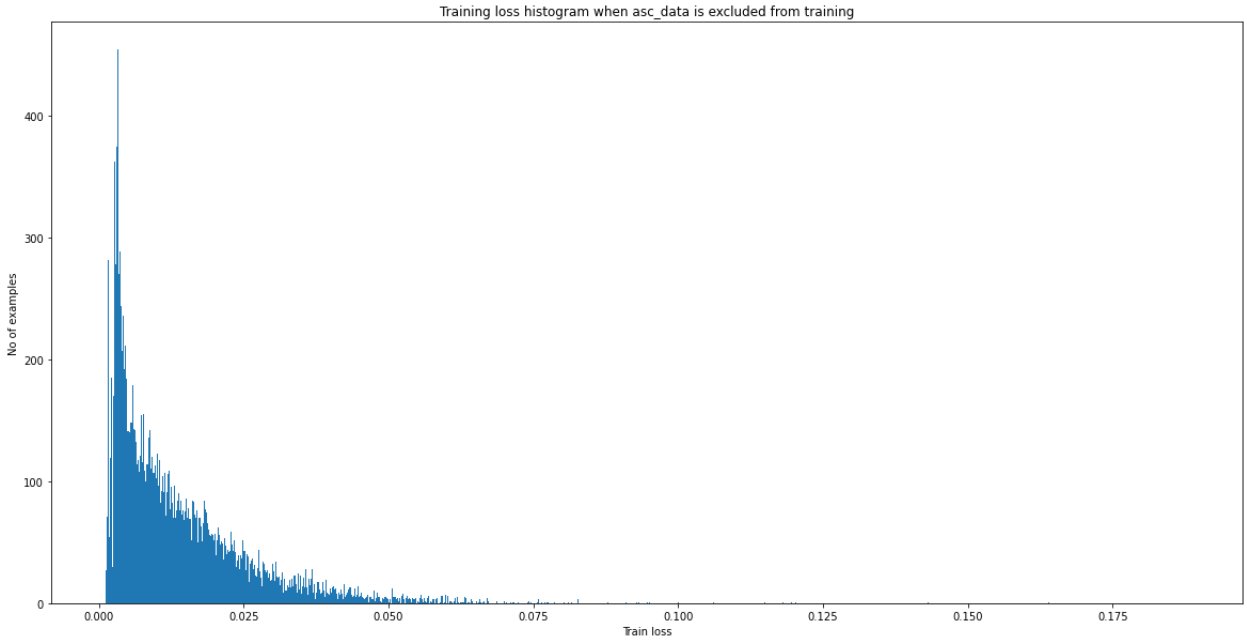
CS 497:Mobile Health and Wearables
Project 2 Report
Manikanta Mandlem

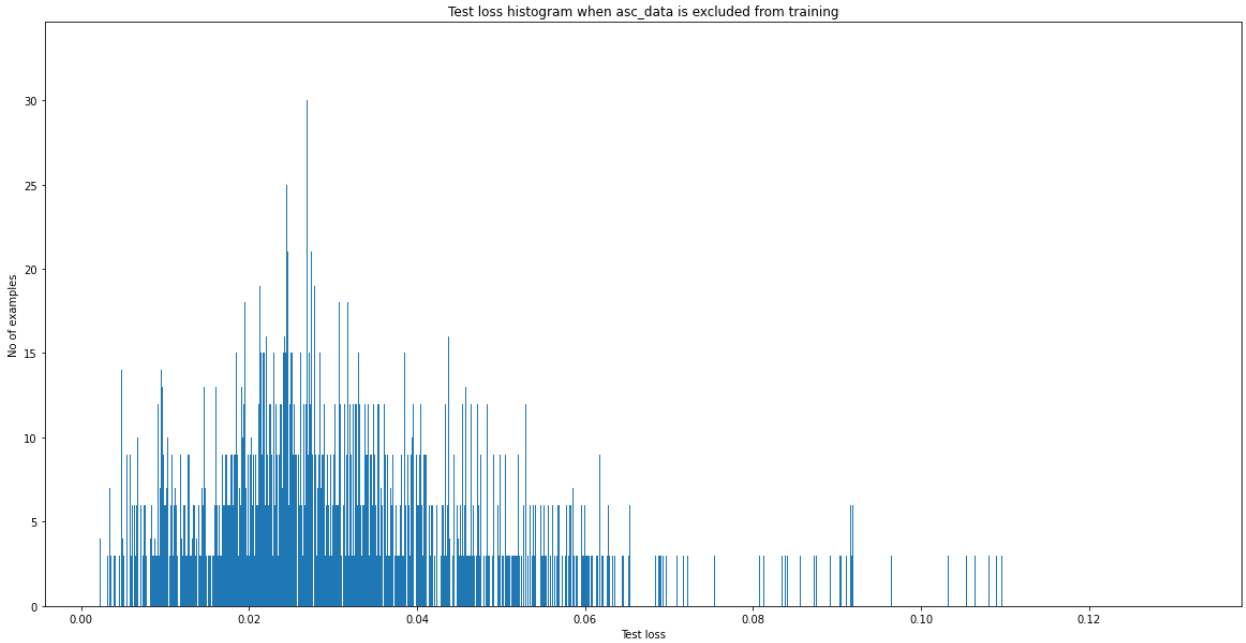
Observations from Autoencoder (Step2):

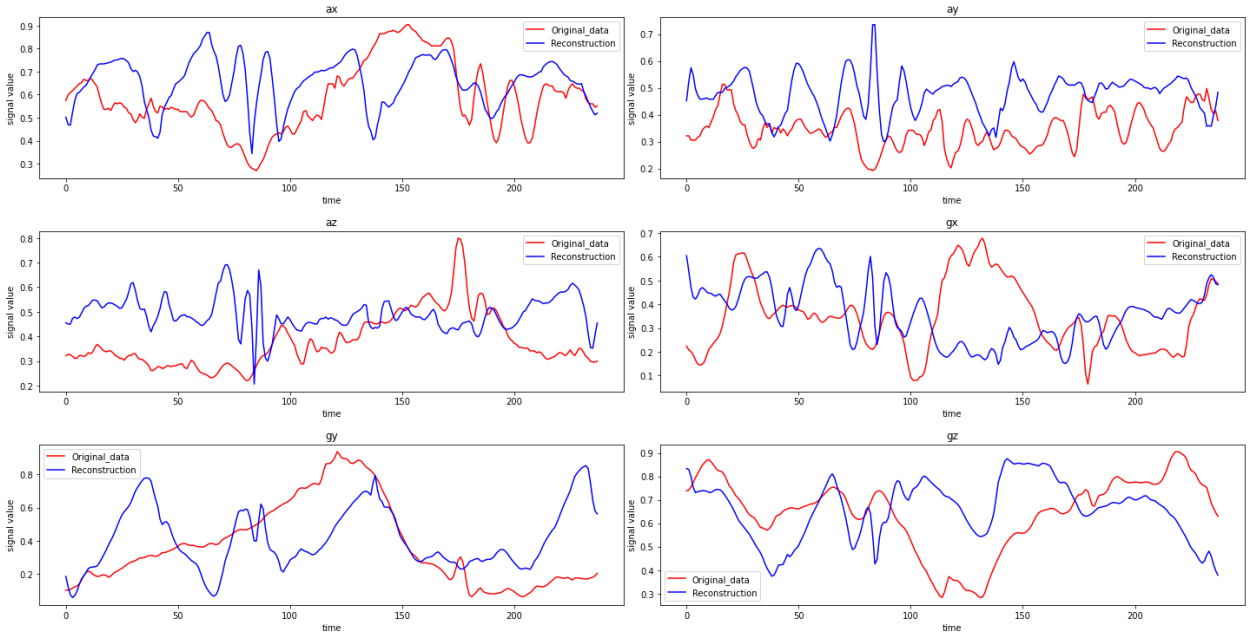
- The autoencoder model architecture that I have used is displayed below:

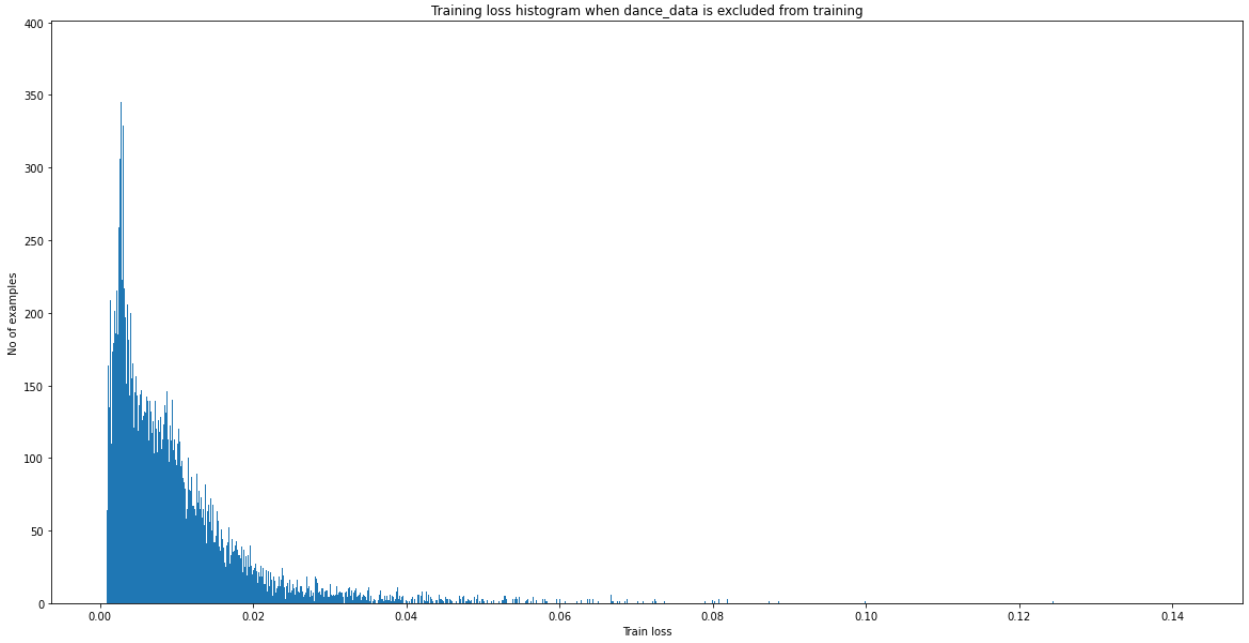
```
self.encoder = tf.keras.Sequential([
                        layers.Dense(128, activation="relu"),
                        layers.Dense(64, activation="relu"),
                        layers.Dense(32, activation="relu"),
                        layers.Dense(16, activation="relu"),
                        layers.Dense(8, activation="relu")])

self.decoder = tf.keras.Sequential([
                        layers.Dense(16, activation="relu"),
                        layers.Dense(32, activation="relu"),
                        layers.Dense(64, activation="relu"),
                        layers.Dense(128, activation="relu"),
                        layers.Dense(6, activation="sigmoid")])
```

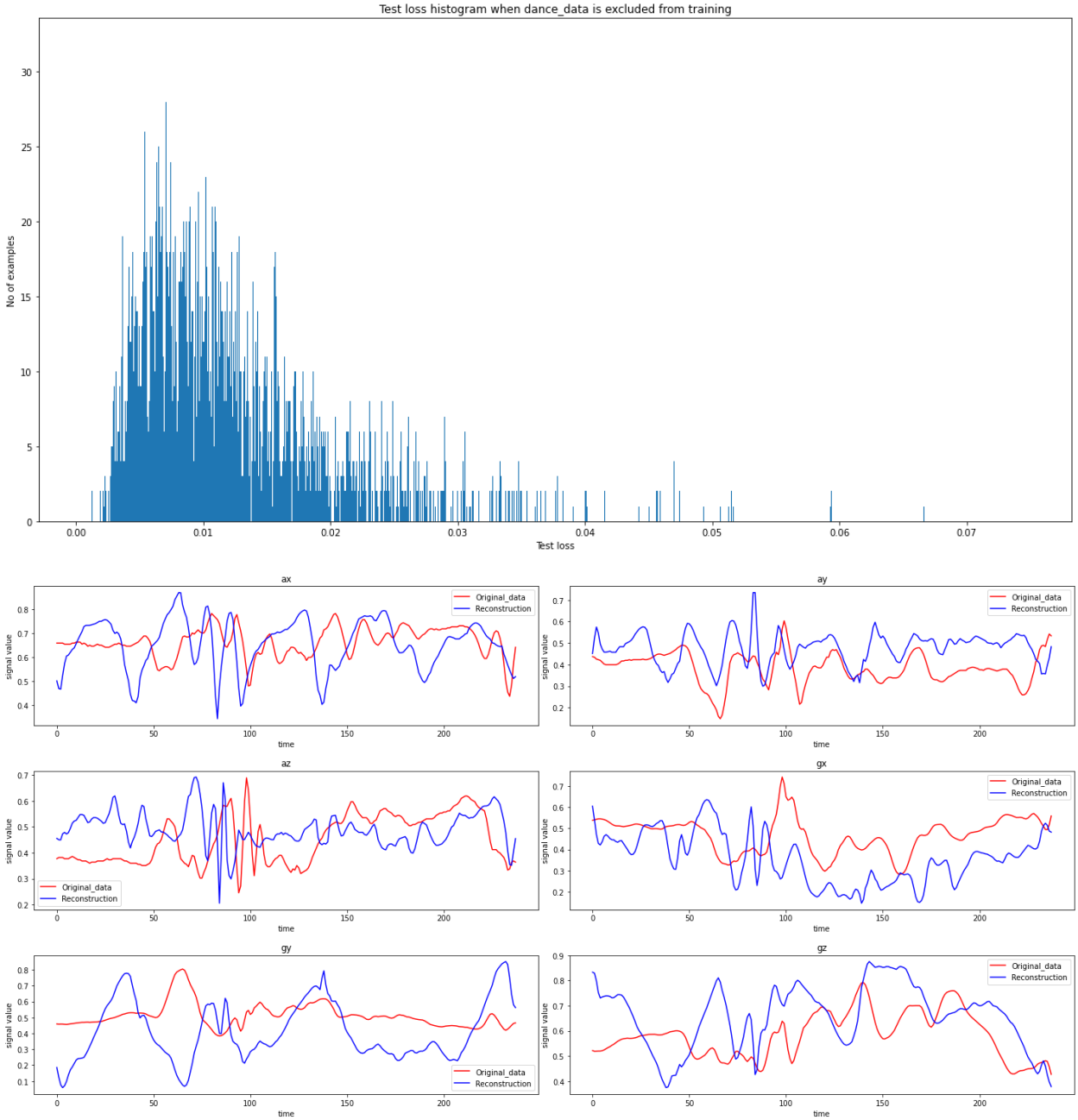
- This architecture seems to do a reasonable job in identifying anomalous signal
- Below are the graphs from 12 different (leave one out) autoencoder experiments. For each triplet of graphs,
    - the first graph shows the histogram of Mean Averaged Error between the original signal and encoded signal for train data
    - The second graph shows the histogram of Mean Averaged Error between the original signal and encoded signal for test data (test activity noted in graph title)
    - The third graph signifies the error between the original test signal and the encoded test signal for a small 2-minute signal segment

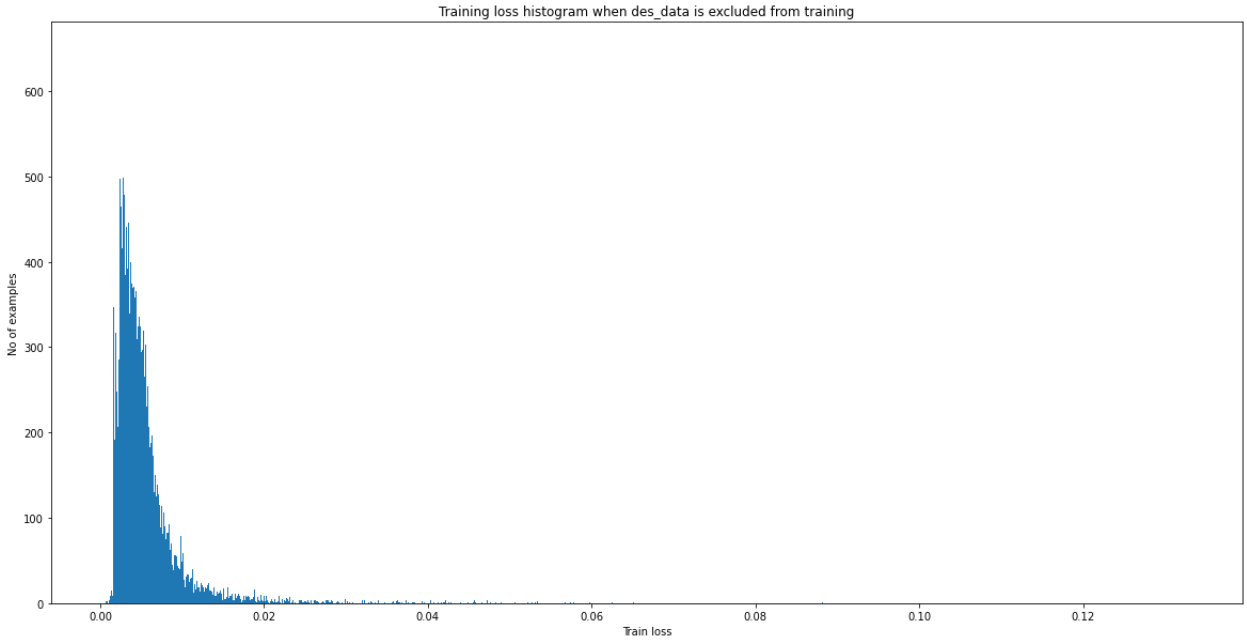

Training loss histogram when asc_data is excluded from training
No of examples
Train loss
0
100
200
300
400
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175

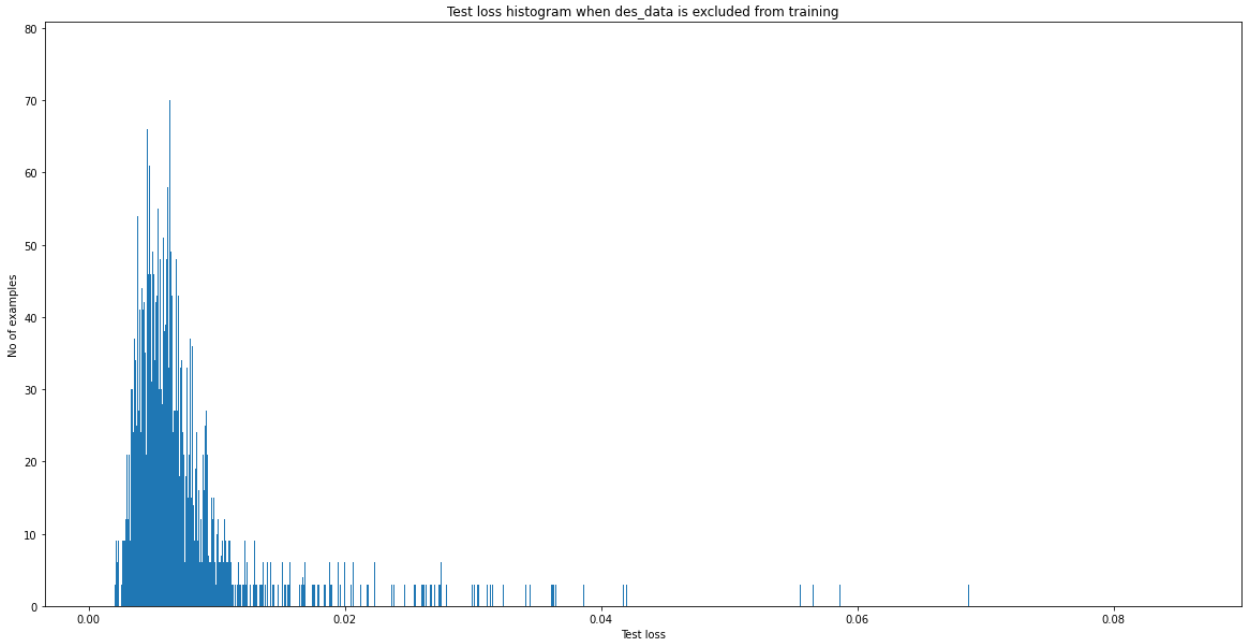

Test loss histogram when asc_data is excluded from training
No of examples
Test loss
0
5
10
15
20
25
30
0.00
0.02
0.04
0.06
0.08
0.10
0.12

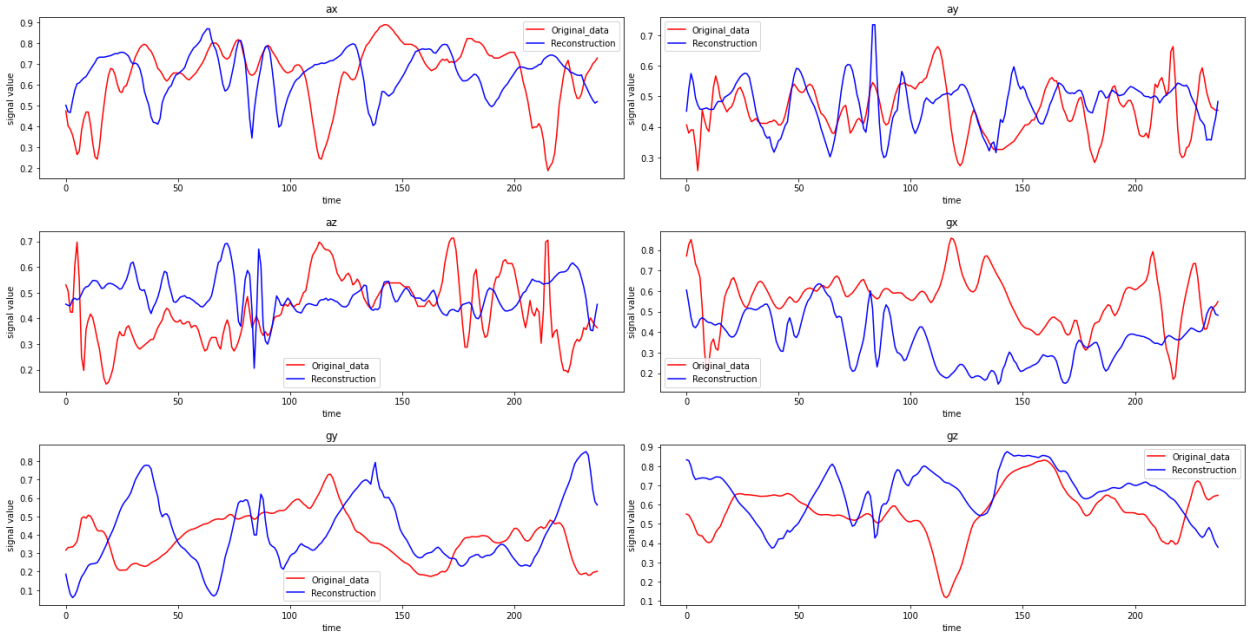

ax
ay
az
gx
gy
gz
Original_data
Reconstruction
signal value
time

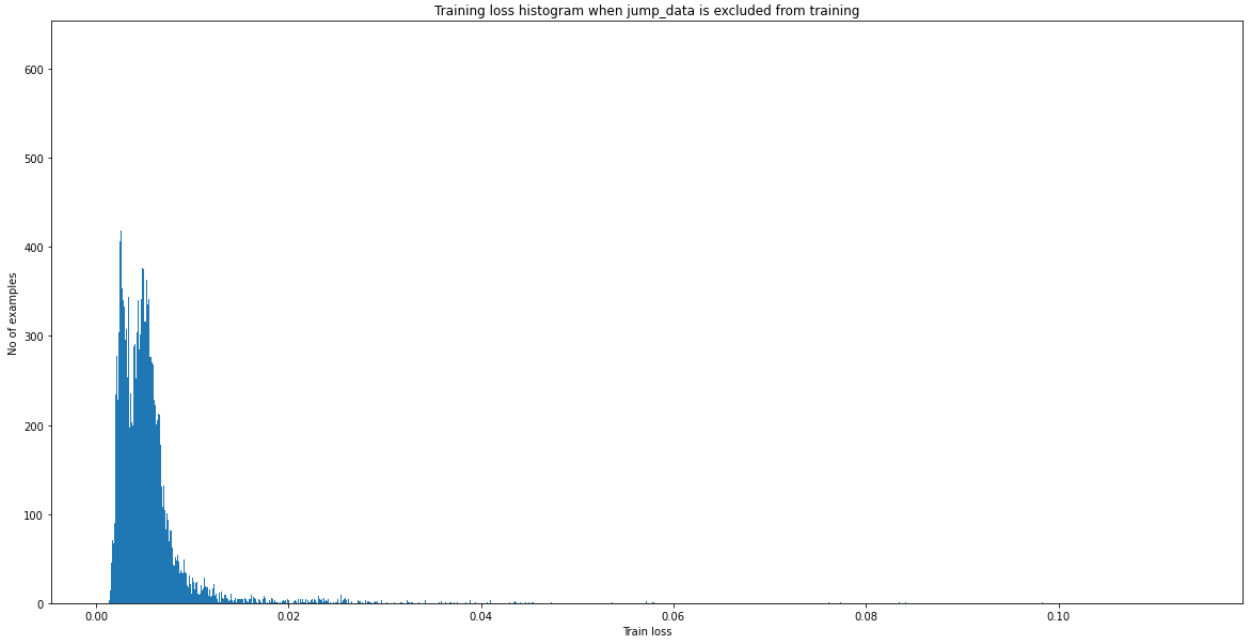

Training loss histogram when dance_data is excluded from training
No of examples
Train loss

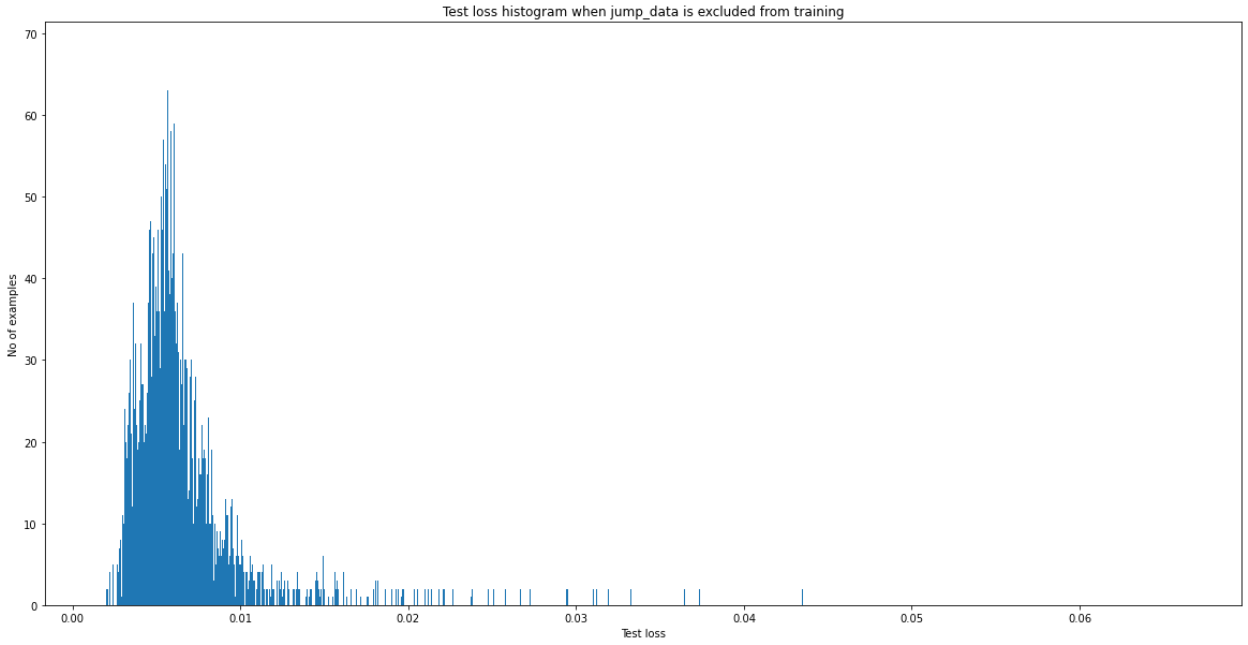

Test loss histogram when dance_data is excluded from training
No of examples
Test loss
ax
ay
az
gx
gy
gz
signal value
time
Original_data
Reconstruction

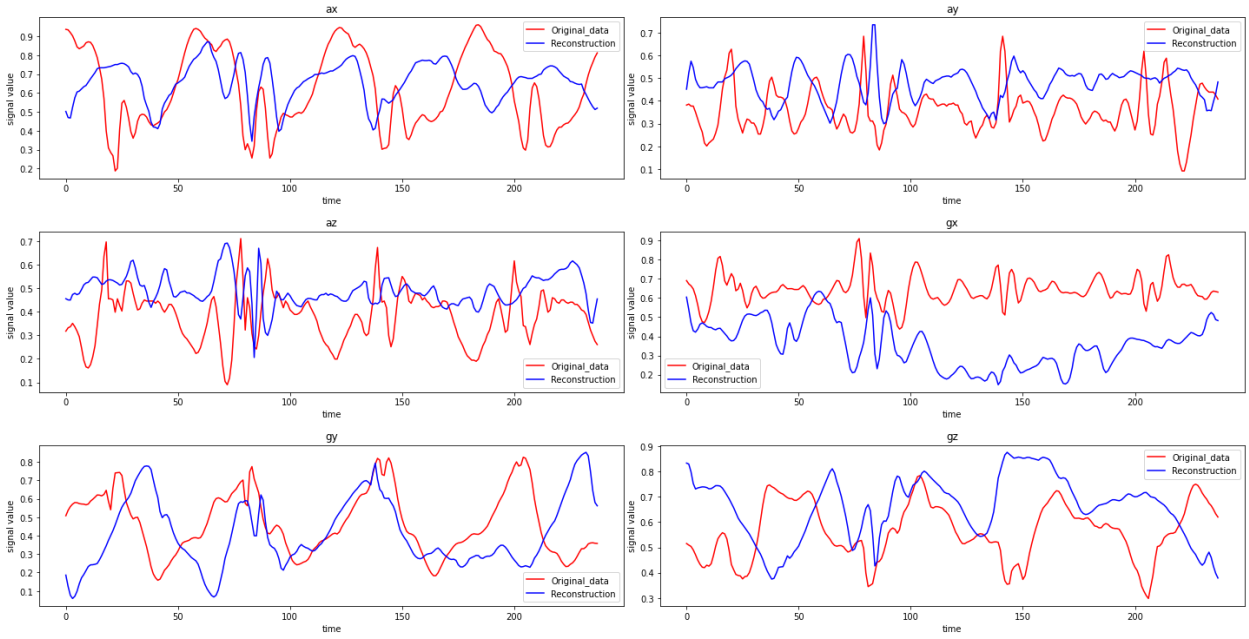

Training loss histogram when des_data is excluded from training
No of examples
Train loss
0
100
200
300
400
500
600
0.00
0.02
0.04
0.06
0.08
0.10
0.12

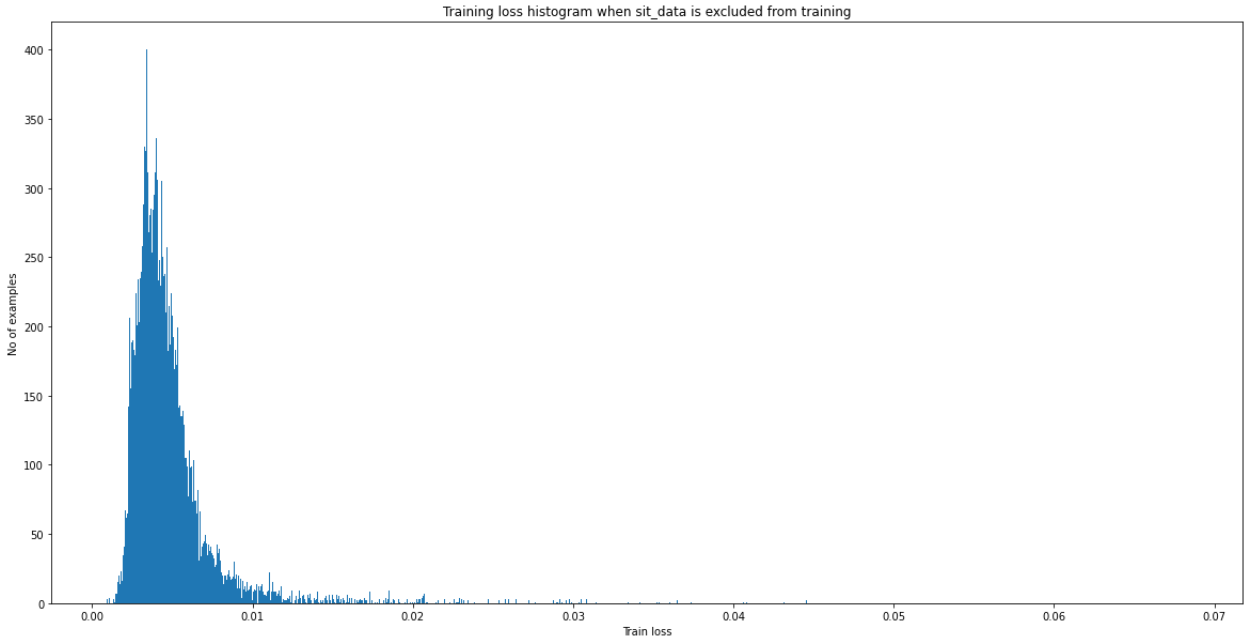

Test loss histogram when des_data is excluded from training
No of examples
Test loss
0
10
20
30
40
50
60
70
80
0.00
0.02
0.04
0.06
0.08

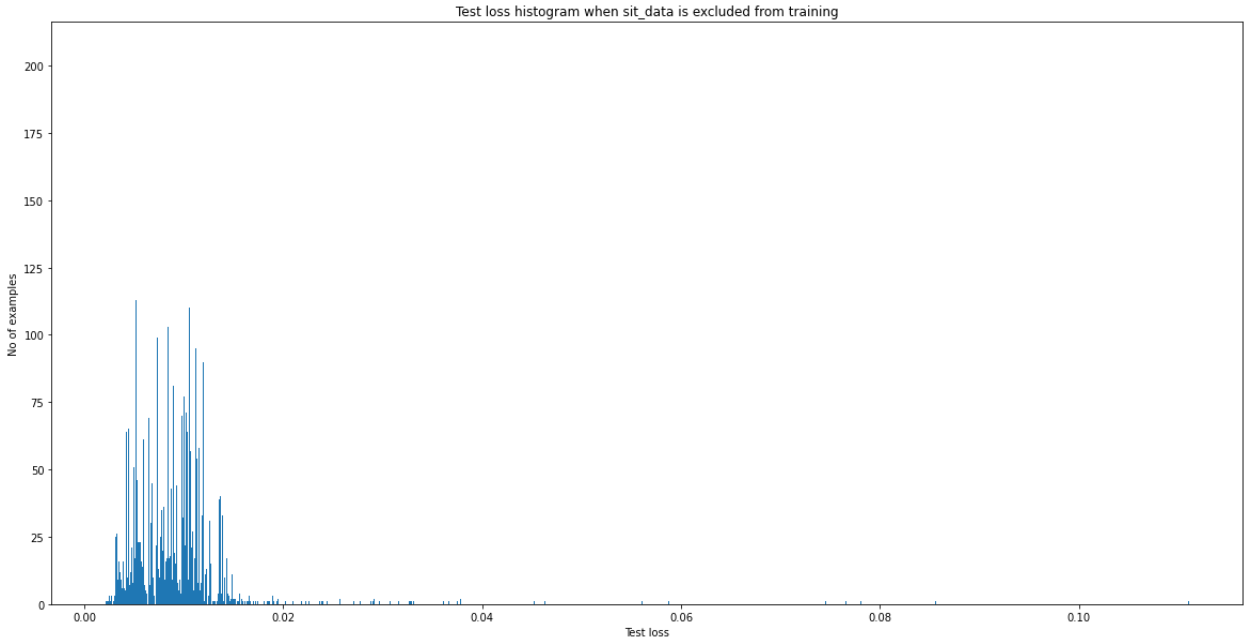

ax
ay
az
gx
gy
gz
Original_data
Reconstruction
signal value
time

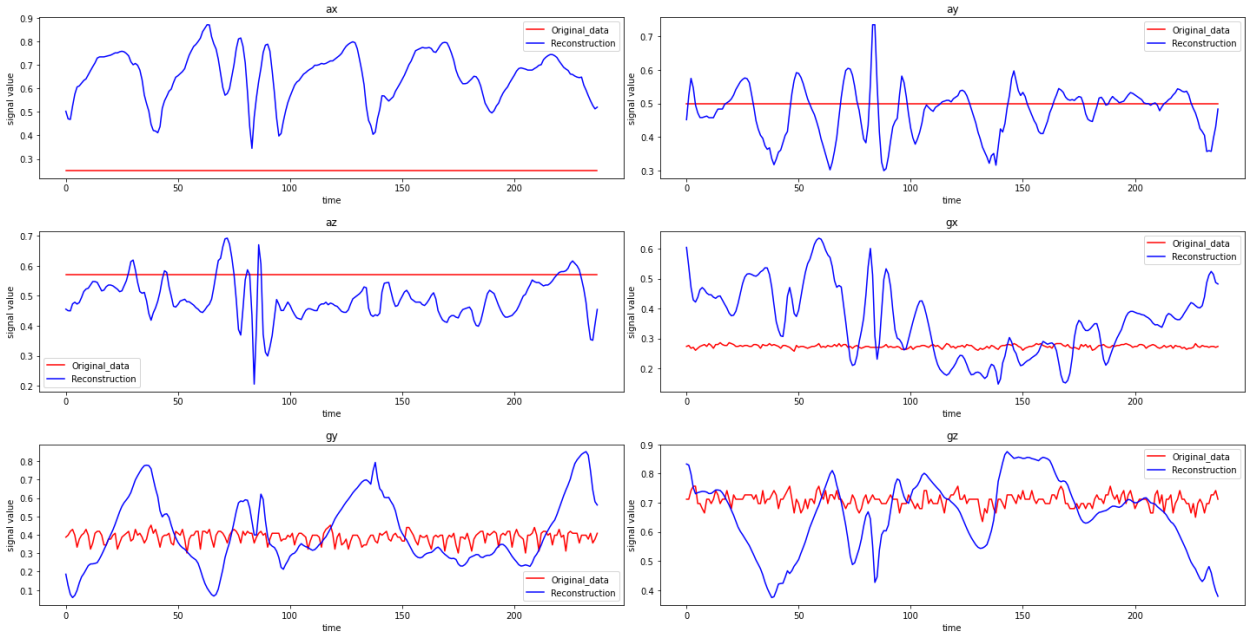

Training loss histogram when jump_data is excluded from training
No of examples
Train loss

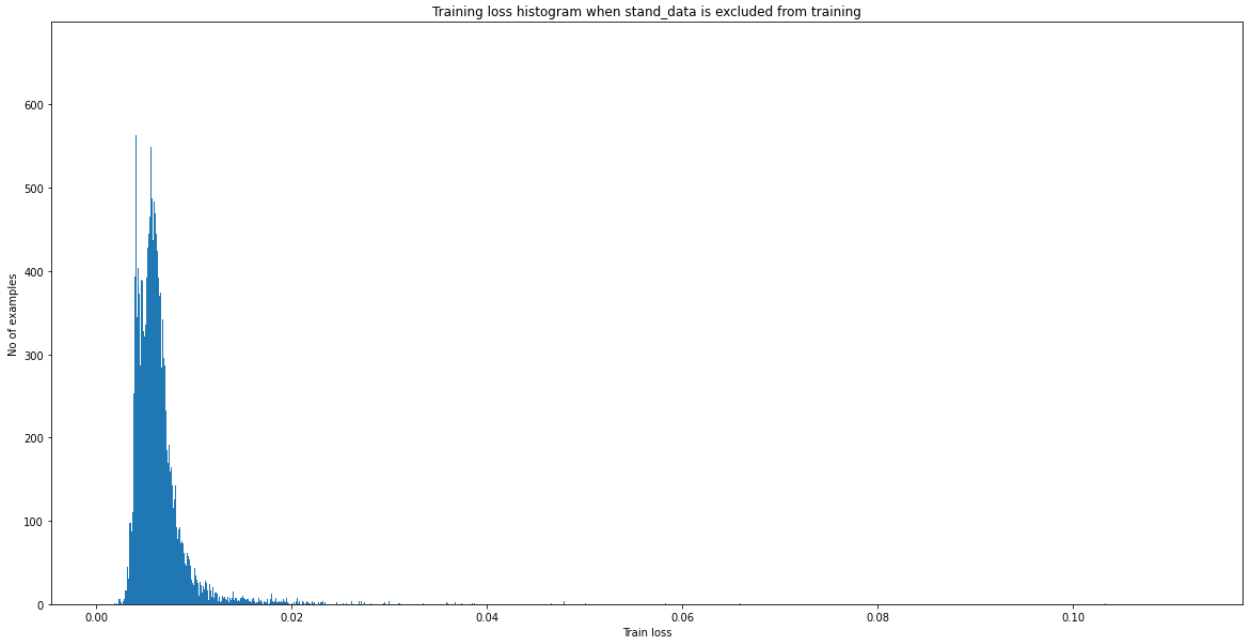

Test loss histogram when jump_data is excluded from training
No of examples
Test loss

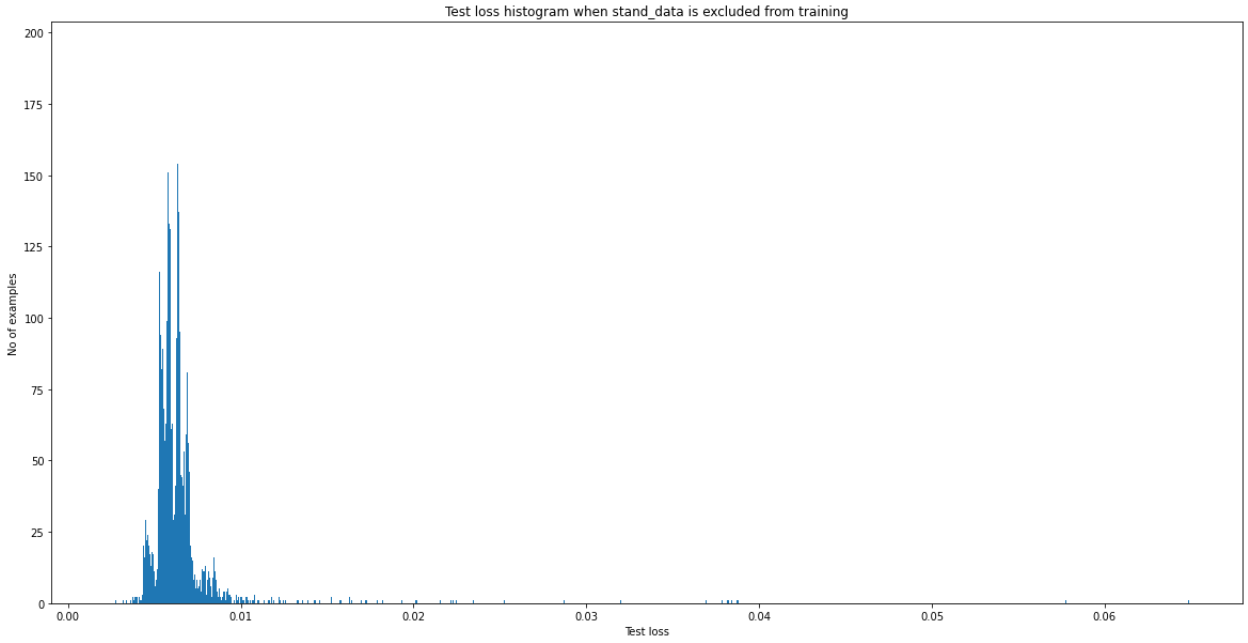

ax
ay
az
gx
gy
gz
signal value
time
Original_data
Reconstruction


Training loss histogram when sit_data is excluded from training
No of examples
Train loss
0
50
100
150
200
250
300
350
400
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07

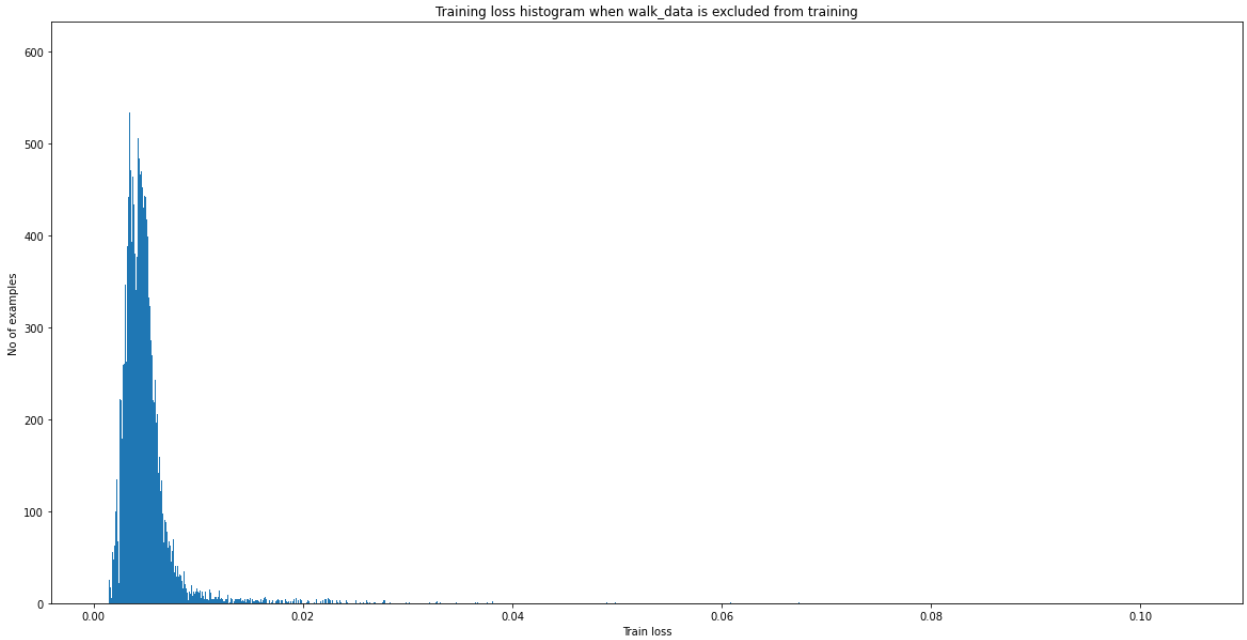

Test loss histogram when sit_data is excluded from training
No of examples
Test loss
0
25
50
75
100
125
150
175
200
0.00
0.02
0.04
0.06
0.08
0.10

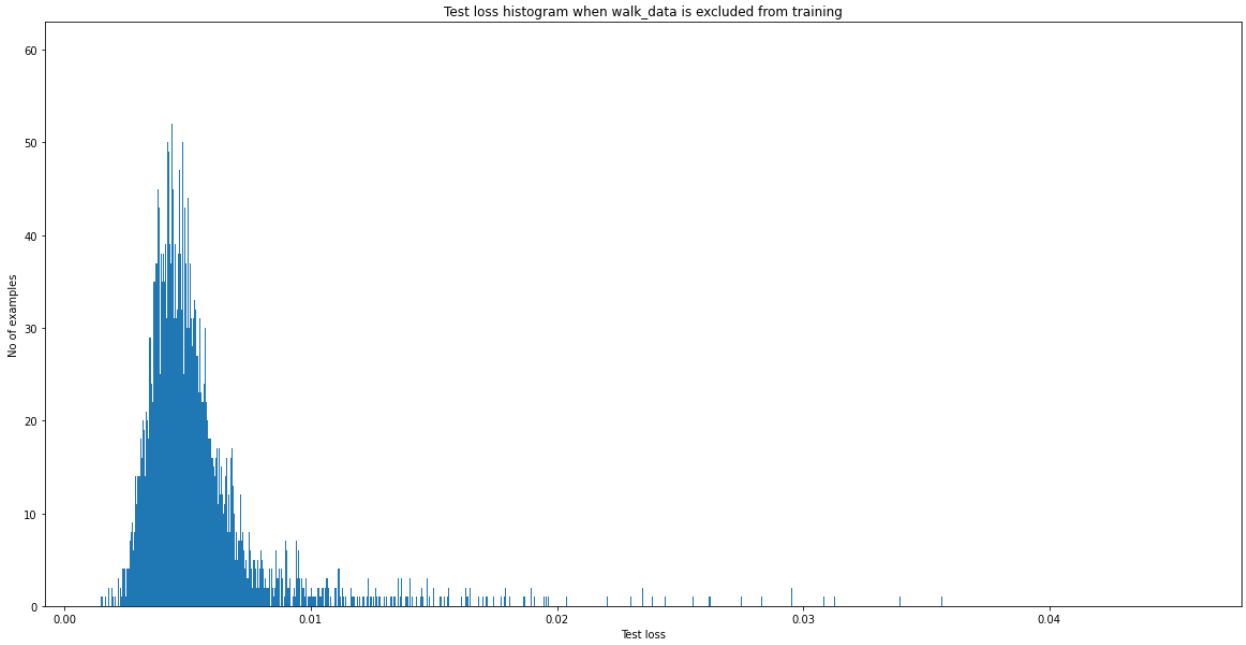

ax
ay
az
gx
gy
gz
Original_data
Reconstruction
signal value
time

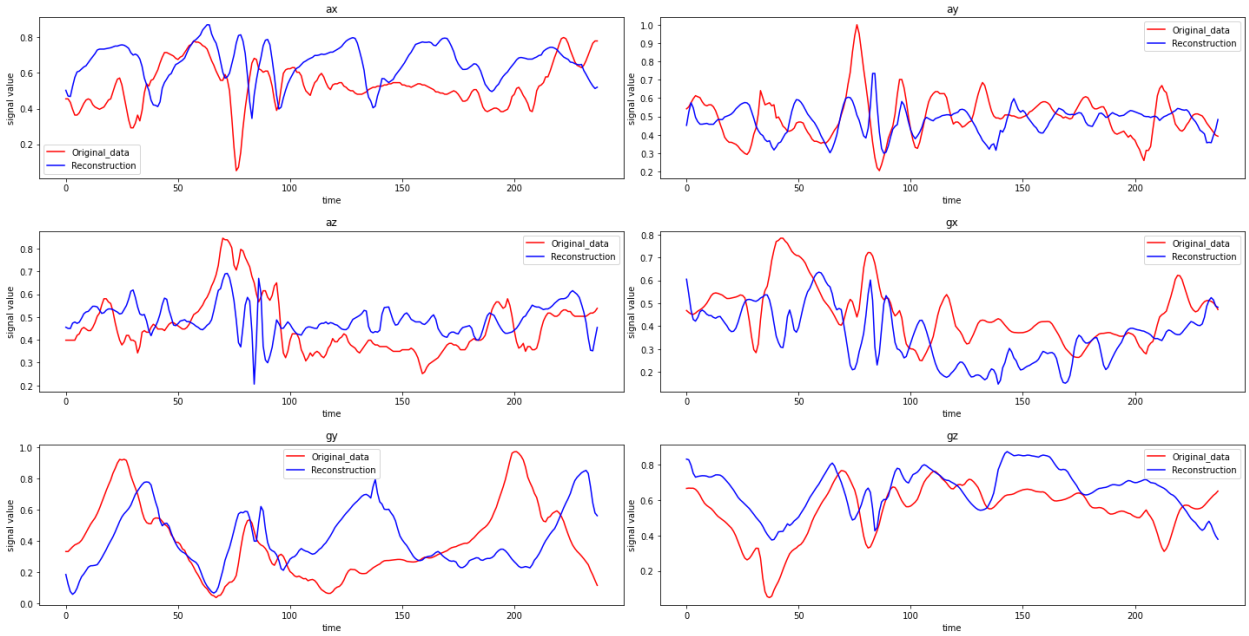

Training loss histogram when stand_data is excluded from training
No of examples
Train loss

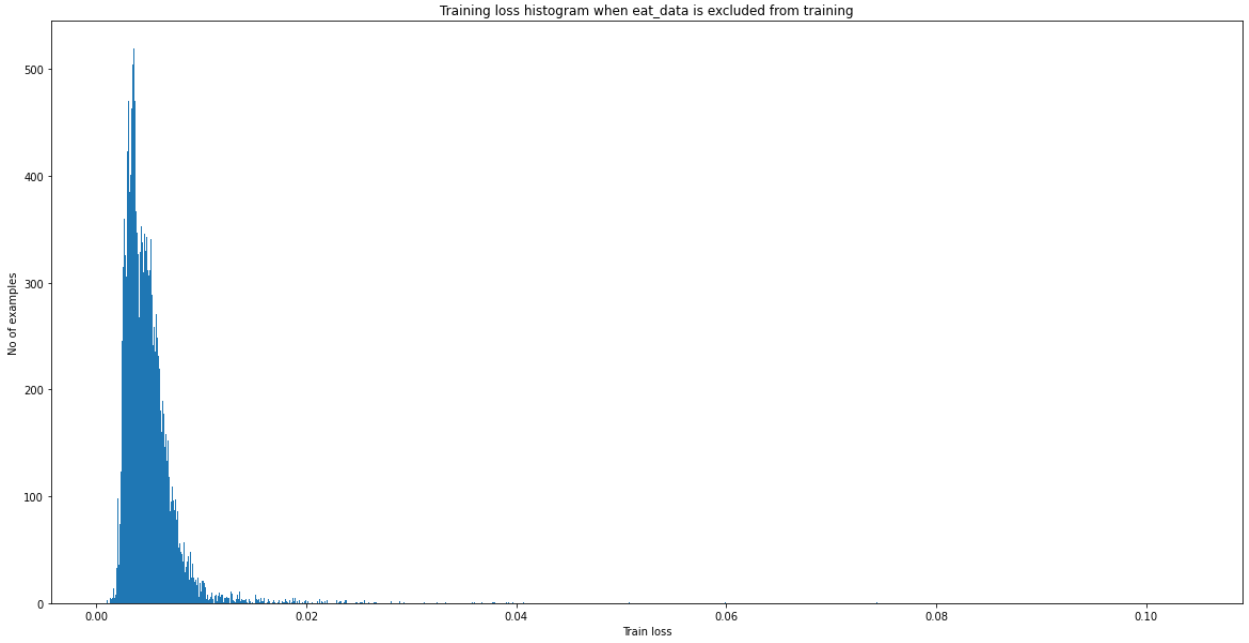

Test loss histogram when stand_data is excluded from training
No of examples
Test loss

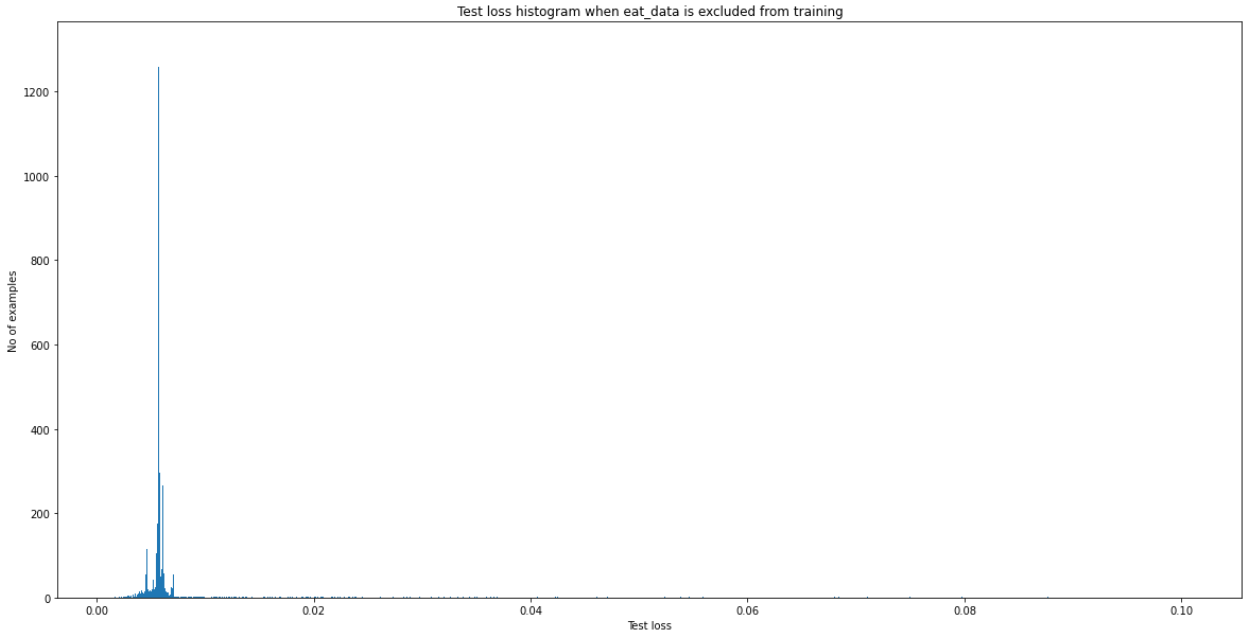

ax
ay
az
gx
gy
gz
signal value
time
Original_data
Reconstruction

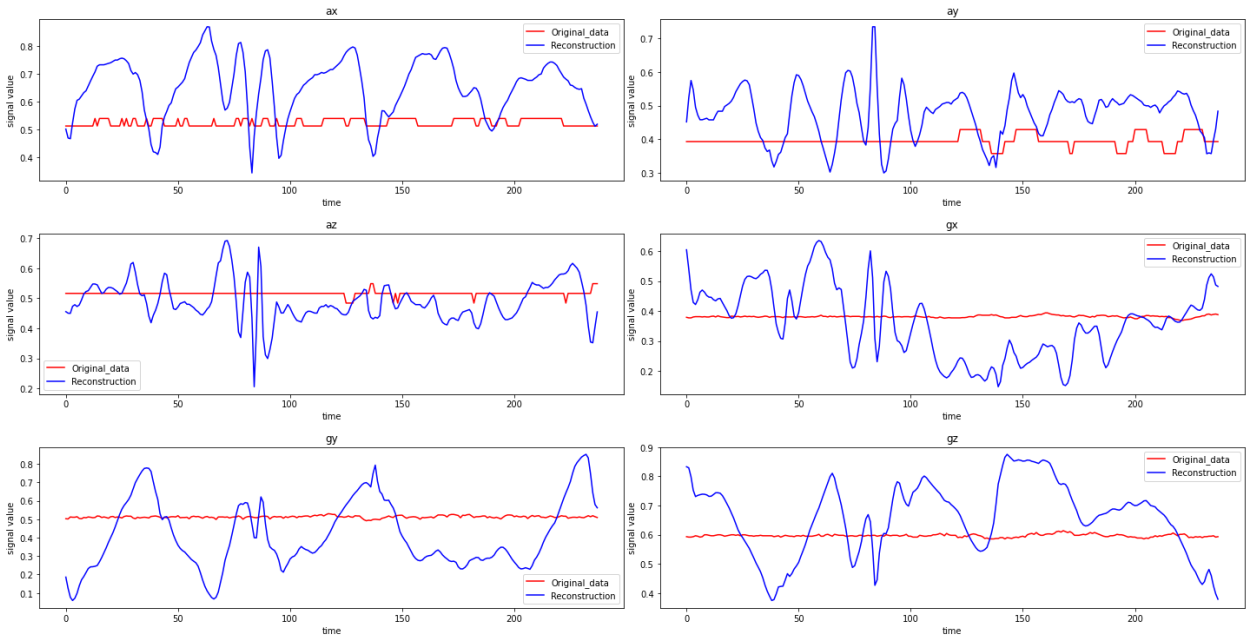

Training loss histogram when walk_data is excluded from training
No of examples
Train loss
0
100
200
300
400
500
600
0.00
0.02
0.04
0.06
0.08
0.10

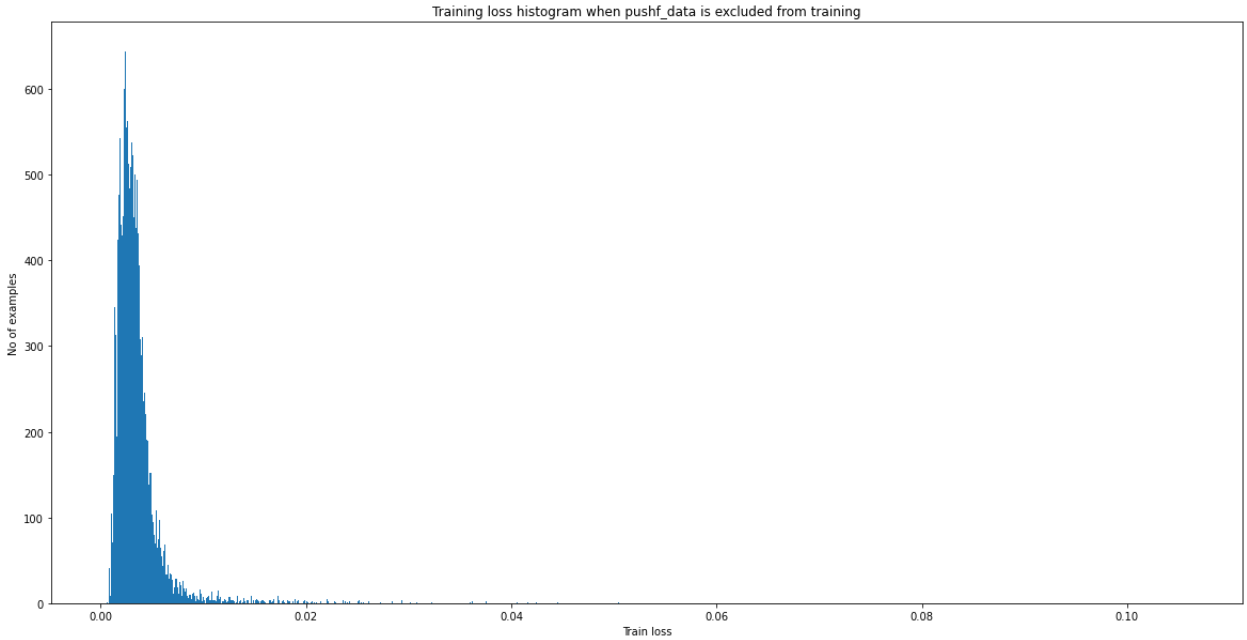

Test loss histogram when walk_data is excluded from training
No of examples
Test loss
0
10
20
30
40
50
60
0.00
0.01
0.02
0.03
0.04

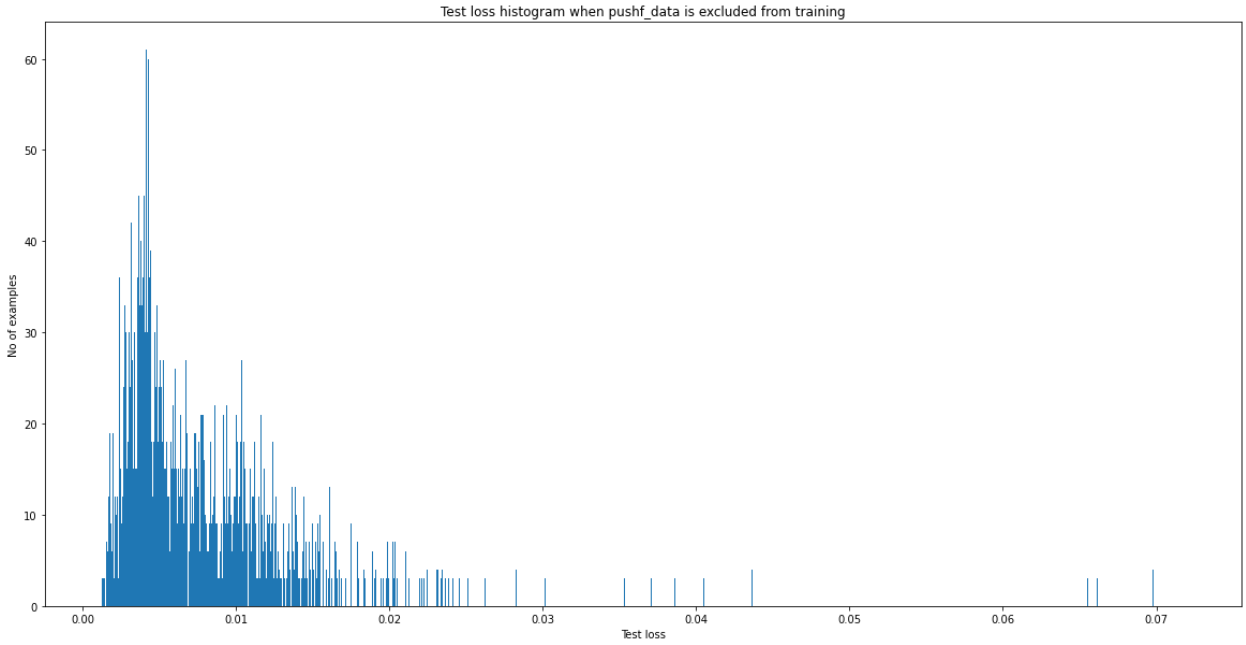

ax
ay
az
gx
gy
gz
Original_data
Reconstruction
signal value
time

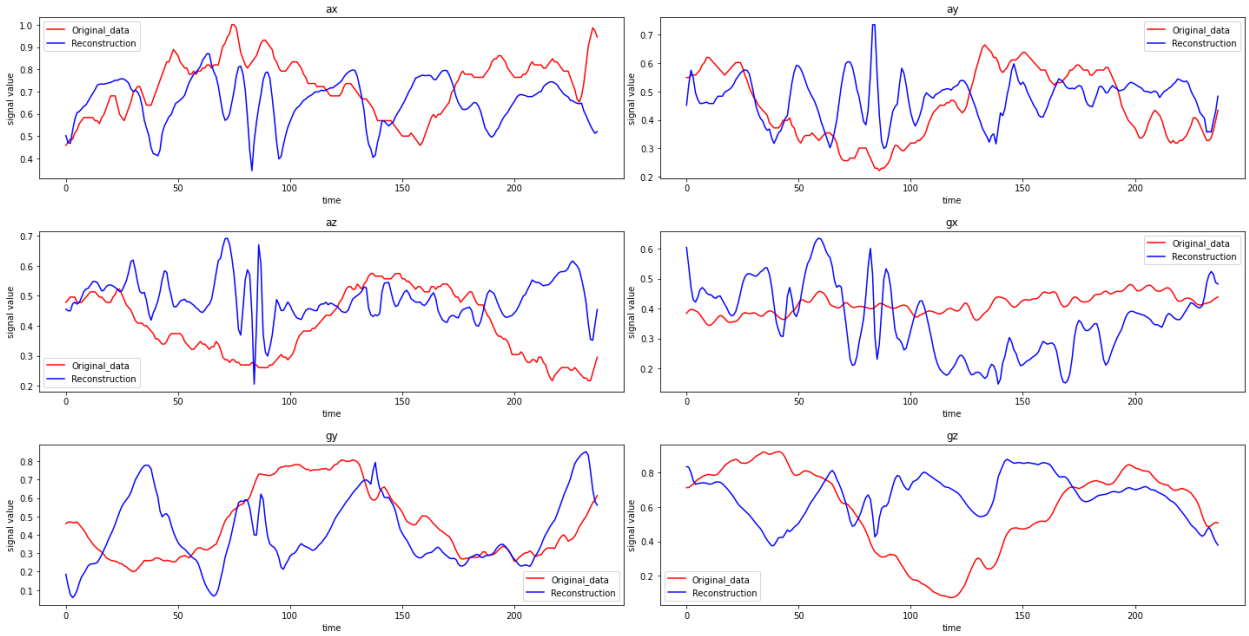

Training loss histogram when eat_data is excluded from training
No of examples
Train loss

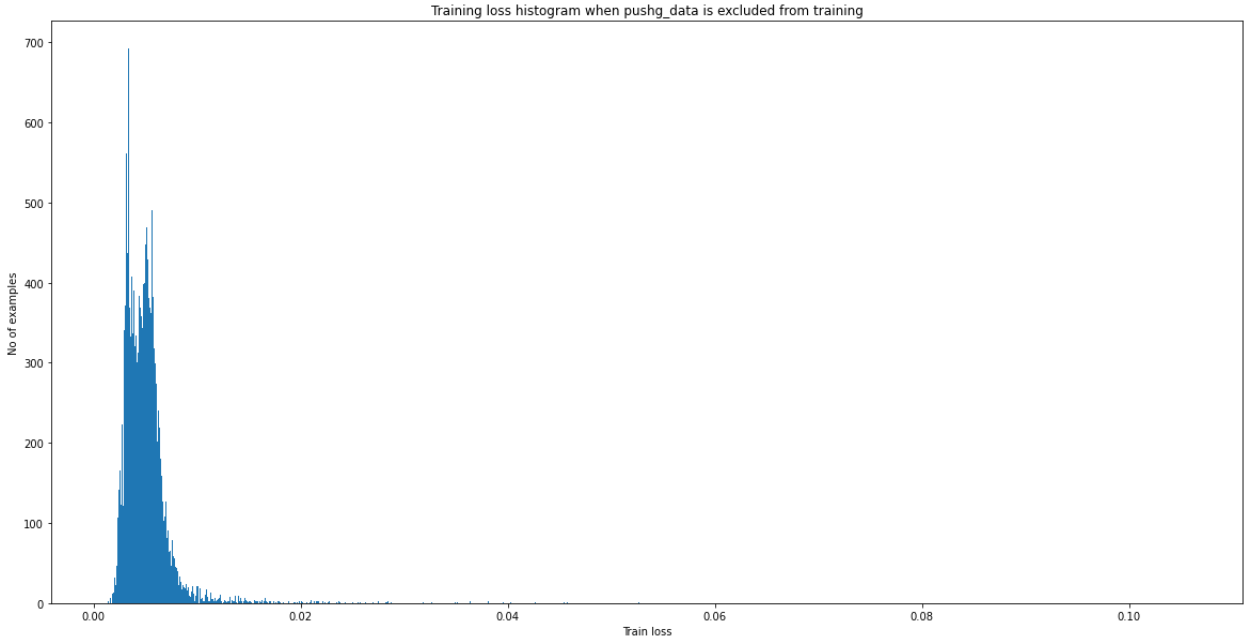

Test loss histogram when eat_data is excluded from training
No of examples
Test loss
0
200
400
600
800
1000
1200
0.00
0.02
0.04
0.06
0.08
0.10

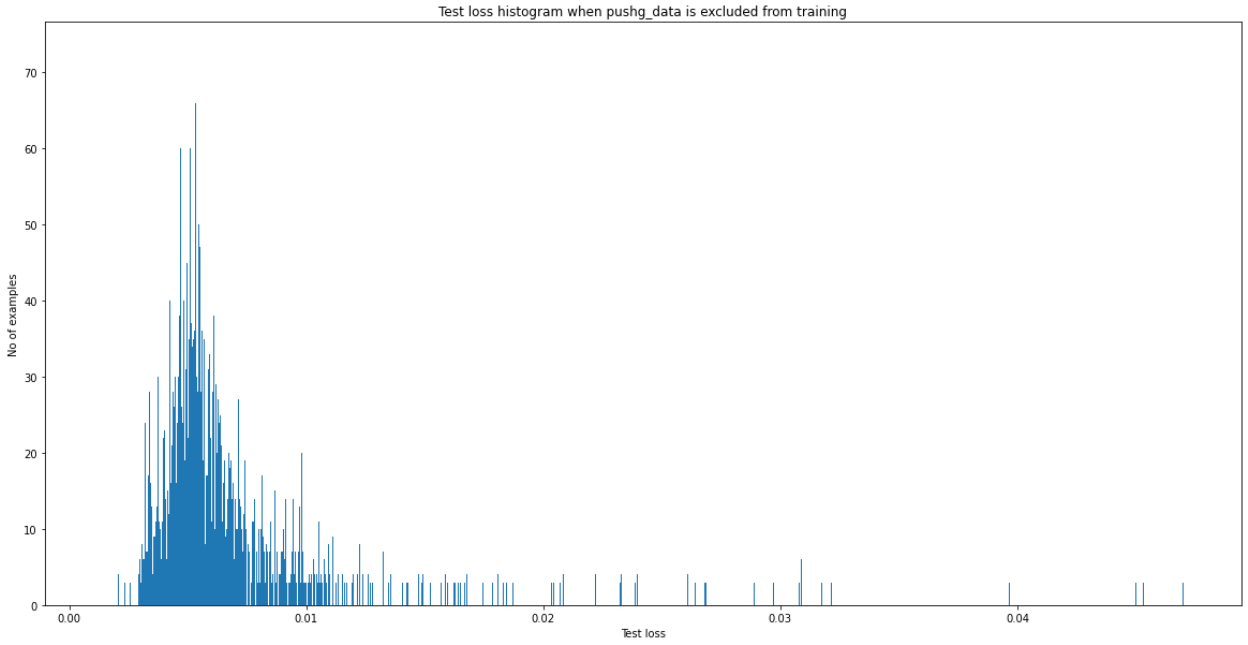

ax
ay
az
gx
gy
gz
Original_data
Reconstruction
signal value
time

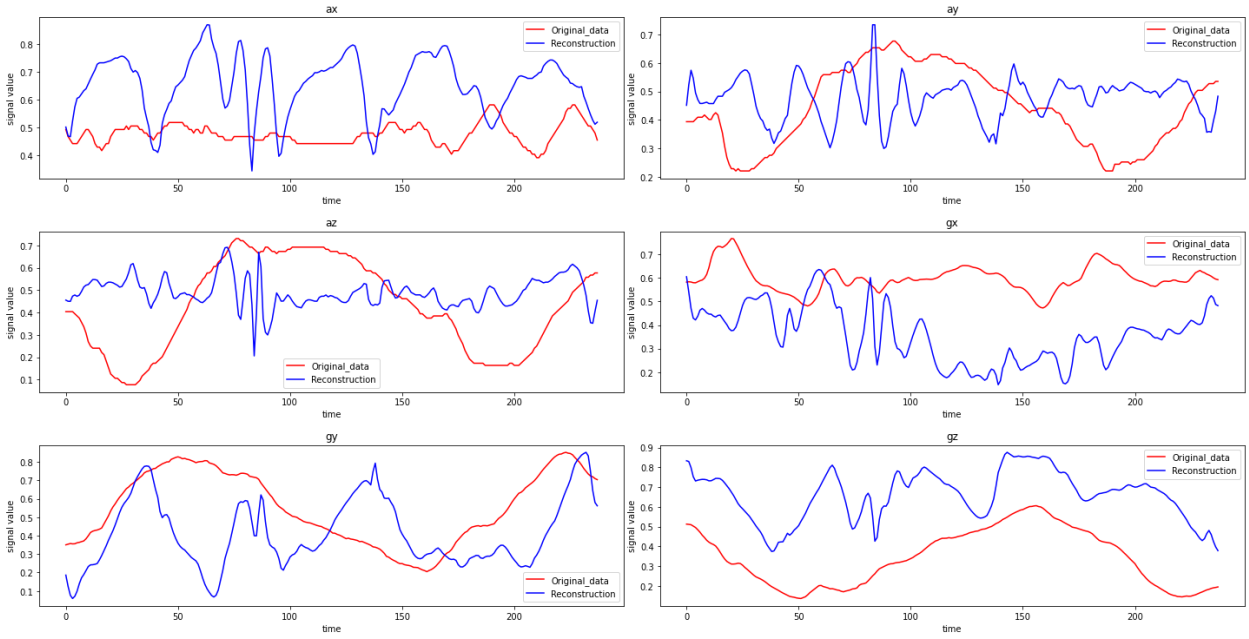

Training loss histogram when pushf_data is excluded from training
No of examples
Train loss
0
100
200
300
400
500
600
0.00
0.02
0.04
0.06
0.08
0.10

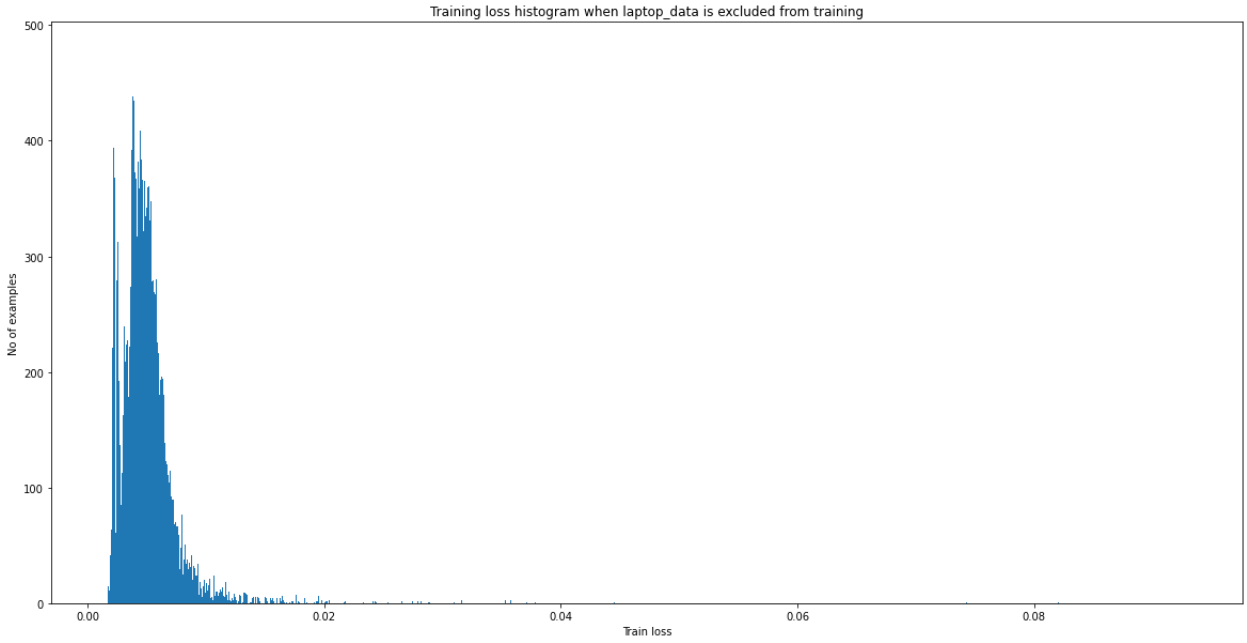

Test loss histogram when pushf_data is excluded from training
No of examples
Test loss
0
10
20
30
40
50
60
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07

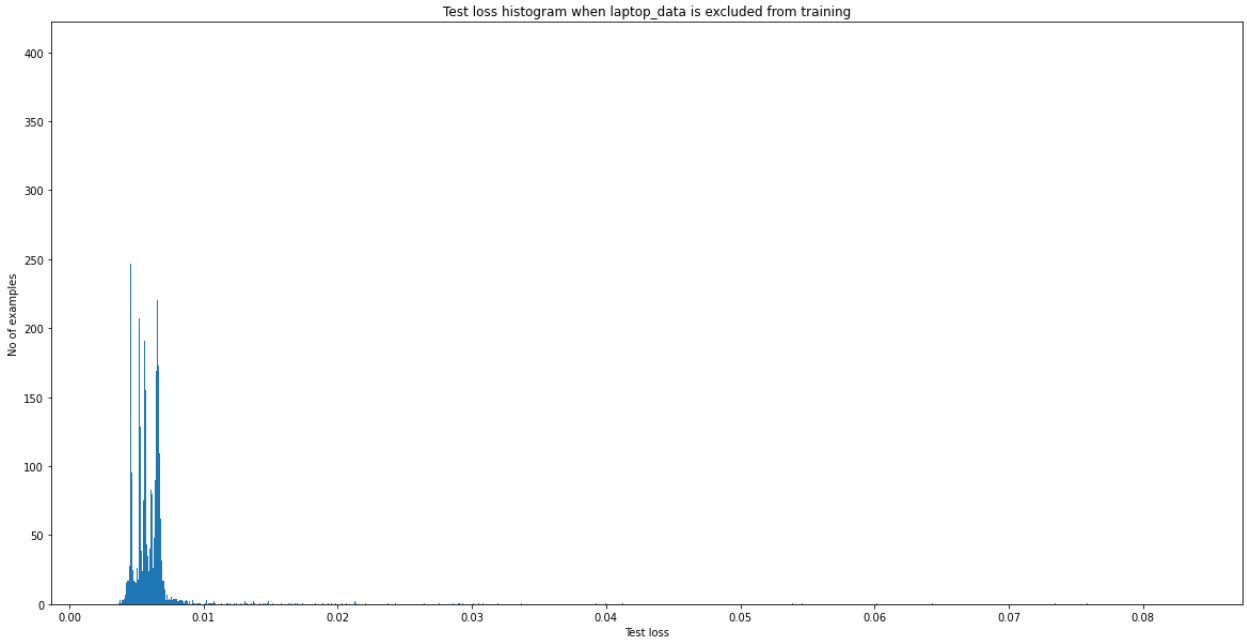

ax
ay
az
gx
gy
gz
Original_data
Reconstruction
signal value
time

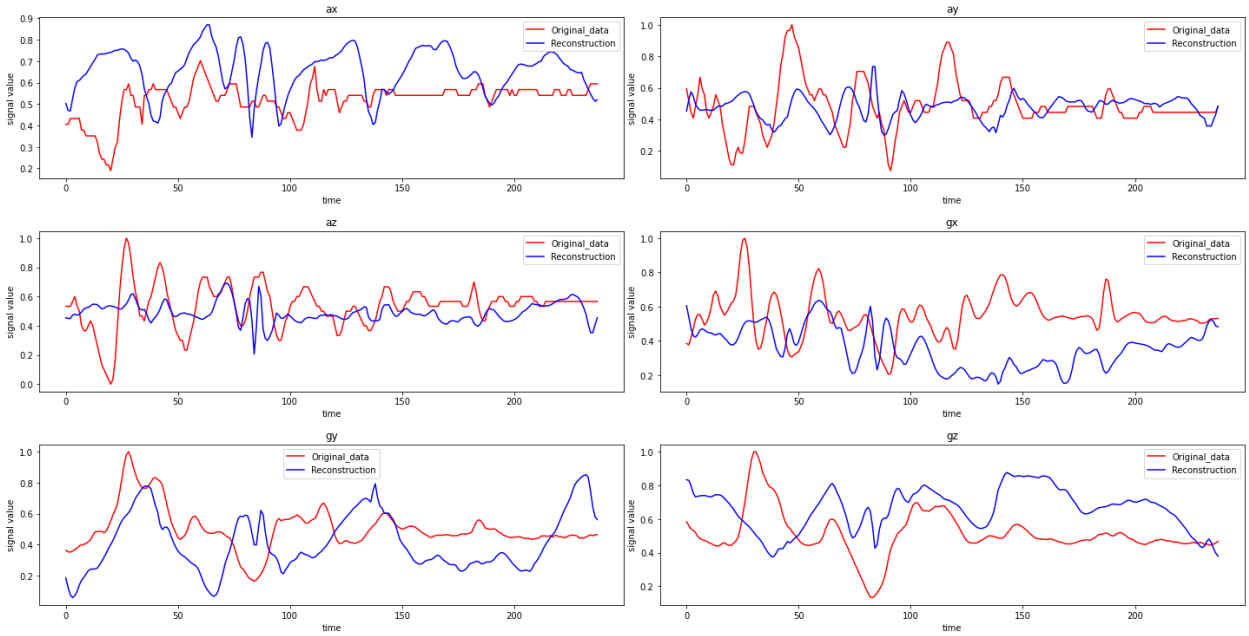

Training loss histogram when pushg_data is excluded from training
No of examples
Train loss

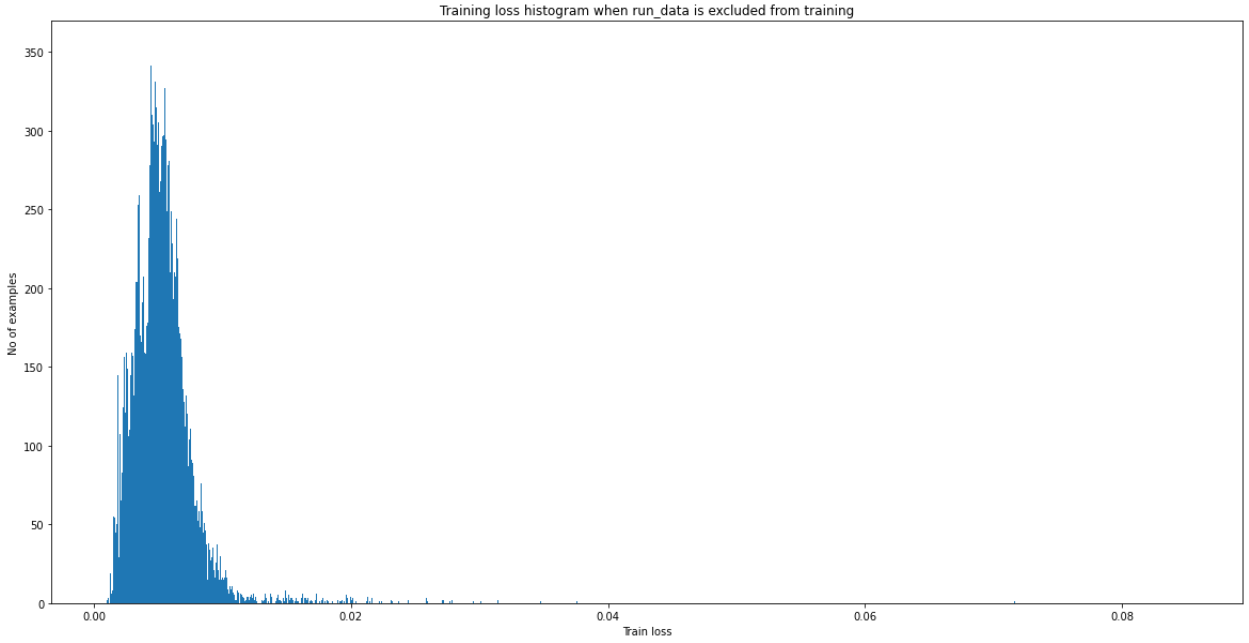

Test loss histogram when pushg_data is excluded from training
No of examples
Test loss

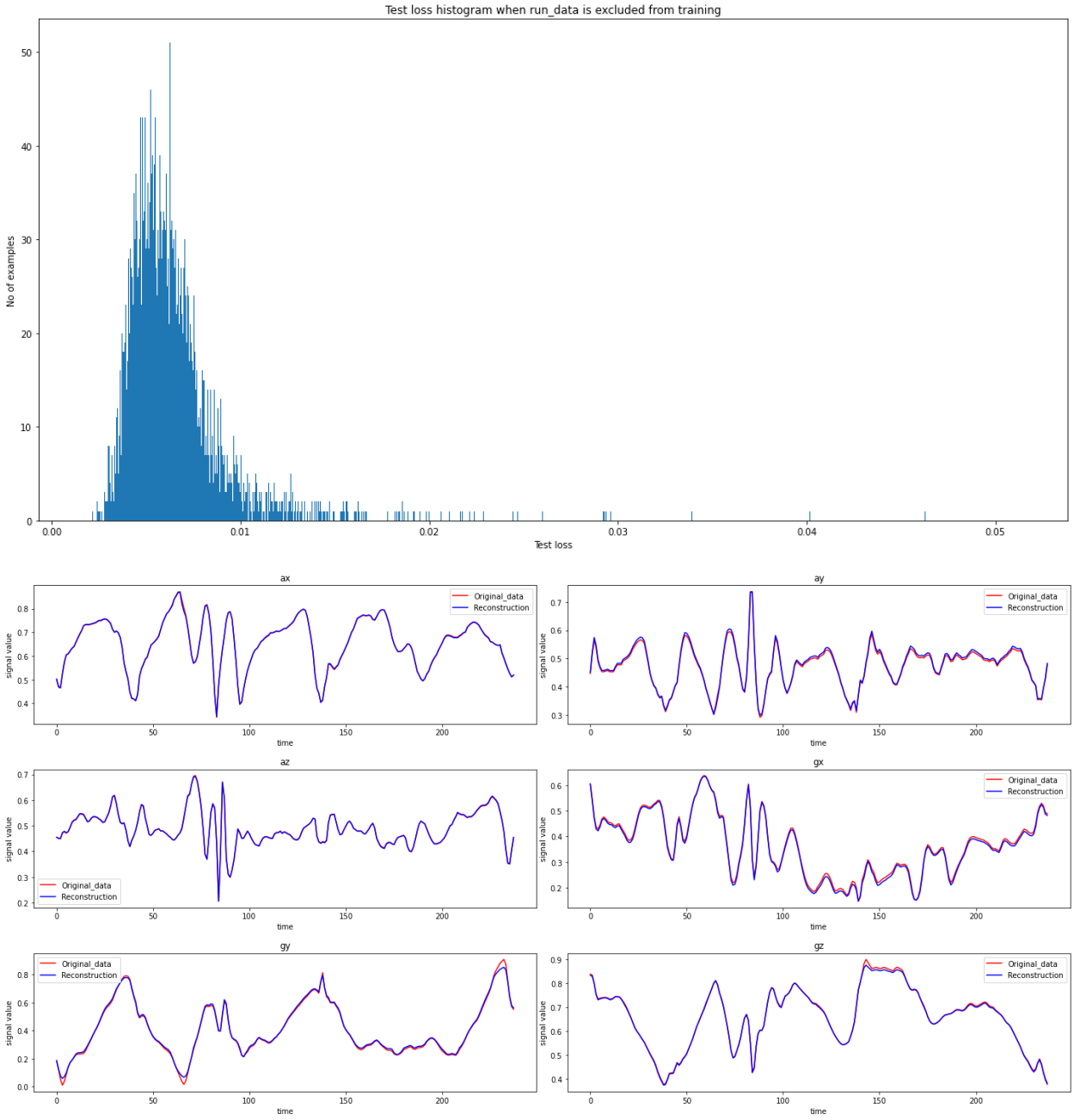

ax
ay
az
gx
gy
gz
signal value
time
Original_data
Reconstruction


Training loss histogram when laptop_data is excluded from training
No of examples
Train loss
0
100
200
300
400
500
0.00
0.02
0.04
0.06
0.08


Test loss histogram when laptop_data is excluded from training
No of examples
Test loss
0
50
100
150
200
250
300
350
400
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08


ax
ay
az
gx
gy
gz
Original_data
Reconstruction
signal value
time


Training loss histogram when run_data is excluded from training
No of examples
Train loss



- The above graphs indicate that all activities (except for running activity) can be detected as an anomalous signal using the autoencoder experiment. One reason that autoencoder is able to nearly perfectly reconstruct running data is that running data might be resembling walking data (or could be dancing data)

Observations from Model Training (Step3):

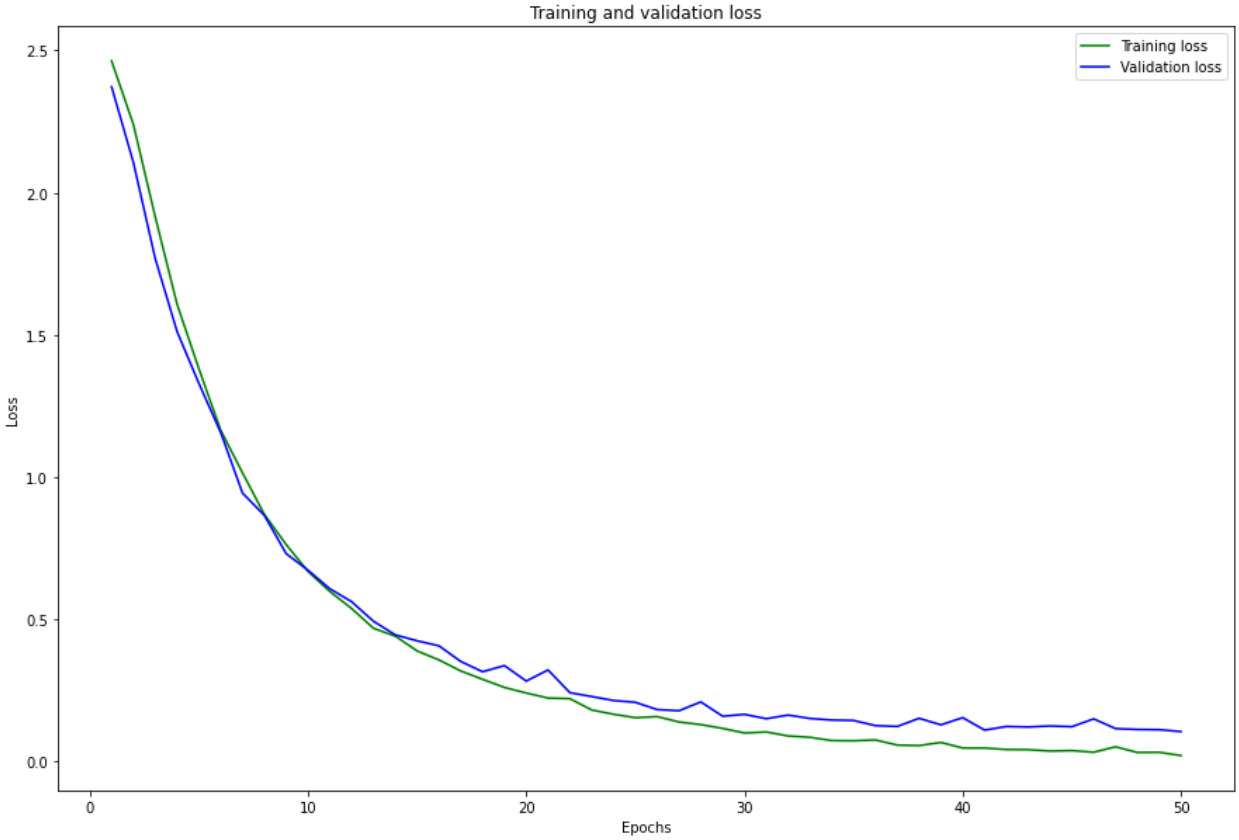
Model1:

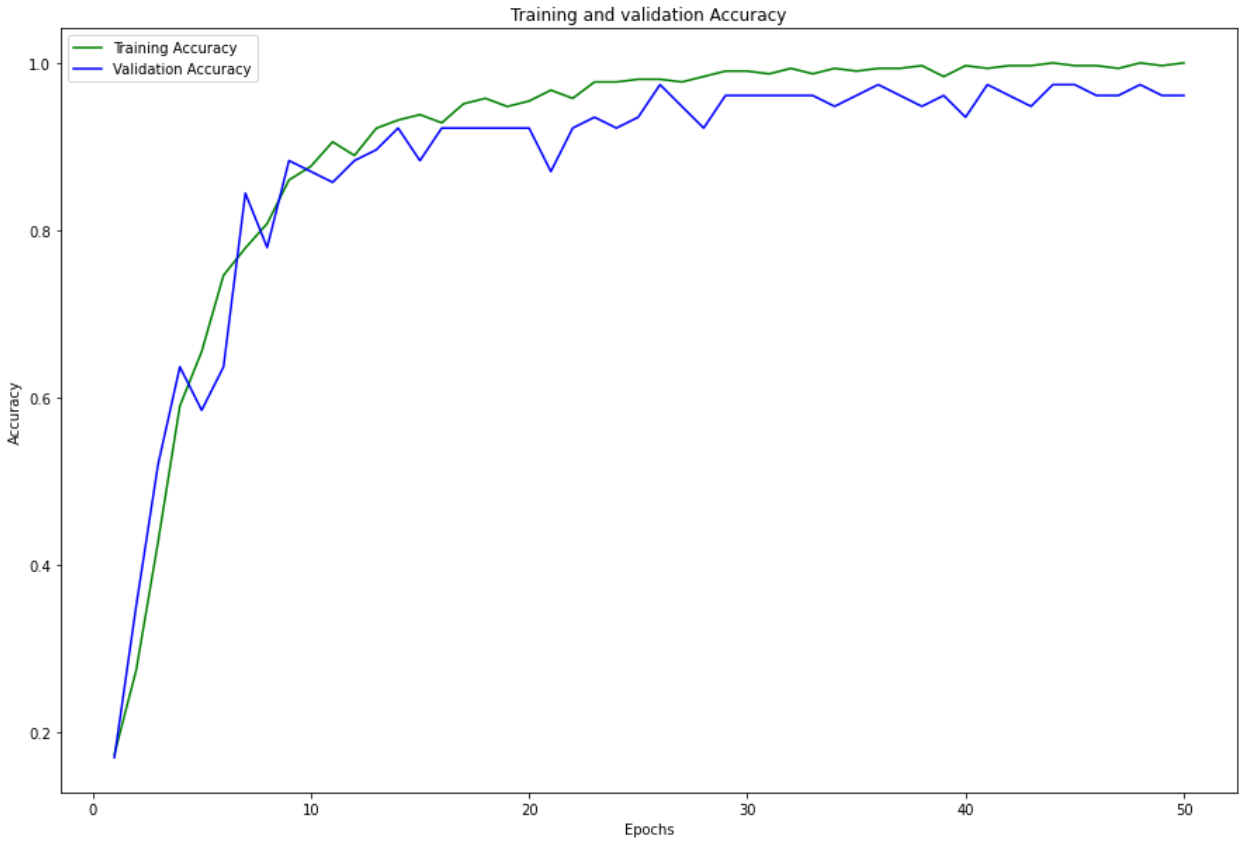
- This model is trained on extracted features using a feature extractor
- Segment length = 3 min → each training example is extracted from 357 samples

- Model was trained for 50 epochs with batch size = 2 (after 50 epochs, model was over training)
- Below are test set metrics for different architectures that I tried

Architecture (nodes per layer)	Test accuracy	Precision	Recall	F1 score
[16, 16]	0.84	0.84	0.85	0.84
[32, 16]	0.83	0.83	0.83	0.83
[32, 32]	0.98	0.98	0.98	0.98
[16, 16, 16]	0.87	0.88	0.87	0.88
[32, 32, 16]	0.94	0.95	0.94	0.94

- From above experiments it seems that [32, 32] neural network worked best for this data
- Below are the loss, accuracy history graphs for [32, 32] neural network





- Below is the test set confusion matrix for the [32, 32] neural network

```
              asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data             9         0           0          0         0           0
des_data             0         9           0          0         0           0
dance_data           0         0           5          0         0           0
jump_data            0         0           0          6         0           0
sit_data             0         0           0          0         9           0
stand_data           0         0           0          0         0           7
walk_data            0         0           0          0         0           0
eat_data             0         0           0          0         0           0
pushf_data           0         0           0          0         0           0
pushg_data           0         0           0          0         0           0
laptop_data          0         0           1          0         0           0
run_data             0         0           0          0         0           0

              walk_data  eat_data  pushf_data  pushg_data  laptop_data  \
asc_data              0         0           0           0            0
des_data              0         0           0           0            0
dance_data            0         0           0           0            0
jump_data             0         0           0           0            0
sit_data              0         0           0           0            0
stand_data            0         0           0           0            0
walk_data            10         0           0           0            0
eat_data              0         9           0           0            0
pushf_data            0         0           8           0            1
pushg_data            0         0           0           8            0
laptop_data           0         0           0           0            6
run_data              0         0           0           0            0

              run_data
asc_data             0
des_data             0
dance_data           0
jump_data            0
sit_data             0
stand_data           0
walk_data            0
eat_data             0
pushf_data           0
pushg_data           0
laptop_data          0
run_data             8
```

- For this network architecture, size of the normal binary executable is 103,314 bytes and size of quantized binary executable is 44,238 bytes

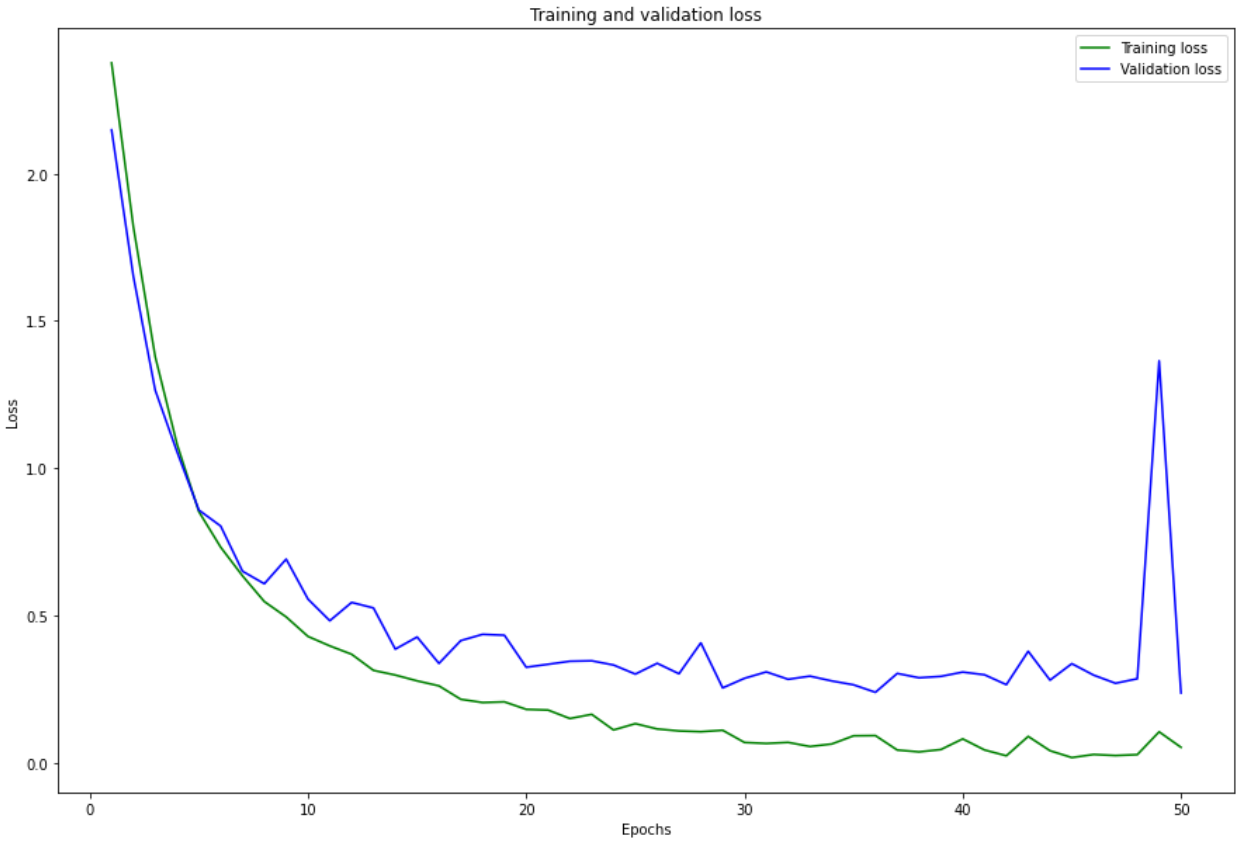
Model2:

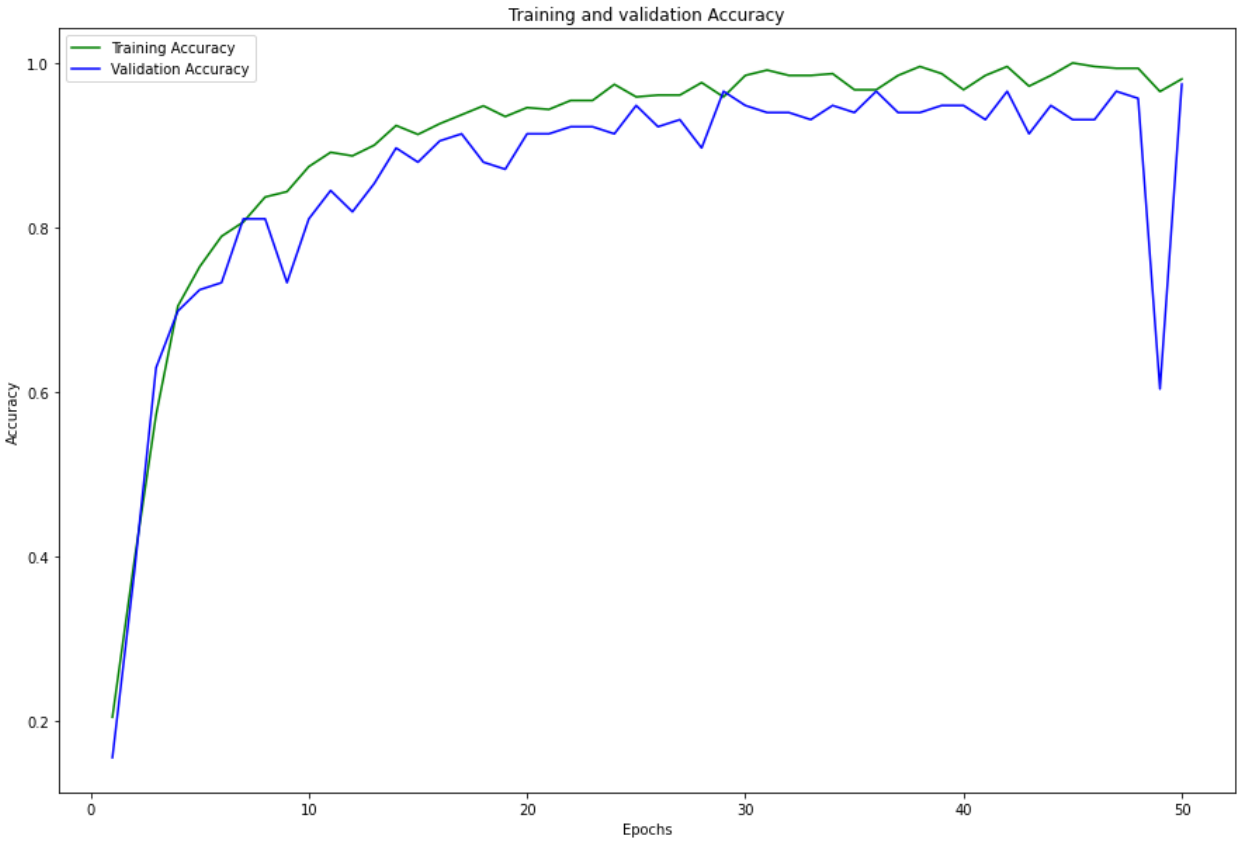
- This model is trained on extracted features using a feature extractor
- Segment length = 2 min → each training example is extracted from 238 samples
- Model was trained for 50 epochs with batch size = 2 (after 50 epochs, model was over training)
- Below are test set metrics for different architectures that I tried

Architecture (nodes per layer)	Test accuracy	Precision	Recall	F1 score

[16, 16]	0.94	0.94	0.94	0.94
[32, 16]	0.94	0.94	0.94	0.94
[32, 32]	0.92	0.93	0.92	0.92
[16, 16, 16]	0.88	0.90	0.88	0.90
[32, 32, 32]	0.98	0.99	0.98	0.99

- From above experiments it seems that [32, 32, 32] neural network worked best for this data
- Below are the loss, accuracy history graphs for [32, 32, 32] neural network





- Below is the test set confusion matrix for the [32, 32, 32] neural network

```
Confusion Matrix:
              asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data            10         0           0          0         0           0
des_data             0        10           0          0         0           0
dance_data           0         0          12          0         0           0
jump_data            0         0           0         15         0           0
sit_data             0         0           0          0        14           0
stand_data           0         0           0          0         0           9
walk_data            0         0           0          0         0           0
eat_data             0         0           0          0         0           0
pushf_data           0         1           0          0         0           0
pushg_data           0         0           0          0         0           0
laptop_data          0         0           0          0         0           0
run_data             0         0           0          0         0           0

              walk_data  eat_data  pushf_data  pushg_data  laptop_data  \
asc_data              0         0           0           0            0
des_data              0         0           0           0            0
dance_data            0         0           0           0            0
jump_data             0         0           0           0            0
sit_data              0         0           0           0            0
stand_data            0         0           0           0            0
walk_data            14         0           0           0            0
eat_data              0        11           0           0            0
pushf_data            0         0          11           0            0
pushg_data            0         0           0          10            0
laptop_data           0         0           1           0           14
run_data              0         0           0           0            0

              run_data
asc_data             0
des_data             0
dance_data           0
jump_data            0
sit_data             0
stand_data           0
walk_data            0
eat_data             1
pushf_data           0
pushg_data           0
laptop_data          0
run_data            11
```

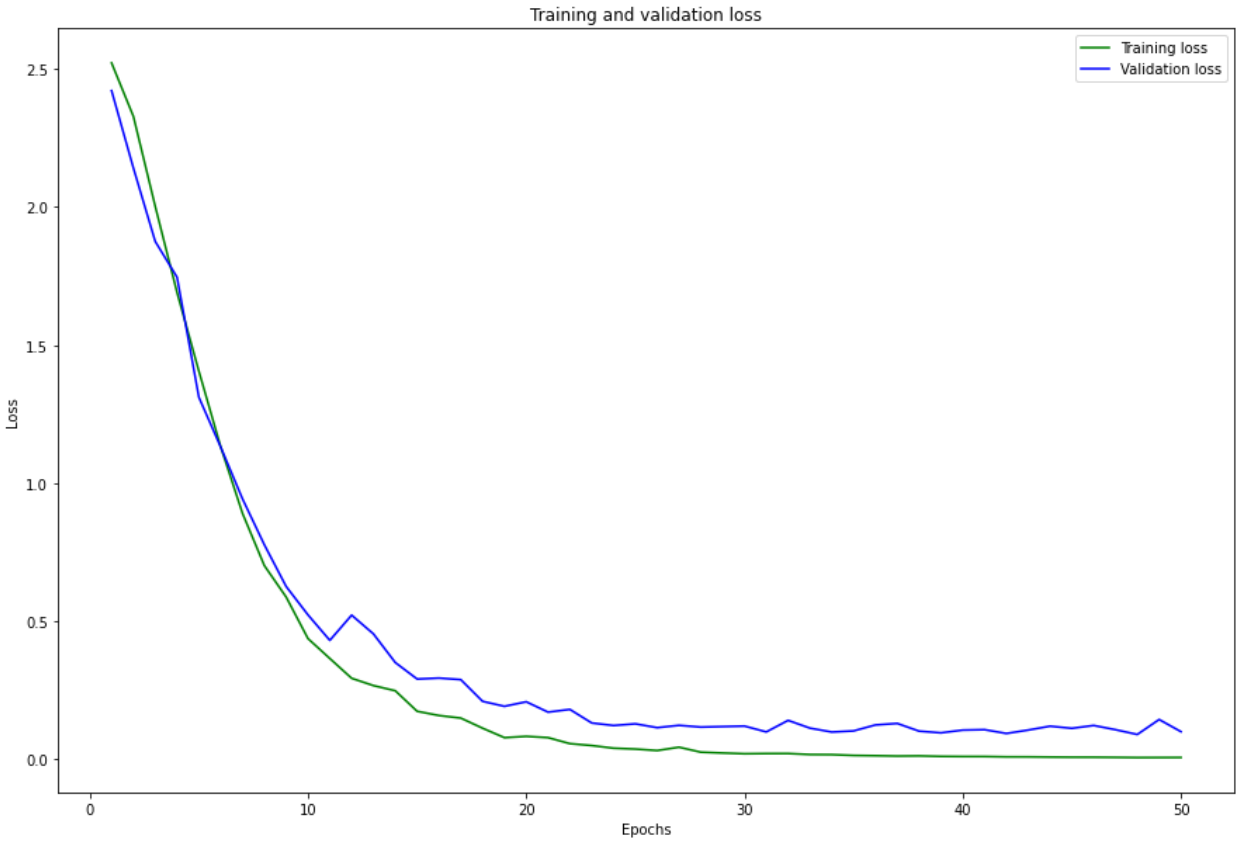
- For this network architecture, size of the normal binary executable is 121,098 bytes and size of quantized binary executable is 48,974 bytes

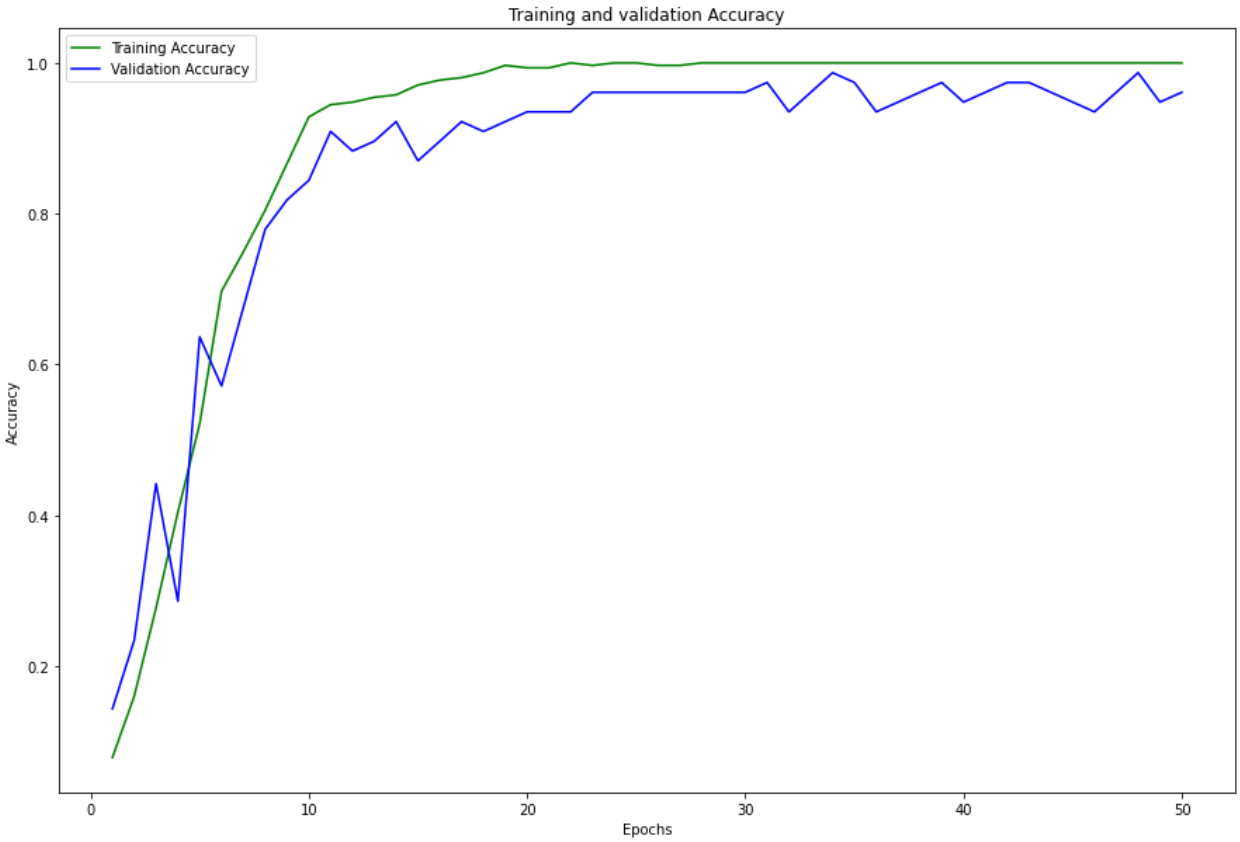
Model3:

- This model is trained on raw data
- Segment length = 3 min → each training example is extracted from 357 samples → 2142 features in each example
- Model was trained for 50 epochs with batch size = 2 (after 50 epochs, model was over training)
- Below are test set metrics for different architectures that I tried

Architecture (nodes per layer)	Test accuracy	Precision	Recall	F1 score
[16, 16]	0.69	0.70	0.70	0.68
[64, 32]	0.94	0.95	0.94	0.94
[64, 64]	0.68	0.68	0.68	0.65
[64, 32, 32]	0.95	0.96	0.95	0.95
[64, 64, 32]	0.92	0.93	0.92	0.91

- From above experiments it seems that [64, 32, 32] neural network worked best for this data
- Below are the loss, accuracy history graphs for [64, 32, 32] neural network





- Below is the test set confusion matrix for the [64, 32, 32] neural network

```
Confusion Matrix:
              asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data             7         0           0          2         0           0
des_data             0         9           0          0         0           0
dance_data           0         1           4          0         0           0
jump_data            0         0           0          6         0           0
sit_data             0         0           0          0         9           0
stand_data           0         0           0          0         0           7
walk_data            0         0           0          0         0           0
eat_data             0         0           0          0         0           0
pushf_data           0         0           0          0         0           0
pushg_data           0         0           0          0         0           0
laptop_data          0         0           0          0         0           0
run_data             0         1           0          0         0           0

              walk_data  eat_data  pushf_data  pushg_data  laptop_data  \
asc_data              0         0           0           0            0
des_data              0         0           0           0            0
dance_data            0         0           0           0            0
jump_data             0         0           0           0            0
sit_data              0         0           0           0            0
stand_data            0         0           0           0            0
walk_data            10         0           0           0            0
eat_data              0         9           0           0            0
pushf_data            0         0           9           0            0
pushg_data            0         0           0           8            0
laptop_data           0         0           0           0            7
run_data              1         0           0           0            0

              run_data
asc_data             0
des_data             0
dance_data           0
jump_data            0
sit_data             0
stand_data           0
walk_data            0
eat_data             0
pushf_data           0
pushg_data           0
laptop_data          0
run_data             6
```

- For this network architecture, size of the normal binary executable is 3,486,002 bytes and size of quantized binary executable is 890,798 bytes. I think model sizes are huge because of very high dimension of the input

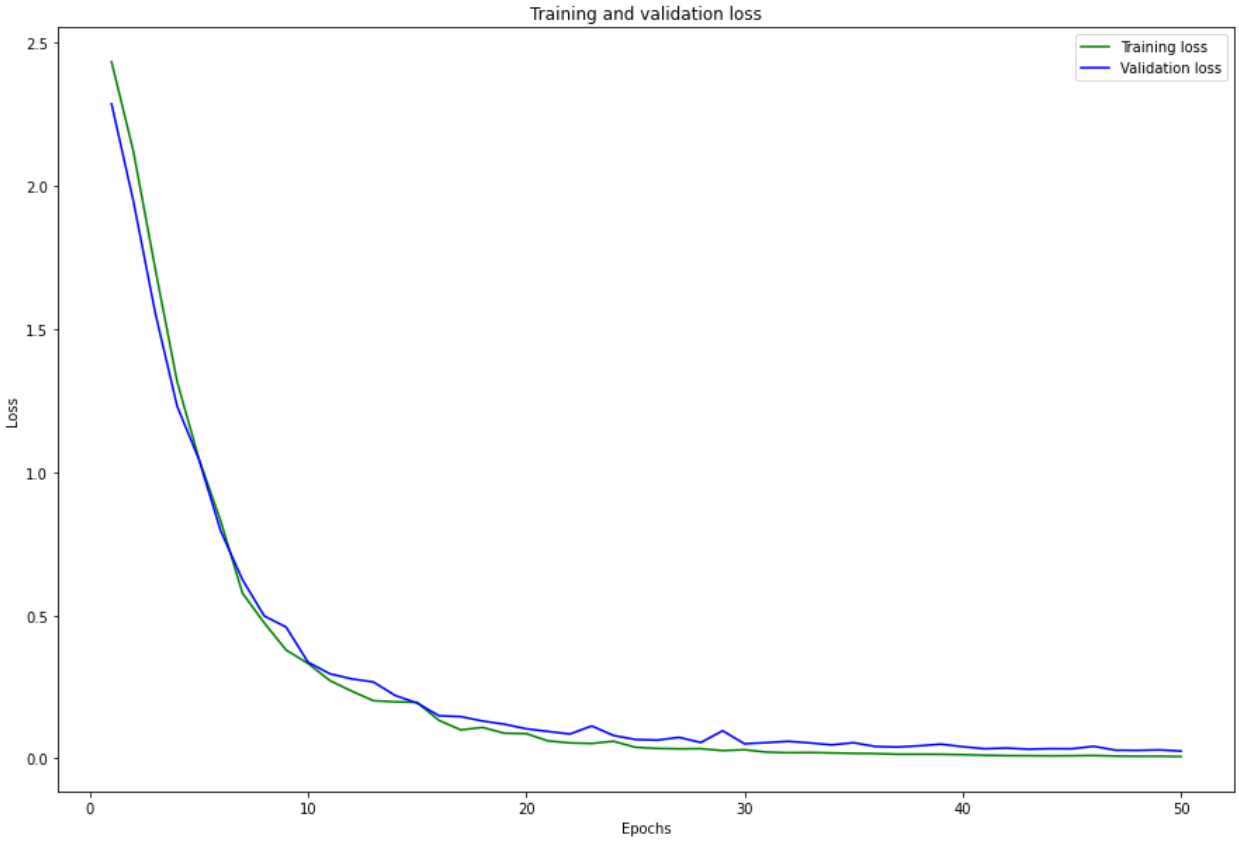
Model4:

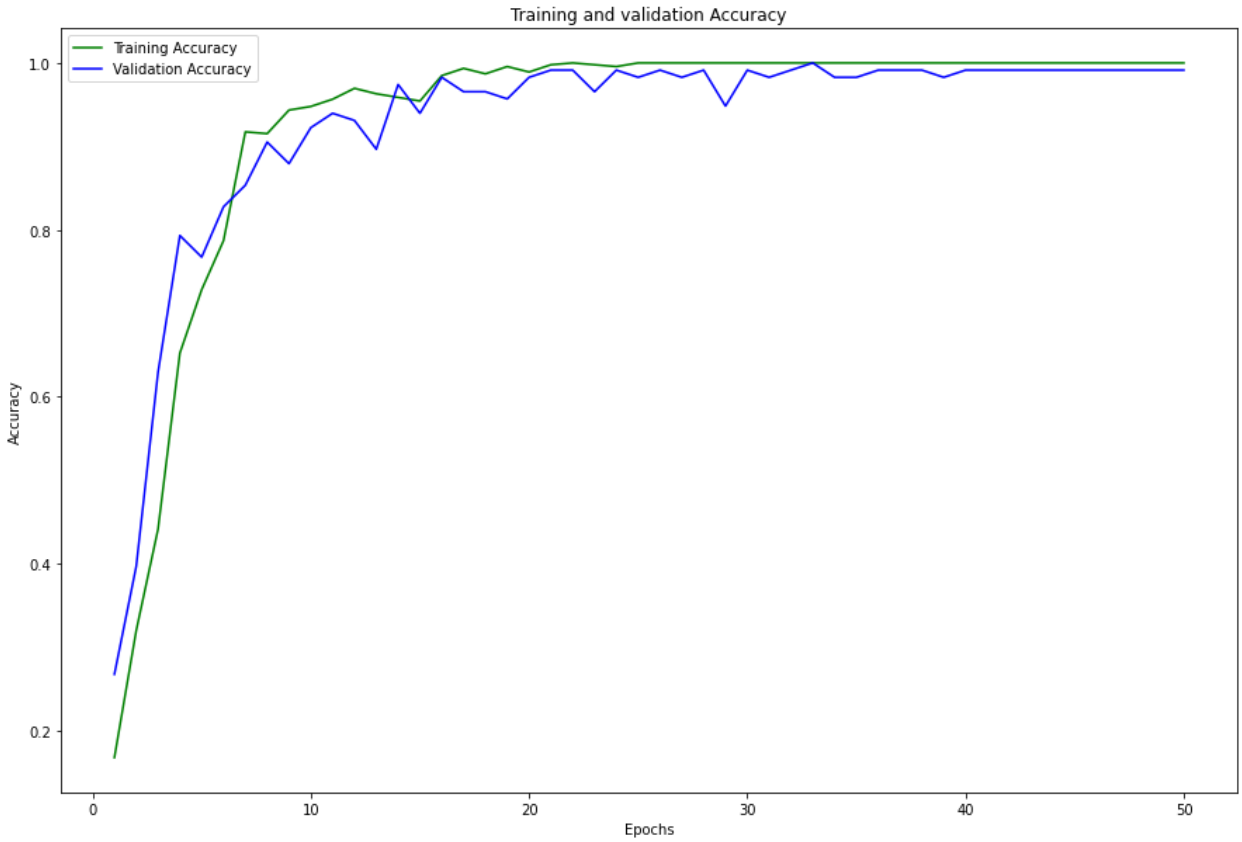
- This model is trained on raw data
- Segment length = 2 min → each training example is extracted from 238 samples → 1428 features in each example
- Model was trained for 50 epochs with batch size = 2 (after 50 epochs, model was over training)

- Below are test set metrics for different architectures that I tried

Architecture (nodes per layer)	Test accuracy	Precision	Recall	F1 score
[32, 32]	0.94	0.95	0.94	0.94
[64, 32]	0.96	0.97	0.96	0.96
[64, 64]	0.98	0.98	0.98	0.98
[64, 32, 32]	0.97	0.97	0.97	0.97
[64, 64, 32]	0.97	0.97	0.97	0.97

- From above experiments it seems that [64, 64] neural network worked best for this data
- Below are the loss, accuracy history graphs for [64, 64] neural network





- Below is the test set confusion matrix for the [64, 64] neural network

```
Confusion Matrix:
              asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data            10         0           0          0         0           0
des_data             0        10           0          0         0           0
dance_data           0         0          11          0         0           0
jump_data            0         0           0         15         0           0
sit_data             0         0           0          0        14           0
stand_data           0         0           0          0         0           9
walk_data            0         0           0          0         0           0
eat_data             0         0           0          0         0           0
pushf_data           0         0           0          0         0           0
pushg_data           0         0           0          0         0           0
laptop_data          0         0           0          0         0           0
run_data             0         0           0          0         0           0

              walk_data  eat_data  pushf_data  pushg_data  laptop_data  \
asc_data              0         0           0           0            0
des_data              0         0           0           0            0
dance_data            0         1           0           0            0
jump_data             0         0           0           0            0
sit_data              0         0           0           0            0
stand_data            0         0           0           0            0
walk_data            14         0           0           0            0
eat_data              0        12           0           0            0
pushf_data            0         0          11           0            1
pushg_data            0         0           0           9            1
laptop_data           0         0           0           0           15
run_data              0         0           0           0            0

              run_data
asc_data             0
des_data             0
dance_data           0
jump_data            0
sit_data             0
stand_data           0
walk_data            0
eat_data             0
pushf_data           0
pushg_data           0
laptop_data          0
run_data            11
```

- For this network architecture, size of the normal binary executable is 2,390,752 bytes and size of quantized binary executable is 614,284 bytes. I think model sizes are huge because of very high dimension of the input

Some Observations on model building:

- Models using raw data were of huge size when compared with models using extracted features. Also, the processing time was not significantly low when compared to extracted features
- Because of less training examples, models were volatile while training, we need more data to get good standard models
- Also, window size of 2 min seems to give good results from available data

4. I had used quantized model typesof model3 and model4 to load on arduino. I thought that I can speed up the prediction process by avoiding the data preprocessing and I see that the memory usage is also within limits of available memory on arduino. Below are some observations

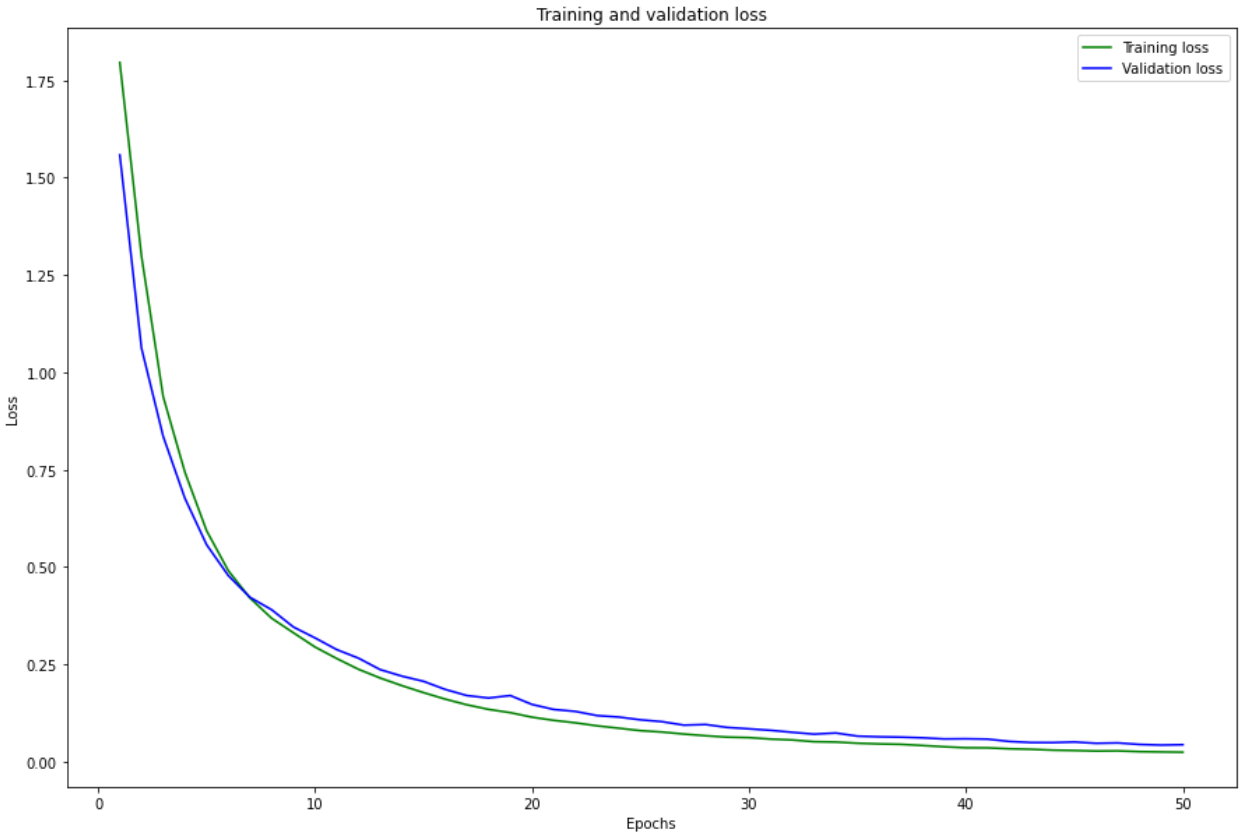
Quantized model trained using 2 min raw data:
Flash Memory required: 349,368 bytes
Average Response Time: 1976 milliseconds

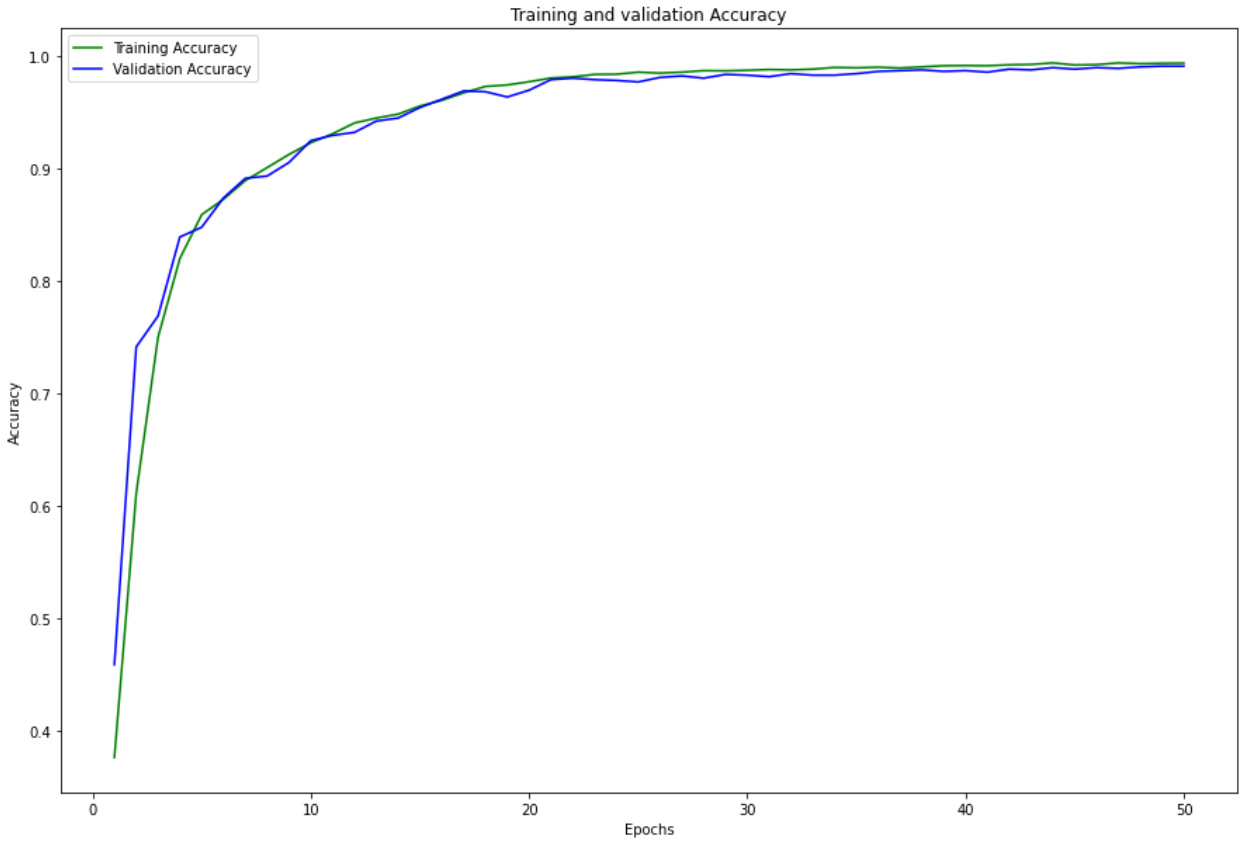
Quantized model trained using 3 min raw data:
Flash Memory required: 394,208 bytes
Average Response Time: 2953 milliseconds

5. I repeated this experiment with master dataset with 7 activities and below are some observations

Model1: Model trained using feature extractor with 2 min data segments

- For this experiment, [32, 32] model architecture gave 99% accuracy for test dataset
- The loss history and accuracy history graphs are as below:





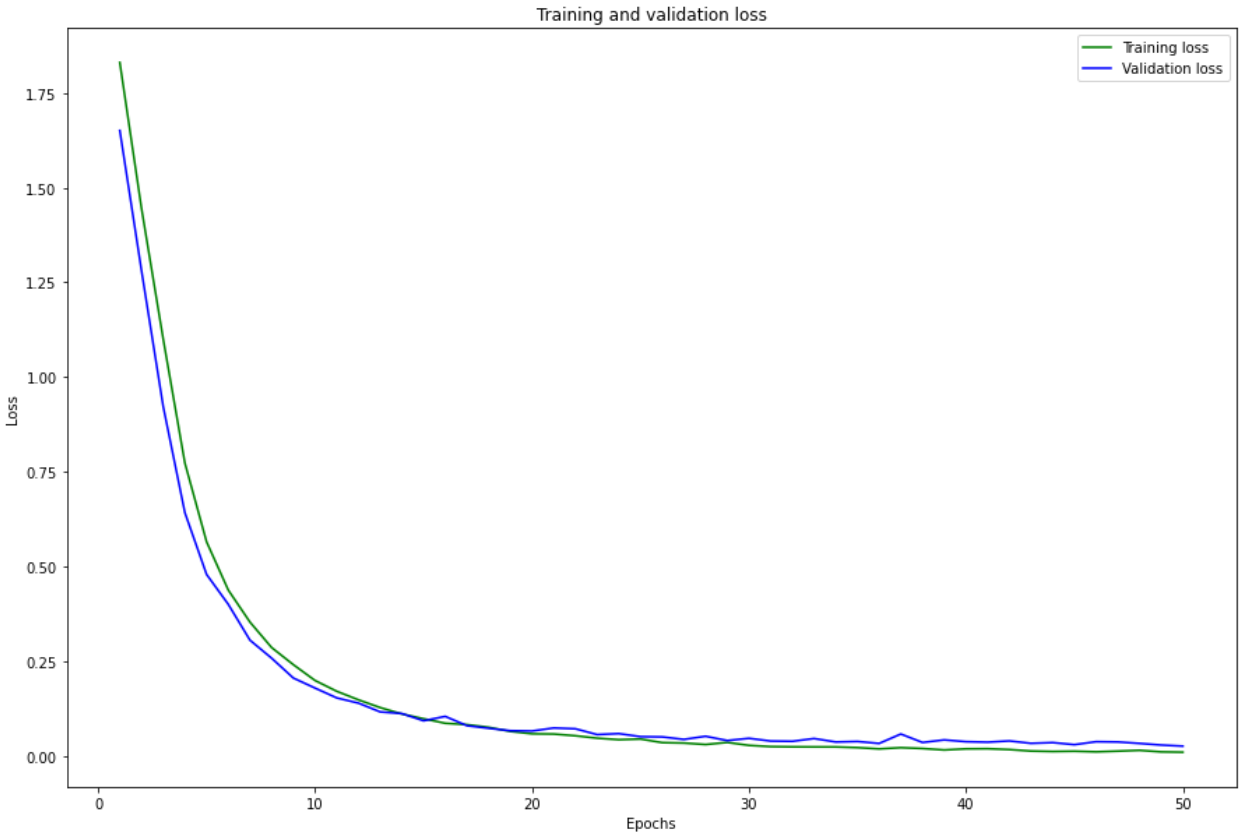
- Below are some metrics about this model:

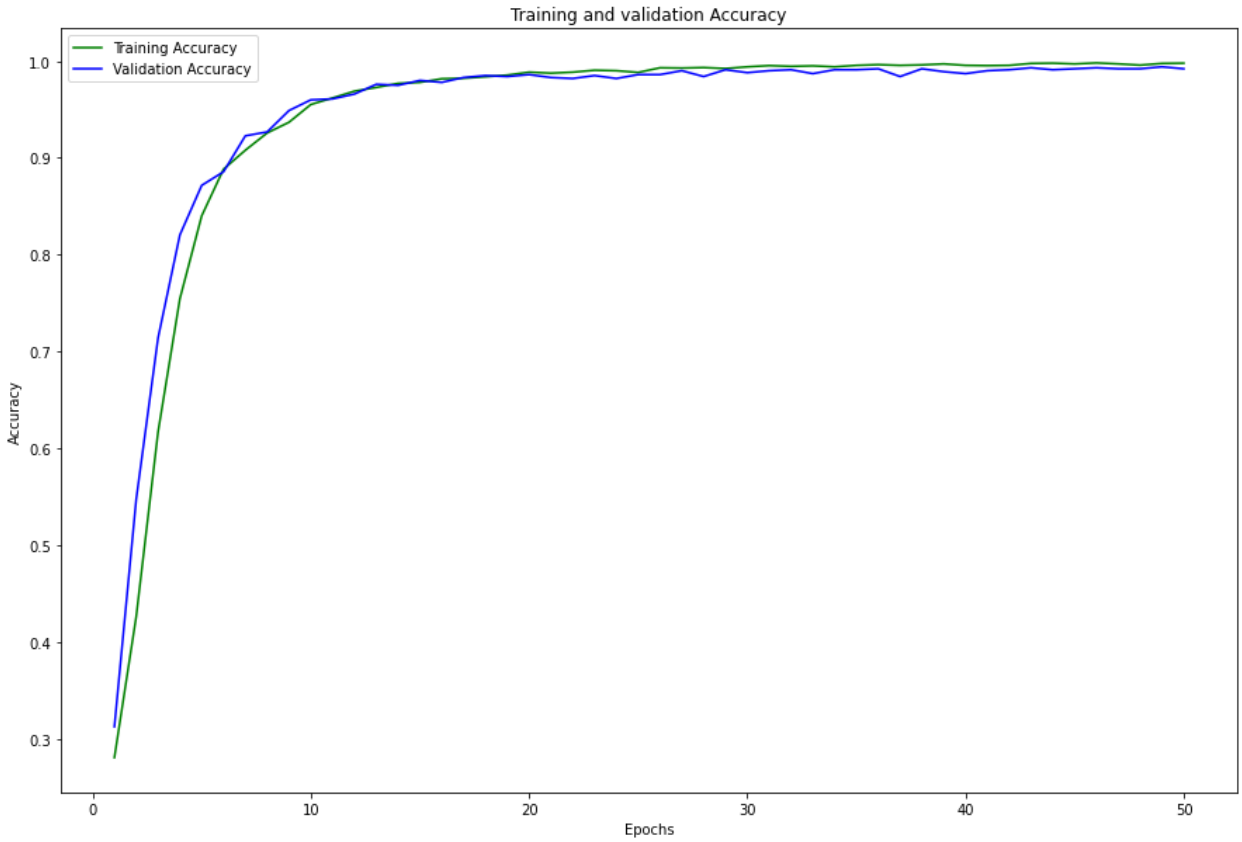
```
accuracy: 0.9919657204070702
Precision:  0.9920246337806007
Recall:  0.9919657204070702
f1_score:  0.9919653069807922
Confusion Matrix:
             asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data          280         1           0          0         0           0
des_data            0       275           0          2         0           0
dance_data          0         0         271          0         0           0
jump_data           0         3           0        251         0           0
sit_data            1         0           0          0       235           0
stand_data          0         0           5          0         0         261
walk_data           0         0           0          0         1           0

            walk_data
asc_data            0
des_data            0
dance_data          0
jump_data           0
sit_data            2
stand_data          0
walk_data         279
```

- When converted to binary .h format, the size of the normal model is 88,070 bytes and size of the quantized model is 37,330 bytes

Model2: Model trained using feature extractor with 3 min data segments

- For this experiment, [32, 32, 32] model architecture gave 99% accuracy for test dataset
- The loss history and accuracy history graphs are as below:





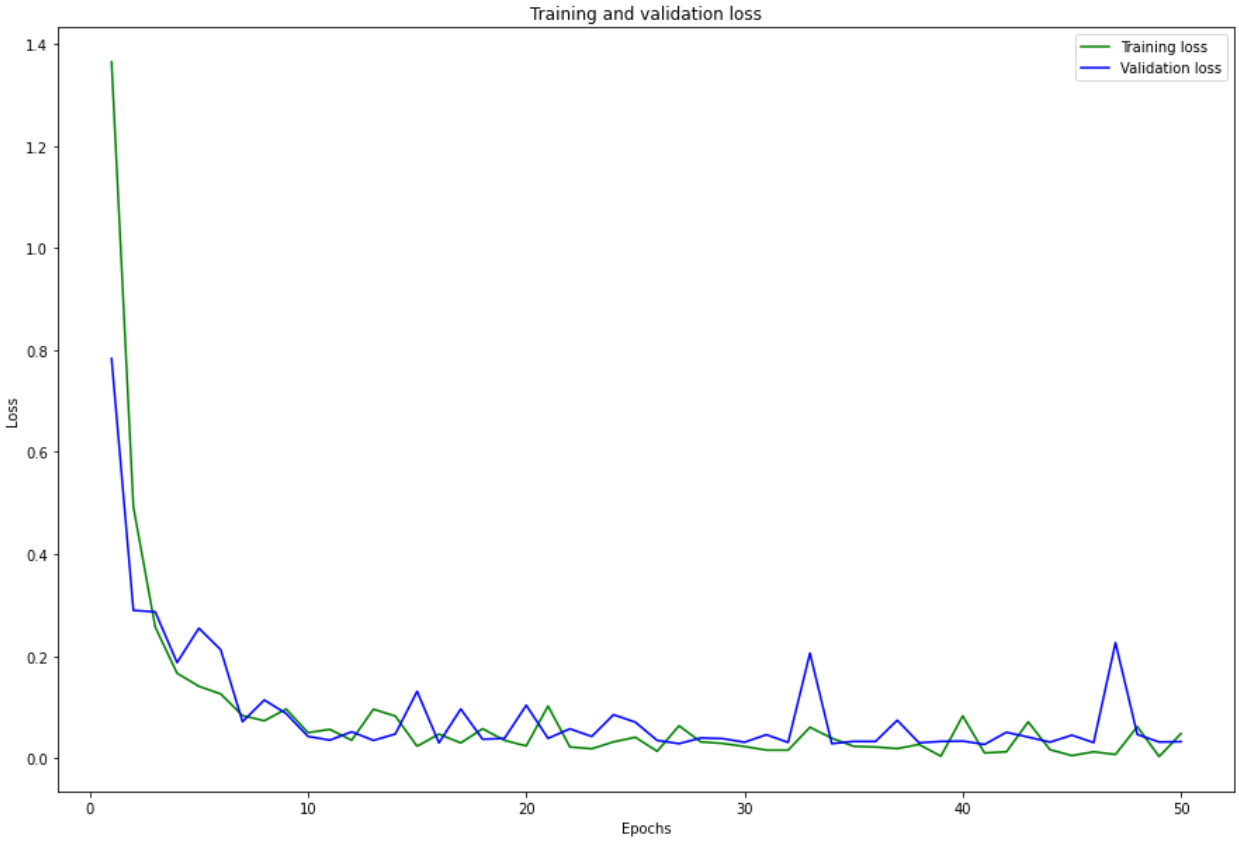
- Below are some metrics about this model:

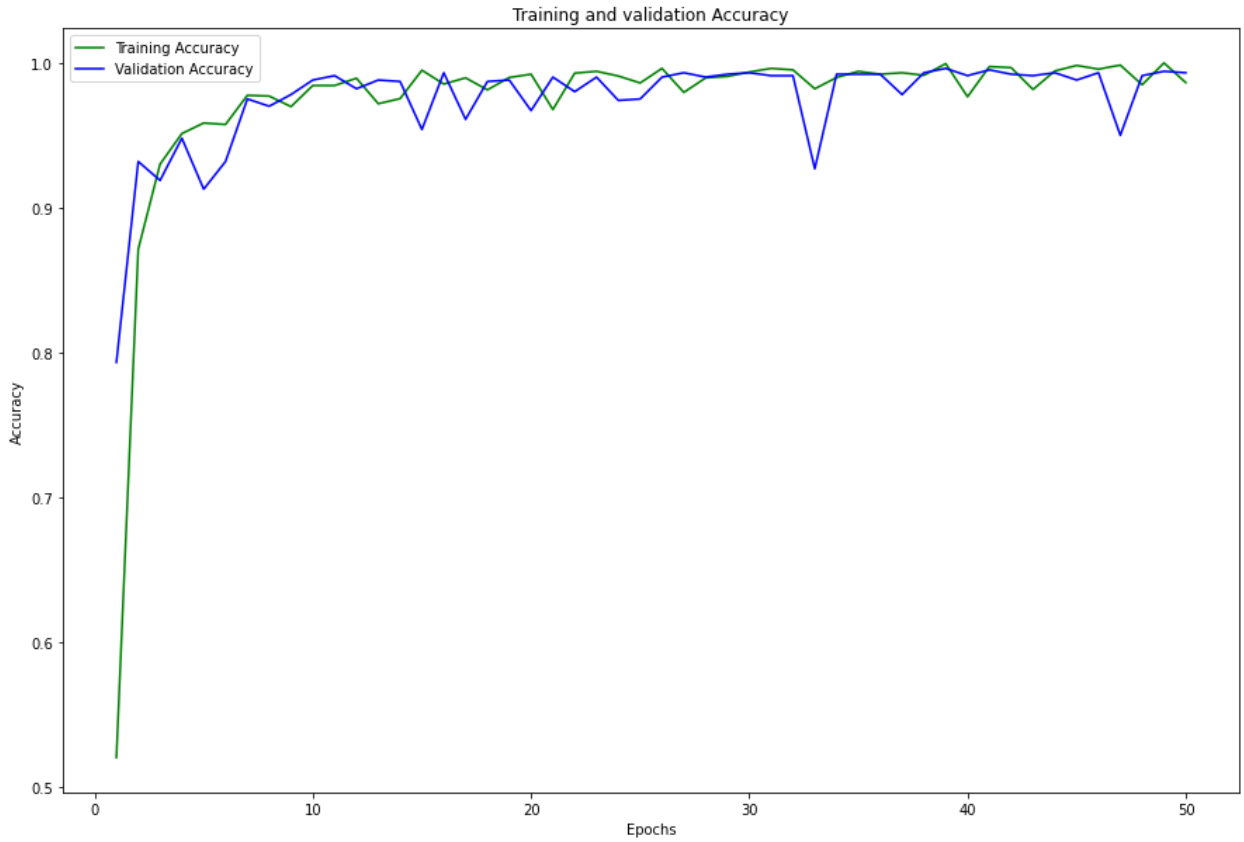
```
accuracy: 0.9959839357429718
Precision:  0.996003846771745
Recall:  0.9959839357429718
f1_score:  0.995982650105555
Confusion Matrix:
             asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data          181         0           0          0         0           0
des_data            0       187           0          1         0           0
dance_data          0         0         184          0         0           0
jump_data           0         3           0        163         0           0
sit_data            0         0           0          0       176           0
stand_data          0         0           0          0         0         165
walk_data           0         0           0          0         0           0

             walk_data
asc_data             0
des_data             0
dance_data           0
jump_data            0
sit_data             1
stand_data           0
walk_data          184
```

- When converted to binary .h format, the size of the normal model is 116,314 bytes and size of the quantized model is 47,246 bytes

Model3: Model trained using raw with 3 min data segments

- For this experiment, [64, 64, 32] model architecture gave 99% accuracy for test dataset
- The loss history and accuracy history graphs are as below:





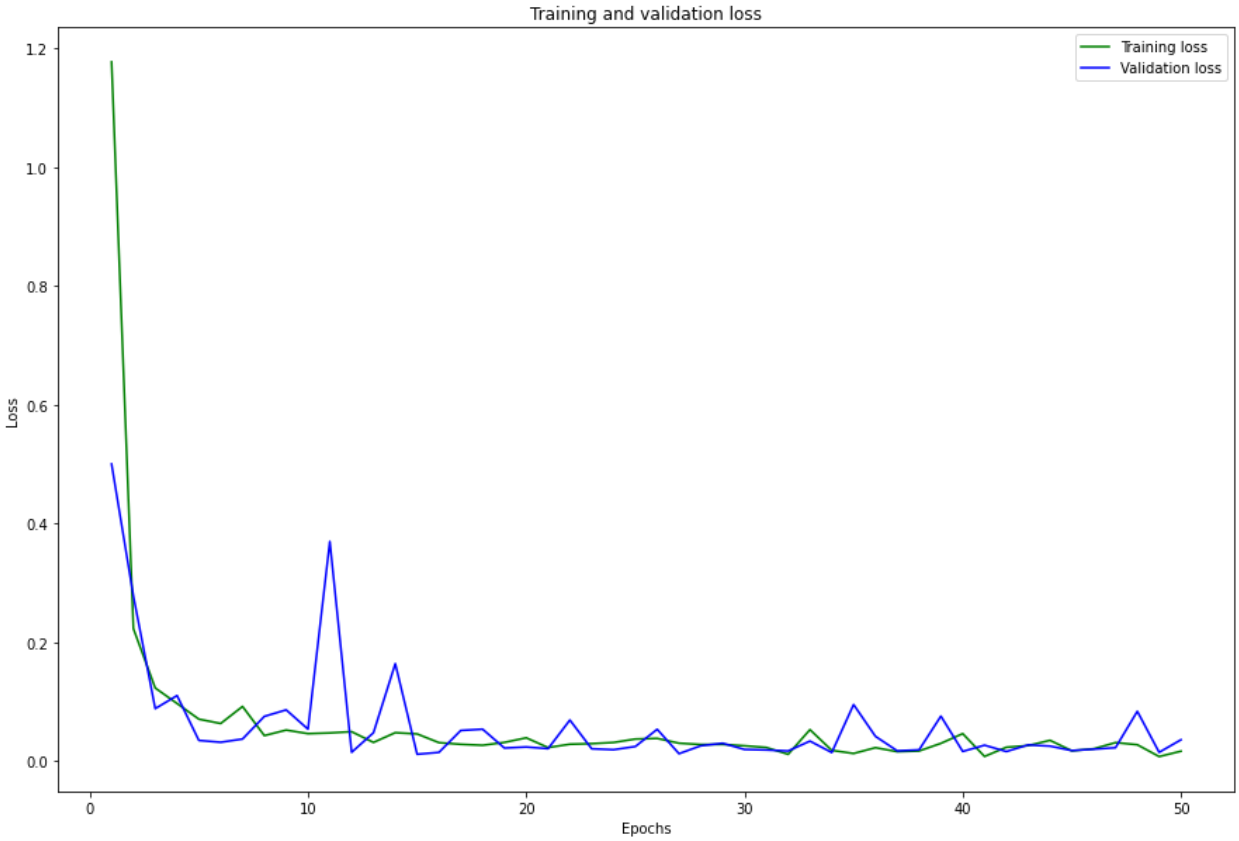
- Below are some metrics about this model:

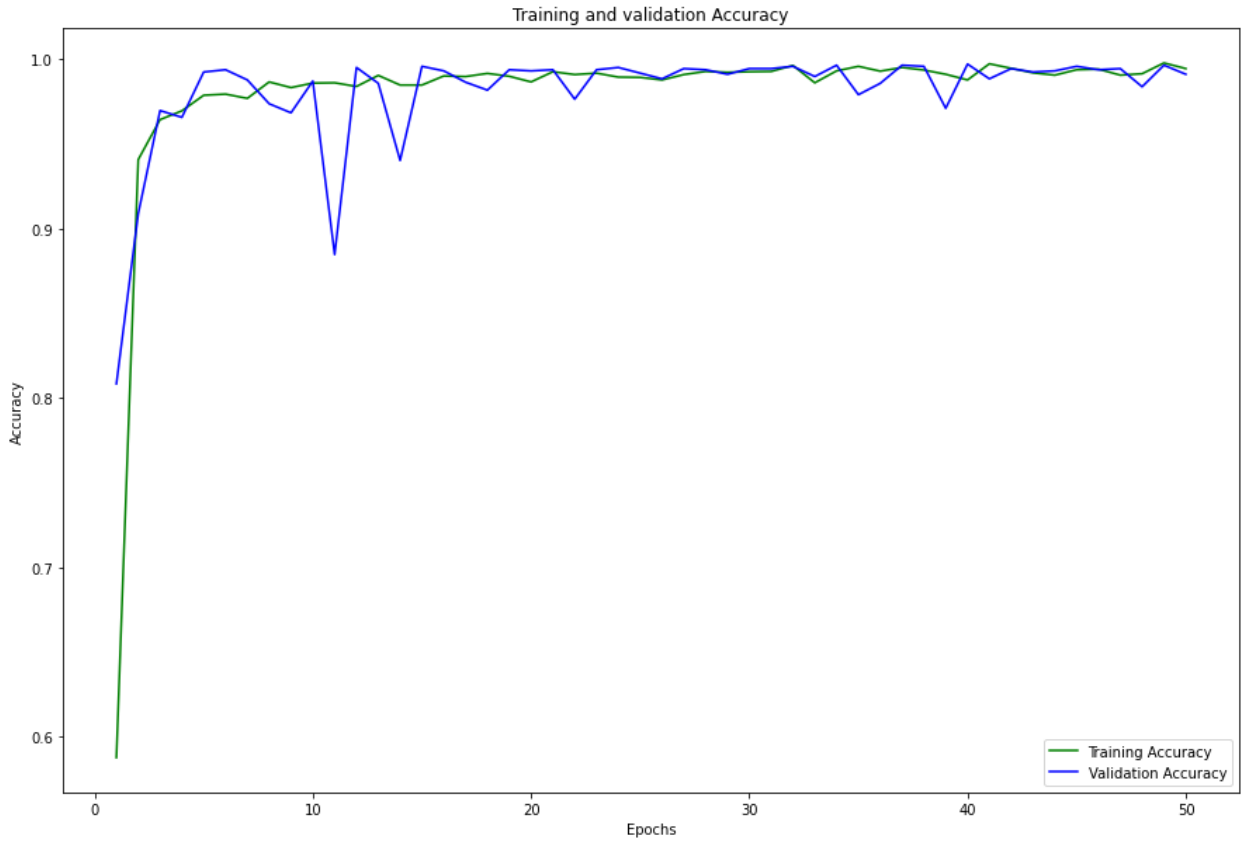
```
accuracy: 0.9959839357429718
Precision:  0.9960140504356076
Recall:  0.9959839357429718
f1_score:  0.9959827960094458
Confusion Matrix:
             asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data          181         0           0          0         0           0
des_data            0       187           0          0         0           0
dance_data          0         0         184          0         0           0
jump_data           0         2           0        164         0           0
sit_data            0         0           0          0       177           0
stand_data          0         0           1          0         0         163
walk_data           0         0           0          0         0           0

            walk_data
asc_data            0
des_data            1
dance_data          0
jump_data           0
sit_data            0
stand_data          1
walk_data         184
```

- When converted to binary .h format, the size of the normal model is 3,485,312 bytes and size of the quantized model is 890,156 bytes

Model4: Model trained using raw with 2 min data segments

- For this experiment, [64, 64] model architecture gave 99% accuracy for test dataset
- The loss history and accuracy history graphs are as below:





- Below are some metrics about this model:

```
accuracy: 0.9898232458489555
Precision:  0.9900092784655659
Recall:  0.9898232458489555
f1_score:  0.9898449995836187
Confusion Matrix:
             asc_data  des_data  dance_data  jump_data  sit_data  stand_data  \
asc_data          277         3           1          0         0           0
des_data            0       273           0          0         0           0
dance_data          0         0         269          0         0           1
jump_data           0         5           0        248         0           0
sit_data            0         0           0          0       235           1
stand_data          0         0           0          0         0         266
walk_data           0         0           0          0         0           0

             walk_data
asc_data             0
des_data             4
dance_data           1
jump_data            1
sit_data             2
stand_data           0
walk_data          280
```

- When converted to binary .h format, the size of the normal model is 2,358,114 bytes and size of the quantized model is 608,364 bytes

I uploaded the quantized models trained using raw data on arduino and below are some observations from that experiments:

Quantized model trained using 2 min raw data:
Flash Memory required: 348,408 bytes
Average Response Time: 1976 milliseconds

Quantized model trained using 3 min raw data:
Flash Memory required: 394,016 bytes
Average Response Time: 2953 milliseconds

Overall, I observed that models trained using combined data are more robust and better in performance when compared with models trained using personal data. Models trained using raw data are performing better than models trained using feature extractions