CS498AML Applied Machine Learning Homework 8

Huamin Zhang (huaminz2), Rongzi Wang(rwang67)

May 11, 2017

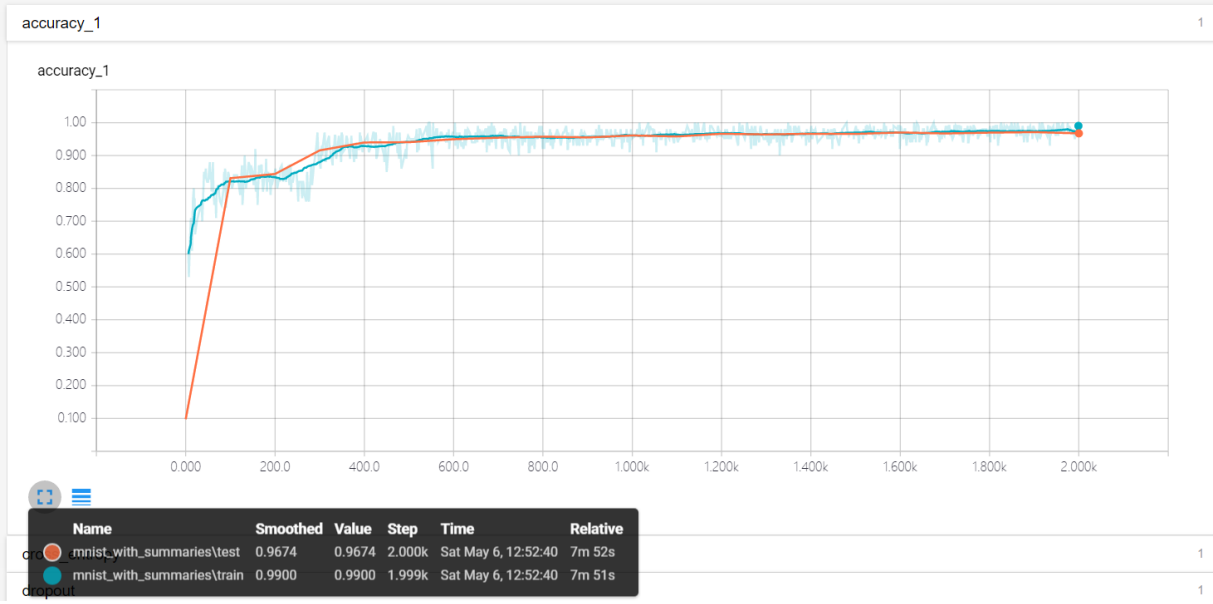
(Source code are included in the folder)

Question 1. MNIST Data Set

Part 1. Insert appropriate lines of code into the tensorflow example to log the accuracy on tensorboard every 100 batches, for at least 2000 batches. You should screen capture the accuracy graph from tensorboard, and submit this.

Answer:

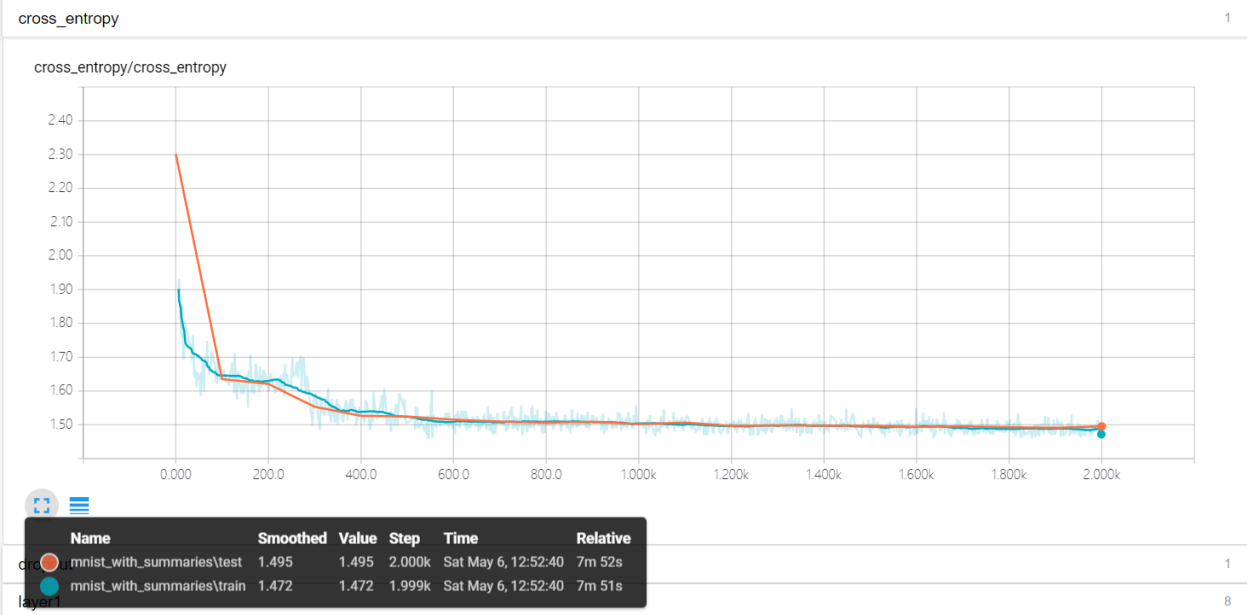
First, we use the tensorboard tutorial code which only include three layers: input layer, hidden layer and output layer. we log the accuracy on both training data and test data every 100 batches for 2000 batches in total. The accuracy and cross-entropy for every step are showed in Picture 1 and Picture 2.



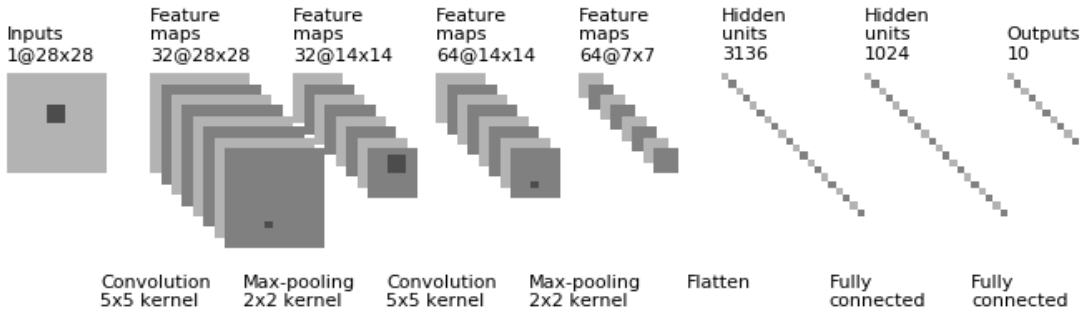
Picture 1. The accuracy on training data and test data for each step

The details about the accuracy on training data and test data are included in the folder. The average of the test data accuracy is 94.56% and the maximum of the test data accuracy is 97.11%.

Second, we use the "DEEP MNIST for experts" tutorial code and insert appropriate lines of code to log the accuracy. The network architecture is shown in Picture 3.



Picture 2. The cross-entropy on training data and test data for each step



Picture 3. The network architecture of CNN model in Question 1 Part 1

(Something that may be modified)

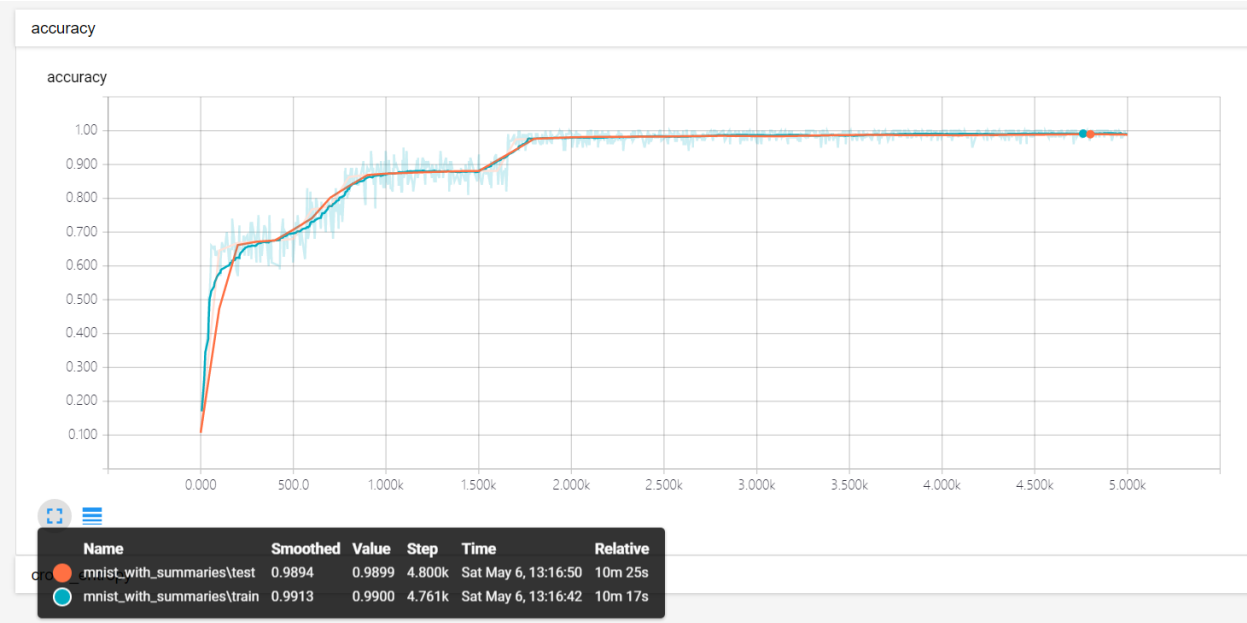
In this CNN model, we set the learning rate to be a constant 0.0001. Also, we use the method of drop-out to avoid overfit in the fully connected layer. The parameter of drop-out is set to be 0.9.

In the model, we calculate cross_entropy as the following:

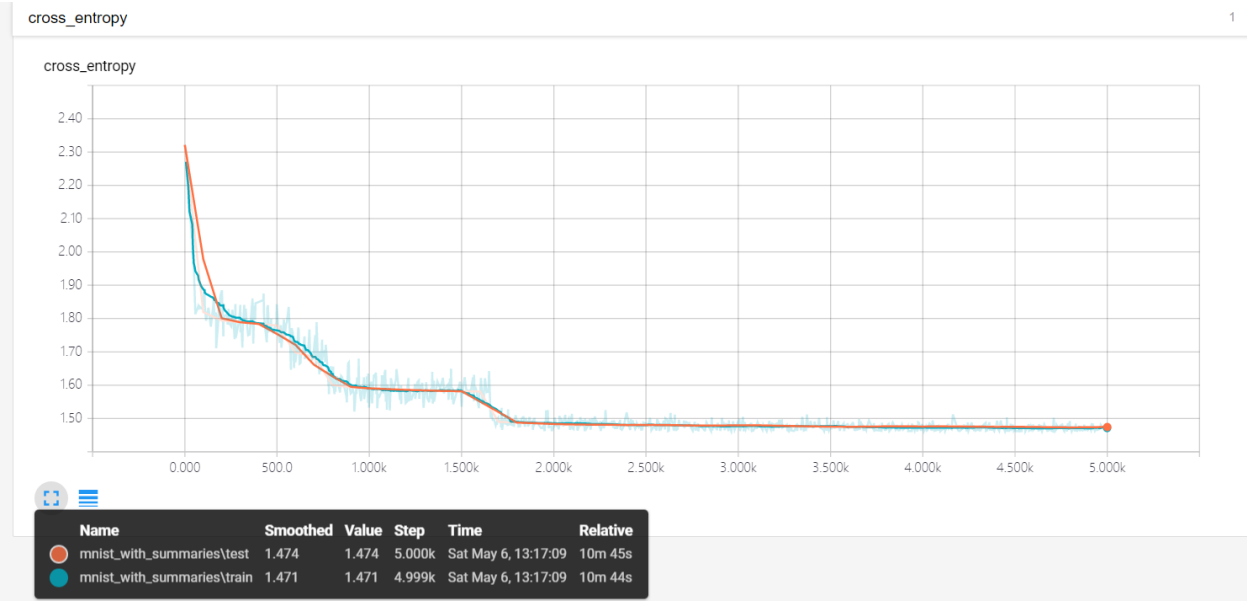
cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=y_, logits=y_conv))

Here we log the accuracy on both training data and test data every 100 batches for 5000 batches in total. The accuracy and cross-entropy on training data and test data for every step are showed in Picture 4 and Picture 5. The details about the accuracy on training data and test data are included in the folder. The average of the test data accuracy is 92.489% and the maximum of the

test data accuracy is 98.99%. Out of the 50 logs, there are 0 log have accuracy greater than 99%, 30 logs having accuracy greater than 98%.



Picture 4. The accuracy on training data and test data for each step

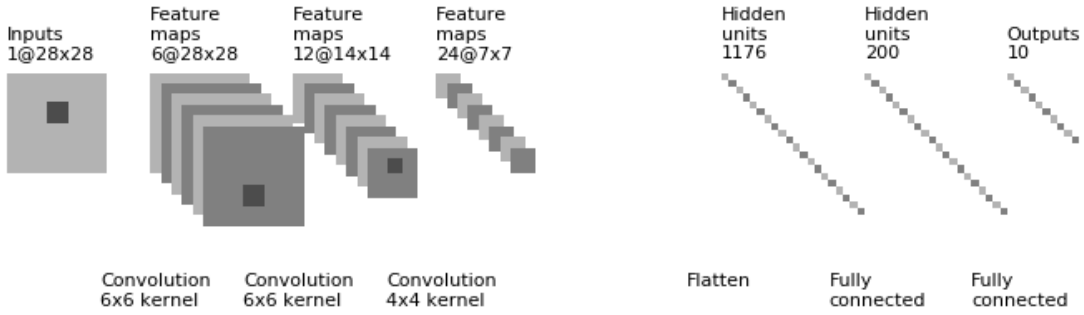


Picture 5. The cross-entropy on training data and test data for each step

Part 2. Modify the architecture that is offered in the DEEP MNIST tutorial to get the best accuracy you can Submit a screen capture of tensorboard graphs of accuracy.

Answer:

In this part, we modify the architecture. The network architecture is shown in Picture 6.



Picture 6. The network architecture of CNN model in Question 1 Part 2

(Something we modified)

In this CNN model, we set the learning rate to be a variable value that depend on the step number. Also, we use the method of drop-out to avoid overfit in the fully connected layer. The parameter of drop-out is set to be 0.75.

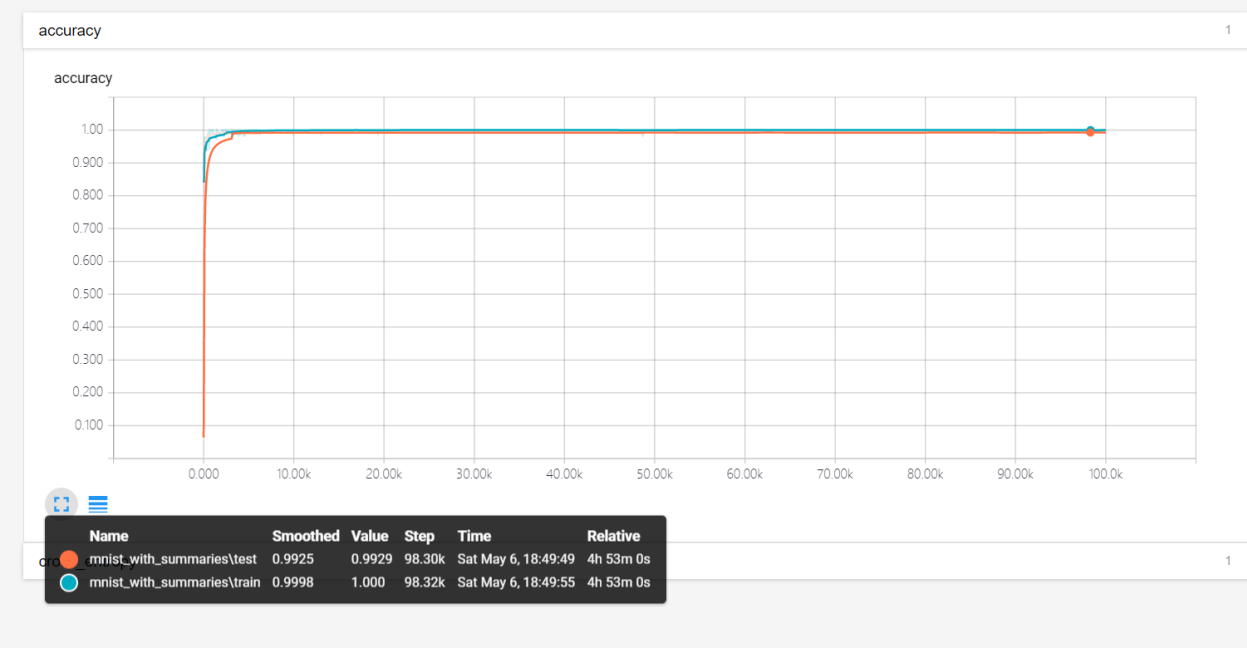
In the model, we calculate cross_entropy as the following:

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=Ylogits, labels=y_)

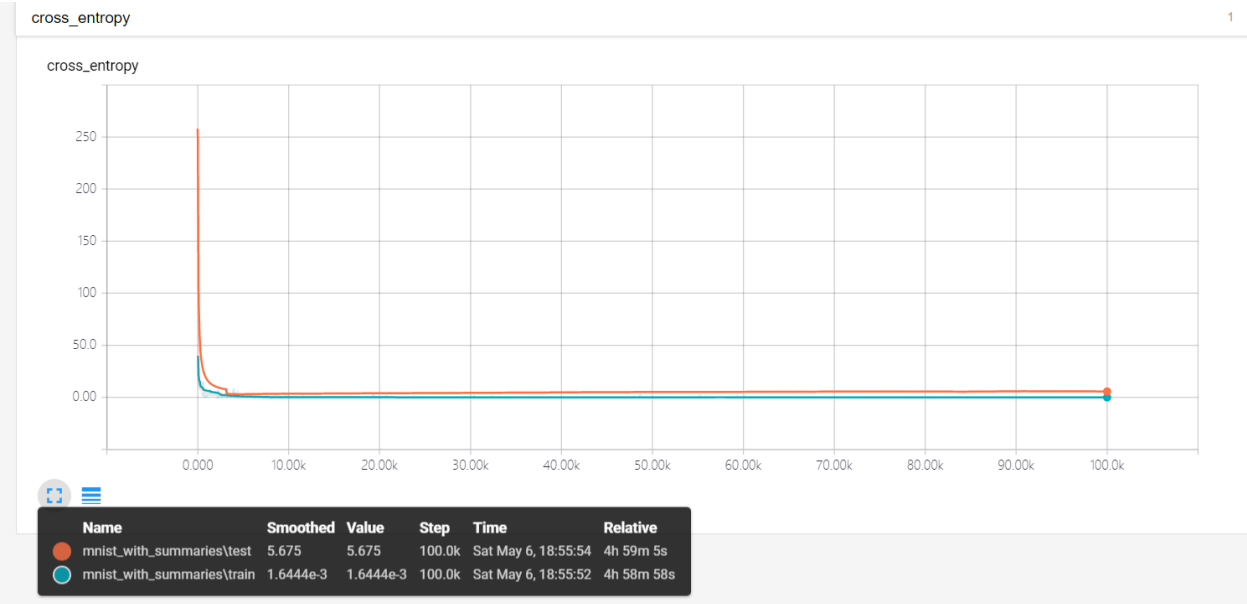
cross_entropy = tf.reduce_mean(cross_entropy)*100

And about the change in CNN architecture, we dropped the max pooling layer and used three convolution layers. In the first convolution layer, the stride is set to be 1 and in the second and third convolution layer, the stride is set to be 2. What's more, we also change the kernel and hidden units (You can find it in Picture 3 and 6.)

Here we log the accuracy on both training data and test data every 100 batches for 100000 batches in total. The accuracy and cross-entropy on training data and test data for every step are showed in Picture 7 and Picture 8. The details about the accuracy on training data and test data are included in the folder. The average of the test data accuracy is 99.1816% and the maximum of the test data accuracy is 99.31%. Out of the 1000 logs, there are 980 logs have accuracy greater than 99%.



Picture 7. The accuracy on training data and test data for each step



Picture 8. The cross-entropy on training data and test data for each step

Discussion:

In part one and two, we can find:

First, according to the two experiment in part one, we find the CNN model can absolutely improve the accuracy.

Second, in part one we run 5000 steps and there is no model that have accuracy over 0.99. In part two the CNN model get accuracy over 0.99 after 2000 steps. So the model in part two is obviously better.

Third, we also run the same network architecture model as that in part two except adding the max-pooling layer, Its performance is not better than the model in part two. So we think drop out the pooling layer will improve the accuracy in this model. The reason we think is that the max-pooling layer will help on the training speed. However, using pooling layer is equivalent to that we lose some information about the training data, so the performance will be worse. If we didn't use the pooling layer and retain the original data, it will only affect the model training speed but do not affect performance

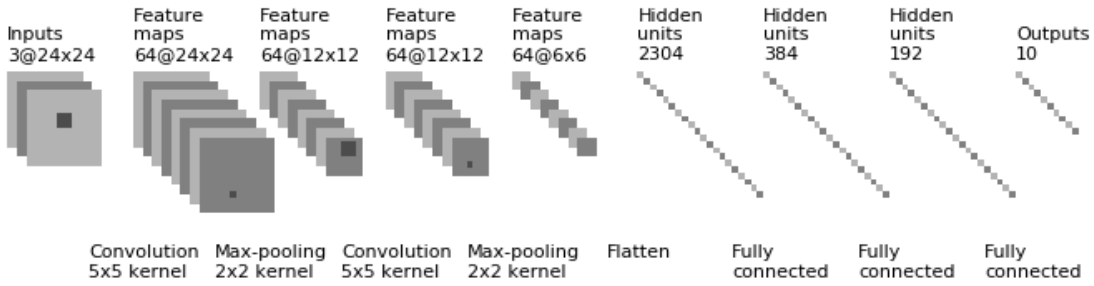
Fourth, we also experimented our implementation by adding more convolutional layers, and observed that 3 convolutional layers generally achieve higher performance than 2 or 1 layer. Also, we try more convolutional layers, but the performance do not improve.

Question 2. CIFAR10 Data Set

Part 1. Insert appropriate lines of code into the tensorflow example to log the accuracy on tensorboard every 100 batches, for at least 2000 batches. You should screen capture the accuracy graph from tensorboard, and submit this.

Answer:

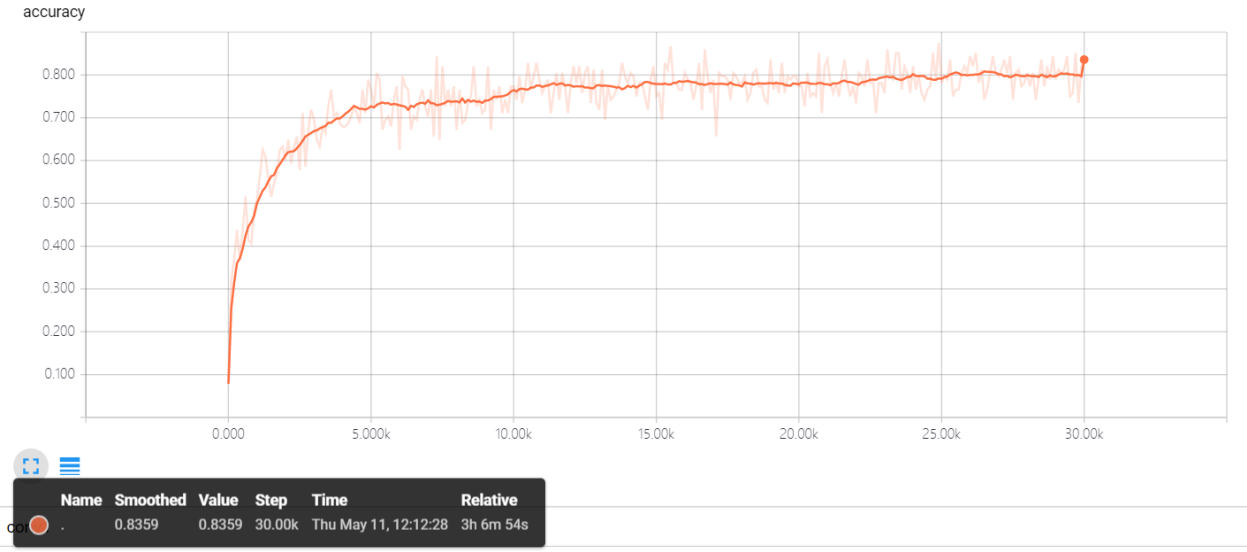
We use the tutorial code and insert appropriate lines of code to log the accuracy. The network architecture is shown in Picture 9.



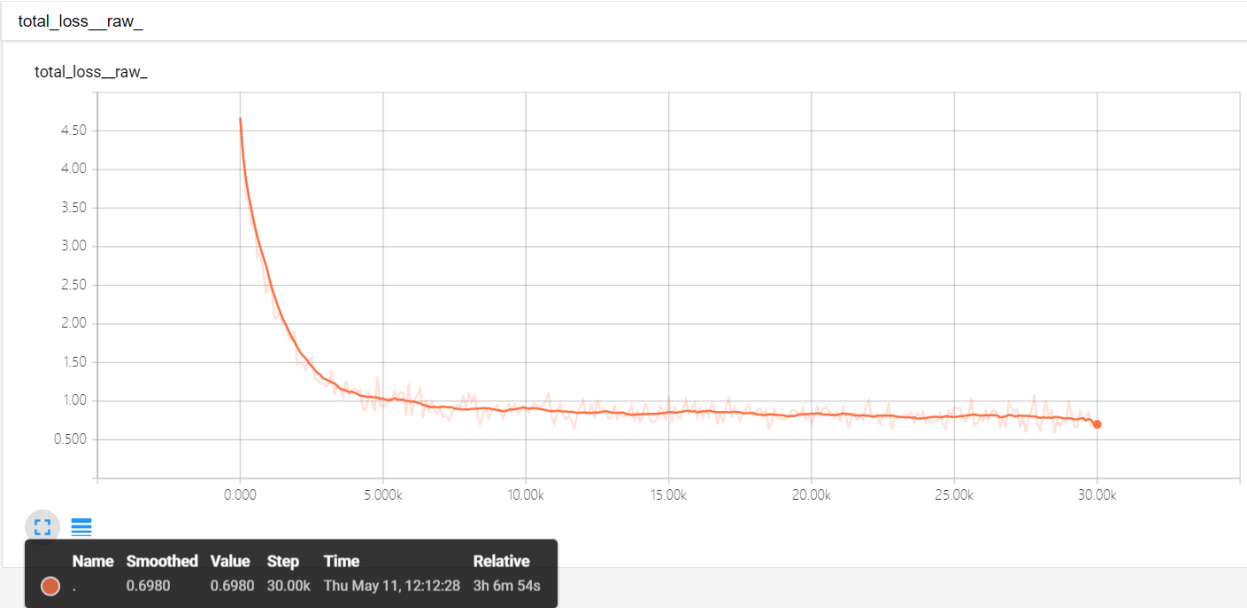
Picture 9. The network architecture of CNN model in Question 2 Part 1

Here we log the accuracy on both training data and test data every 100 batches for 30000 batches in total. We insert some code to make sure that the script will evaluate every 100 batch. (in the tutorial code, it depends on time.sleep() that decide how often to run the eval scripy)

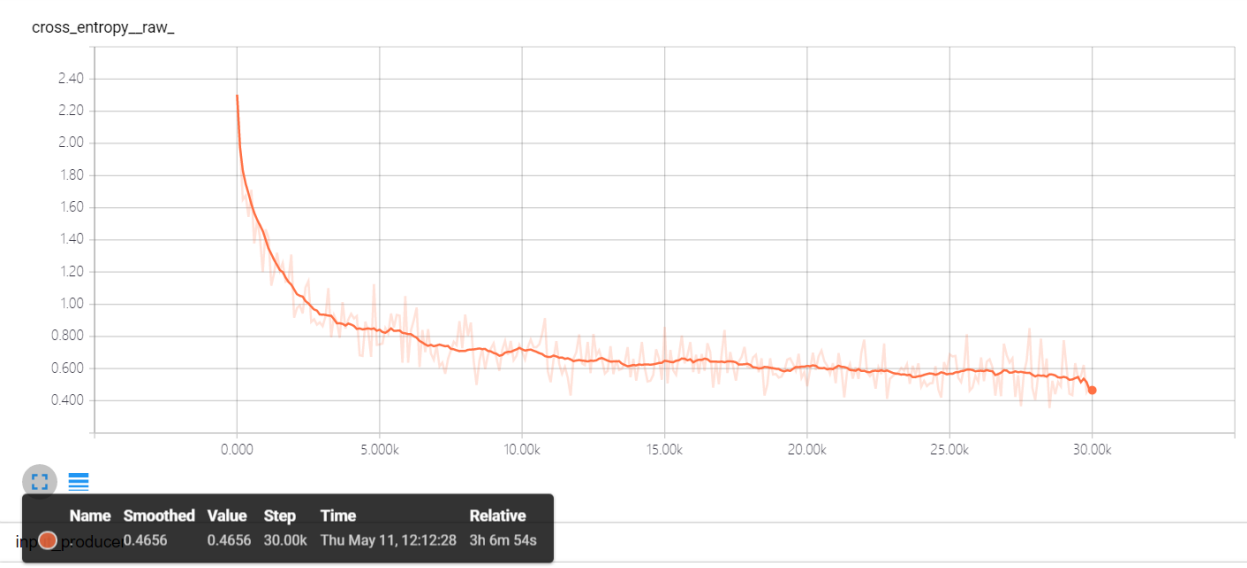
The accuracy, loss, and cross-entropy on training data for every step are showed in Picture 10, Picture 11 and Picture 12. The accuracy on test data for every step are showed in Picture 13. The details about the accuracy on training data and test data are included in the folder. The average of the training data accuracy is 74.66% and the maximum of the training data accuracy is 87.5%. The average of the test data accuracy is 80.17% and the maximum of the test data accuracy is 84.879%.



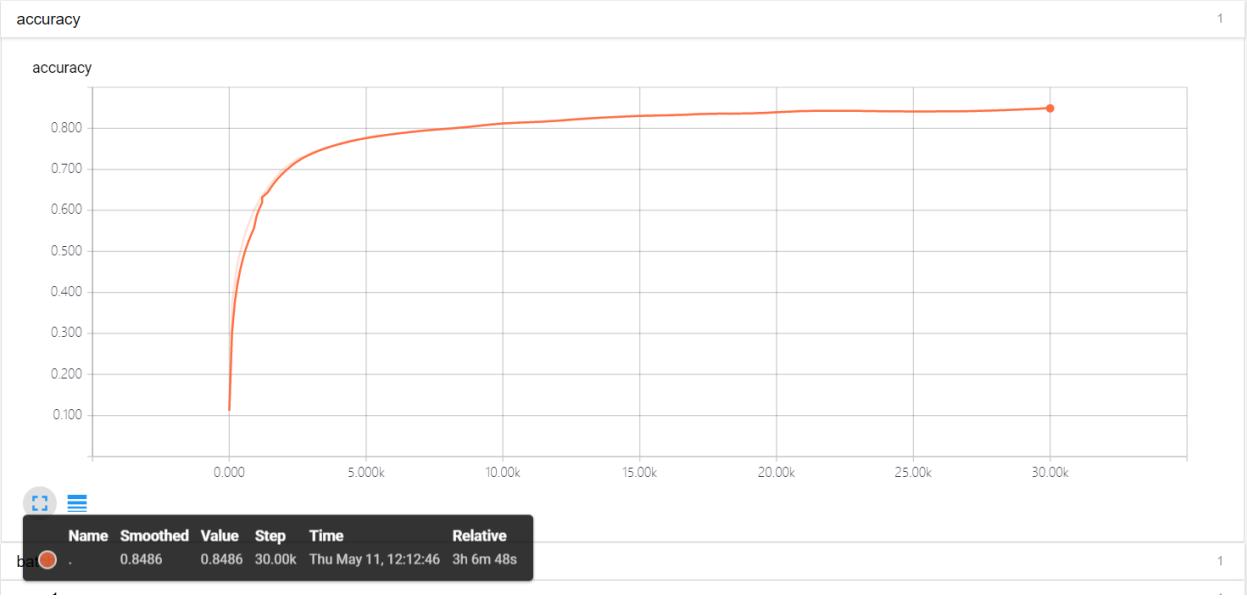
Picture 10. The accuracy on training data for each step



Picture 11. The loss on training data for each step



Picture 12. The cross-entropy on training data for each step

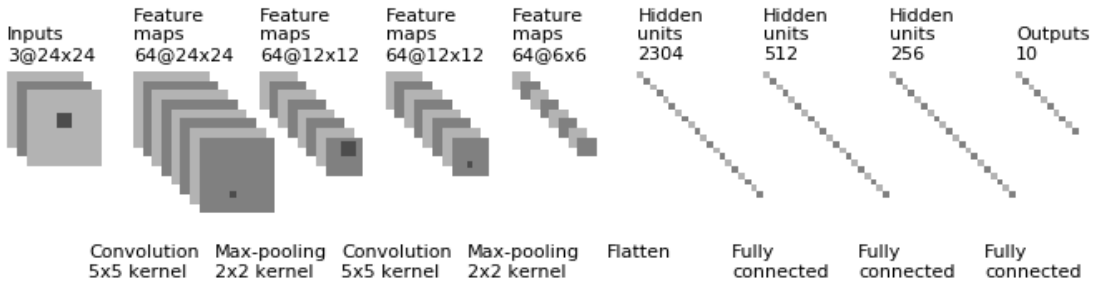


Picture 13. The accuracy on test data for each step

Part 2. Modify the architecture that is offered in the CIFAR-10 tutorial to get the best accuracy you can. Anything better than about 93.5% will be comparable with current research. Be aware that people with bigger computers will likely do better at this exercise (so I won't make grades depend on accuracy). Submit a screen capture of tensorboard graphs of accuracy.

Answer:

In this part, we modify the architecture. The network architecture is shown in Picture 14.



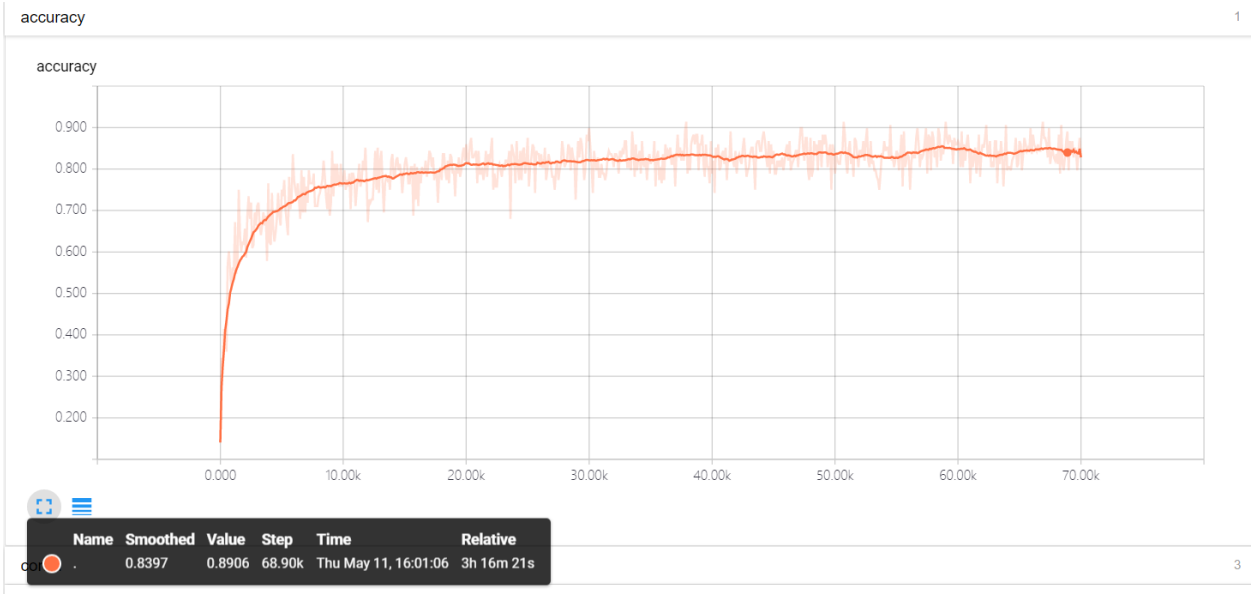
Picture 14. The network architecture of CNN model in Question 2 Part 2

(Something we modified)

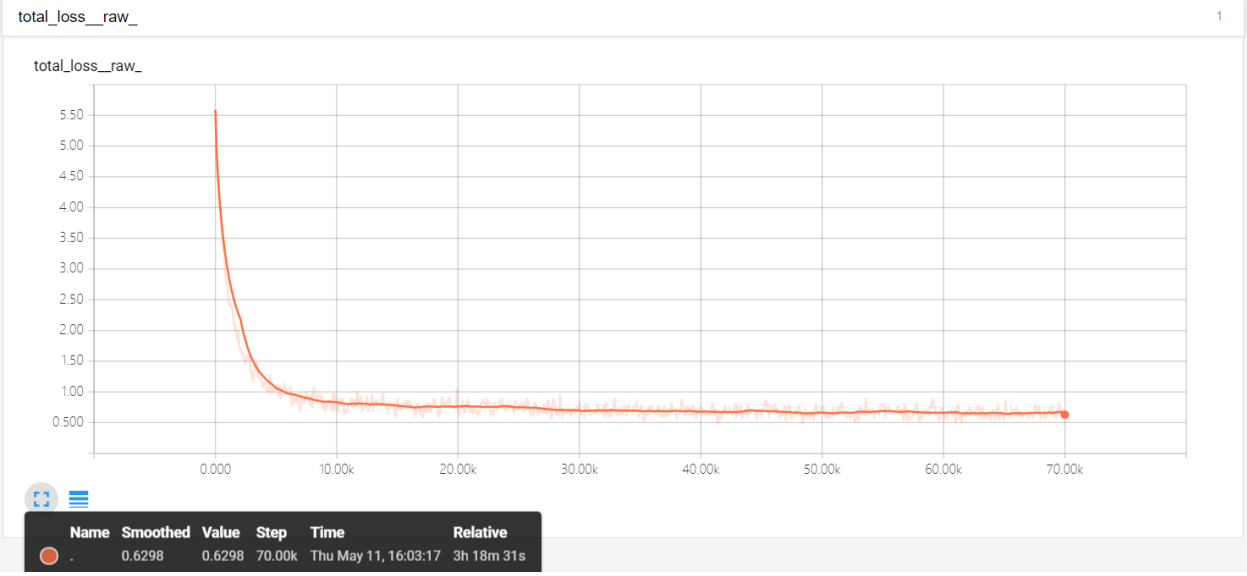
In this CNN model, we set the learning rate to be a variable value that depend on the step number.

And about the change in CNN architecture, the only thing we did is to use the method of batch normalization before every activation set (before take relu) in the two convolution layers.

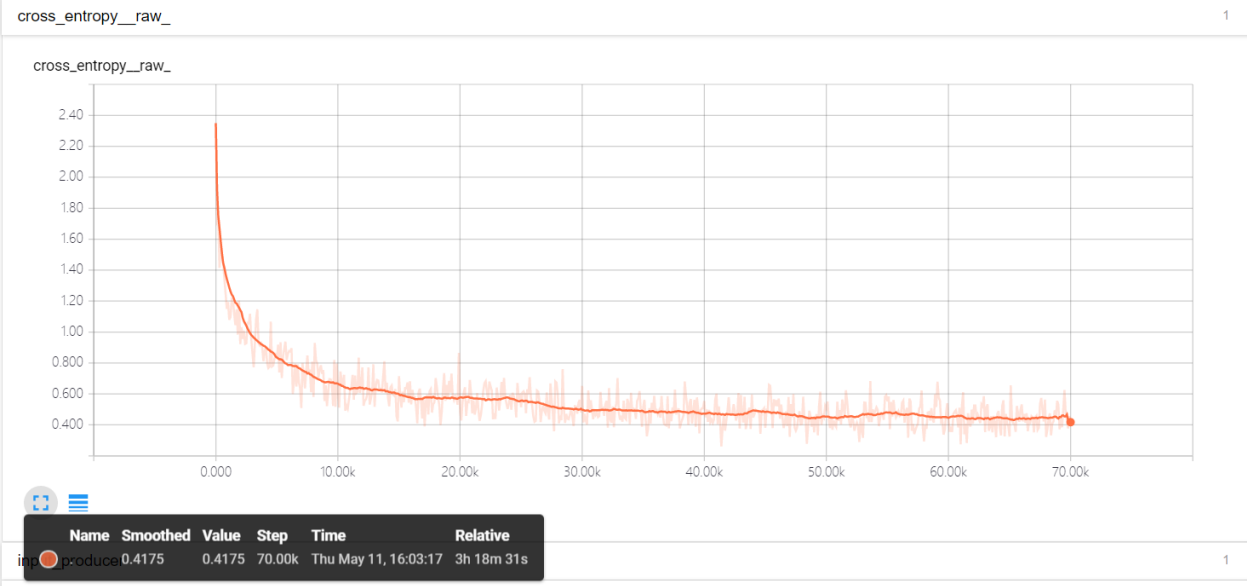
Here we log the accuracy on both training data and test data every 100 batches for 70000 batches in total. The accuracy, loss, and cross-entropy on training data for every step are showed in Picture 15, Picture 16 and Picture 17. The accuracy on test data for every step are showed in Picture 18.



Picture 15. The accuracy on training data for each step



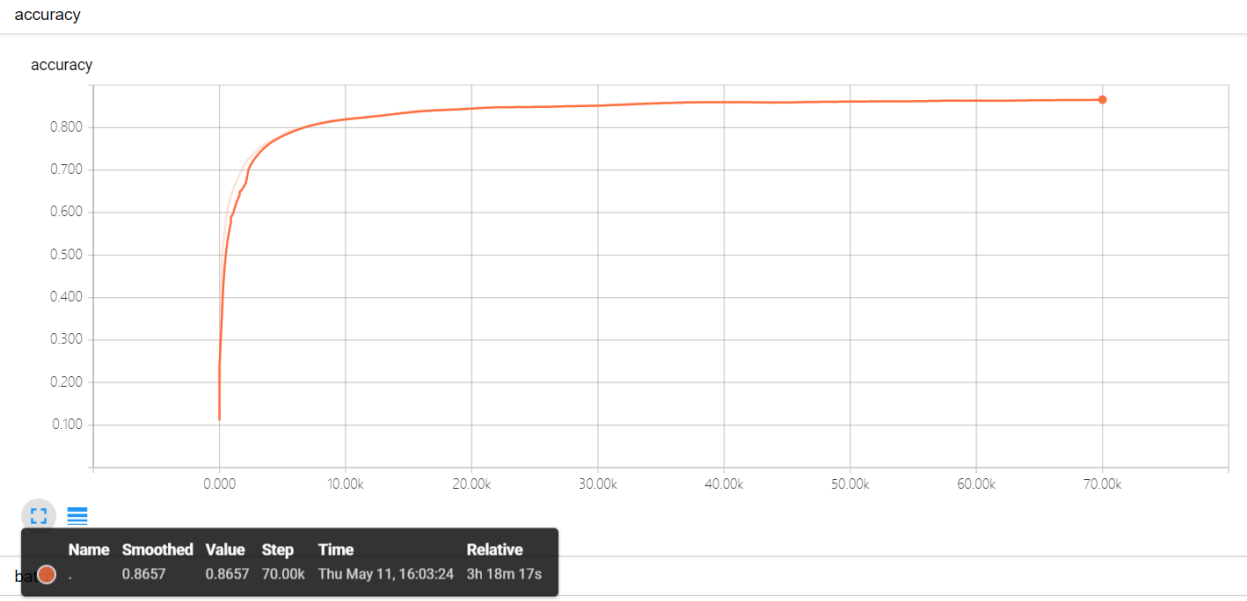
Picture 16. The loss on training data for each step



Picture 17. The cross-entropy on training data for each step

And about the change in CNN architecture, we used the method of batch normalization. What's more, we also change the hidden units of fully connected layers. (You can find it in Picture 9 and 14.)

The details about the accuracy on training data and test data are included in the folder. The average of the training data accuracy is 80.2837 % and the maximum of the training data accuracy is 91.4062 %. The average of the test data accuracy is 83.8724 % and the maximum of the test data accuracy is 86.61 %.



Picture 18. The accuracy on test data for each step

Discussion:

In part one and two, we can find:

First, in part one, there is no model that have accuracy over 85% and in part two the CNN model get accuracy over 85% after 36800 steps. The accuracy of the last step in part one (30000 steps) is 84.9% and the accuracy of 30000 steps in part two is 85.1%. So, we think the model in part two is better.

Third, we also experimented our implementation by adding more convolutional layers, and observed that 3 convolutional layers generally achieve higher performance than 2 or 1 layer. Also, we try more convolutional layers, but the performance does not improve.

Fourth, we did not use local_response_normalizationuse (norm1, norm2), but use the method of batch normalization in part two. We know that normalization is often used as a pre-processing step to make the data comparable across features. “As the data flows through a deep network, the weights and parameters adjust those values, sometimes making the data too big or too small again - a problem the authors refer to as "internal covariate shift". By normalizing the data in each mini-batch, this problem is largely avoided. ”[1] Thus, batch normalization potentially helps in two ways: faster learning and higher overall accuracy.

[1] https://www.quora.com/Why-does-batch-normalization-help