**Multimedia Communications**

EEE408

Lab 1 - report

Name: Cong Yin

ID Number: 1715471

**Introduction**

Convolutional Neural Network is a feed-forward neural network. The artificial neurons of that can respond to a number of surrounding units in a range of coverage, and it has excellent performance for large image processing. Caffe is a clear and efficient framework for deep learning, which was produced by Dr. Yangqing Jia. Caffe is a pure C++/CUDA architecture that support command line, Python and MATLAB interface. It also can seamlessly switch between CPU and GPU.

The objective of this lab is use the kind of database that named cifar-10 for clearly understand the working process of Convolutional Neural Network (CCN).

**Instrument**

**Preparatory work**

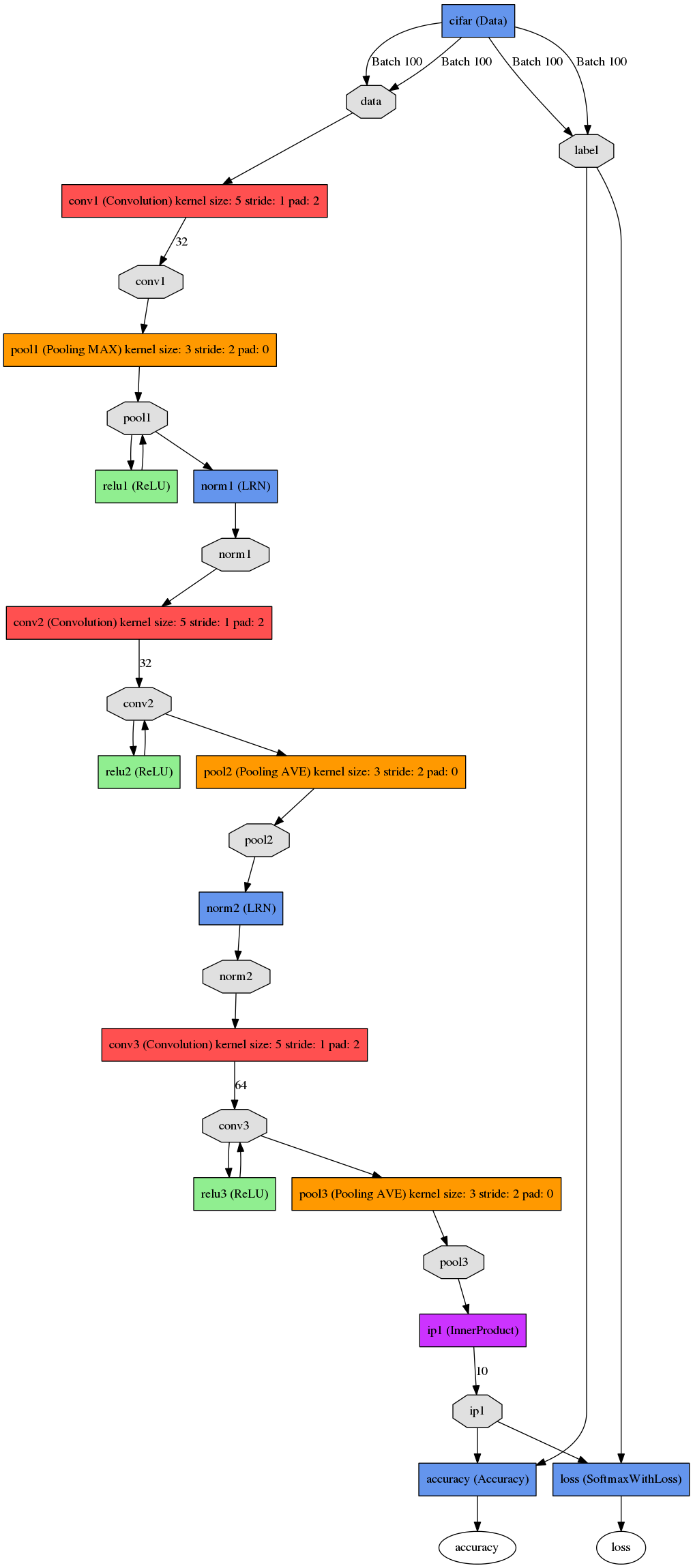
Run the “get\_cifar10.sh” and “create\_cifar10.sh” is the first step of experiment, which can create two files named “cifar10\_test\_lmdb” and “cifar10\_train\_lmdb”.

**Network structure**

There are two ways to draw the network structure. The first kind of method is Netscope that is an on-line visualization tools. It can be used to visualize the network structure of a kind of file, which format is “prototxt” in Caffe structure. Another method is “python/draw\_net.py”. The function of this file is draw the network model, which makes the “prototxt” file become a picture. For example, Figure 1-1 and Figure 1-2 shows the same network structure with different method.



**Figure 1-1 The network structure of cifar-10 (Netscope)**



**Figure 1-2 The network structure of cifar-10 (“python/draw\_net.py”)**

For the two methods, the network structure drawn by Netscope is intuitionistic and succinct that is easy to understand the network model quickly. However, it is lack of detailed information in each layer. The network structure drawn by “draw\_net.py” saves the parameter information, and the details of each layers are more abundant, but the structure is not so clear. This is particularly evident on the large model.

**Curve plotting**

The first step is record the training log. Add new line parameters in the command of the training process, which can implement a record of log as shown below.

‘GLOG\_logtostderr=0 GLOG\_log\_dir=examples/cifar10/log/ \’

And then analyze the training log. Copy the three scripts to the log folder. And run the “parse\_log.sh” can get the two files which have different format.

caffe-master/tools/extra/parse\_log.sh

caffe-master/tools/extra/extract\_seconds.py

caffe-master/tools/extra/plot\_training\_log.py.example

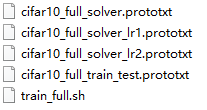
Finally, create an image. Run the “plot\_training\_log.py.example” can get the line chart, and we can choose 0 to 7 before “example” in the script to create the different type of graph.

**Task 1**

**Problem description**

“Please use the “train\_full\_no\_LRN.sh” to train the image classification convolutional neural network (CNN). The file “cifar10\_full\_train\_test\_no\_LRN.prototxt” specifies the network structure. Please draw an illustrative CNN network configuration figure for it, and explain the feature size in each network layer. Please also write down the number of parameters and number of neurons in each layer. (20%)”

In this part, students should understand the basic knowledge of the image classification convolutional neural network (CNN). And then, draw the network structure diagram. Finally, calculate the number of parameters and number of neurons in each layer. For this lab, there are five files will often be used by us as shown in the figure1-3



**Figure 1-3 The files will often be used**

The “cifar10\_full\_solver.prototxt”, “cifar10\_full\_solver\_lr1.prototxt” and “cifar10\_full\_solver\_lr2.prototxt” means the training solution from 0 to 60000, from 60000 to 65000 and from 65000 to 70000. The “cifar10\_full\_train\_test.prototxt” is the whole network structure of cifar-10. “train\_full.sh” is the executable file for the training process.

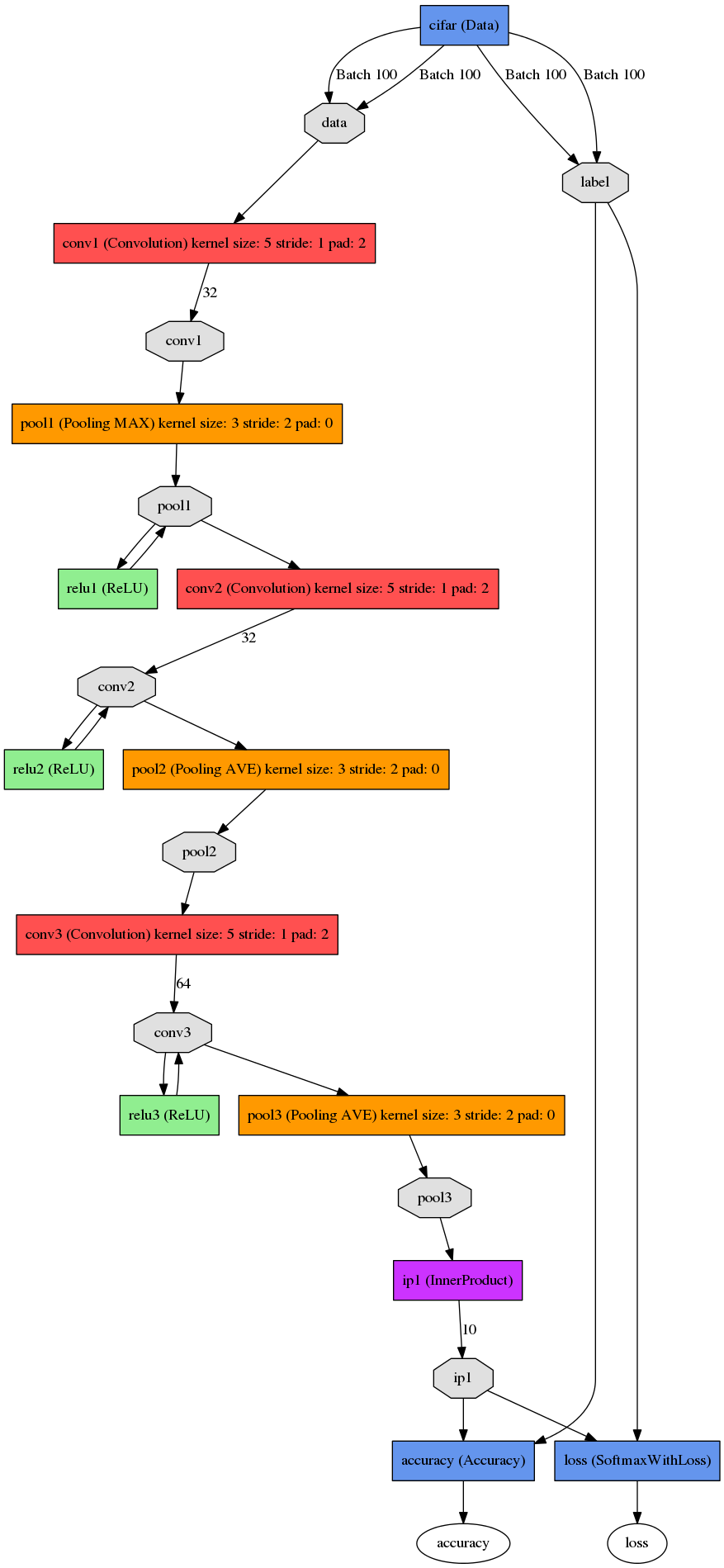
**Experimental process**

The “cifar10\_full\_train\_test\_no\_LRN.prototxt” means delete the Local Response Normalization (LRN) layer. So I delete the all LRN layers in the “cifar10\_full\_train\_test.prototxt” as shown in the figure 1-4. And then I rechristen all of the five active files for this lab in attachment.

**Results and analysis**

The network structure clearly shows there are three convolution layers with Rectified Linear Unit (ReLu) and three pooling layers. According to the following function, we can calculate the number of parameters and number of neurons in each layer.

The “N” means the size of image, and the “F” means kernel size.



**Figure 1-4 The network structure of cifar-10 (no LRN layer)**

According to the comparison of two kinds of network structure, one has Local Response Normalization (LRN) layer, and the other not have Local Response Normalization (LRN) layer and then, we can get the similar result. So, I think the Local Response Normalization (LRN) layer is not so important for model to improve the accuracy.

The size of image is . For the first convolution layer, the number of neurons will turn into , through the first pooling layer, it's usually a good rule to round down to the next number. So, the number of neurons will turn into . For the second convolution layer, the number of neurons is , through the second pooling layer, the number of neurons will turn into . For the third convolution layer, the number of neurons is , through the third pooling layer, the number of neurons will turn into .

For the first convolution layer, the feature size equals to number of parameters. And the number of parameters is . Actually the “3” is the depth of previous data, because the size of the image is . Through the first pooling layer, the number of parameters is . For the second convolution layer, the number of parameters is . through the second pooling layer, the number of parameters is . For the third convolution layer, the number of parameters is . through the third pooling layer, the number of parameters is .

**Task 2**

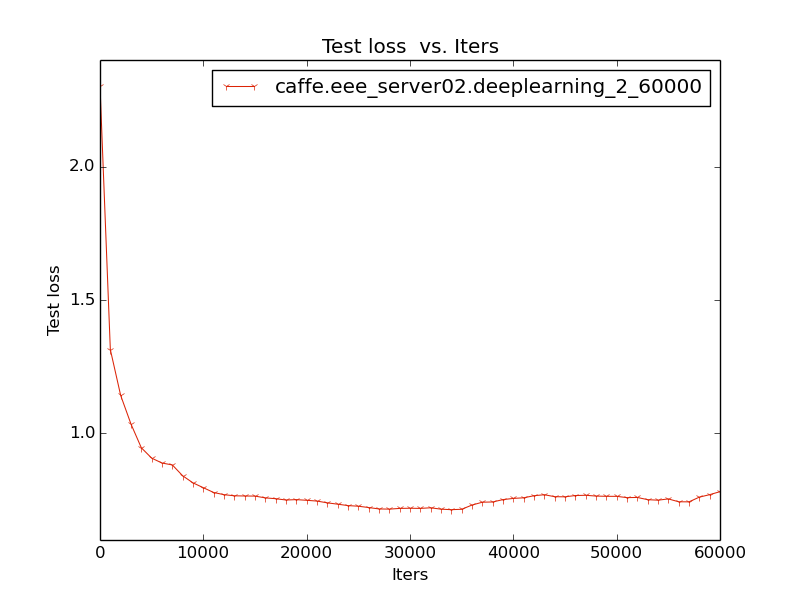
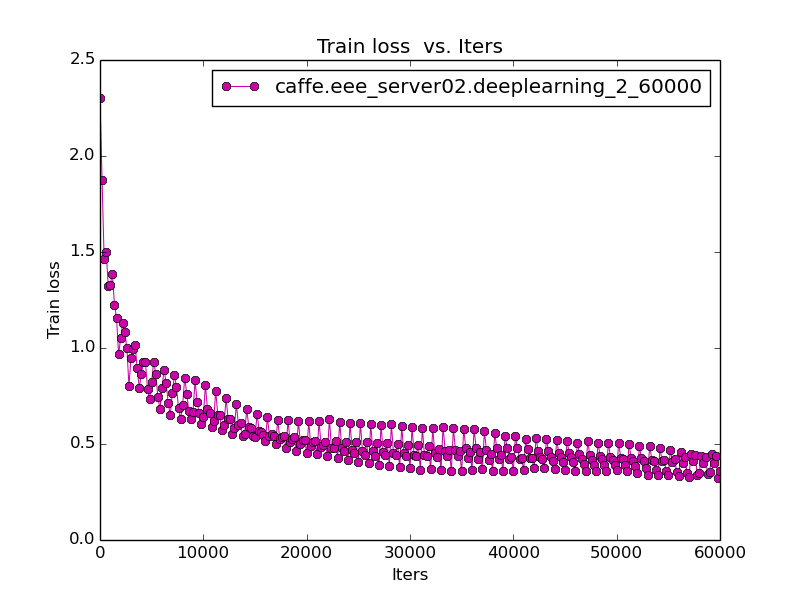
**Problem description**

“Plot a figure to show how training loss and testing loss change with training iteration number. Compare the training loss with the testing loss, and explain why this happens. (20%)”

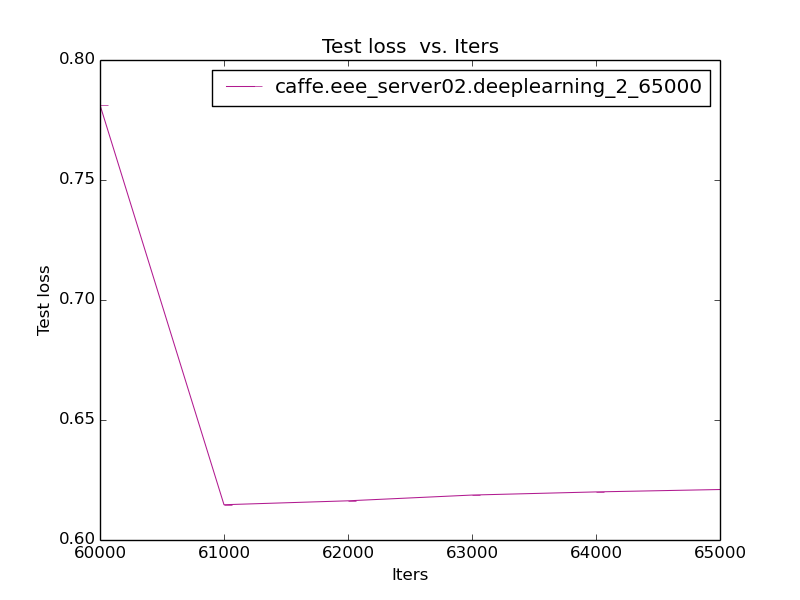
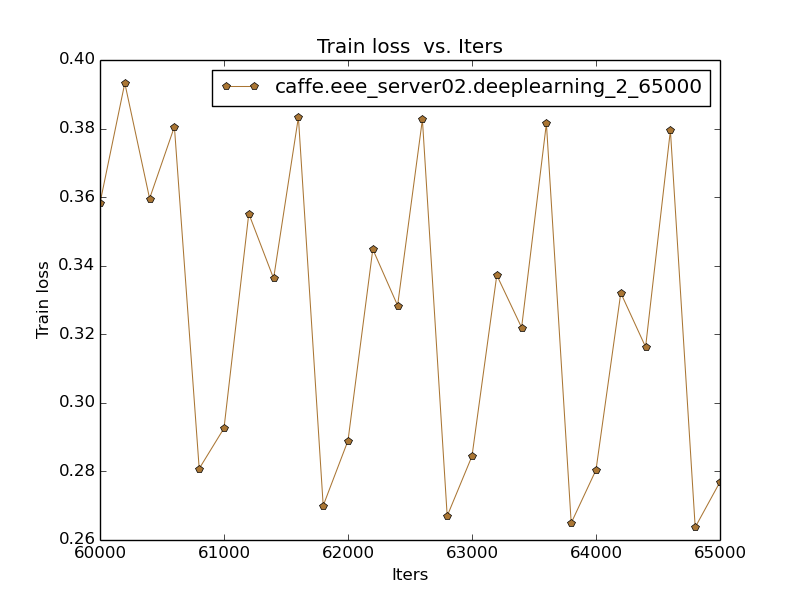
**Results and analysis**

****

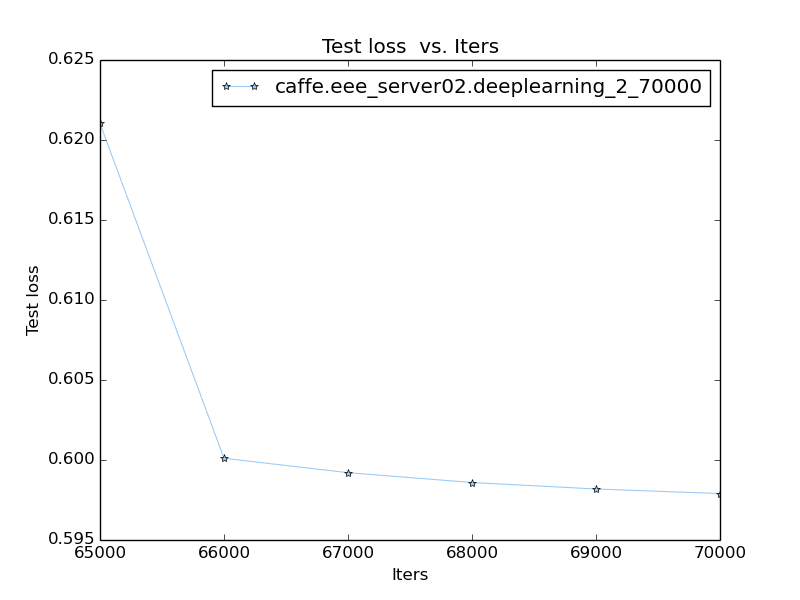
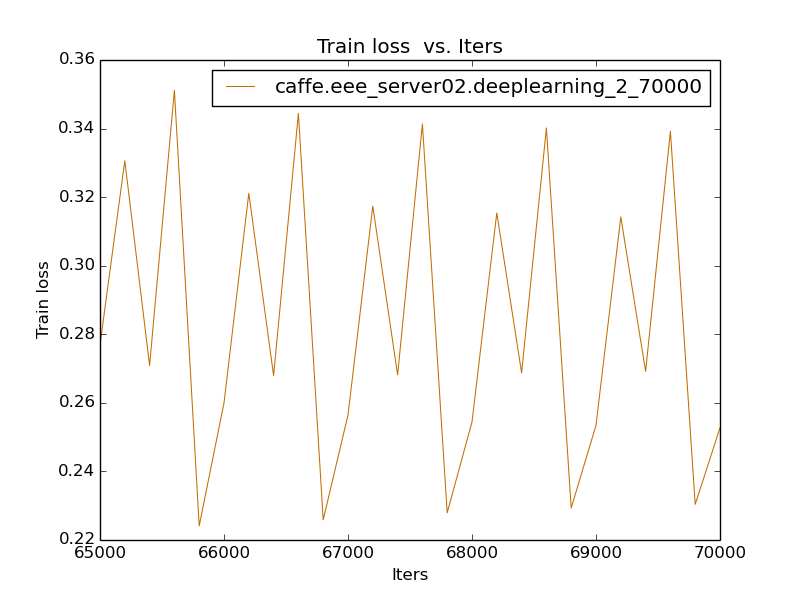
**Figure 2-1 The result of cifar-10 (no LRN layer)**

****

**Figure 2-2 The train loss (left) and test loss (right) with learning rate that equals to 0.001**

****

**Figure 2-3 The train loss (left) and test loss (right) with learning rate that equals to 0.0001**

****

**Figure 2-4 The train loss (left) and test loss (right) with learning rate that equals to 0.00001**

The figure 2-2 previously shows that the train loss and test loss have the same downtrend. The test loss is getting into a kind of stationary state after a certain time that as same as the right of figure 2-3 and figure 2-3. Because it has its own loss value, which adapt it. Even if the graph of train loss has abundant fluctuation, but the overall trend is same as test loss. All of that means it meets the bottleneck of the learning process, which should reduce the learning rate. Overall, the train loss is less than test loss.

**Task 3**

**Problem description**

“CNN network structure will affect the image classification performance. Please change the network structure by modifying the prototxt and provide the test accuracy with each new network structure. The prototxt for all the network models should be included in the report. (20%)

(The modifications should include 4 items: (a) half the channel number in each layer; (b) double the channel number in each layer; (c) change to CNN layer number from 3 to 0 and 1; (d) change the CNN layer number from 3 to 4 and 5.)”

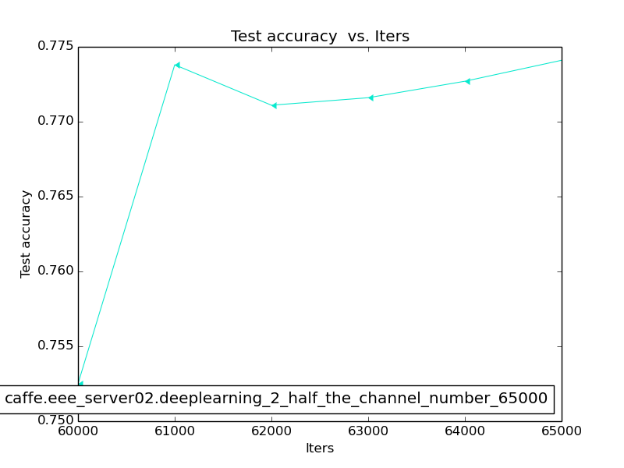
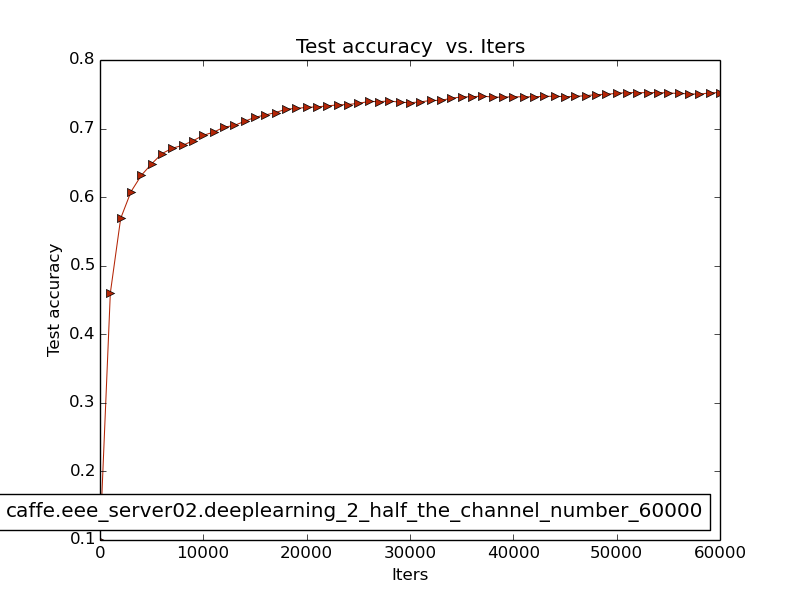
**Experimental process**

The changing of channel number of each layer can change the value of “num\_output” in the “cifar10\_full\_train\_test.prototxt”. The changing of CNN layer number is also add or subtract the CNN layer in the “cifar10\_full\_train\_test.prototxt”.

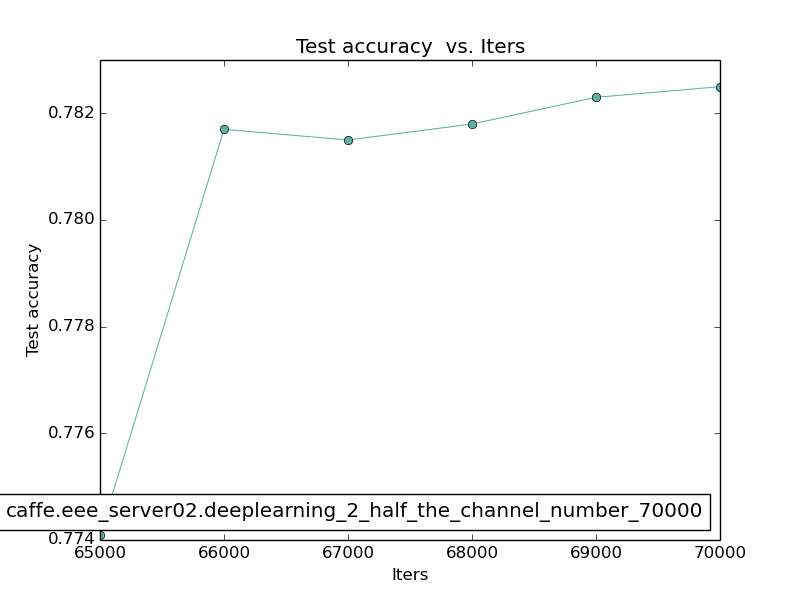
Caused by the randomness of experimental data, each experiment can get the unique experimental result. Compared with line chart, the result of test accuracy has a little bit difference because of my mistake. The graph and result are not from the same experiment. However, the difference is negligible that cannot affect the experimental result.

**Results and analysis**

1. Half the channel number in each layer



**Figure 3-1 The test accuracy from 0 to 60000 (left) and from 60000 to 65000 (Half the channel number in each layer)**

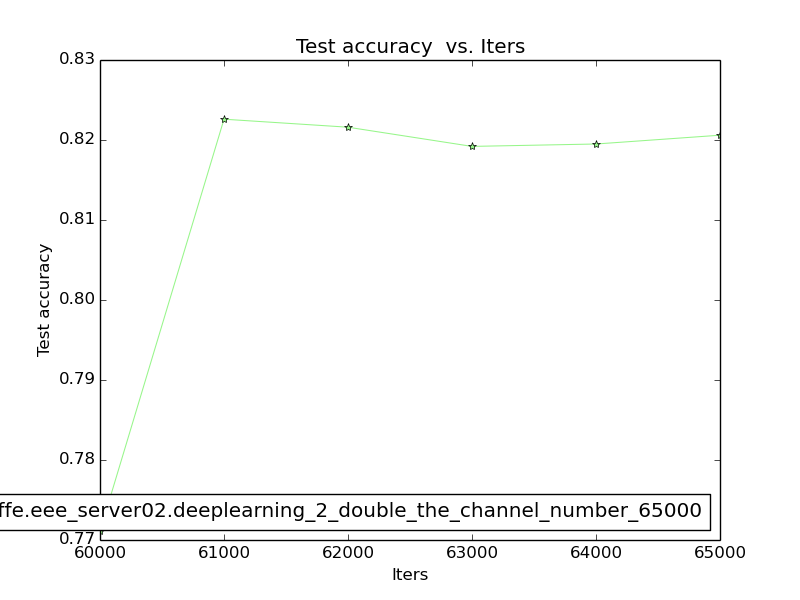
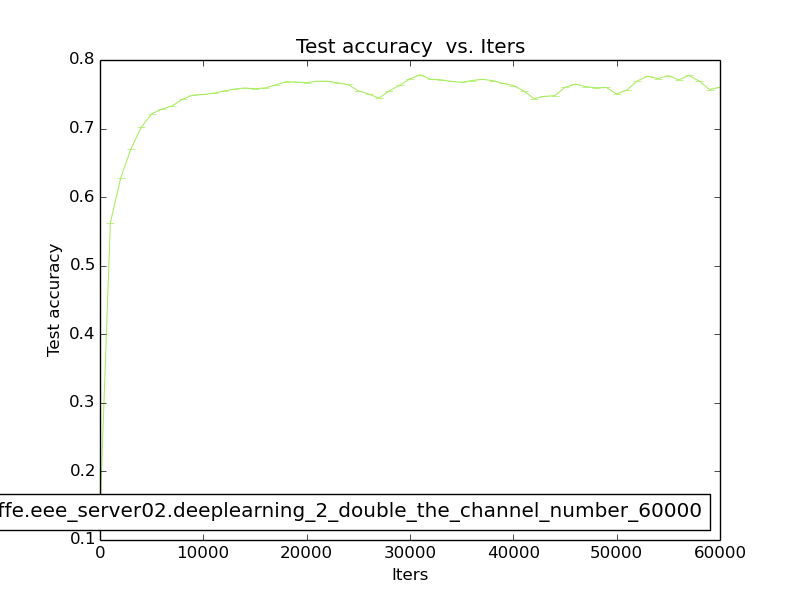


**Figure 3-2 The test accuracy from 65000 to 70000 (Half the channel number in each layer)**

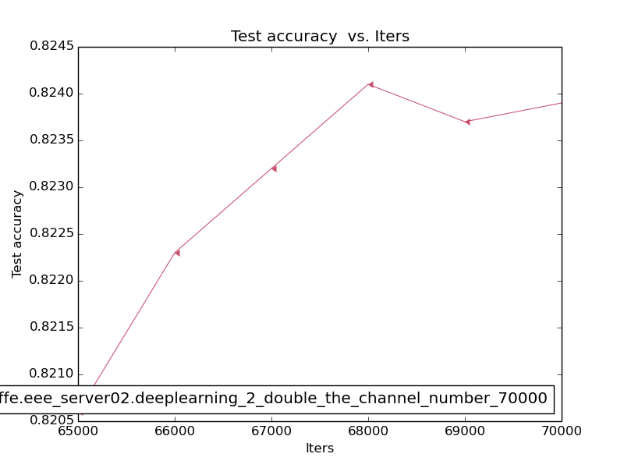


**Figure 3-3 The result of cifar-10 (Half the channel number in each layer)**

1. Double the channel number in each layer



**Figure 3-4 The test accuracy from 0 to 60000 (left) and from 60000 to 65000 (Double the channel number in each layer)**

****

**Figure 3-5 The test accuracy from 65000 to 70000 (Double the channel number in each layer)**



**Figure 3-6 The result of cifar-10 (Double the channel number in each layer)**

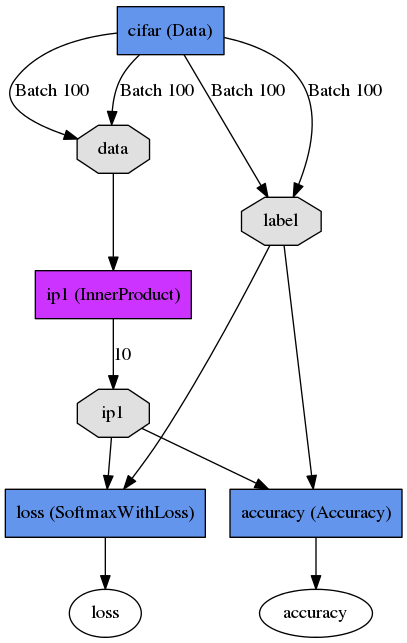
**Table 3-1 The results of different number of channels**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Half the channel number in each layer | Original channel number in each layer | Double the channel number in each layer |
| Accuracy | 0.7845 | 0.8069 | 0.8188 |
| Loss | 0.645845 | 0.596766 | 0.593456 |

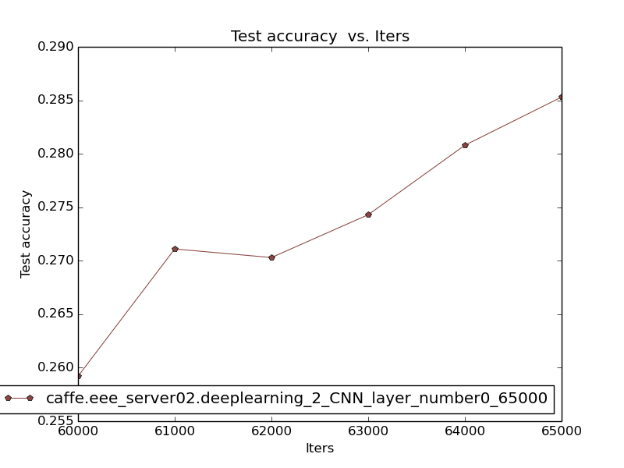
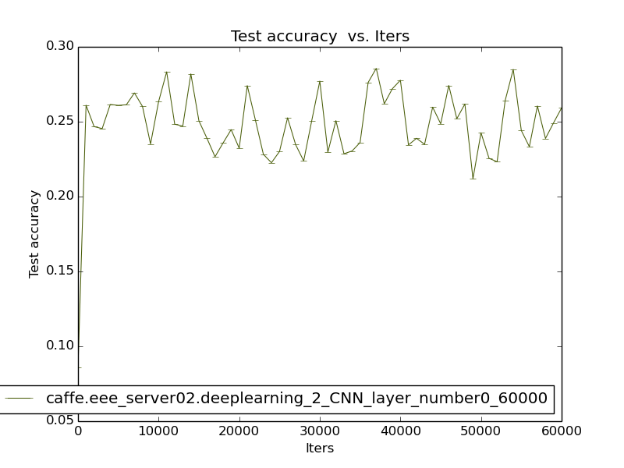
Compared half the channel number with double the channel. It is obviously shows that they can work normally, and they are all have the similar tendency caused by the same network structure. According to the figures, which from figure 3-1 to figure 3-6 and table 3-1. Compare with original number of channel, the accuracy of double number of channel is higher than other methods. Maybe improve the number of channel in each layer is one of the effective measure to improve the accuracy.

1. Change to CNN layer number from 3 to 0 and 1

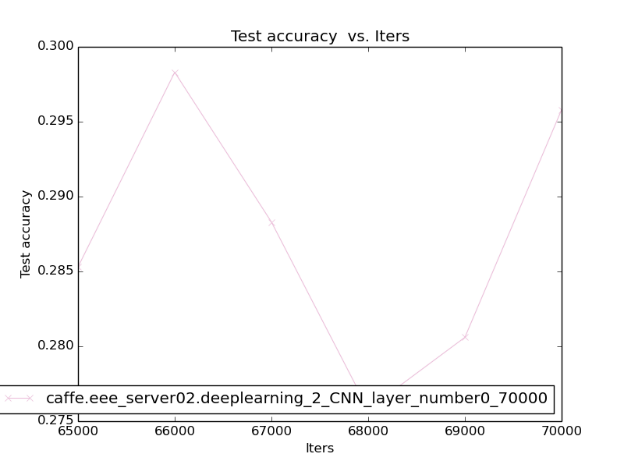
When the CNN layer number equals to 0



**Figure 3-7 The network structure of the CNN layer number equals to 0**

****

**Figure 3-8 The test accuracy from 0 to 60000 (left) and from 60000 to 65000 (CNN layer number equals to 0)**

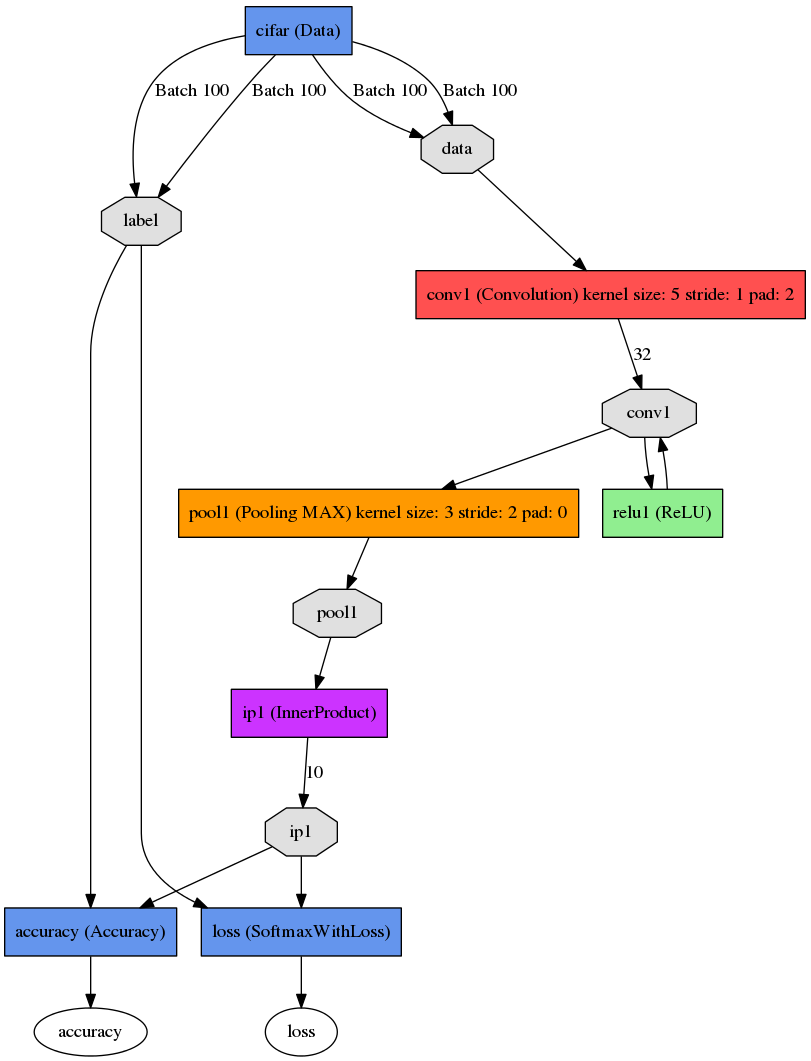
****

**Figure 3-9 The test accuracy from 65000 to 70000 (CNN layer number equals to 0)**

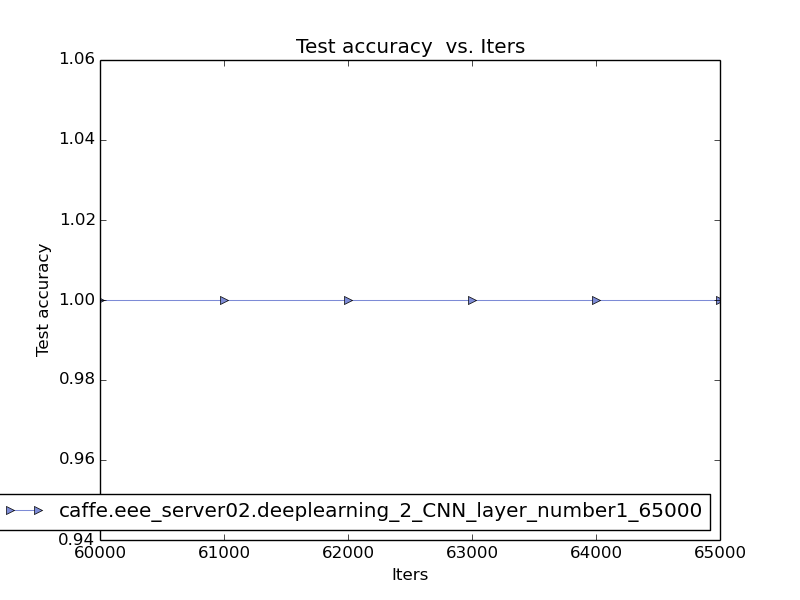
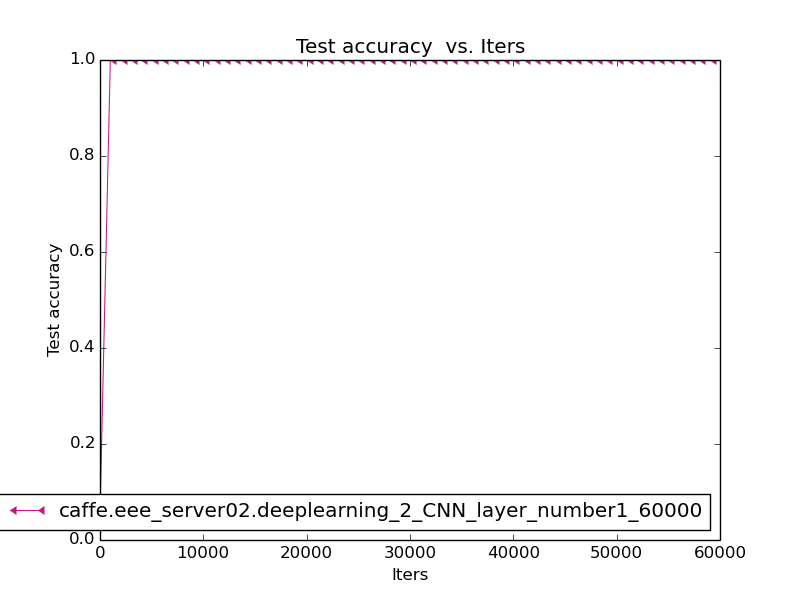
****

**Figure 3-10 The result of cifar-10 (CNN layer number equals to 0)**

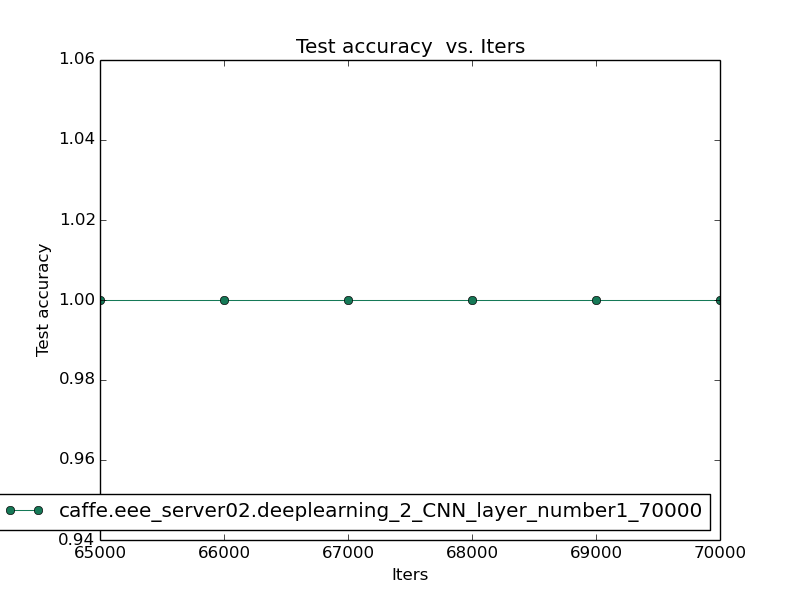
When the CNN layer number equals to 1

****

**Figure 3-11 The network structure of the CNN layer number equals to 1)**

****

**Figure 3-12 The test accuracy from 0 to 60000 (left) and from 60000 to 65000 (CNN layer number equals to 1)**

****

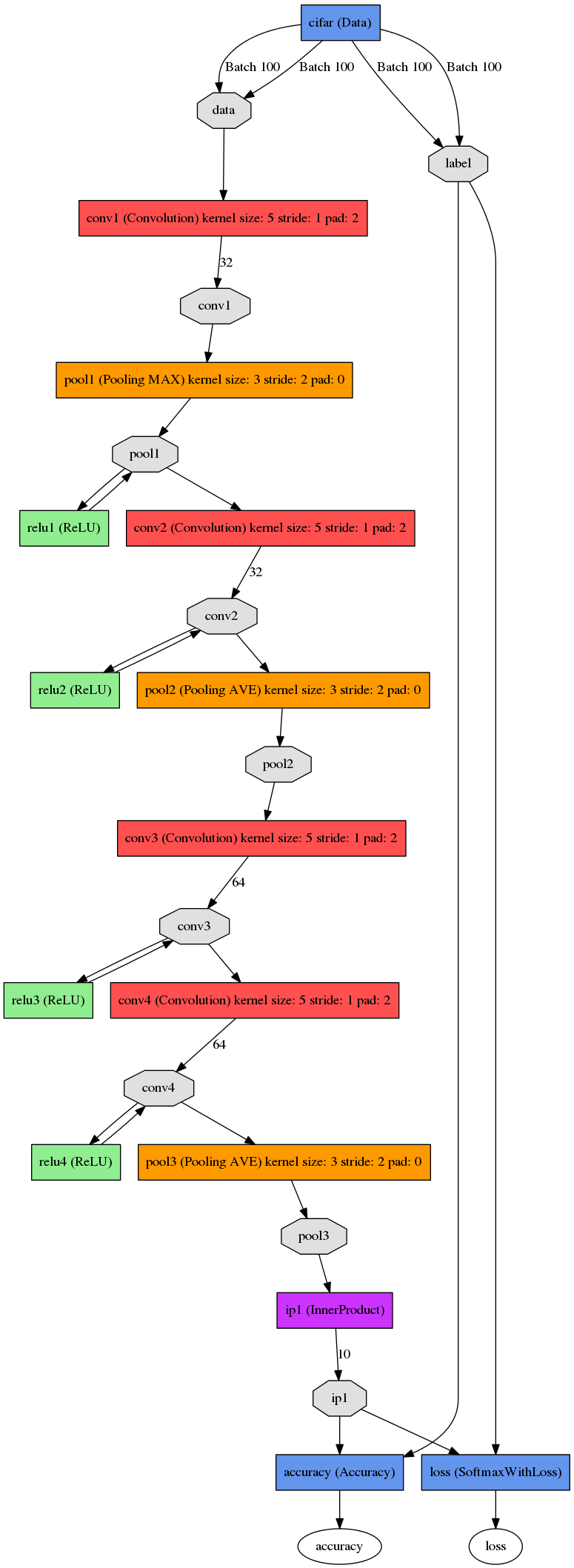
**Figure 3-13 The test accuracy from 65000 to 70000 (CNN layer number equals to 1)**

****

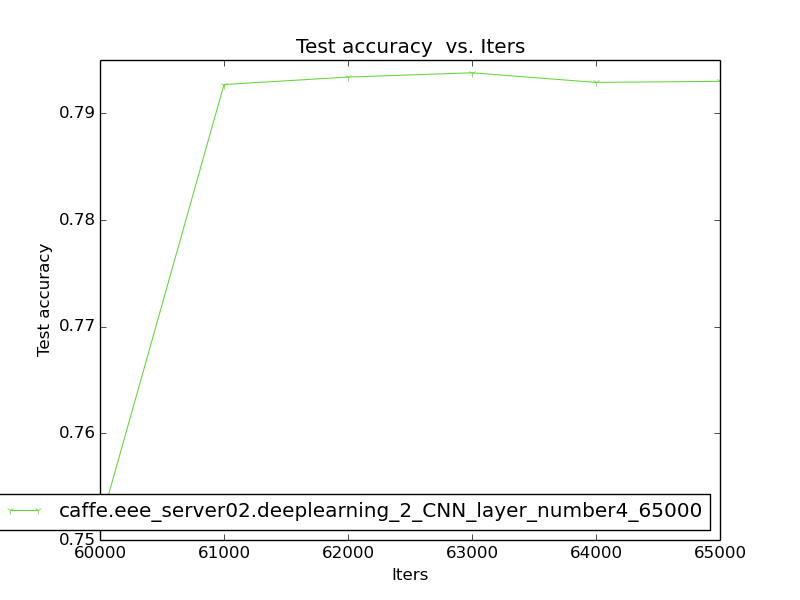
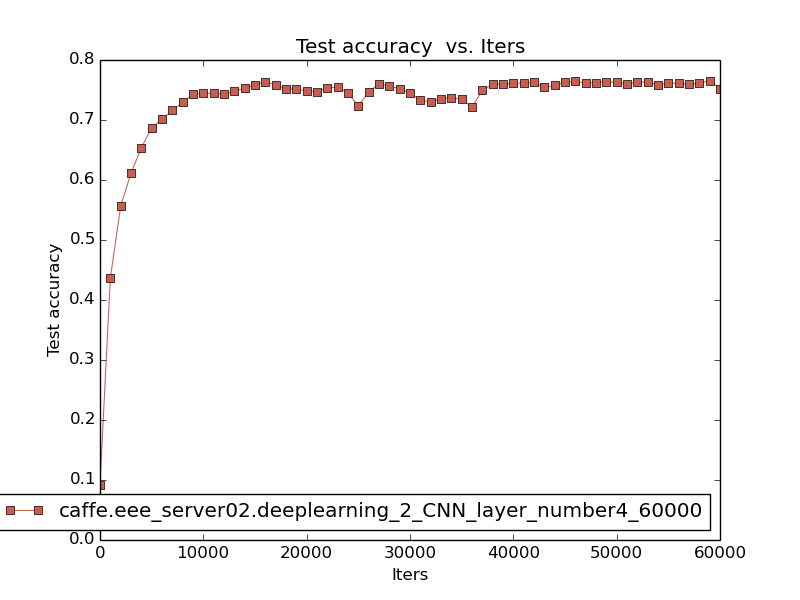
**Figure 3-14 The result of cifar-10 (CNN layer number equals to 1)**

1. Change to CNN layer number from 3 to 4 and 5

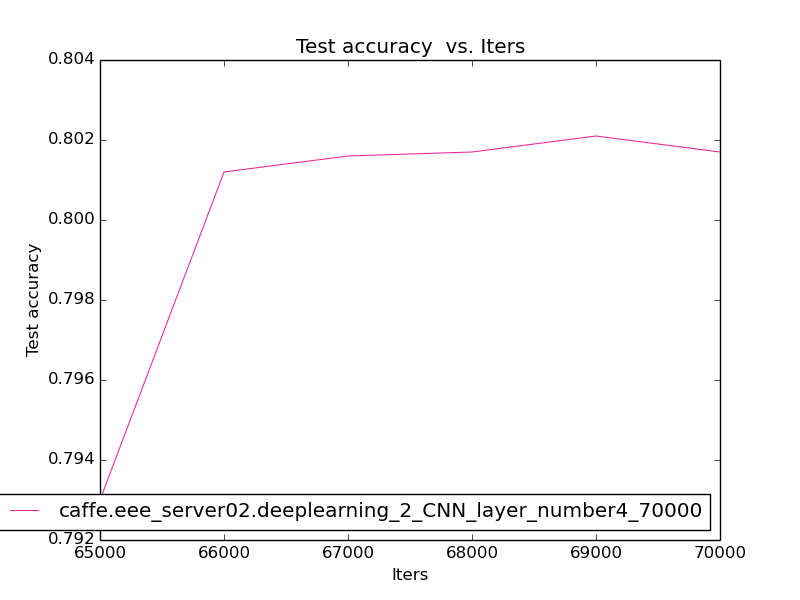
When the CNN layer number equals to 4



**Figure 3-15 The network structure of the CNN layer number equals to 4)**

****

**Figure 3-16 The test accuracy from 0 to 60000 (left) and from 60000 to 65000 (CNN layer number equals to 4)**

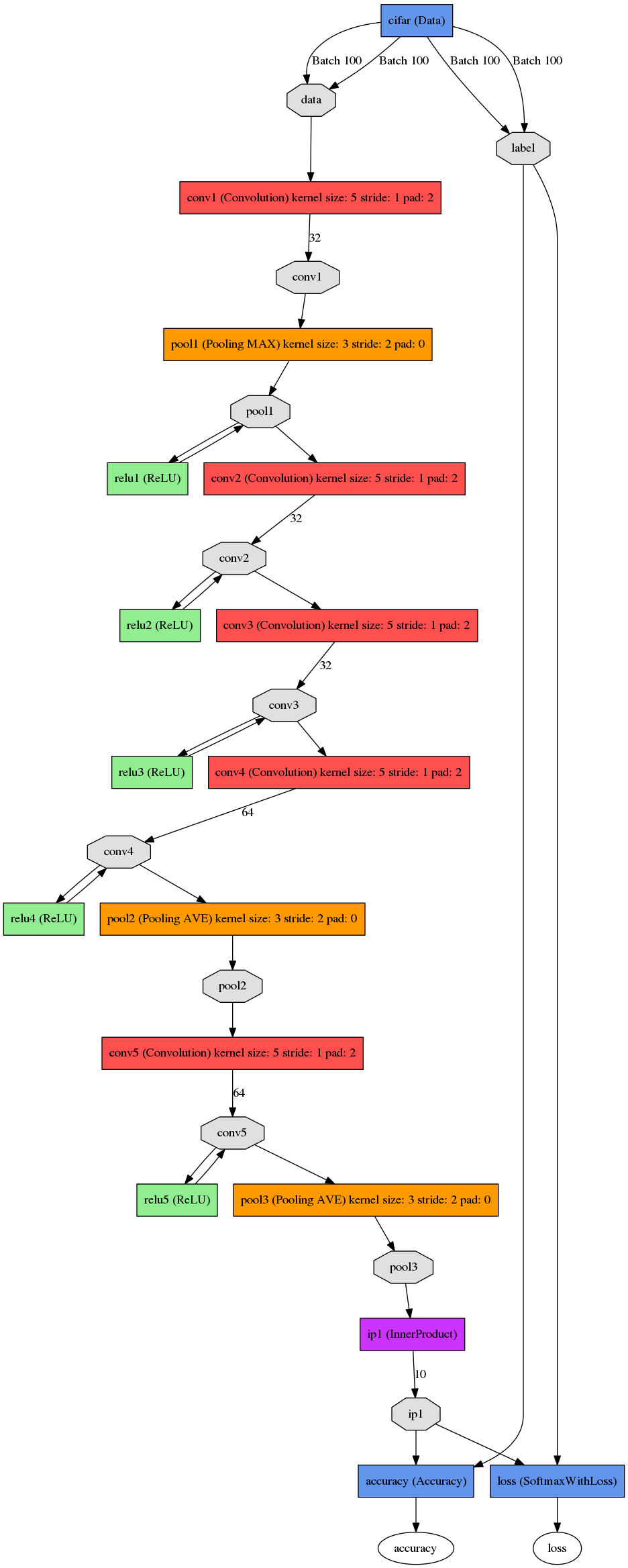
****

**Figure 3-17 The test accuracy from 65000 to 70000 (CNN layer number equals to 4)**

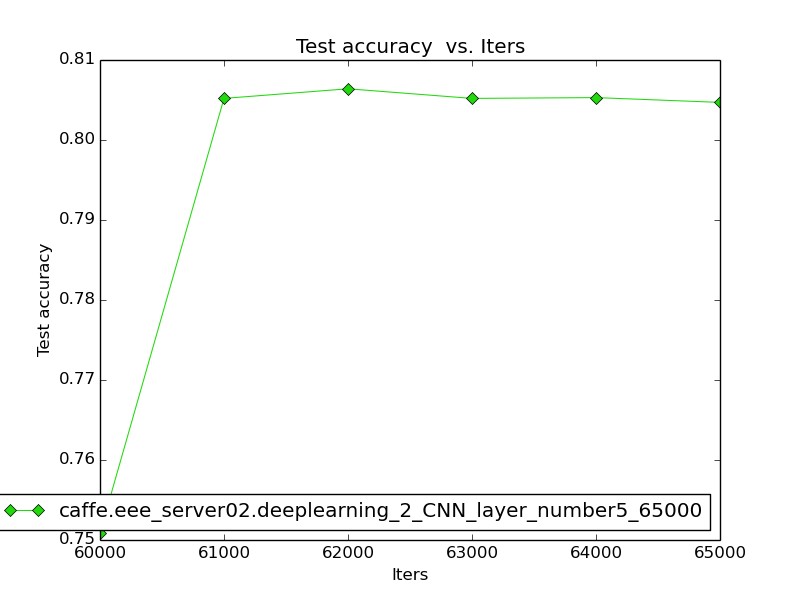
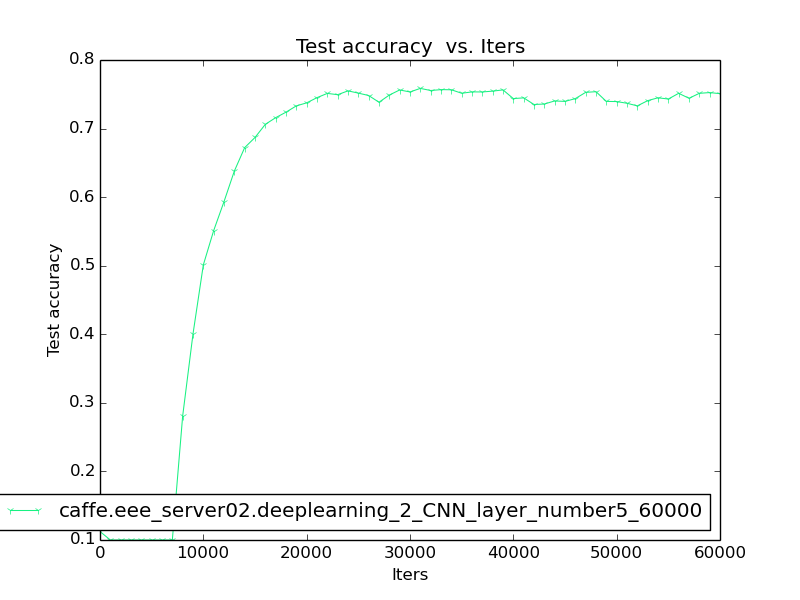
****

**Figure 3-18 The result of cifar-10 (CNN layer number equals to 4)**

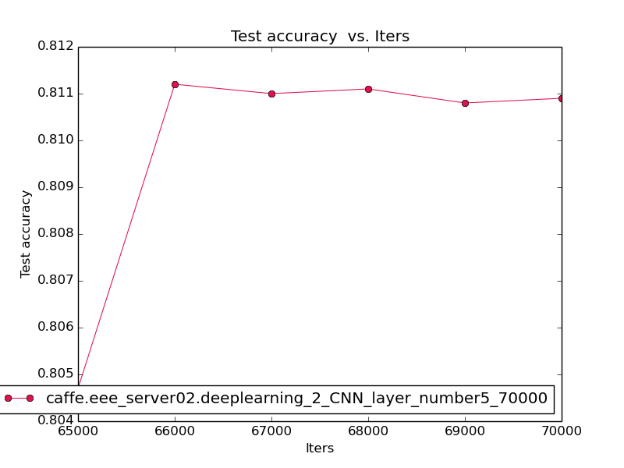
When the CNN layer number equals to 5

****

**Figure 3-19 The network structure of the CNN layer number equals to 5)**

****

**Figure 3-20 The test accuracy from 0 to 60000 (left) and from 60000 to 65000 (CNN layer number equals to 5)**

****

**Figure 3-21 The test accuracy from 65000 to 70000 (CNN layer number equals to 5)**

****

**Figure 3-22 The result of cifar-10 (CNN layer number equals to 5)**

**Table 3-2 The results for different number of CNN layer (from 3 to 0, 1, 4 and 5)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | CNN layer number equals to 0 | CNN layer number equals to 1 | CNN layer number equals to 3 | CNN layer number equals to 4 | CNN layer number equals to 5 |
| Accuracy | 0.2731 | 1 | 0.8069 | 0.7901 | 0.8093 |
| Loss | 13.1993 | 87.3365 | 0.596766 | 0.771032 | 0.721898 |

There are two questions, which from (c) to (d) cannot change the values of each layer. The table 3-2 clearly shows that the number of layer can get the different accuracy and loss. When the number of layer from 3 to 0, the accuracy has significantly decreased. Because, when the data turn into the model, there are not any layer to process all of the data. So, the whole process for this experiment is just a kind of random assortment. This process also affect the line graph that makes the line have abundant fluctuation as shown in the left of figure 3-8. When the number of layer from 3 to 1, the accuracy becomes to 1. The figure 3-12 and figure 3-13 shows that the tendency will become a straight line. Because there is overfitting. So, this model can only just have used for this dataset. When the number of layer from 3 to 4 and 5, the result of that are similar as original model. According to this part that previously shows that reduce the number of layer cannot improve the accuracy and then, it might even bring about the overfitting. However, if we optionally add the number of layer that cannot improve the accuracy. In other words, although it is a kind of possible way to improve the accuracy, we also need carefully add some layer to original model.

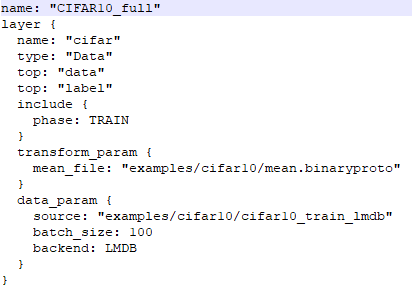
**Task 4**

**Problem description**

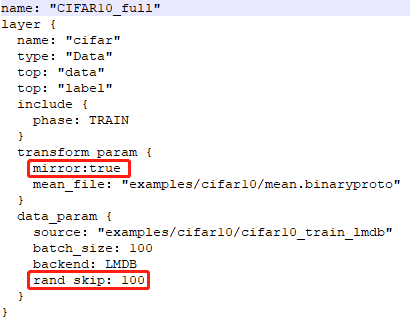
“Data augment technique is commonly used for the deep learning training process. Please propose a data augment method, and train the CNN network specified in “cifar10\_full\_train\_test\_no\_LRN.prototxt” with the augmented dataset. Please write down the adopted data augment method and record the final test accuracy. (20%)”

**Experimental process**

In this task, there are two common methods that named mirroring and random skip has been used by my me as shown in the figure 4-1 and figure 4-2.

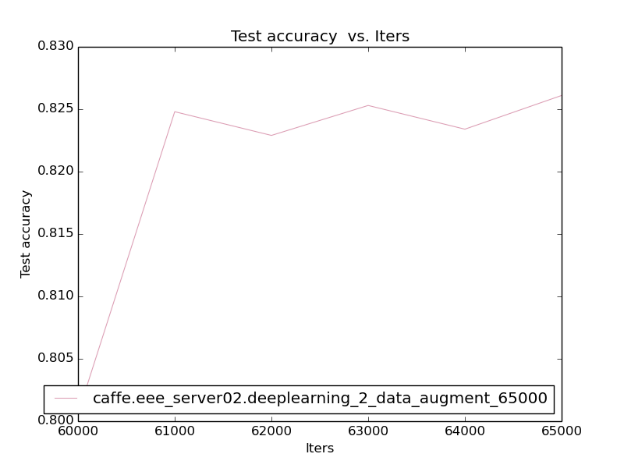
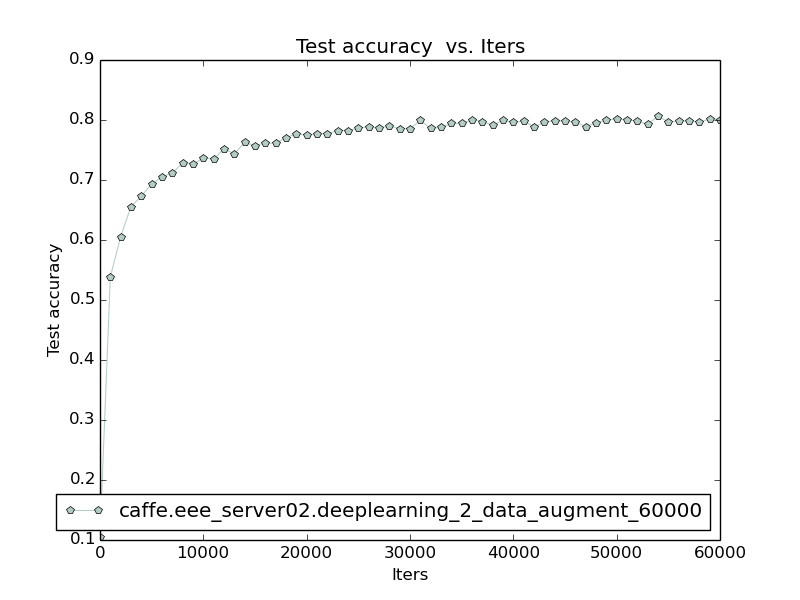
****

**Figure 4-1 The original first layer of cifar-10**

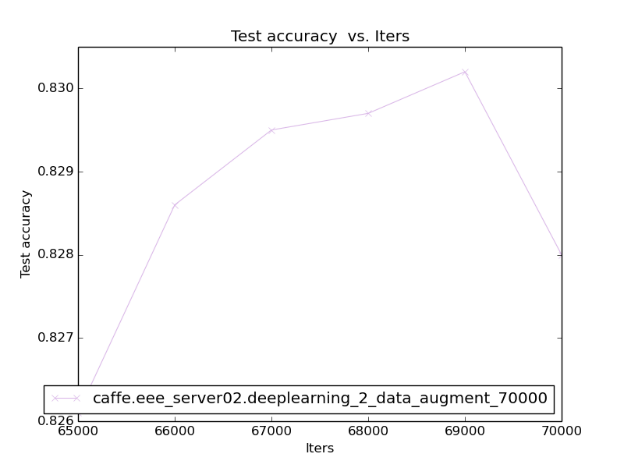
****

**Figure 4-2 Add the mirroring and random skip in the first layer of cifar-10**

**Results and analysis**

****

**Figure 4-3 The test accuracy from 0 to 60000 (left) and from 60000 to 65000 (Data enhancement)**

****

**Figure 4-4 The test accuracy from 65000 to 70000 (Data enhancement)**

****

**Figure 4-5 The result of cifar-10 (Data enhancement)**

The figure 4-3, figure 4-4 and figure 4-5 shows the accuracy has previously improvement. The mirroring is a basic method of data augment technique, which can prevent overfitting. This method can increase the sample size that have certain effect for improve the accuracy according to this lab. The random skip is randomly skip some files that also can help to improve the accuracy.

**Task 5**

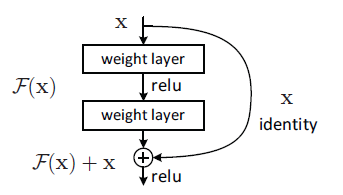
**Problem description**

“Please propose a method to boost the image classification performance, and detail your method and report the final image classification performance that your achieved. (20%)”

**Experimental process**

For this task, I used the ResNet model for our dataset according to consult the references.

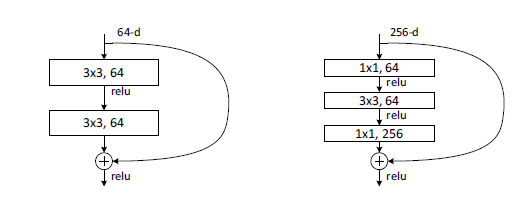
I changed a little bit value that makes this model to be similar with this lab. ResNet was put forward in 2015 and it won the first place in the classification part which competition is belongs to ImageNet. The structure of PlainNet is mainly based on VGG modification and the main structure of the ResNet is the same as PlainNet. Compare with PlainNet, ResNet just have much more connections of shortcut. The whole ResNet can be seen that was stacked by many residual block stacks according to the shortcut that as shown in the figure 5-1.



**Figure 5-1 Shortcut connection**

The figure 5-2 shows that different design of shortcut for ResNet-34 (left) and ResNet-50/101/152 (right). Generally, the structure was called "building block". Among of that, the right can also have named “bottleneck design”. The function of bottleneck design is reduced the number of parameters. The first convolution () make the dimension of channel to reduce from 256 to 64. And then, it will recover to original dimension according to another convolution . The whole number of parameters is . However, if that cannot use the bottleneck, the whole number of parameters is .

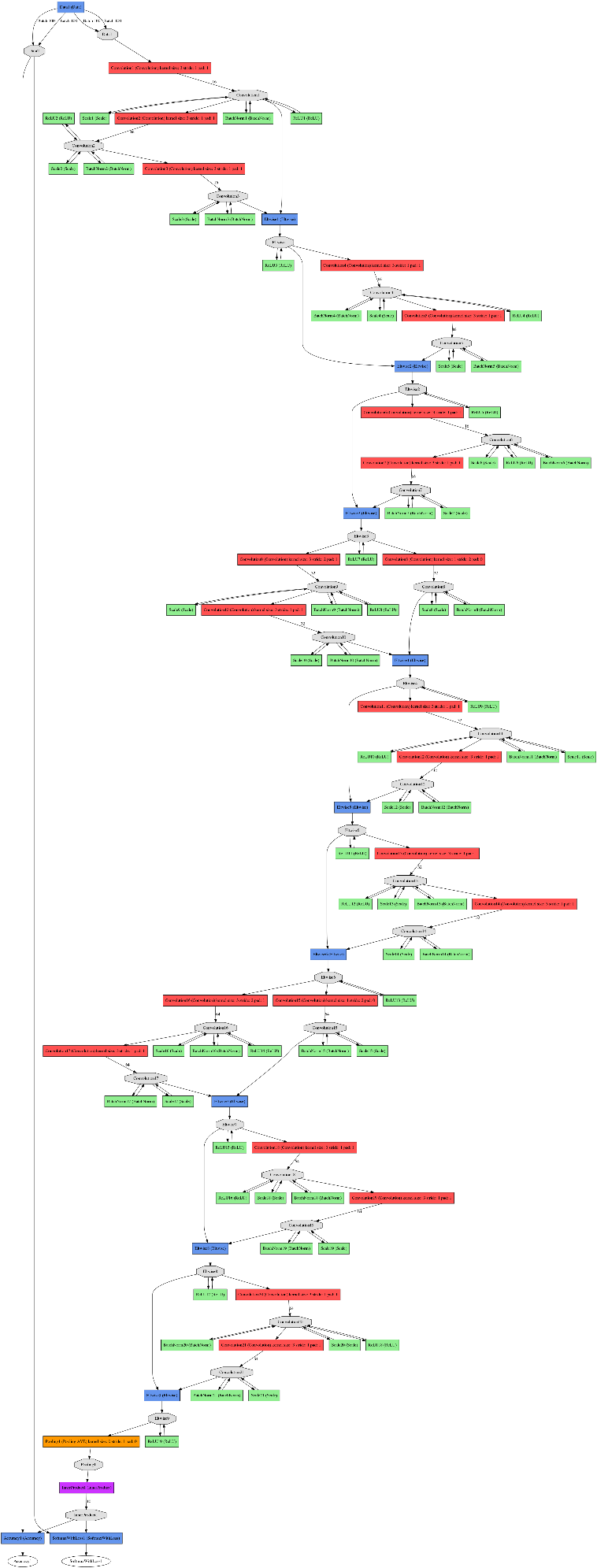
For conventional ResNet, it can be used in 34 layer or other networks with less layer. The bottleneck design is usually used in deeper networks such as 101 layer, the purpose of that is to reduce the amount of calculation and parameters

****

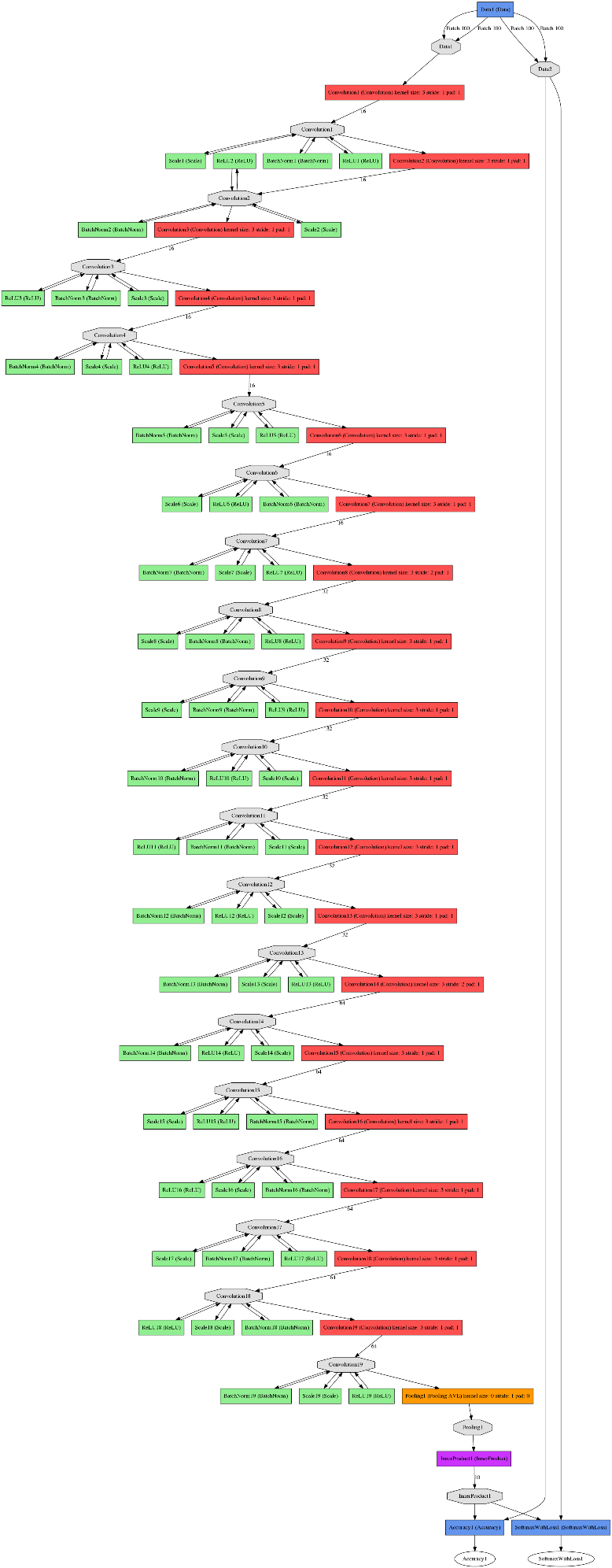
**Figure 5-2** **Building block**

**Results and analysis**

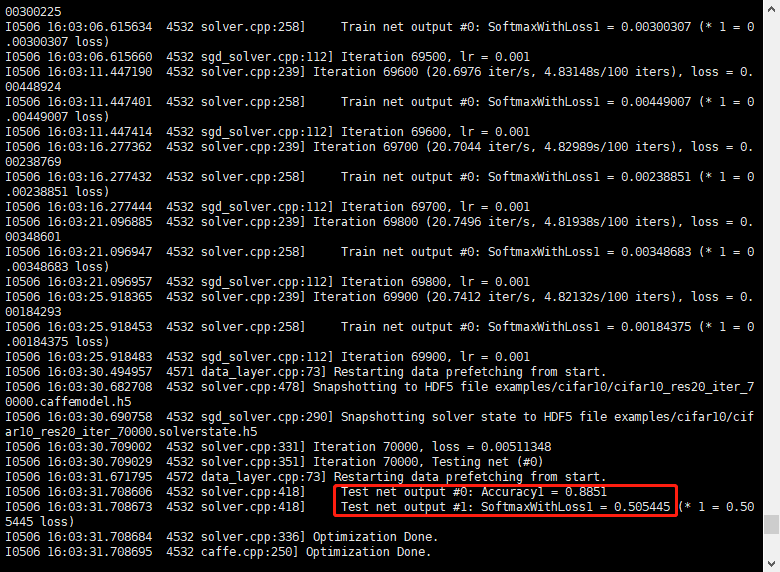
According to the figure 5-3 and figure 5-4, that shows the similar suture of ResNet and PlainNet. Compare with original structure of this lab, the accuracy of these two methods has significantly improved



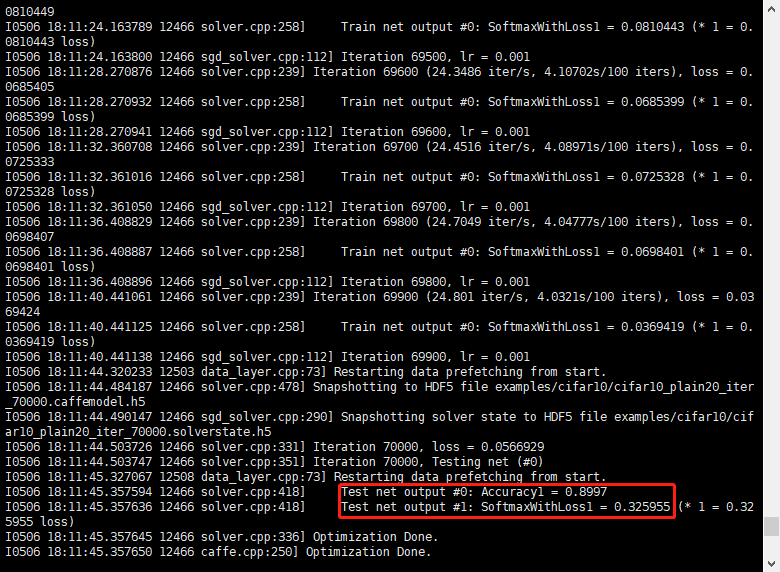
**Figure 5-3 The network structure of the ResNet (20 layers)**



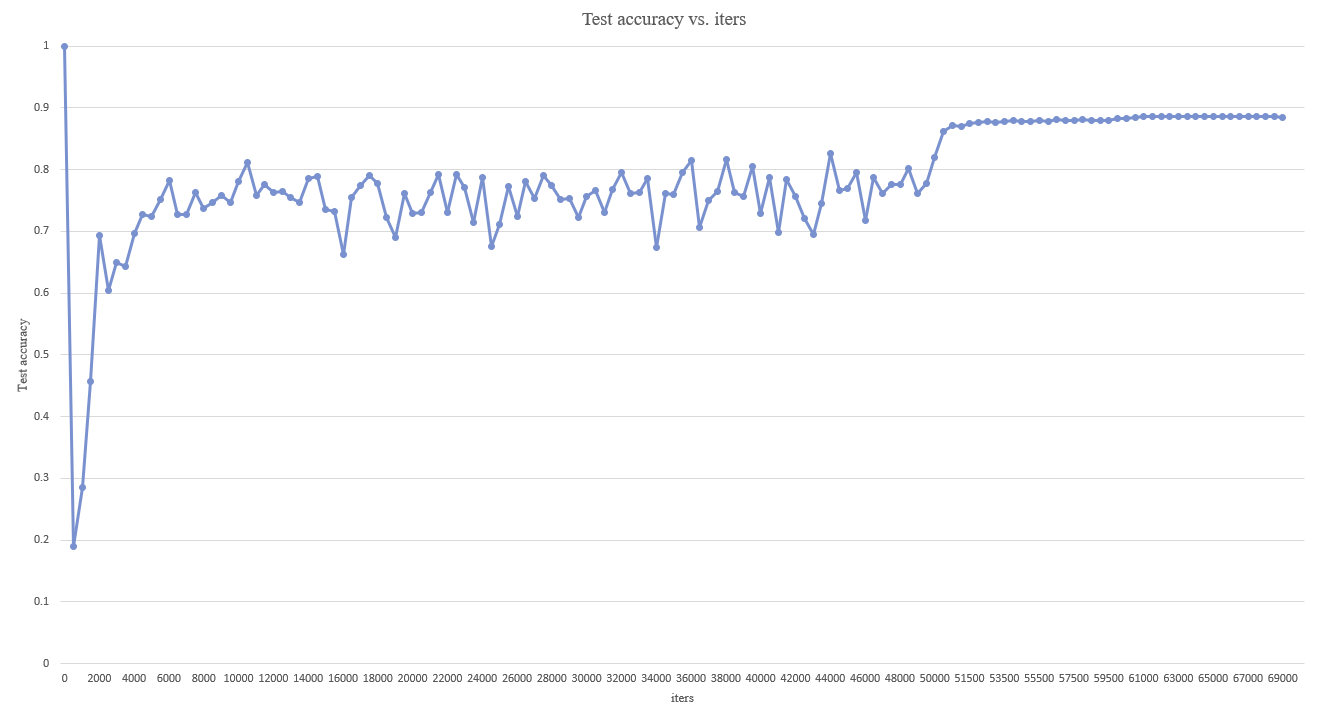
**Figure 5-4 The network structure of the PlainNet (20 layers)**



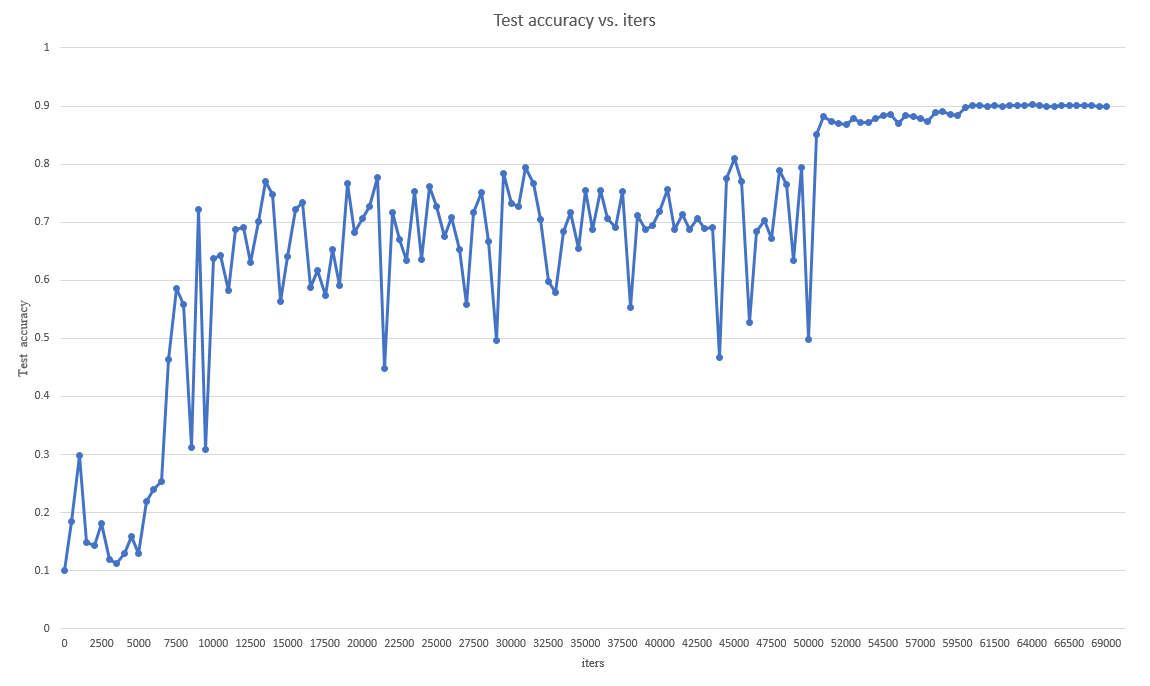
**Figure 5-5 The result of ResNet (20 layers)**

****

**Figure 5-6 The result of PlainNet(20 layers)**

****

**Figure 5-7 The test accuracy from 0 to 70000 (ResNet)**

****

**Figure 5-8 The test accuracy from 0 to 70000 (PlainNet)**

**Conclusion**

This lab helps students to clearly understand the Convolutional Neural Network (CCN), And then, compared different number of CCN layer according to the cifar-10 dataset. Thirdly, this lab requires students to improve the accuracy by the data augment technique. Finally, the last task make students understand different models to improve the accuracy.

**Reference**

(He et al., 2015)HE, K., ZHANG, X., REN, S. & SUN, J. 2015. Deep Residual Learning for Image Recognition. 770-778.