USC Viterbi
School of Engineering

DEPARTMENT
Electrical and Computer Engineering

EE 569
Homework #5 Problem 2:
EE569 Competition
CIFAR10 Classification using CNN

Name: Zhiwei Deng
Date: 04/29/2020

Contents

1. Problem 2

1.1 Motivation and Logics

1.1.1 Motivation

Data Augmentation: To improve the generalization ability of the model, the data augmentation is necessary for the training data. The augmentation of the images provides more training samples of the data. This technique can also eliminate the correlations between the objects and its image locations. In this problem, image horizontal flip, rotation, width and height shift are used. While too much augmentation might make the model takes longer time to converge.

VGG Net: VGG Net is a deep learning network which can perform better than AlexNet in image classification tasks. It replaces the large convolutional kernels of the AlexNet with multiple small convolutional kernels. In AlexNet, the kernel sizes are 11 by 11, 7 by 7 and 5 by 5. While in VGG, these kernel sizes are replaced by all 3 by 3 kernel size. For example, the 7 by 7 kernels are replaced by three 3 by 3 kernels, and 5 by 5 kernels are replaced by two 3 by 3 kernels. In other words, to certain receptive, multiple small kernels can outperform single large kernel. Because this manner can help the system get more complicated in-linearity and cost less computation resources. More specifically, the 7 by 7 kernel contains 49 parameters, while three 3 by 3 kernels only have 27 parameters to train. Also, using VGG Net can increase the networks' depth and extract deeper features than AlexNet. The architecture of VGG Net are shown as below in Figure 1.

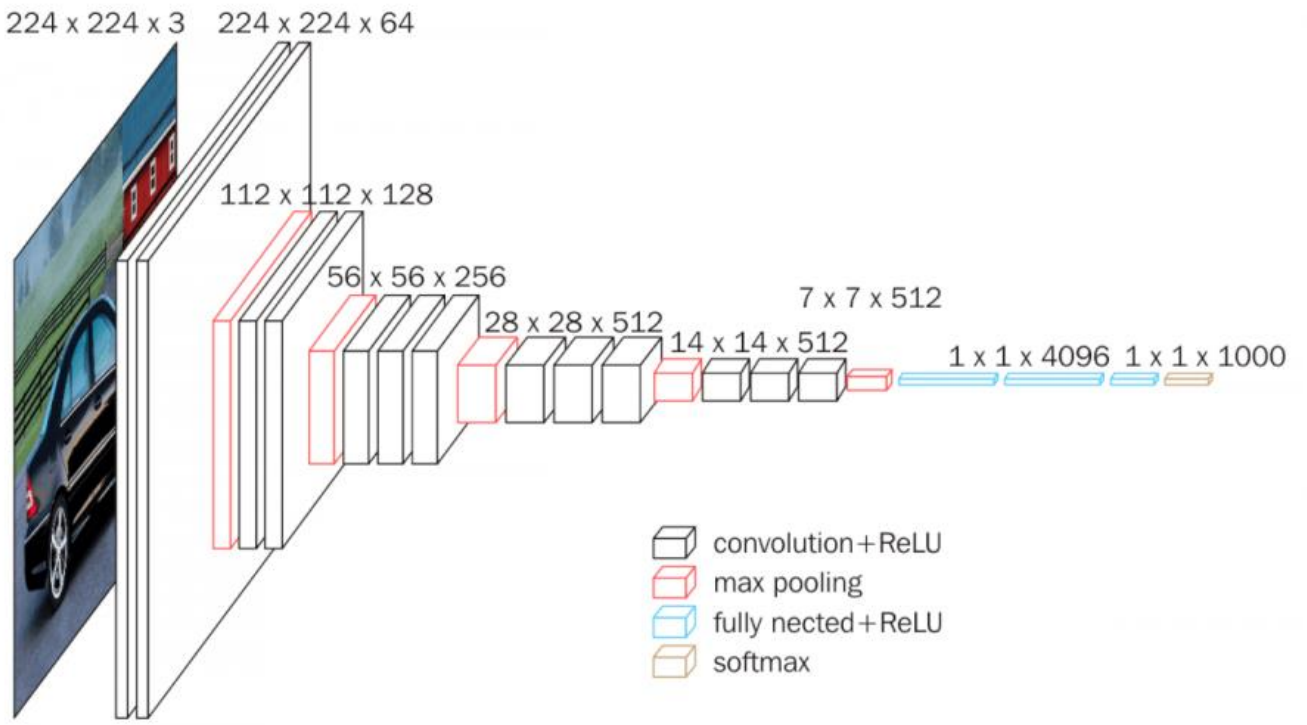

Figure 1. VGG 16 Network Architecture [3]

Boosted LeNet-4[2]: For any accurate model, there still has some errors in it. While since the errors of different model can distributed differently, the average errors of different models can be smaller if we train them separately. The boosted LeNet-4 is proposed in this manner, it trains 3 different LeNet-4 networks and uses the output of each network to make the decision. Since the error distributes differently due to the random initialization, the average confidence of true label should be strengthened, and the errors should be weakened. So, the performance should improve in general. According to [2], the error rate for MNIST dataset of single LeNet-4 is 1.1%, and for boosted LeNet-4 is 0.7%. The architecture of boosted LeNet-4 is shown as below in Figure 2. While this network adds the output of each subnet directly rather than compute the average.

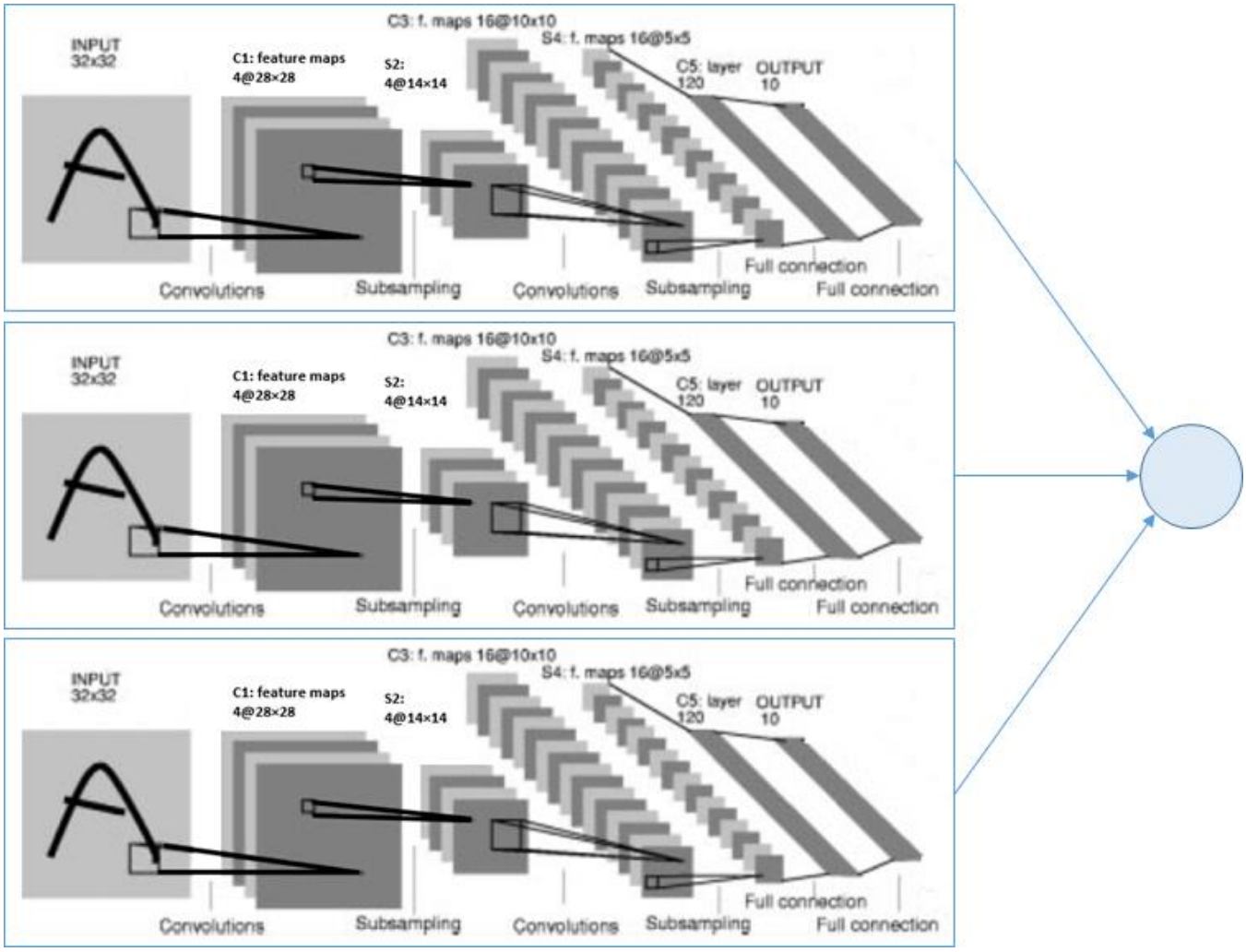

Figure 2. Boosted LeNet-4 Architecture

1.1.2 Logics

More Filters: In the baseline LeNet-5 model, for each convolutional layer, the number of the filters are small. In general, more filters extract more features of the image. Although, the filter number is bigger means that there are more parameters need to train. Also, based on the baseline LeNet-5 model, I add 3 more convolutional layers which make the network deeper and more robust. If the network goes deeper, the features that extracted are more representative, so there are usually more filters in later layers.

Less FC Layers: There are experiments that state that the performance of VGG Net will be influenced little even the FC layers are removed. However, most of the parameters are produced by the first FC layer. So, we can remove the FC layer to an average layer and link it directly to the softmax layer. This manner can reduce the model size greatly and have little influence on the model, which can accelerate the training process.

Batch Normalization: If the distributions of the training data vary in a big range, the model will train to fit every distribution of the data. While if the data are normalized in each layer's input, it will be a fix distribution N (0,1), which will be easier for the model to fit the data. For the same reason, weights regularization is also used in every convolutional layer, and Exponential Linear Unit is used as the activation function.

Learning Rate Degradation: As the training process goes on, the loss function gets closer to the global minimum and the gradients will decrease. If the learning rate is same as the beginning, it will be most likely to be too big and miss the global minimum or even back optimized. So, when the epoch number is bigger, the learning rate of the model should decrease gradually. A suitable learning rate scheduler should be set. However, the learning rate cannot be too small, so at some point, the learning rate should just keep. In this assignment, the learning rate is computed by equation (1) and (2).

$$LR = 0.00105 - epoch \times 0.00001 \tag{1}$$

$$LR = 0.00005 \qquad if\ LR < 0.00005 \tag{2}$$

1.1.3 Network Architecture

The single network of the model is shown as below in Figure 3. There are total 5 convolutional layers with 32, 32, 64, 64, 128 filters, 3 max-pooling layers, 2 drop out layers.

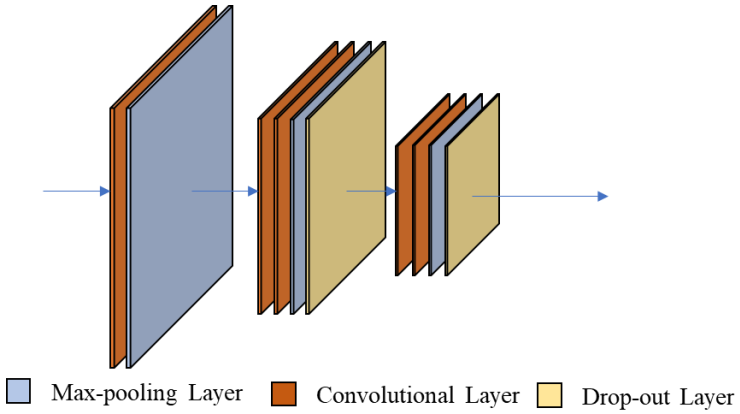

Figure 3. Single Subnet Model

The whole model is constructed by 3 subnets as shown as below in Figure 4.

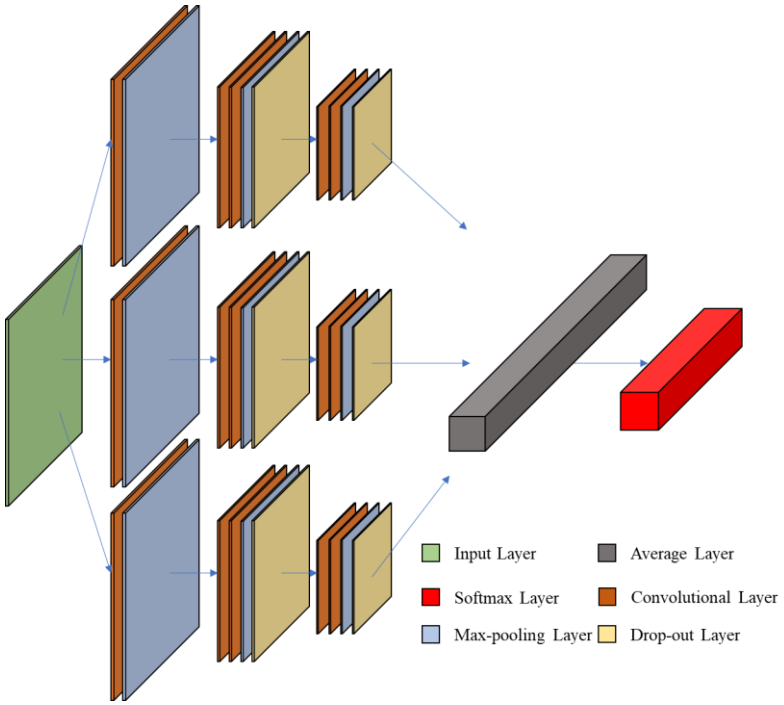

Figure 4. Model Architecture

1.1.4 Discussion

Parameters: The parameters of the network and the training process is show as below in Table 1.

Table 1. Parameters of Training Process

Parameters	Value
Epochs	100
Learning Rate	0.00105 (Initial)
Batch Size	64
Augmentation	Rotation = 20 Width Shift = 0.15 Height Shift = 0.15 Horizontal Flip
Conv1 Filter Num	32
Conv2 Filter Num	32
Conv3 Filter Num	64
Conv4 Filter Num	64
Conv5 Filter Num	128
Dropout1 Factor	0.3
Dropout2 Factor	0.4

Steps: The first conv layer is to extract the features in the first stage and reduce the dimension of the input data in order to reduce the model size. The second, third and fourth, fifth conv layers are expected to extract medium and deepest features form the tensor data respectively. They are concatenated to replace the large kernels using small kernels (3,3) as state above in VGG section. All conv layers are followed by the batch normalization layer to

prepare the data as the normal distributed input for the next layer's input. The boosted process uses 3 independent networks to train for the same data, while the confusion or error has small probability to occur at the same place or classes. So, averaging the outputs of all subnets can lower these errors and provide a more accurate result.

Sources of Improvements: As described above, there are many techniques that improve the generalization ability and robustness of the model. In general, this model outperforms the LeNet-5 is due to more filters and the deeper network architecture. More layers and more filters give the model a better ability to fit the non-linearity of the data, this is the main reason why the model can exceed the LeNet-5.

Generalization: On the other hand, compared with the Problem 1 (b), the generalization ability of the model is better since the data augmentation is applied on the training dataset. Also, drop out is used in this network, which can prevent the dead neurons in the network. The generalization ability is better mean the model can perform more accurate on the data that it hasn't seen before, which means the test accuracy can be higher.

Robustness: The robustness is guaranteed by the batch normalization, the same distribution of the image data can make the model focus more on the feature properties rather than spatial distributions. Also, the learning rate degradation process promises that the model will not overfit the training data or miss the global minimum. Also, the removement of the FC layers contributes to the reduction of the parameter numbers and can prevent the

over-fitting problem. The small kernel window means that there is more local information are preserved compared with the 5 by 5 windows in LeNet-5, which might also contribute to the higher accuracy of the model.

1.2 Classification Accuracy

1.2.1 Accuracy

In this part, I use 3 models to train the training data, single subnet, two subnets and three subnets. The accuracy of each model is shown as below in Table 2.

Table 2. Accuracy and Times of Different Model

Model	Single Subnet	Two Subnets	Three Subnets
Accuracy	87.05%	89.87%	90.10%
Epochs	120	120	120
Train Time / Epoch	560s on CPU 12s on GPU	610s on CPU 16s on GPU	650s on CPU 19s on GPU
Train Time Total	~18.5h on CPU ~0.4h on GPU	~20h on CPU ~0.5h on GPU	~21.5h on CPU ~0.6h on GPU
Inference Time / Epoch	112s on CPU 1s on GPU	115s on CPU 1s on GPU	117s on CPU 1s on GPU

Note: The training time on CPU is estimated by 10 epochs, but not trained by whole epochs. While the training time on GPU is calculated directly. The accuracy values converges after 120 epochs.

For the model with the highest accuracy, the training and testing accuray curves are shown as below in Figure 5 and 7. The training and testing loss curves are shown as below in Figure 6 and 8. From Table 2, we can see that three subnets and 2 subnets' accuracies are quiet close. The test accuracy of 1/4, 1/8, 1/16, 1/32 original training samples is plotted as below in Figure 9.

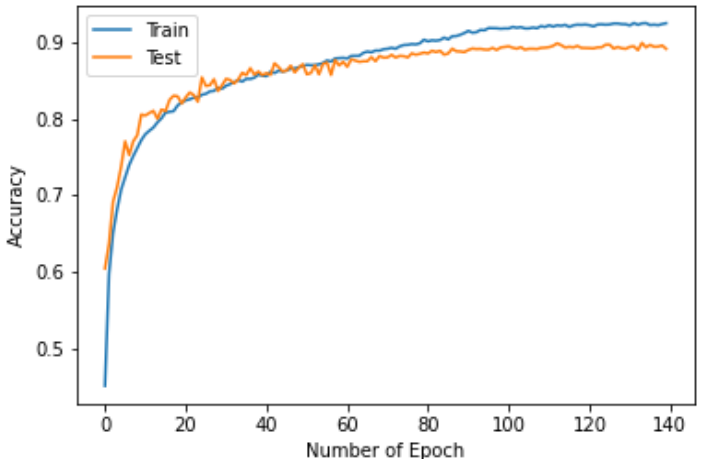

Figure 5. Acc-Epochs (3 Subnets, LR = 0.00105, Batch = 64, Epoch = 140)

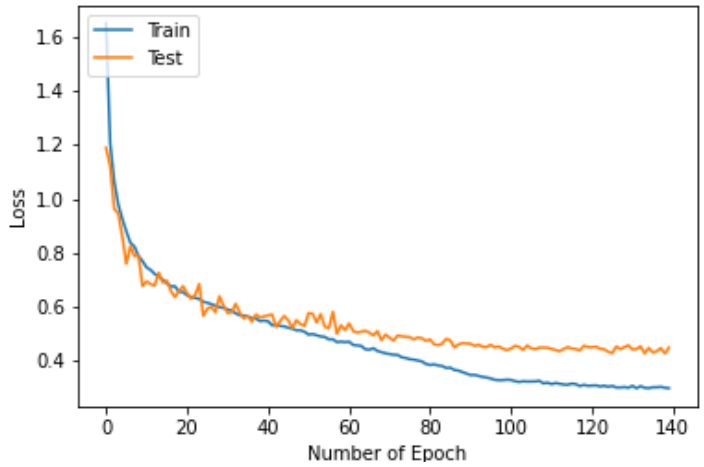

Figure 6. Loss-Epochs (3 Subnets, LR = 0.00105, Batch = 64, Epoch = 140)

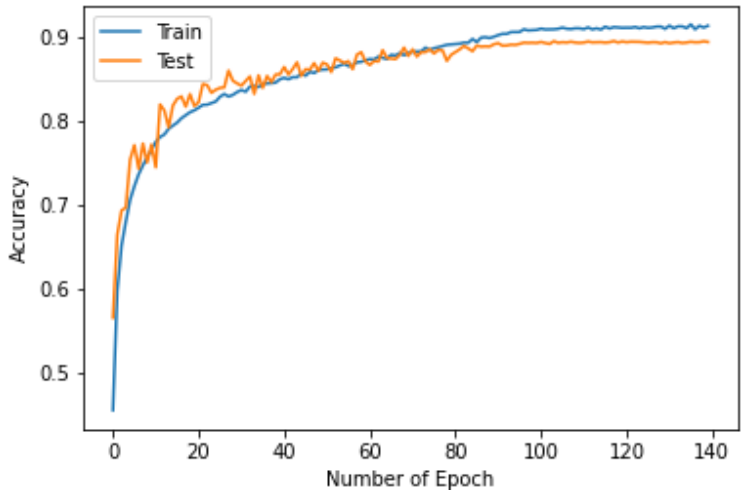

Figure 7. Acc -Epochs (2 Subnets, LR = 0.00105, Batch = 64, Epoch = 140)

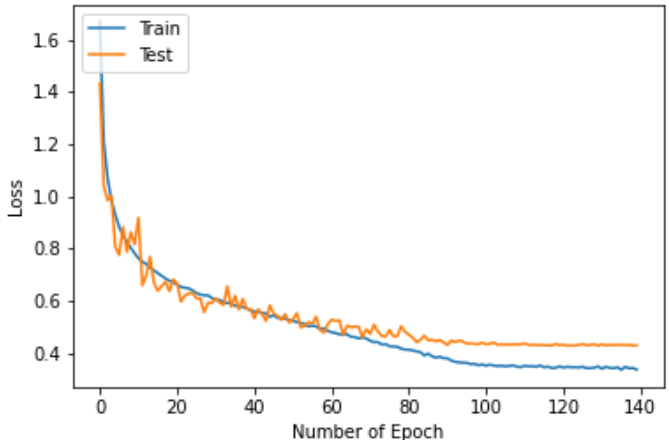

Figure 8. Loss-Epochs (2 Subnets, LR = 0.00105, Batch = 64, Epoch = 140)

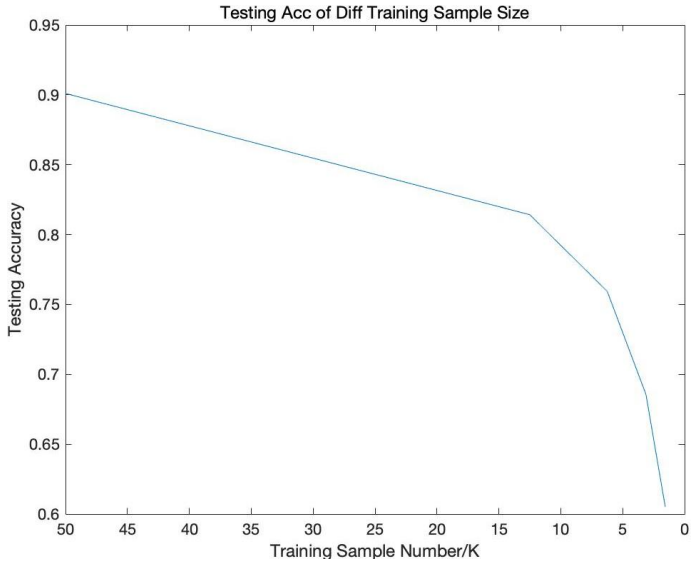

Figure 9. Test Accuracy – Training Sample Size(3 Subnets)

1.2.2 Model Size

The model size of single subnet is shown as below in Table 3.

Table 3. The Model Sizes of Different Model

Model	Single Subnet	Two Subnets	Three Subnets
Model Size	161,196	301,898	442,602
Trainable Size	160,554	300,618	440,682
Untrainable Size	640	1,280	1,920

Since the FC layers are all removed, the model sizes of three models are not very large. This scale of model can easily be trained or applied on the mobile devices. Also, since in the two subnets architectuer and three subnets architeture, the model's parameter is no proportional with the single subnet since the subnets share one same softmax layer, which contains about 20 thousand paramters.

In conclusion, balance the performances of the model sizes, and performances of three models, the 2-Subnet model might be the best model among all. Since it posses medium model size and almost same testing accuracy with 3 subnets. And from the experiment we can see that the boosting techniques can help models to improve their performances.

1.2.3 Discussion

Hyper-parameters: In this assignment, I tried different hyper parameters to see the effects of different hyper-parameters. The performance comparison between hyper-parameter is shown as below, all of the parameters are tested on 3 subnets architecture.

Epochs: The epochs number's influence can be observed from Figure 5 to 9. While the epochs number is larger, the accuracy tends to be higher and loss tend to be lower. However, since the model's property might not be the best. The loss and accuracy converges after certain amount of epochs. In this case, the convergence happens between 100 and 120 epochs. If the epochs number is too small, the model will underfit.While this model will not overfit even the trainong goes on, which can show the robustness of this model.

Optimizers: In this work, I tried two different optimizers to train the network. The comparison between them related with accuracy can be shown as below in Figure 10. From Figure 10, we can see that SGD's performace is slightly worse than Adam, but the final accuracy is basicly same. Also, SGD's step is slower than Adam when the epoch numbers are small, but the during the training processing going on, the SGD is more stable than Adam optimizer.

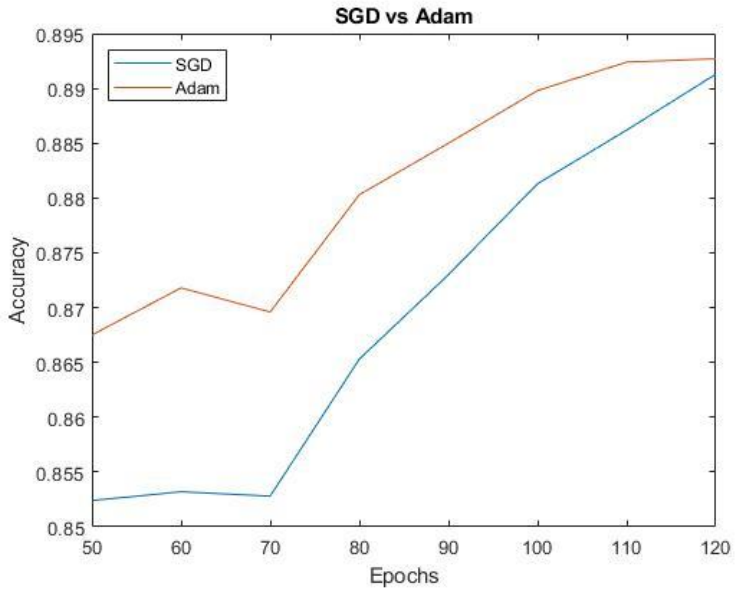

Figure 10. SGD vs Adam

Learning Rate: In this assignment, the learning rate is changing continuously until to some threshold. This method can effectively prevent the overfitting occurrence since the steps of later epochs are much smaller than the beginning. However, this might make the learning process slower but more stable.

Technical Specs: GPU:GTX1080 GRAM:8Gi, CPU:Intel Core-I9-7700K, RAM: 8Gi, Library:Keras 2.3.1, TF:1.14.0.

References

[1] LeCun,Yann,etal. "Gradient based learning applied to document recognition." Proceedings of the IEEE86.11(1998):2278-2324

[2] LeCun,Yann,etal. " Comparison of Learning Algorithms for Handwritten Digit Recognition", ICANN (1995)

[3] Karen S., Andrew Z. "Very Deep Convolutional Networks for Large-Scale Image Recognition", CVPR (2015)