Rules of Homework 2

* **You need to hand in your python program and separate it into different folder. Make sure every program is compilable!!!（Ex:1A、1B、2A、2B…）。**
* **If question B needs the question A’s code, please just copy question A into question B to make it correct.**
* **If you need to create a save model or logs file, make sure to use relative path instead of full path!**

1. Deep Learning
2. In question A, you need to define a function to create a DNN model.

The model needs to have **five hidden layers.**

You don’t need to print out anything, but make sure the code is compilable.

Scoring standard: Correctness of DNN model.

1. In question B, you need to train a model using question A’s DNN model and MNIST dataset **on digits 0 to 4**. You can get the digit 0-4 by using following code:

X\_train1 = mnist.train.images[mnist.train.labels < 5]

y\_train1 = mnist.train.labels[mnist.train.labels < 5]

X\_valid1 = mnist.validation.images[mnist.validation.labels < 5]

y\_valid1 = mnist.validation.labels[mnist.validation.labels < 5]

X\_test1 = mnist.test.images[mnist.test.labels < 5]

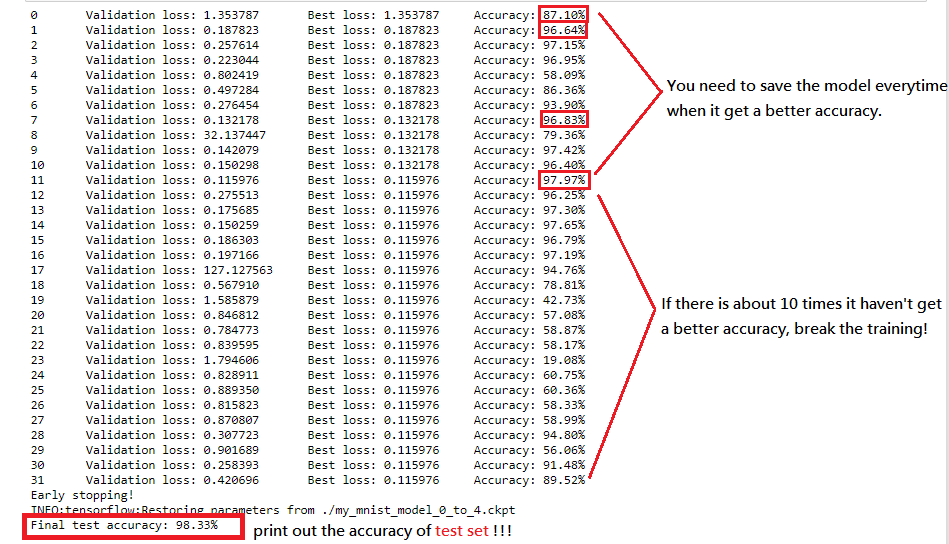
y\_test1 = mnist.test.labels[mnist.test.labels < 5]

Make sure you use the validation set to evaluate the training process!

You need to save the model as the best model when you get a better accuracy with validation set, and stop the training process when there are at least 10 epoch that the validation accuracy can’t be better than the best model.

Please print out your learning process include the accuracy of validation set, you need to print out the accuracy of test set in the end of the training process.

For example:



Scoring standard: Show the training process and the accuracy.

1. In question C, you need to find out the best hyperparameters using cross-validation!

You can choose several candidates with changing these parameters:

Number of neurons: [10, 30, 50, 70, 90, 100, 120, 140, 160]

batch size: [10, 50, 100, 500]

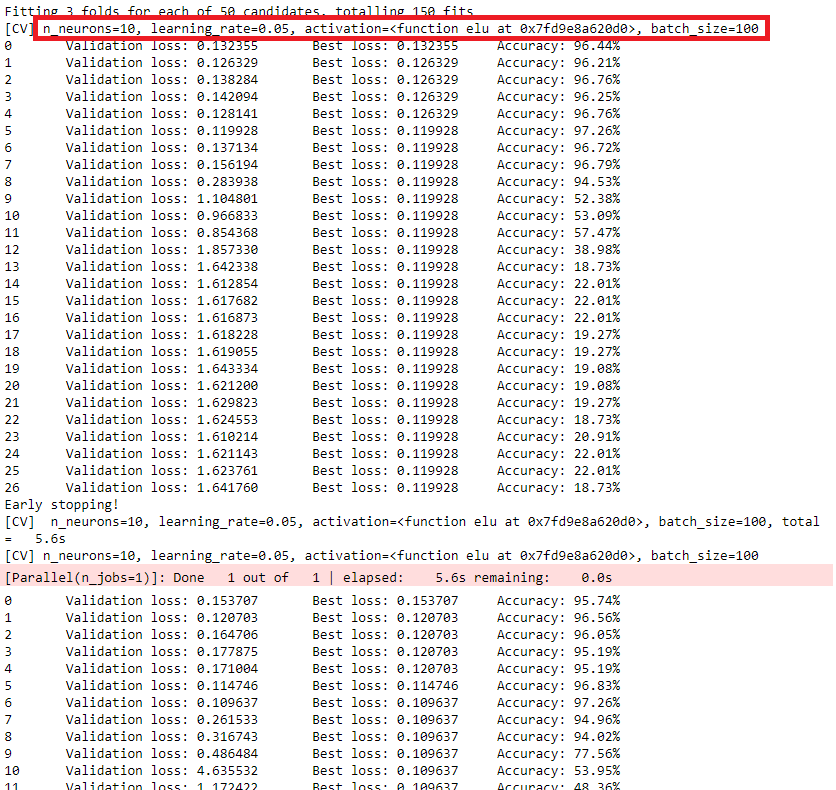
learning rate [0.01, 0.02, 0.05, 0.1]

activation function :[ relu, elu, leaky\_relu, tanh]

Train the model with your chosen parameters, and use cross-validation to estimate the accuracy rate of each candidates. Takes 3-flod cross-validation for example: You need to separate your train set into 3 part, and use first part, second part, third part as validation set, respectively.

That means you need to train 3 times per every candidates and save the best model as result. Therefore, you can just choose about 5 candidates to compare their accuracy rate instead of trying every possible hyperparameters combination( or you will need to train the model at least 9\*4\*4\*4\*3 times! Don’t waste you time :D).

You need to print out your training process (which should be really long) and print the hyperparameters you used as following:



When all the training is done, print out the best hyperparameters combination and the accuracy of test set using that model.

For example:



Scoring standard: The cross-validation process, the chosen hyperparameters, and the accuracy.

1. In question D, use the best hyperparameters combination to build the DNN model and try to add Batch Normalization layer after every hidden layer and see what the learning rate will become!

You need to run two training process with and with out batch normalization method and show the learning curve (using loss value or accuracy) on the Tensorboard.

The graph of learning curve will be like



Scoring standard: Learning curve between two model. Using tensorboard (50% discount if you didn’t).

1. In question E, first you need to find out whether the model is overfitting or not.

First, train the model by the previous question., After the training process is done, print out the accuracy of both train set and test set.

For example:



You can figure out that the training accuracy is 99.92% when the testing accuracy only have 99.16%. Now that is the overfitting problem.

Next step, add drop out method into your DNN model and also print out the accuracy of both train set and test set.

For example:



You can see the difference between two accuracy become smaller. But if you want to get a better accuracy, you should do cross-validation again to find out a better hyperparameters combination. However, in this question, you can just add the drop out method and print out the accuracy.

Scoring standard: The accuracy of train and test set before/after adding drop out to the model.

1. Transfer learning
2. In question A, you need to reload one of the model you save at question 1. You can use “tf.train.import\_meta\_graph” to reload the graph and “get\_collection” to get the five hidden layer from the reloaded model. Make sure to keep only the output layer's trainable variables! You don’t need to print out anything in this question.

Scoring standard: Whether you reload the previous model and do not make the weight and bias of 5 hidden layer trainable.

1. In question B, you need to train the model using question A’s DNN model and MNIST dataset **on digits 5 to 9**. You can get the digit 5-9 by using following code:

X\_train2\_full = mnist.train.images[mnist.train.labels >= 5]

y\_train2\_full = mnist.train.labels[mnist.train.labels >= 5] - 5

X\_valid2\_full = mnist.validation.images[mnist.validation.labels >= 5]

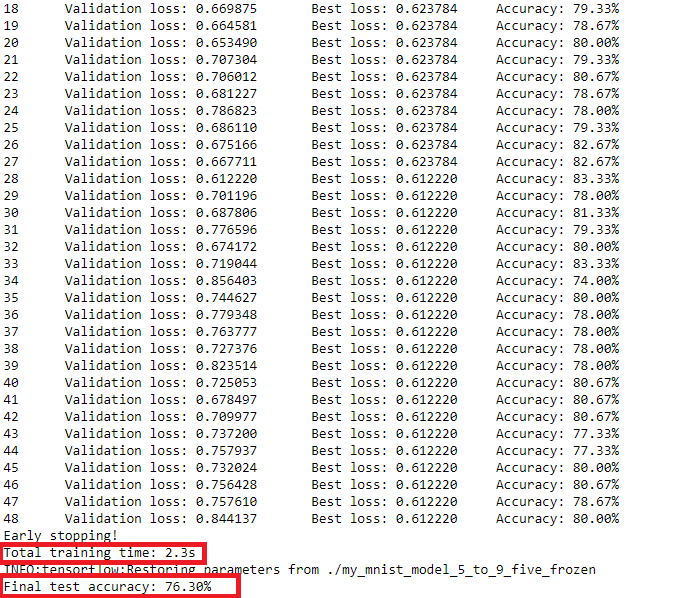
y\_valid2\_full = mnist.validation.labels[mnist.validation.labels >= 5] - 5

X\_test2 = mnist.test.images[mnist.test.labels >= 5]

y\_test2 = mnist.test.labels[mnist.test.labels >= 5] – 5

Then, start a training using the frozen model and print out the learning process and accuracy of the test set. You can print out all the global variable in your session before and after the training process to make sure 5 hidden layer’s weight and bias doesn’t change! Furthermore, please use time package to print out the training time in the end.

For example:

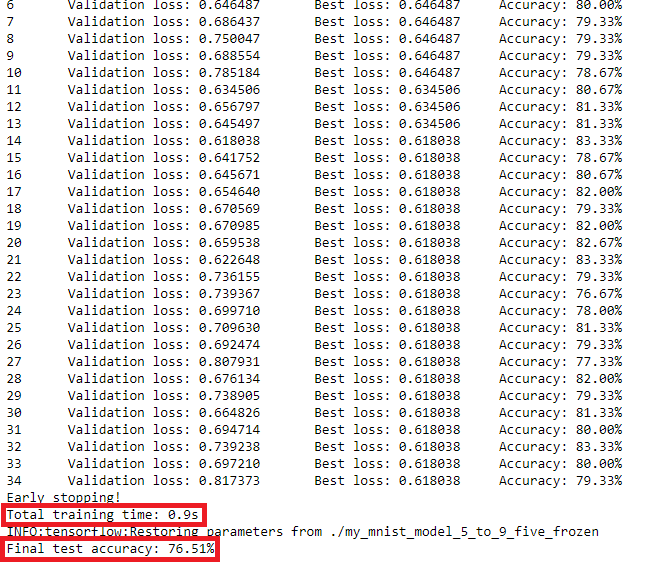


Scoring standard: The code of frozen model. Also the accuracy and time. Make sure frozen layers’ weight and bias doesn’t change!

1. In question C, we need to train the model using roughly the same code as earlier. The difference is that we compute the output of the top frozen layer at the beginning (both for the training set and the validation set), and we cache it.

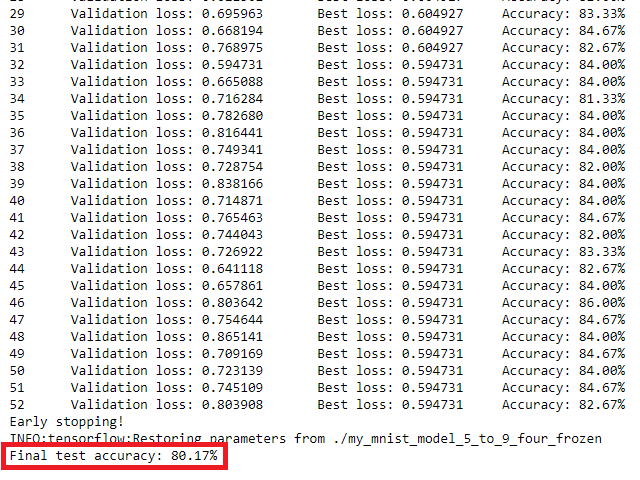
Also print out the learning process, accuracy and the training time in this question.

For example:



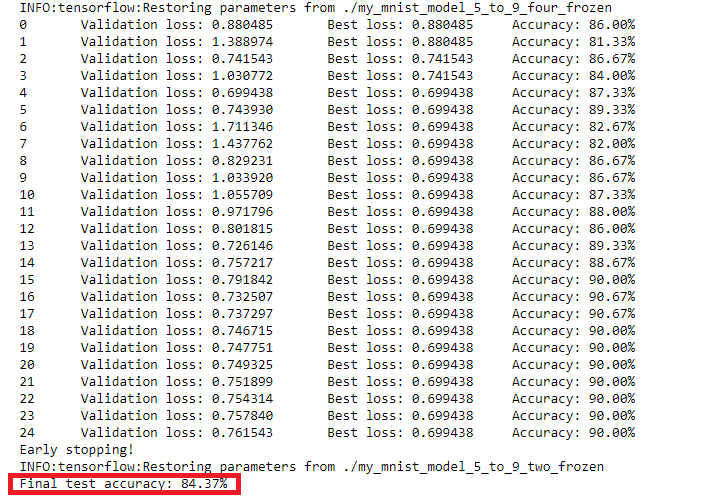
Scoring standard: The code of caching model. Also the accuracy and time.

1. In question D, just freeze four hidden layers and make the fifth one trainable! You need to print out the learning process and accuracy.



Scoring standard: The code of frozen model. Also the accuracy and time. Make sure frozen layers’ weight and bias doesn’t change!

1. In question E, just like question D but freeze 2 hidden layers and make others trainable! You need to print out the learning process and accuracy.



Scoring standard: The code of frozen model. Also the accuracy and time. Make sure frozen layers’ weight and bias doesn’t change!