried to set learning rates equal 0.1,0.01 and 0.001 to exam the performance. The experiment result in table shows when learning rate = 0.001 obtained better performance than other learning rates.

Table 1. The learning rate vs Model accuracy.

|  |  |  |  |
| --- | --- | --- | --- |
| Learning rates | 0.1 | 0.01 | 0.001 |
| Accuracy | 61% | 72% | 88.5% |

The dropout rate is a also important hyperparameter, the high dropout rate will result low accuracy of models, because the dropout rate decide how many neurons need to be abandoned. High dropout rate means high information lost which is not appropriate for my experimental setup. Therefore, I decreased the number of dropout layer to 1 and lower the dropout rate to 0.4 in the proposed model.

For the batch size, I choose 512 images as 1 batch. Even though the large batch size will accelerate the training process, it also decreased the learning ability of deep learning model. Because the size of feature map should expand as the batch size expand, if not the network cannot preserve too many learned features, so the performance will be lower.

### 6.1.2. Training acceleration techniques

The training period is limited to 10 minutes, so I’m only able to train the proposed model with limited number of epochs. To achieve higher performance within limited epochs, I used scheduler to decrease the learning rate value every 4 epochs with 0.5 decay rate which can allow this model to converge fast at beginning and train model more accurate when the training lose close to its global minimal.

The second technique I used is the accumulator, which allow model updates its parameters by accumulating gradients based on the sum of gradients computed over several mini-batches instead of a single mini-batch [8]. The technique effectively increases the batch size, leading to more stable updates and potentially improved convergence. Therefore, proposed model can balance the advantages and disadvantages of batch size. As a result, the proposed model can train 15 epochs without sacrifice the training accuracy and depth of neural network.

# 7. Conclusion

## 7.1. The future work

Even though the proposed method obtained acceptable accuracy, the method still has lots of space for improvement. I designed the method by mixing the VGG architecture with ResNet architecture. These two architectures are classic and powerful, but they are not state-of-art architecture. According to the data from the benchmark website “Paper with code”, the best method for CIFAR10 dataset classification is VIT-H/14 [9], published onICLR 2021 conference. It is a transformer architecture-based deep learning model and is able to achieve 99.5% accuracy. Therefore, updating the old ResNet architecture to the latest transformer architecture is a worthy direction to research in the future.

# References

1. Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791
2. Krizhevsky, Alex, Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional Neural Networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

[3] Kumar, B. (2021, August 31). Convolutional Neural Networks: A brief history of their evolution. Medium. https://medium.com/appyhigh-technology-blog/convolutional-neural-networks-a-brief-history-of-their-evolution-ee3405568597

[4] He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. <https://doi.org/10.1109/cvpr.2016.90>

[5] G. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of ´ linear regions of deep neural networks. In NIPS, 2014.

[6] Krizhevsky, A. (n.d.). The CIFAR-10 dataset. CIFAR-10 and CIFAR-100 datasets. <https://www.cs.toronto.edu/~kriz/cifar.html>

[7] Brownlee, J. (2020, September 11). Understand the impact of learning rate on neural network performance. MachineLearningMastery.com. https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/

[8] Nikita Kozodoi. (2021, February 19). Gradient accumulation in pytorch. Nikita Kozodoi. https://kozodoi.me/python/deep%20learning/pytorch/tutorial/2021/02/19/gradient-accumulation.html

[9] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., &amp; Houlsby, N. (2021, June 3). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv.org. <https://arxiv.org/abs/2010.11929>

# Appendix

1. The detail structure of my network structure:

Layer (type) Output Shape Param #

==========================================

Conv2d-1 [-1, 64, 32, 32] 1,792

ReLU-2 [-1, 64, 32, 32] 0

BatchNorm2d-3 [-1, 64, 32, 32] 128

Conv2d-4 [-1, 64, 32, 32] 36,928

ReLU-5 [-1, 64, 32, 32] 0

BatchNorm2d-6 [-1, 64, 32, 32] 128

MaxPool2d-7 [-1, 64, 16, 16] 0

Conv2d-8 [-1, 128, 16, 16] 73,856

ReLU-9 [-1, 128, 16, 16] 0

BatchNorm2d-10 [-1, 128, 16, 16] 256

Conv2d-11 [-1, 128, 16, 16] 147,584

ReLU-12 [-1, 128, 16, 16] 0

BatchNorm2d-13 [-1, 128, 16, 16] 256

Conv2d-14 [-1, 128, 16, 16] 147,584

BatchNorm2d-15 [-1, 128, 16, 16] 256

ReLU-16 [-1, 128, 16, 16] 0

MaxPool2d-17 [-1, 128, 8, 8] 0

Conv2d-18 [-1, 256, 8, 8] 295,168

ReLU-19 [-1, 256, 8, 8] 0

BatchNorm2d-20 [-1, 256, 8, 8] 512

MaxPool2d-21 [-1, 256, 4, 4] 0

Conv2d-22 [-1, 512, 4, 4] 1,180,160

ReLU-23 [-1, 512, 4, 4] 0

BatchNorm2d-24 [-1, 512, 4, 4] 1,024

Conv2d-25 [-1, 512, 4, 4] 2,359,808

ReLU-26 [-1, 512, 4, 4] 0

BatchNorm2d-27 [-1, 512, 4, 4] 1,024

AdaptiveAvgPool2d-28 [-1, 512, 1, 1] 0

Linear-29 [-1, 4096] 2,101,248

ReLU-30 [-1, 4096] 0

Linear-31 [-1, 4096] 16,781,312

ReLU-32 [-1, 4096] 0

Dropout-33 [-1, 4096] 0

Linear-34 [-1, 10] 40,970

================================================================

Total params: 23,169,994

Trainable params: 23,169,994

Non-trainable params: 0

----------------------------------------------------------------

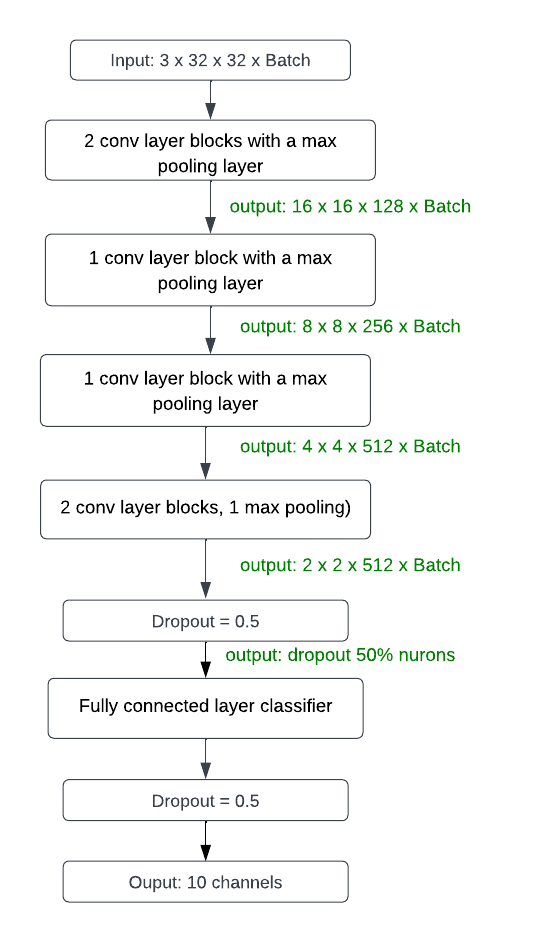
Input size (MB): 0.01

Forward/backward pass size (MB): 6.38

Params size (MB): 88.39

Estimated Total Size (MB): 94.78

2. The network structure of assignment 2:



3. The network structure in graph:

