Yuan Hong (Bill) Sun
1003039838

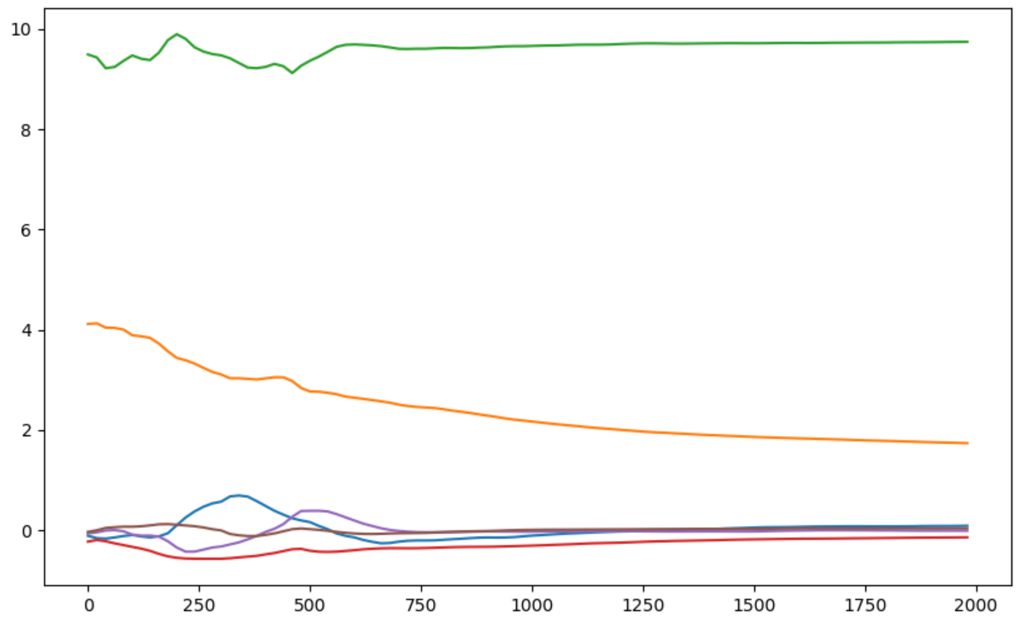
MIE324 Assignment 3

2. Data Pre-processing

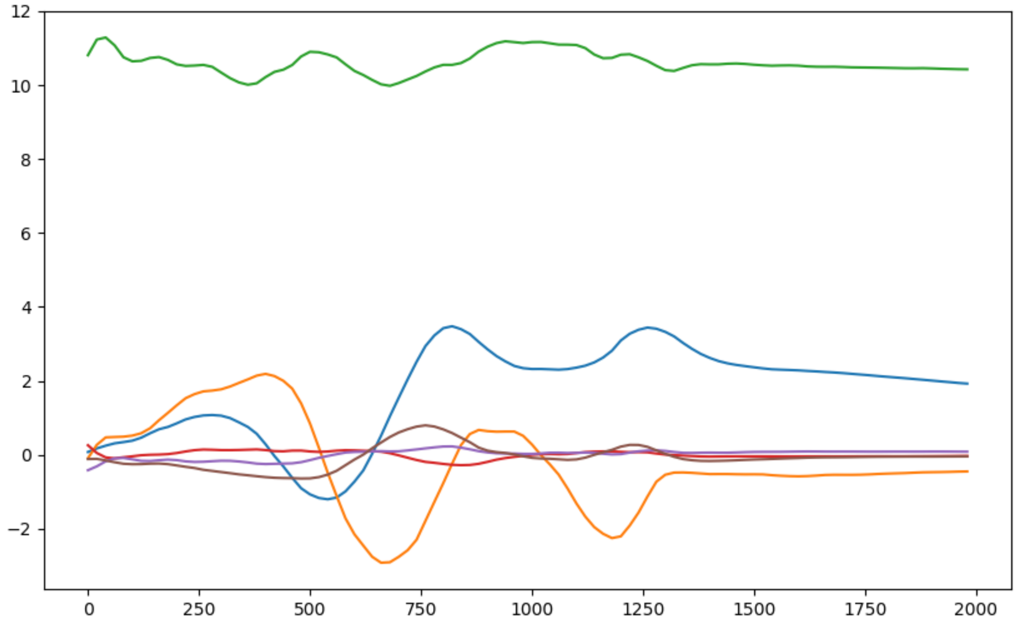
2.2 Understanding the dataset

Letter "a":

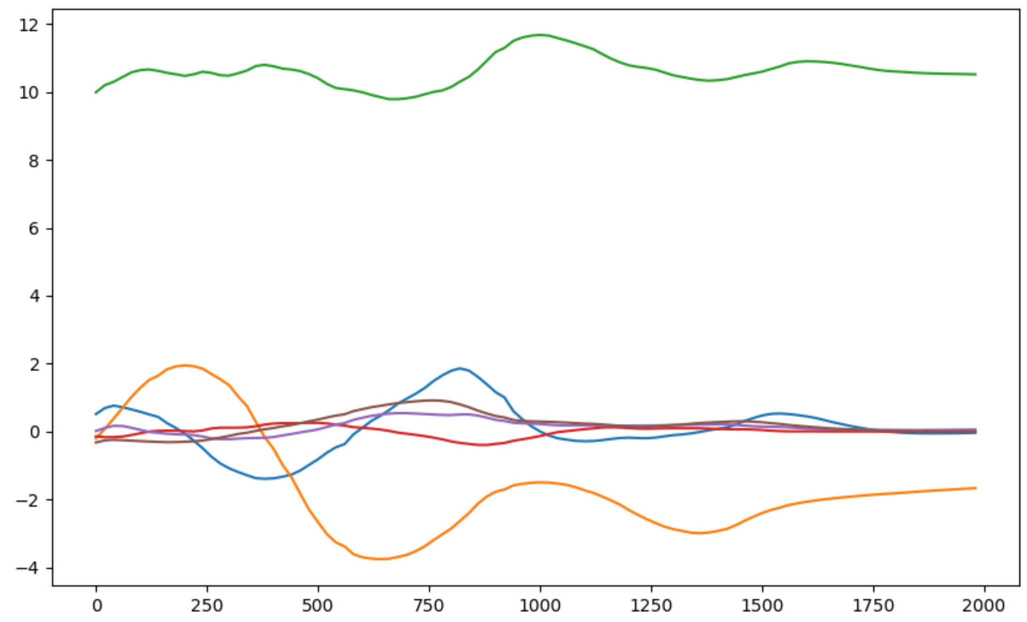
1.



2.

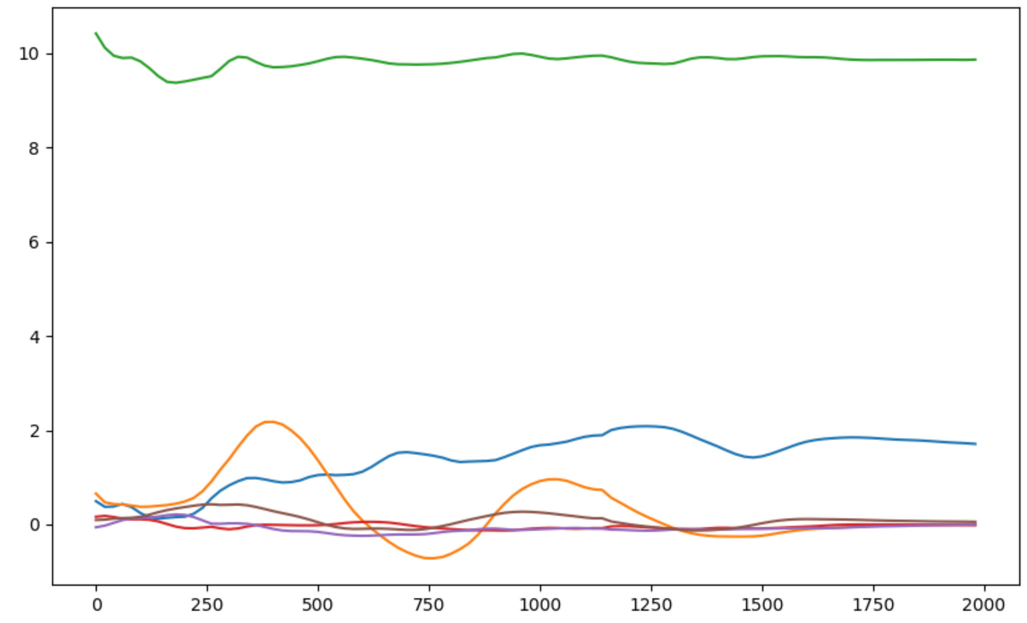


3.


12
10
8
6
4
2
0
−2
−4
0
250
500
750
1000
1250
1500
1750
2000

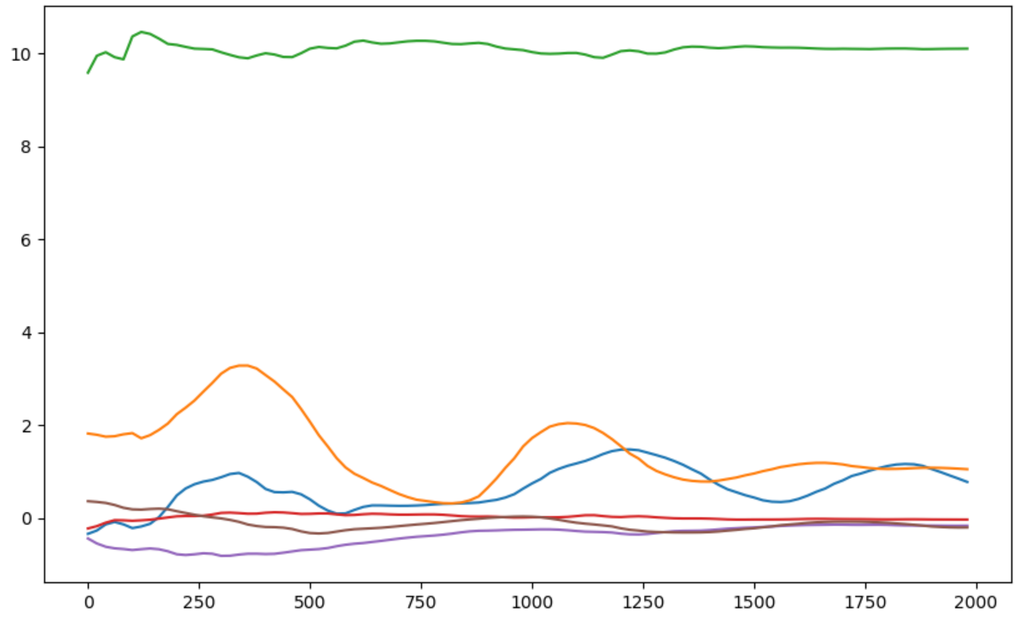
Letter "b":

1.

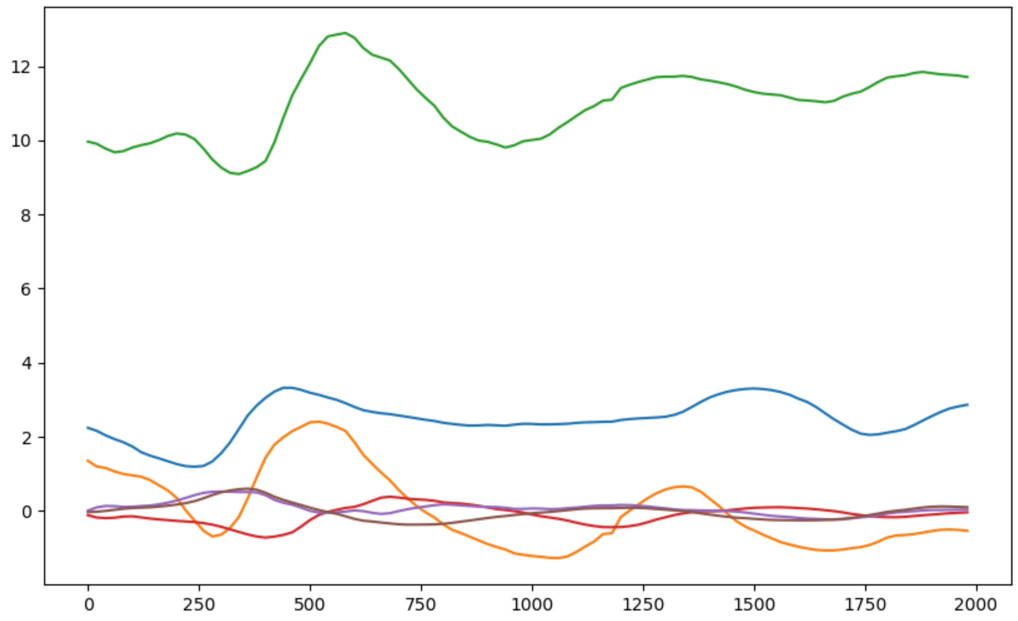

10
8
6
4
2
0
0
250
500
750
1000
1250
1500
1750
2000

2.

Yuan Hong (Bill) Sun
1003039838



3.



1. The data appears to be inconsistent across the three instances of each of the letters. In letter a, in the first graph, the orange line decreases asymptotically while in the other two instances, there are two clear peaks. The blue line also showed similar patterns in the latter two but did not do so in the first instance. This may be because the action was completed very quickly/roughly in the first instance.

In letter b, the orange line has consistently two peaks across all three instances, and the blue line shows roughly 3 peaks (though this is less clear). Other than the shape of the peaks, there is no clear difference in the movements between letters a and b.

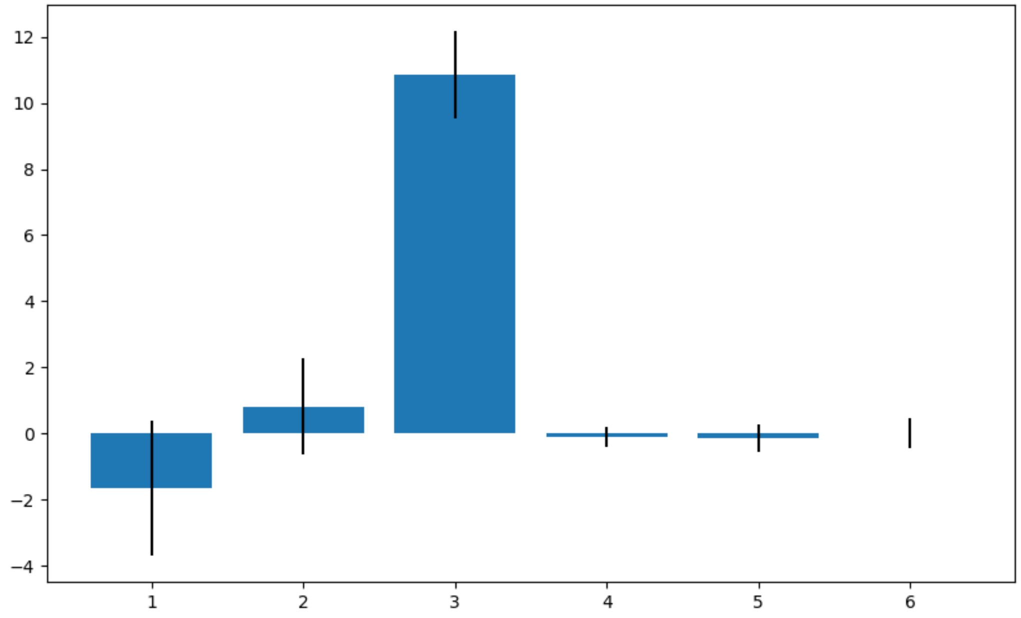
2. The values make sense since the gestures have a near-constant value for every value other than Ax and Ay. This is because there are no rotations (Wx, Wy, Wz), nor there are any vertical movement of the phone (Az), as instructed when collecting the data. Az, however, is close to 10 for each of the gestures, since there is gravitational acceleration (~10 m/s^2).

3. Since the data is hardly distinguishable just by looking at individual values, the variance of the data is an important differentiator. Therefore, the receptive field of the kernels towards the end of the CNN needs to be small, since capturing individual data points is not likely to give any distinguishable patterns.

2.3 Basic statistics

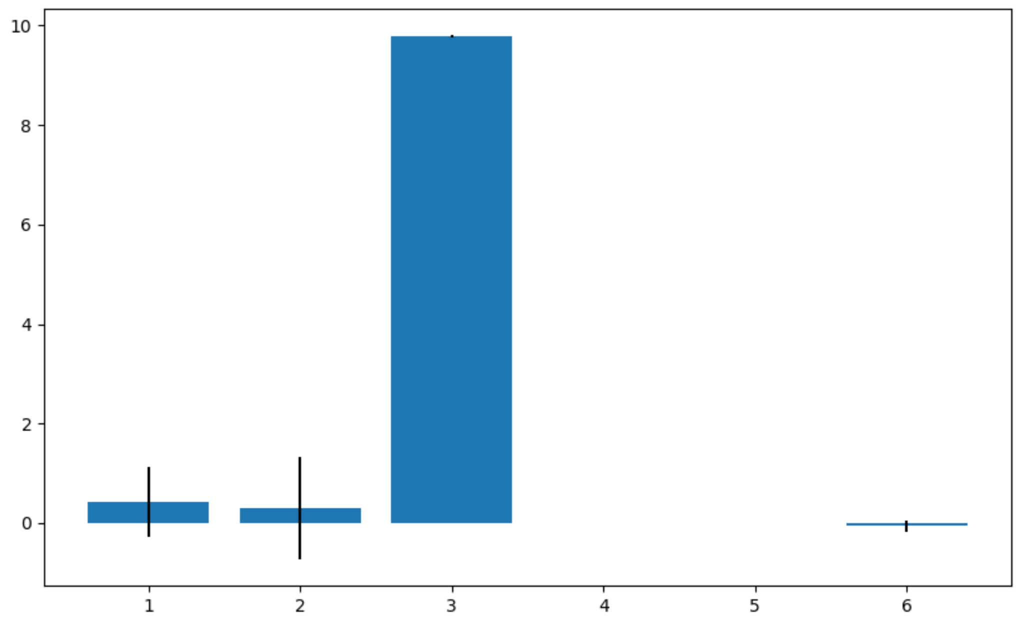
(Order of bars: Ax, Ay, Az, Wx, Wy, Wz)

Sample letter a:

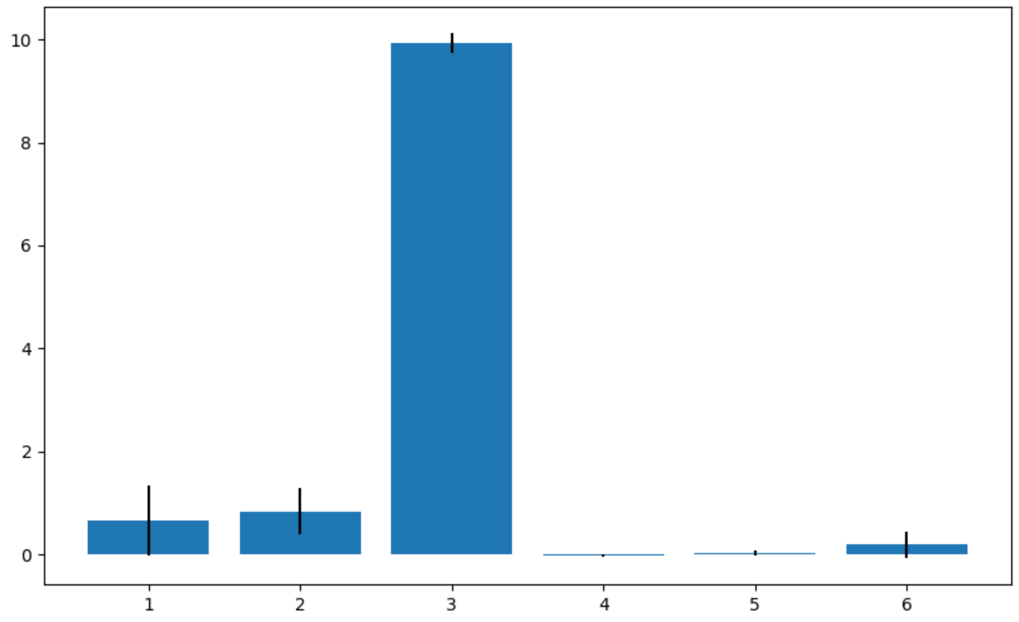


Sample letter b:

Yuan Hong (Bill) Sun
1003039838



Sample letter c:



1. Using this method, it was easier to identify the gesture since the first and second attributes (Ax and Ay) has different means in each letter. Az is around 10 which is expected (as discussed in 2.2.3), and the rotation attributes were close to zero, which is also expected. However, all of this cannot be attributed to the gesture since there are large variations in the data, created by the inconsistency during data collection.

2. It is more challenging for the neural network to classify the values based on the average statistics, since there is less difference in the means overall from gesture to gesture (i.e. between b and c), so it may be more difficult to make such classifications. The original data presents more differentiators.

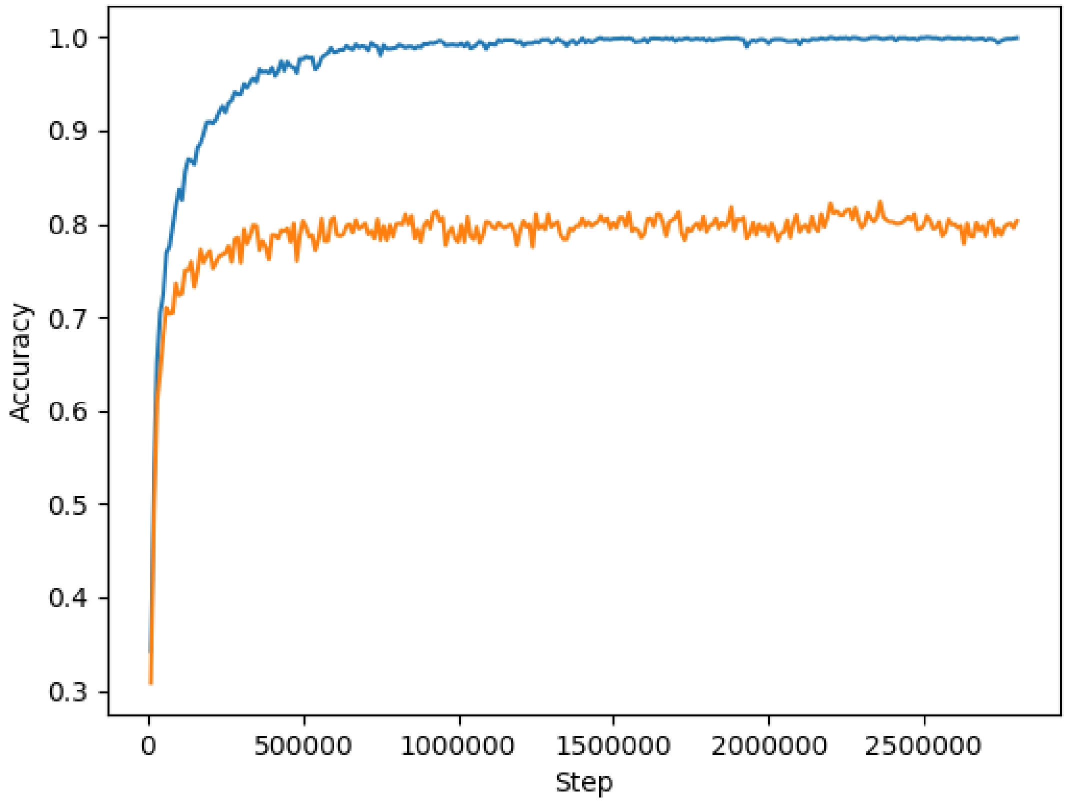
Yuan Hong (Bill) Sun
1003039838

2.5 Train-validation split

1. The validation set is only an indicator of the model performance during training (aside from training data). After training is complete, using a new set of data to test the model ensures that the model is performing well and not overfitting / underfitting on the training and validation data. This is especially true if the validation data set is small or if it includes many outliers. This will decrease the overall quality of the model.

The training-validation split is set to 70% and 30%. The reason for using more validation data (compared to previous assignments, which used an 80-20 split) is because no test dataset is given, and a more reliable evaluation result from the validation data is needed to ensure a higher model performance.

3. Training



*The step number is incorrect and need to be divided by 100 each (error in code)

Overall, the best validation accuracy was around 82%, however, after smoothing the graph it is between 80% and 81% when the graph flattened out.

There is some overfitting since the training accuracy reached 100% but the validation accuracy only approached 80%.

Yuan Hong (Bill) Sun
1003039838

4. Test performance

First, a bare-bones structure with one convolutional layer and two linear layers were used, to ensure that there was a working model. The convolutional layer was 1D since the data needs to be traversed linearly (this was applied to all convolutional layers afterwards). This bare-bones model resulted in less than 70% validation accuracy.

As for the hyperparameters, it was determined through trial-and-error that a batch size of 32, a learning rate of 0.01, and an epoch range around 200 gives the best results. These hyperparameter values were set to default and used throughout the remaining model changes and runs.

Next, one convolutional and one linear layer were added. This model now had two convolutional layers and three linear layers. In addition, MaxPool1d was used to decrease the runtime on the convolutional layers. ReLU was used as the primary activation function. Output from each layer was also normalized using BatchNorm1d. This model was able to achieve 74% validation accuracy.

One more convolutional and linear layer each were then added. Using the default setup from before (hyperparameters, MaxPool1d, ReLU, BatchNorm1d), a maximum of 81% validation accuracy was achieved. Several different changes were made to the model and then tested, such as changing the activation ReLU to leaky ReLU, getting rid of BatchNorm1d or MaxPool1d. However, all these changes resulted in slightly lower validation accuracies (between 76% and 79%). Also, when softmax was implemented at the end, it significantly delayed the training process (i.e. it took a lot longer to stabilize to 80% validation accuracy compared to not using softmax), so therefore it was left out. Log_softmax was tried as well, but it gave a lower validation accuracy.

Furthermore, the sizes of convolutional and linear layers were given such that each successive convolutional layer doubles in size, while each linear layer afterwards decreases in size by roughly half. This was done in the belief that a generally uniform increase and decrease in layer sizes retains more information in the data compared to random changes in layer size.

In another experiment, two more convolutional layers were added to the model, bringing the total number of convolutional layers to 5. However, this did not give a validation accuracy greater than 79%. This may be because the dimensions of each layer were not optimal (as I did not have enough time to fully test each case), but regardless, these two layers were later removed, and the model described above (3 convolutional, 4 linear layers) was taken as the best model overall.

5. Feedback

a). This assignment took me more than 30 hours to complete, including the model runtime.

b). The challenging part is trying to find the best model without much guidance or instructions. It is also the pressure of seeing other classmates getting higher validation accuracies. It also took me some time to understand the different activation functions and their requirements, whether or not to use one-hot encoding, and the setting the dimensions/size of the layers.

Yuan Hong (Bill) Sun
1003039838

c). The coding part was relatively straightforward and I enjoyed that part. Also, it was entertaining watching the validation accuracy increase and stabilize.

d). The most confusing part is trying to understand different activation functions and how each of them affects the model performance. Also, I had to look up how to calculate layer dimensions after MaxPool.

e). Some instructions were given to help us create the best model, which was somewhat helpful, although it could be elaborated on what each function does.