CS 519-006: Deep Learning

Assignment #3 Report

Michael Lam
March 3, 2016

This assignment explores tuning a convolutional neural network using Keras. We investigate various tuning methodologies and report the results.

1 Question 1

Remove the dropout layer after the fully-connected layer (5 points). Save the model after training.

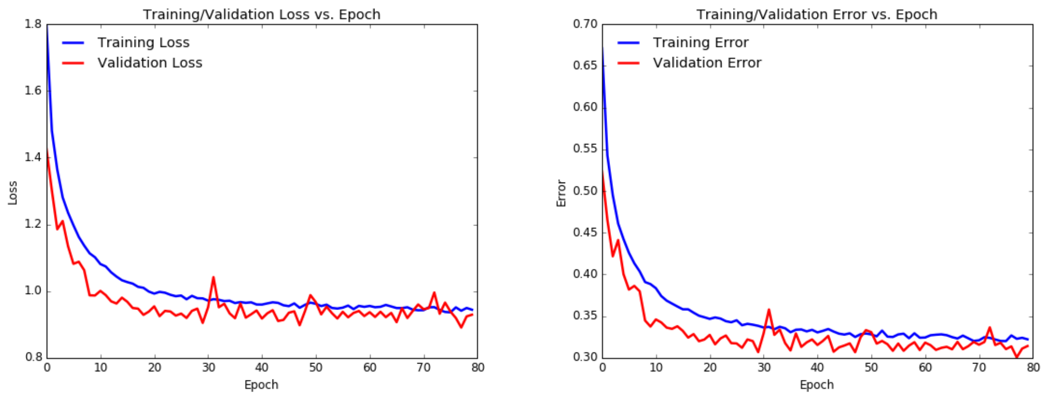
We removed the dropout layer after the fully-connected layer. Figure 1 shows the training/validation loss and error before doing anything; we will denote this as the baseline. Figure 2 shows the result of removing the dropout layer after the fully-connected layer. Interestingly, the training error from removing the dropout layer is lower than the training error from the baseline. However, the validation error remains about the same. This makes sense as expected since the dropout layer adds regularization during training so training would be slightly "harder" and so the training curve would be slightly worse. However, the validation error is lower, which shows that the model with dropout is not as prone to overfitting.

2 Question 2

Load the model you saved at step 1 as initialization to the training. Add another fully connected layer with 512 filters at the end (10 points). Train and save the model.

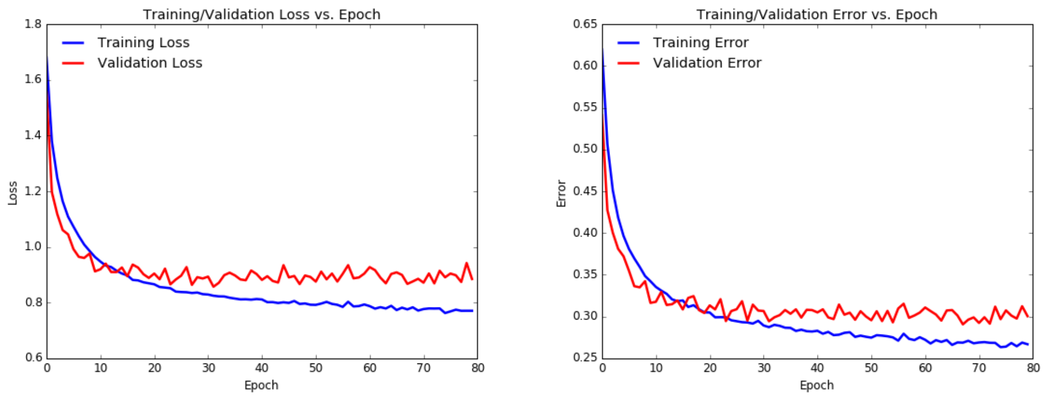
We loaded the model from step 1 as initialization. We then removed the last fully connected layer–the one converting to the number of classes–and softmax layer. We added another fully connected layer with 512 filters, then added back 10-node fully connected layer and softmax. We did not add back the weights for the 10-node fully connected layer from step 1. (We also experimented with adding weights back from step 1 but the results were similar.)

Figure 3 shows the loss and error plots. Interestingly, the training error is slightly better than the one from question 1 but the training curve is not as smooth as before. In addition, the validation error is now chaotic and increasing. We probably implemented the code correctly because the training curve is still reasonable. One explanation is that the model is overfitting very early: there is a slight decrease in the validation loss at first but it then it increases.



(a) (b)

Figure 1: Before question 1 and before doing anything. (a) Training and validation loss. (b) Training and validation error.



(a) (b)

Figure 2: Question 1: remove dropout layer. (a) Training and validation loss. (b) Training and validation error.

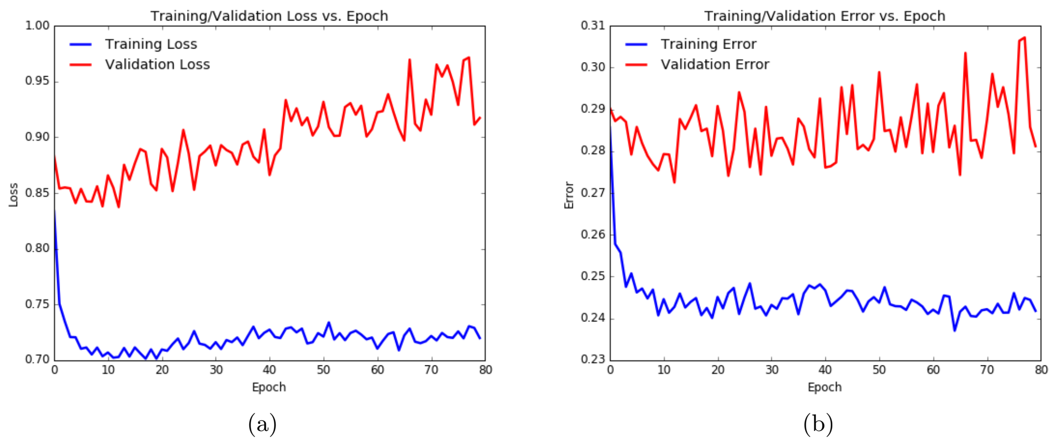


Figure 3: Question 2: add additional dense layer at the end. (a) Training and validation loss. (b) Training and validation error.

3 Question 3

Load the model you saved at step 2 as initialization. Add dropout layers to both fully-connected layers (10 points), re-train the model. (Hint: in this case you may need to manually move the weights to the correct corresponding locations in the new model, but some has mentioned that you can ”pop” a layer, so it might be easier than that).

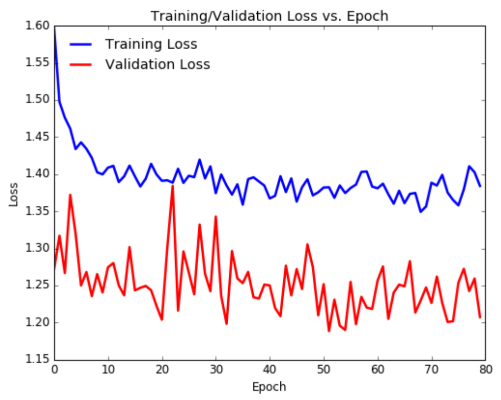
We loaded our mdoel from question 2 as initialization. We added dropout layers after both fully-connected layers.

Figure 4 shows the result after adding drop out back to all the fully connected layers. The training and validation curves are higher than those in question 2. The training curve still looks reasonable but the error is now around 0.45 at epoch 80 rather than around 0.24 from question 2. The validation error, while still looking chaotic, is lower than the training error. It would be reasonable that since dropout layers add regularization, the training performance would be worse at first. We could also expect some “chaotic” behavior since we initialized the weights from question 2, which also had chaotic results.

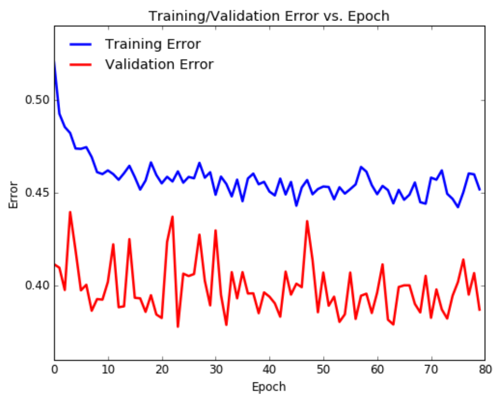
4 Question 4

Re-train the final model (after the model changes in tunings 1-3) from scratch. Try to use an adaptive schedule to tune the learning rate, you can choose from RMSprop, Adagrad and Adam (Hint: you don’t need to implement any of these, look at Keras documentation please) (5 points).

Figure 5 shows the result of training 1-3 from scratch. We trained with the Adam optimizer with the default settings. We see a much nicer curve in contrast to the plots in question 3. We see much better curves than from the previous questions. It seems that



(a)



(b)

Figure 4: Question 3: add dropout after fully connected layers. (a) Training and validation loss. (b) Training and validation error.

training entirely from scratch is potentially better than fine-tuning the top layers, or at least fine-tuning in the sequence as prescribed by questions 1-3. We also note that the performance is similar to the baseline. Adding an extra layer does not help that much in this case.

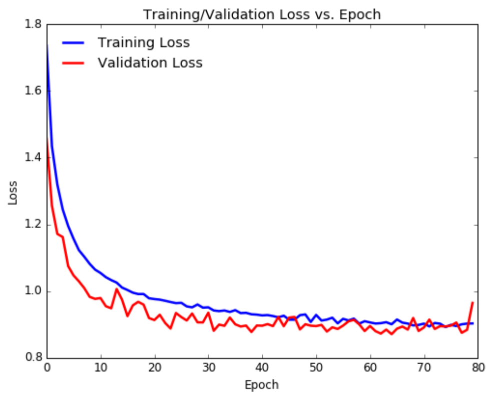
5 Question 5

Try to tune your network in two other ways (10 points) (e.g. add/remove a layer, change the activation function, add/remove regularizer, change the number of hidden units) not described in the previous four. You can start from random initializations or previous results as you wish.

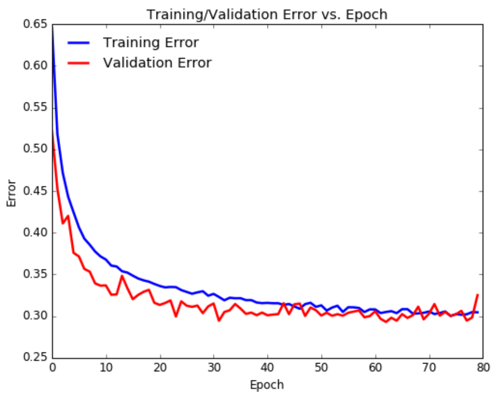
We explore two modifications. The first modification is to add more convolutional layers. Specifically, instead of two 3×3 convolutional layers with 32 filters, we add a third one before the first max pooling. We also add another 3×3 convolutional layer with 64 filters before the second max pooling layer. This is the first modification. The second modification is to simply modify question 4 such that all the fully connected layers contain 256 hidden nodes rather than 512.

Figure 6 shows the result of adding more convolutional layers. Compared to the baseline, it seems to perform slightly worse. This probably makes sense since from question 4, adding another dense layer did not help. Perhaps removing layers would have been a better experiment.

Figure 7 shows the result of using 256 hidden nodes instead of 512 hidden nodes. The training error is worse than the baseline but the validation error is about the same. In this case, making the model more restrictive gave slightly worse results. Thus model flexibility needs to be explored.

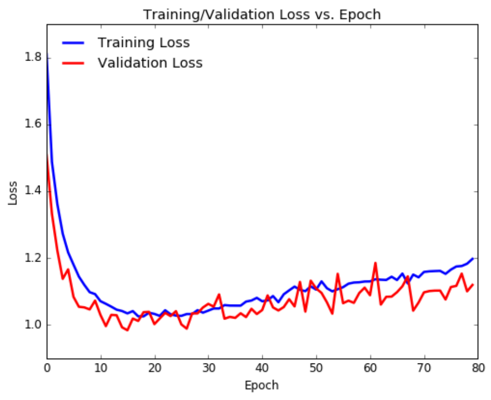


(a)

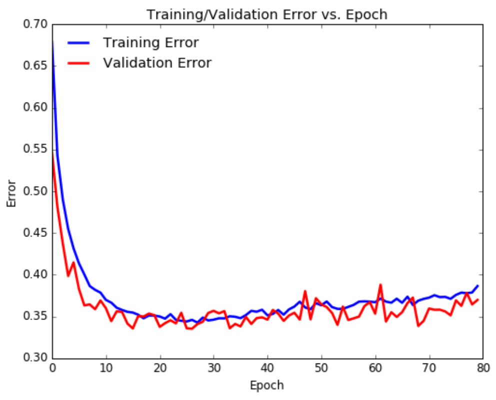


(b)

Figure 5: Question 4: train with changes 1-3 from scratch. (a) Training and validation loss. (b) Training and validation error.

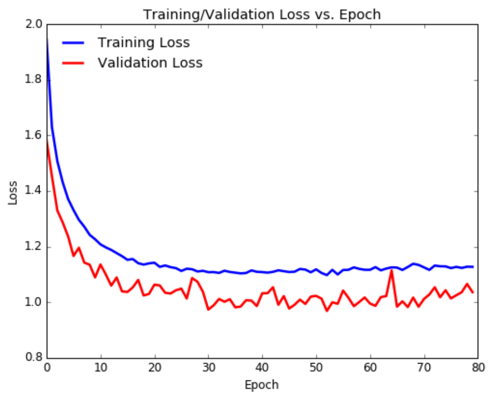


(a)

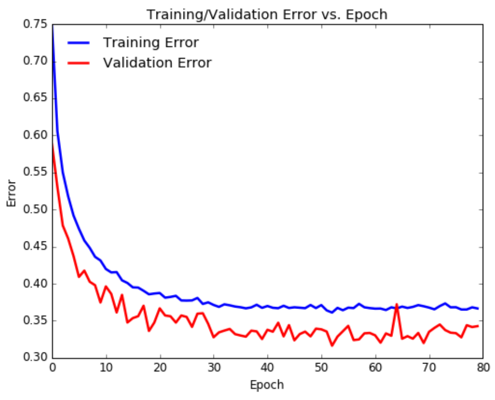


(b)

Figure 6: Question 5: add more convolutional layers. (a) Training and validation loss. (b) Training and validation error.



(a)



(b)

Figure 7: Question 5: change dense layers of question 4 to 256 hidden units. (a) Training and validation loss. (b) Training and validation error.