ResNet and MobileNet V1

qifengc2@uci.deu/ID:57004415

May 1, 2022

Contents

1 Residual connection and Residual block

1.1 Residual connection

In traditional feedforward neural networks, data flows through each layer sequentially: The output of a layer is the input to the next layer. Residual connection provides another path for data to reach latter parts of the neural network by skipping some layers.

1.2 The importance of using Residual block

Deep learning trains neural networks by means of error backpropagation. As the depth of the network increases, Since the model cannot train more layers as an identity mapping, there will be problem of network degradation. As a result, the training error and test error of the model with deeper network layers are larger.
Residual Block adds an identity mapping to ensure that the input and output of this layer are basically the same. It can solve the problem of network degradation to a certain extent

2 Overfitting and Degradation

2.1 Overfitting

Overfitting refers to the phenomenon in which the training error is small and the test error is relatively large in deep learning. Means that the model performs well on the training set and poorly on the test set.

2.2 Degradation

Degradation refers to the phenomenon that as the number of network layers increases, the training accuracy gradually tends to be saturated and begins to decline.

2.3 Difference

The main reason of Overfitting is that the model learns the characteristics of the training data as a commonality, resulting in poor generalization ability. Since the test data does not have the characteristics of the training data, the test results perform poorly.
The main reason of Degradation is that the redundant network layers are not identically mapping.

3 Standard Convolution, 1x1 Convolution and Depthwise Separable Convolution

3.1 Standard Convolution

The depth of the convolution kernel of the standard convolution is the same as the number of channels of the input data, the depth of the output feature

map is 1, and the number of feature maps (the number of output channels) is determined by the number of convolution kernels.
Assumption: the input data size is n×n×m, the convolution kernel size is k×k×m, the number is l, the padding is p, and the step stride is s.
Then the size of the output feature maps is calculated as: (n+2×p-k)/s, and the number is l.

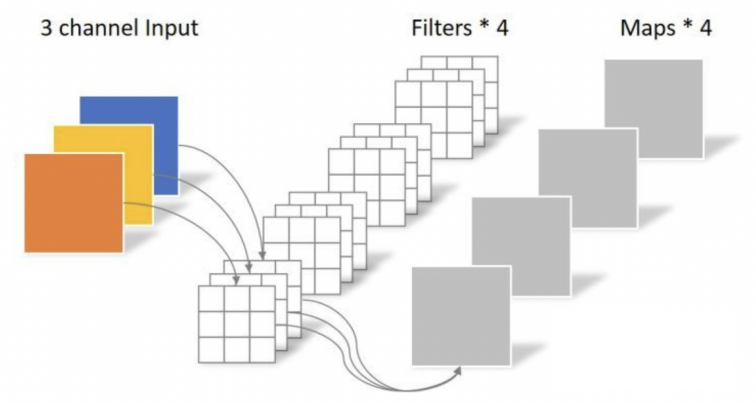

Figure 1: Standard Convolution

3.2 1x1 Convolution

The feature maps calculation of 1x1 Convolution is basically the same as the calculation process of standard convolution, and will not be repeated here. The difference is that the convolution kernel size of 1x1 Convolution is fixed to 1x1.

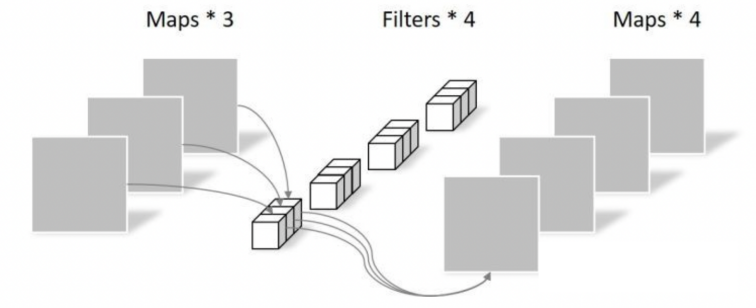

Figure 2: 1x1 Convolution

3.3 Depthwise Convolution

The number of Depthwise Convolution kernels is determined by the number of input channels (the number of output channels is the same as the number of

input channels). The depth of the convolution kernel is fixed to 1, and each convolution kernel processes data in one channel.

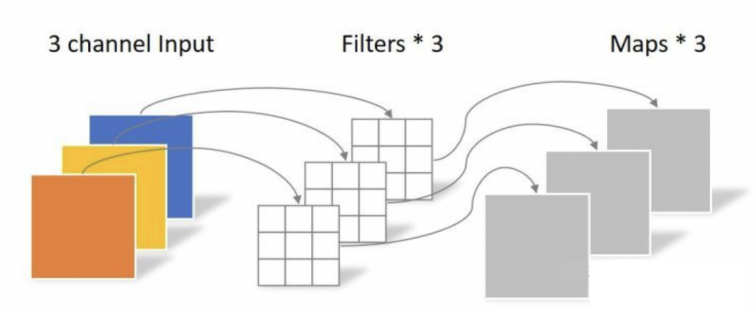

Figure 3: Depthwise Convolution

3.4 Difference

3.4.1 Difference between Standard Convolution and 1x1 Convolution

The difference between Standard Convolution and 1x1 Convolution mainly lies in the size of the convolution kernel. The size of the convolution kernel of Standard Convolution is not fixed, while the size of the convolution kernel of 1x1 Convolution is fixed to 1x1.

3.4.2 Difference between Standard Convolution and Depthwise Convolution

The difference between Standard Convolution and Depthwise Convolution is the depth of the convolution kernel and the number of convolution kernels. The number of Standard Convolution input channels determines the depth of the convolution kernel. The number of convolution kernels (the number of output channels) is not determined by the number of input channels. Each convolution kernel processes the data of all input channels at the same time. The number of Depthwise Convolution input channels determines the number of convolution kernels (the number of output channels), and the depth of each convolution kernel is fixed at 1, and each convolution kernel only processes data in one input channel.

3.4.3 Difference between 1x1 Convolution and Depthwise Convolution

The difference between 1x1 Convolution and Depthwise Convolution is that the size of the convolution kernel of 1x1 Convolution is fixed to 1x1, each convolution kernel processes data of all channels at the same time, and the number of convolution kernels (the number of output channels) is not determined by the number of input channels. The depth of each Depthwise Convolution kernel is fixed to 1, each convolution kernel only processes data in one channel, and the

number of input channels determines the number of convolution kernels (the number of output channels).

4 ResNet

4.1 Model Architectures

My model contains two residual block.

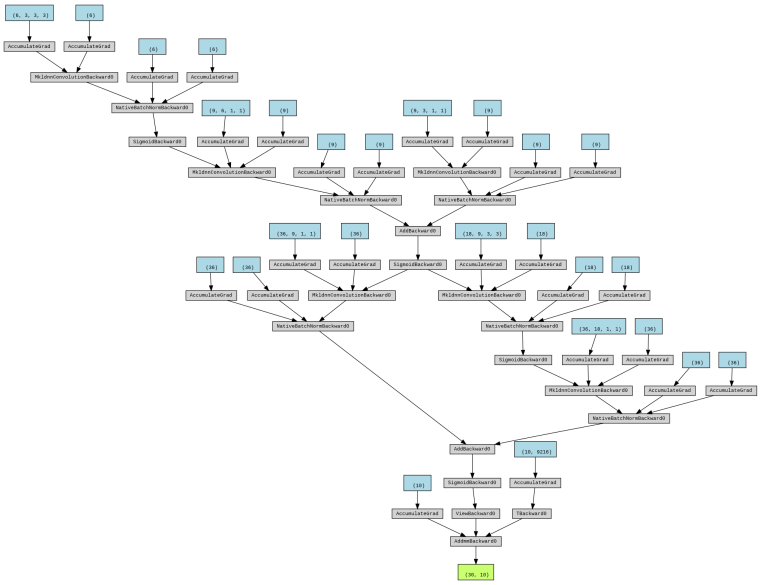

Figure 4: ResNet Model Architecture

4.2 Residual Block and Fully Connected Layer

4.2.1 Residual Block Architecture

The composition of the residual block is as follows: a shortcut and a regular network layer are superimposed and passed through the activation function. The shortcut contains a convolutional layer and a batch normalization. The regular network layer consists of two modules: the first module concludes the convolutional layer, batch normalization and activation function. The second module concludes convolutional layer and batch normalization.

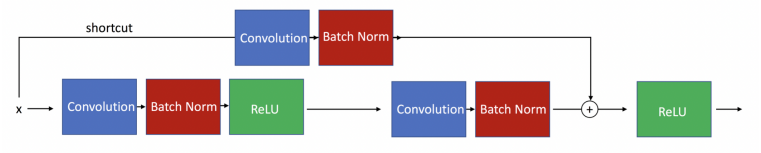

Figure 5: Residual Block Architecture

4.2.2 First Residual Block

The size of the first convolution kernel in first residual block regular network layer is 3x3, the default stride is 2, and the padding is 1.
The size of the second convolution kernel in first residual block regular network layer is 1x1, the default stride is 1, and the padding is 0.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Cov1	3	3×3	3×64×64	6×32×32
BatchNorm1	N\A	N\A	6×32×32	6×32×32
ActF1	N\A	N\A	6×32×32	6×32×32
Cov2	6	1×1	6×32×32	9×32×32
BatchNorm2	N\A	N\A	9×32×32	9×32×32

Table 1: The Table Of The Regular Network Layer In First Residual Block

The size of the convolution kernel in first residual block shortcut is 1x1, the default stride is 2, and the padding is 0.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Cov1	3	1×1	3×64×64	9×32×32
BatchNorm1	N\A	N\A	9×32×32	9×32×32

Table 2: The Table Of The Shortcut In First Residual Block

4.2.3 Second Residual Block

The size of the first convolution kernel in second residual block regular network layer is 3x3, the default stride is 2, and the padding is 1.
The size of the second convolution kernel in second residual block regular network layer is 1x1, the default stride is 1, and the padding is 0.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Cov1	9	3×3	9×32×32	18×16×16
BatchNorm1	N\A	N\A	18×16×16	18×16×16
ActF1	N\A	N\A	18×16×16	18×16×16
Cov2	18	1×1	18×16×16	36×16×16
BatchNorm2	N\A	N\A	36×16×16	36×16×16

Table 3: The Table Of The Regular Network Layer In Second Residual Block

The size of the convolution kernel in second residual block shortcut is 1x1, the default stride is 2, and the padding is 0.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Cov1	9	1×1	9×32×32	36×16×16
BatchNorm1	N\A	N\A	36×16×16	36×16×16

Table 4: The Table Of The Shortcut In Second Residual Block

4.2.4 Fully Connected Layer

The fully connected layer is one layer, the size of the kernel is 16x16, and the output is 10.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Fc	10	16×16	36×16×16	10

Table 5: The Table Of The Fully Connected Layer

4.3 Training the model

In order to study the influence of different factors on the accuracy of the results, I choose to control the variables to train the model.

4.3.1 Activation Functions effect

To study the influence of activation function on Inference Time, Training Time and accuracy, I will take the method of controlling Optimizer and Learning Rate and changing the Activation Function of the model for training.
I select Adam as the Optimizer and 3e-4 as the Learning Rate. I will use ReLU, Sigmoid, LeakyReLu, ReLU6, and GELU as activation function to train the model.

4.3.2 Different optimizer effect

To study the effect of the Optimizer and different epochs on the accuracy, I will take the method of controlling the Activation Function and Learning Rate, and changing the Optimizer and epochs of the model for training.
I will choose Adam as the Activation Functio and 3e-4 as the Learning Rate value. I will use SGD, Adam, AdamW, Adadelta, Adamax, ASGD as Optimizer and train the model for 10 epochs, 20 epochs, 50 epochs for each Optimizer.

4.3.3 Different learning rate effect

To study the effect of Learning Rate and different epochs on accuracy, I will take control of Activation Function and Optimizer, and change the Learning Rate and epochs of the model for training.
I will choose Adam as the Activation Function and Adam as the Optimizer. The Learning Rate values are 3e-8,3e-6, 3e-4, 3 and 30 and the models will be trained for 10 epochs, 20 epochs, and 50 epochs for each value.

4.4 Results

4.4.1 Activation Functions effect

I measure the training time and inference time of ReLU and set ReLU's ratio as unit 1. By measuring the training time and inference time of other activation functions, comparing them with the training time and inference time of ReLU and calculating the percentage.

Activation Function	ReLU	Sigmoid	LeakyReLu	ReLU6	GELU
Inference Time	1	0.11%	0.09%	0.1%	1%
Training Time	1	33.69%	18.07%	12.96%	16.26%
Accuracy	98.07%	96.92%	97.96%	98.31%	98.38%

Table 6: Activation Functions effects on Trainng time and Accuracy

4.4.2 Different optimizer effect

Optimizer	SGD	Adam	AdamW	Adadelta	Adamax	ASGD
10 epochs	92.48%	98.07%	97.69%	83.35%	97.25%	92.73%
20 epochs	93.57%	98.45%	98.49%	85.14%	97.74%	94.52%
50 epochs	96.03%	99.27%	99.28%	90.36%	99.02%	98.97%

Table 7: Different optimizer and how the accuracy changes

4.4.3 Different learning rate effect

LR	3e-8	3e-6	3e-4	3	30
10 epochs	6.87%	84.37%	98.07%	51.18%	48.80%
20 epochs	2.62%	88.67%	98.20%	48.80%	48.75%
50 epochs	0.07%	94.68%	99.01%	41.13%	40.29%

Table 8: Different learning rate and how the learning rate change the results

4.5 Summary

The effects of different activation functions, different optimizers, and different learning rates on the model have been discussed in previous reports. These effects are also well reflected in the residual network model, so I won't go into details here.

This time I want to study the difference between residual network and general network(without residual network). By comparing the operation time, I found that the residual network needs longer training time, I think because it is more complex than the ordinary network. However, the test and verification time of

the residual network is shorter than that of the general network.
At the same time, when the activation functions, optimizers, learning rates and training times are the same, the accuracy obtained by the residual network is higher than that of the general network.
I think the characteristic of residual network is that although it increases the computation time, it trains a better model. As a result, the accuracy of model verification is improved and the verification time is shortened.

5 MobileNet V1

5.1 Model Architectures

My model contains one Convolutional Layer and one Fully Connected Layer.

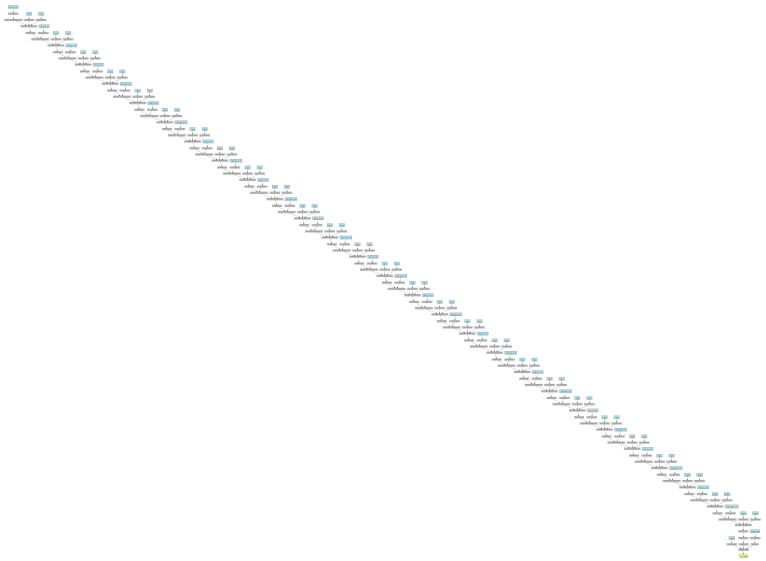

Figure 6: MobileNet V1 Architecture

5.2 Convolutional Layer

The convolutional layer contains a standard convolution and thirteen Depthwise Separable Convolutions and an Adaptive Average Pool.

5.2.1 Standard Convolution

Standard Convolution concludes convolutional layer, batch normalization and activation function.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Cov1	3	3×3	3×64×64	32×32×32
BatchNorm1	N\A	N\A	32×32×32	32×32×32
ActF1	N\A	N\A	32×32×32	32×32×32

Table 9: The Table Of The Standard Convolution

5.2.2 Depthwise Separable Convolution

Depthwise Separable Convolution concludes Depthwise Convolution and 1x1 Convolution. They both have convolutional layer, batch normalization and activation function.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Cov1	32	3×3	32×32×32	32×32×32
BatchNorm1	N\A	N\A	32×32×32	32×32×32
ActF1	N\A	N\A	32×32×32	32×32×32
Cov2	32	1×1	32×32×32	64×32×32
BatchNorm2	N\A	N\A	64×32×32	64×32×32
ActF2	N\A	N\A	64×32×32	64×32×32
Cov3	64	3×3	64×32×32	64×16×16
BatchNorm3	N\A	N\A	64×16×16	64×16×16
ActF3	N\A	N\A	64×16×16	64×16×16
Cov4	64	1×1	64×16×16	128×16×16
BatchNorm4	N\A	N\A	128×16×16	128×16×16
ActF4	N\A	N\A	128×16×16	128×16×16

Table 10: The Table Of Part of The Depthwise Separable Convolution

5.2.3 Adaptive Average Pool

The Adaptive Average Pool, the size of the kernel is 1x1, and the output is 1024.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
AdaptiveAvgPool	1024	1×1	1024×2×2	1024

Table 11: The Table Of The Adaptive Average Pool

5.3 Fully Connected Layer

The fully connected layer is one layer, the size of the kernel is 1x1, and the output is 1000.

Layer Name	Channel Size	Kernel Size	Input Size	Output Size
Fc	1000	1×1	1024×1×1	1000

Table 12: The Table Of The Fully Connected Layer

5.4 Training the model

In order to study the influence of different factors on the accuracy of the results, I choose to control the variables to train the model.

5.4.1 Activation Functions effect

To study the influence of activation function on Inference Time, Training Time and accuracy, I will take the method of controlling Optimizer and Learning Rate and changing the Activation Function of the model for training.
I select Adam as the Optimizer and 3e-4 as the Learning Rate. I will use ReLU, Sigmoid, LeakyReLu, ReLU6, and GELU as activation function to train the model.

5.4.2 Different optimizer effect

To study the effect of the Optimizer and different epochs on the accuracy, I will take the method of controlling the Activation Function and Learning Rate, and changing the Optimizer and epochs of the model for training.
I will choose Adam as the Activation Functio and 3e-4 as the Learning Rate value. I will use SGD, Adam, AdamW, Adadelta, Adamax, ASGD as Optimizer and train the model for 10 epochs, 20 epochs, 50 epochs for each Optimizer.

5.4.3 Different learning rate effect

To study the effect of Learning Rate and different epochs on accuracy, I will take control of Activation Function and Optimizer, and change the Learning Rate and epochs of the model for training.
I will choose Adam as the Activation Function and Adam as the Optimizer. The Learning Rate values are 3e-8,3e-6, 3e-4, 3 and 30 and the models will be trained for 10 epochs, 20 epochs, and 50 epochs for each value.

5.5 Results

5.5.1 Activation Functions effect

Activation Function	ReLU	Sigmoid	LeakyReLu	ReLU6	GELU
Inference Time	1	0.78%	0.39%	0.97%	0.58%
Training Time	1	1.50%	7.84%	12.97%	15.71%
Accuracy	94.59%	98.18%	95.32%	94.28%	95.61%

Table 13: Activation Functions effects on Trainng time and Accuracy

5.5.2 Different optimizer effect

Optimizer	SGD	Adam	AdamW	Adadelta	Adamax	ASGD
10 epochs	75.79%	94.59%	95.03%	69.49%	92.95%	72.95%
20 epochs	78.20%	96.52%	96.96%	72.13%	93.57%	74.29%
50 epochs	85.46%	98.74%	98.98%	79.61%	96.51%	80.37%

Table 14: Different optimizer and how the accuracy changes

5.5.3 Different learning rate effect

LR	3e-8	3e-6	3e-4	3	30
10 epochs	0.00%	71.35%	94.59%	51.64%	51.18%
20 epochs	0.00%	81.00%	95.96%	54.75%	51.53%
50 epochs	0.00%	93.21%	98.03%	60.59%	52.26%

Table 15: Different learning rate and how the learning rate change the results

5.5.4 Comparison of different model

```
================================================================
Total params: 4,231,976
Trainable params: 4,231,976
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.57
Forward/backward pass size (MB): 115.43
Params size (MB): 16.14
Estimated Total Size (MB): 132.15
----------------------------------------------------------------
```

Figure 7: Size of MobileNet V1

```
================================================================
Total params: 61,100,840
Trainable params: 61,100,840
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.57
Forward/backward pass size (MB): 8.38
Params size (MB): 233.08
Estimated Total Size (MB): 242.03
----------------------------------------------------------------
```

Figure 8: Size of Alexnet

5.6 Summary

Through comparison, we found that compared with normal convolution, the model trained by separable convolution has higher accuracy after verification. At the same time, the separable convolution contains fewer parameters, and the required memory space of the model is smaller.