Term Project

Ramona Devi

May 1, 2021

Abstract

Goal of this project is to perform Classification on Taiji Dataset.

Contents

1 Introduction

Classify all the 43 key-poses based on MoCAP + Footpressure dataset. Below are the two main things I performed with dataset:

1. Adding in 1st and/or 2nd order kinematic pose information into classification while testing a range of temporal windows for velocity and acceleration calculation. Velocity and Acceleration of joints at different temporal windows may have impact or not on classification accuracy of any ML technique. Basically can position, velocity, and acceleration and different low pass filtering windows impact classification accuracy.

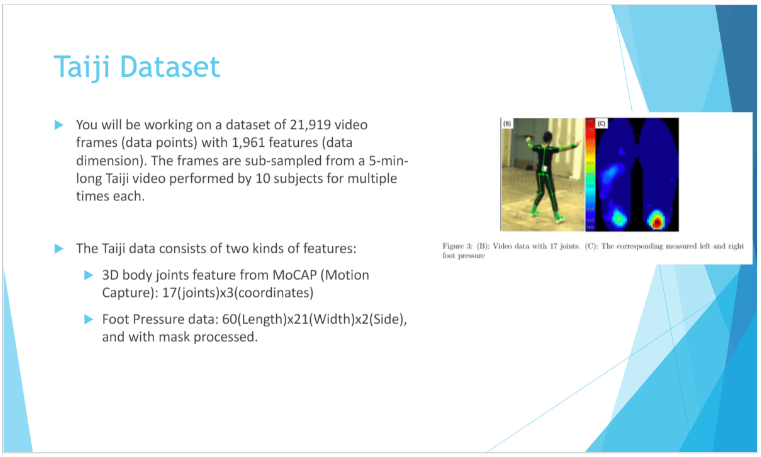
2.Evaluate the impact of foot pressure resolution (spatial scale) on classification accuracy. Holding all other aspects constant, and see 'Does reducing the resolution of the foot pressure impact the classification accuracy?' Systematically reduce resolution from current 60x21x2 (2000+) pixels to less than 20 pixels and see classification accuracy.

1.1 Dataset

Dataset: 130019 data points with 2064 features including 17*3*2(for velocity and acceleration), 17*3 for position and rest for foot pressure pixels.

Velocity and acceleration data has been generated from videos and foot pressure data using sensors on shoes.

Labels: 43 key forms



Taiji Dataset

- You will be working on a dataset of 21,919 video frames (data points) with 1,961 features (data dimension). The frames are sub-sampled from a 5-min-long Taiji video performed by 10 subjects for multiple times each.
- The Taiji data consists of two kinds of features:
  - 3D body joints feature from MoCAP (Motion Capture): 17(joints)x3(coordinates)
  - Foot Pressure data: 60(Length)x21(Width)x2(Side), and with mask processed.

Figure 3: (B): Video data with 17 joints. (C): The corresponding measured left and right foot pressure

Figure 1: Details for Taiji Dataset

Below I show the data points distribution in all 43 classes.

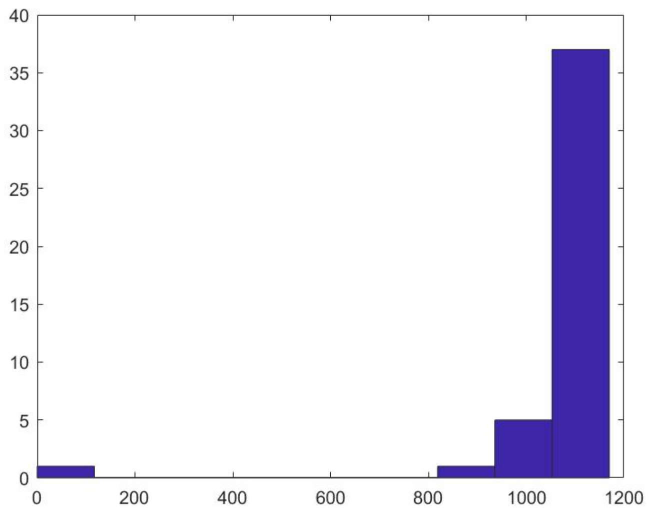


Figure 2: Data distribution

2 Implementation/Flow chart

Step1: Reshaping of data

I had to reshape my data into 4D array to feed it to 2D CNN model.

Step 2: Splitting the Data

This step includes splitting of data into Train/Test/Val data and classifying the frames to a particular case(based on m and n values)

Step 3: Feature Engineering

In this step I used Variance ratio(Filter method) to select the top 50 percent feature

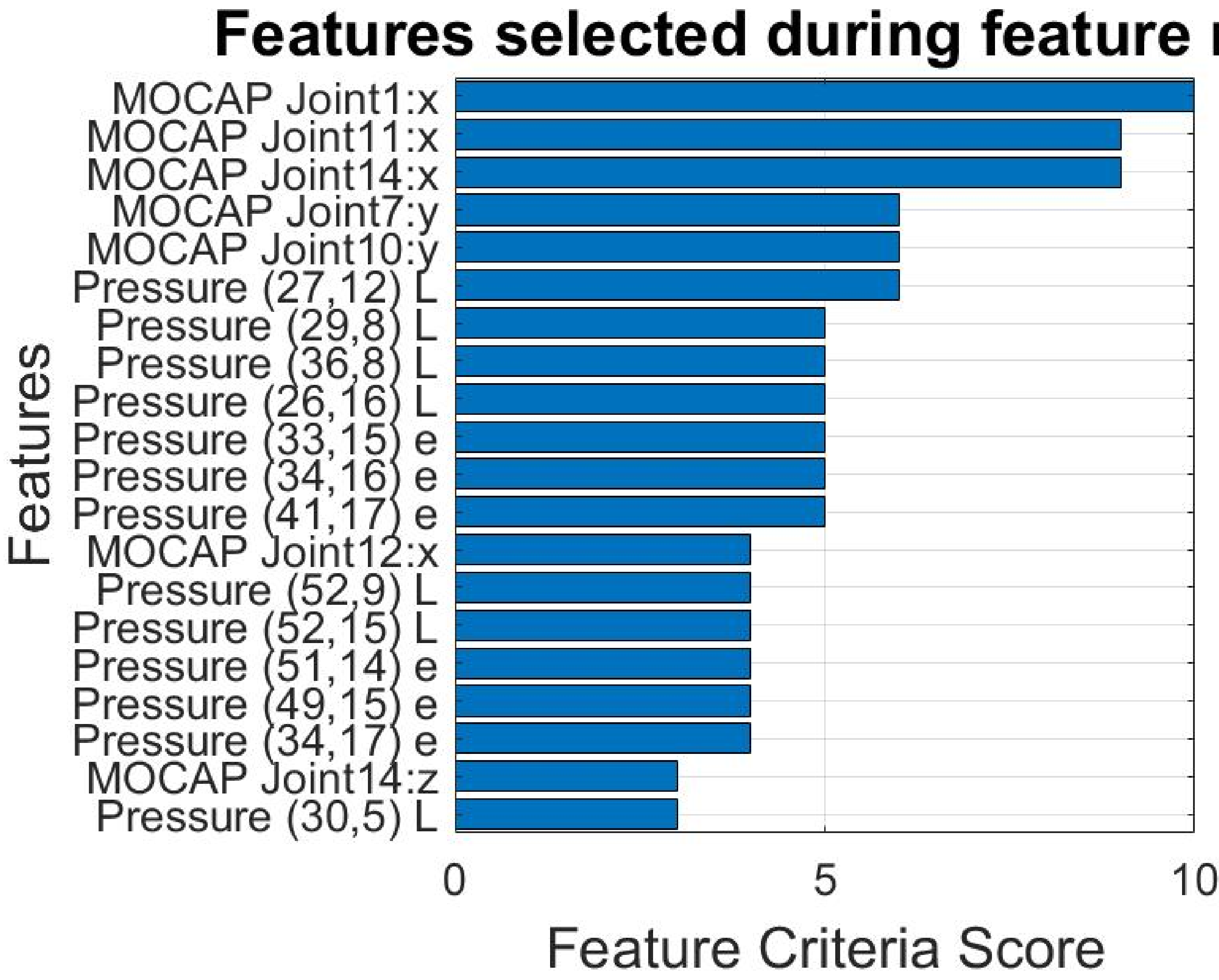


Figure 3: Feature selected by VR filter method

Step 4: Running my model on raw data

I tried different combination of filter size and number of filters in my network architecture.

Step5: Adding extra features

I added additional velocity and acceleration features in data and ran my model on it.

Step6: Training on reduced feature dataset

I removed foot pressure features less than 20 pixel and ran my model on this.

Step7: Visualisation of data

Used TSNE visualisation for 1st layer of my CNN network on all three modified dataset.

3 Results

In this section Following plots are shown for each Dataset:

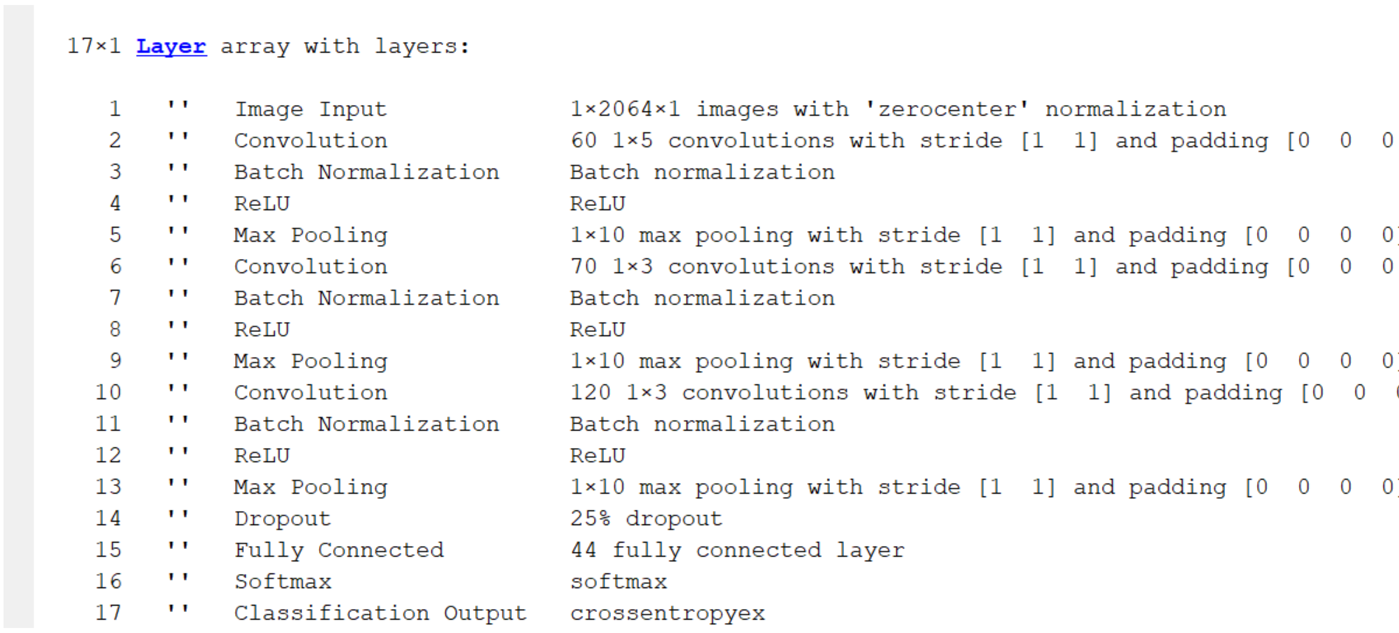
1. Training Accuracy vs Loss
2. TSNE visualisation for Test, Train and Validation data.
6. Used 20 epochs for each network with 3e-4 learning rate

Number of epochs: 20

Learning rate: 3e-4

Batch size: 160

Note: I forget to save AnalyseNetwork at the time of training and it takes a lot of time to train again so I am just attaching a screenshot of my network.



```
  17×1 Layer array with layers:

     1   ''   Image Input             1×2064×1 images with 'zerocenter' normalization
     2   ''   Convolution             60 1×5 convolutions with stride [1  1] and padding [0  0  0
     3   ''   Batch Normalization     Batch normalization
     4   ''   ReLU                    ReLU
     5   ''   Max Pooling             1×10 max pooling with stride [1  1] and padding [0  0  0  0
     6   ''   Convolution             70 1×3 convolutions with stride [1  1] and padding [0  0  0
     7   ''   Batch Normalization     Batch normalization
     8   ''   ReLU                    ReLU
     9   ''   Max Pooling             1×10 max pooling with stride [1  1] and padding [0  0  0  0
    10   ''   Convolution             120 1×3 convolutions with stride [1  1] and padding [0  0
    11   ''   Batch Normalization     Batch normalization
    12   ''   ReLU                    ReLU
    13   ''   Max Pooling             1×10 max pooling with stride [1  1] and padding [0  0  0  0
    14   ''   Dropout                 25% dropout
    15   ''   Fully Connected         44 fully connected layer
    16   ''   Softmax                 softmax
    17   ''   Classification Output   crossentropyex
```

Figure 4: Feature selected by VR filter method

3.1 Results for Original Dataset

1. Training Accuracy : around 70 percent
2. Testing Accuracy : around 65 percent
3. Validation Accuracy : around 64 percent

Time taken for Training Network: 10 mins on gpu

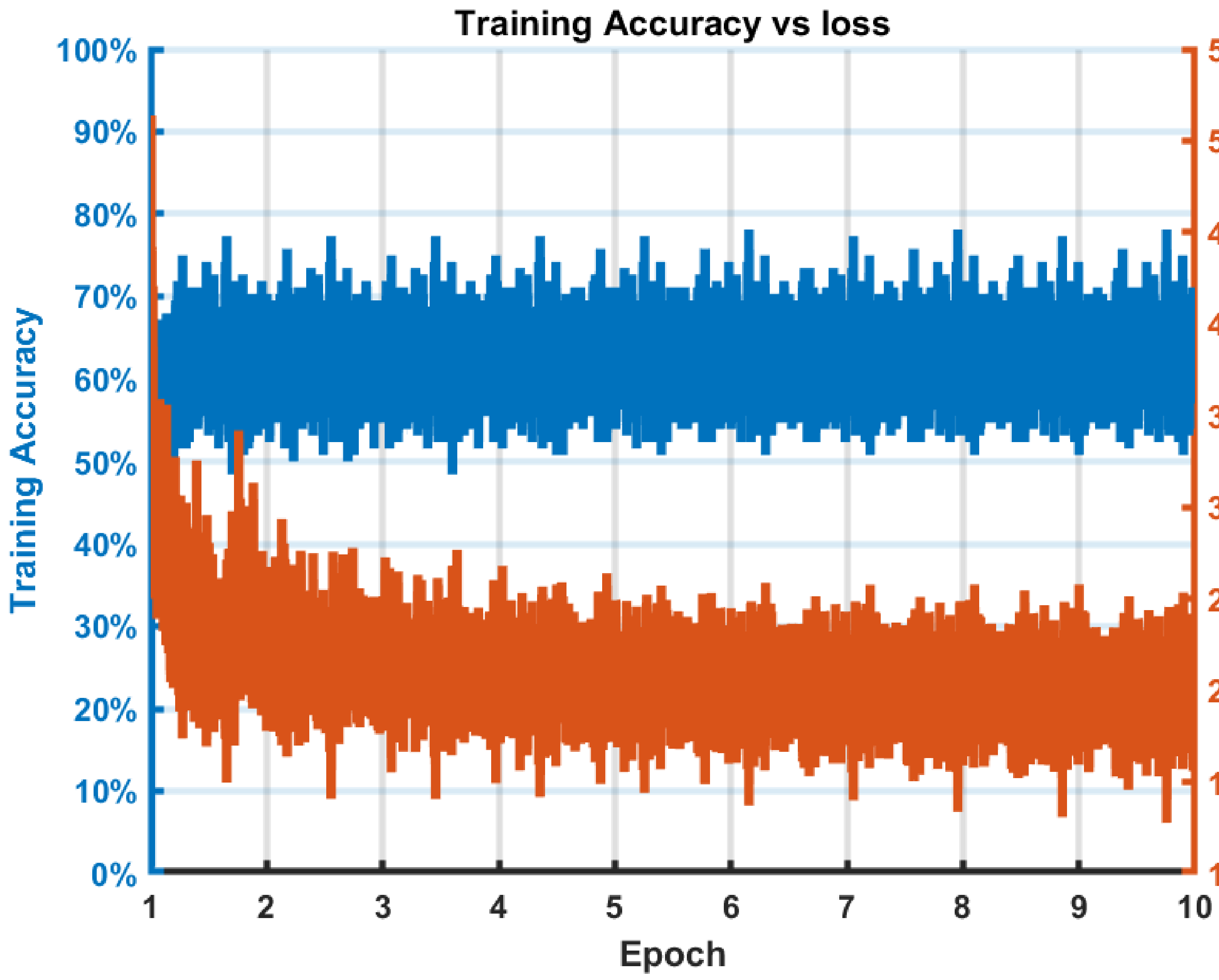


Figure 5: Accuracy vs Loss

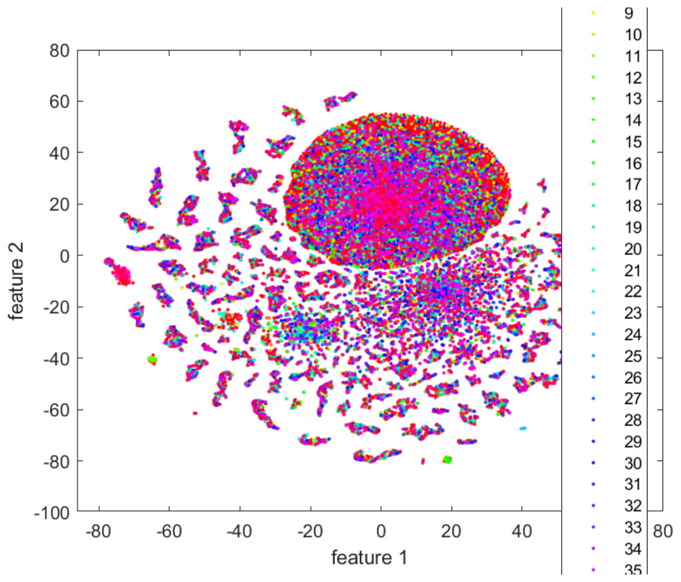


Figure 6: Training TSNE visualisation

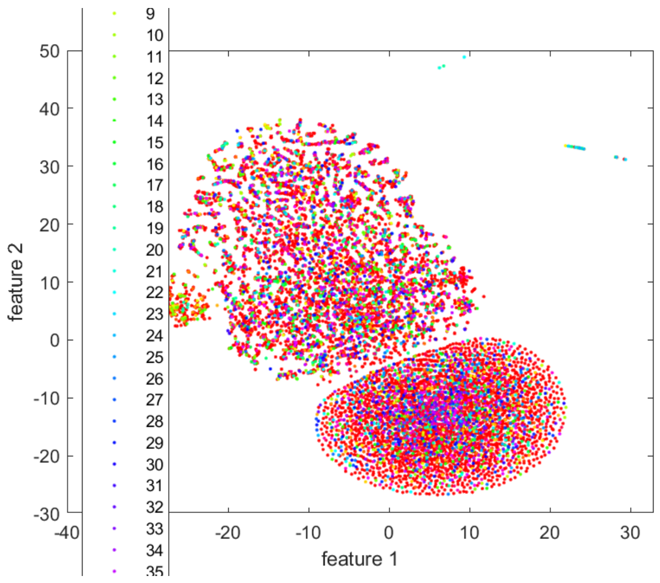


Figure 7: Testing TSNE visualisation

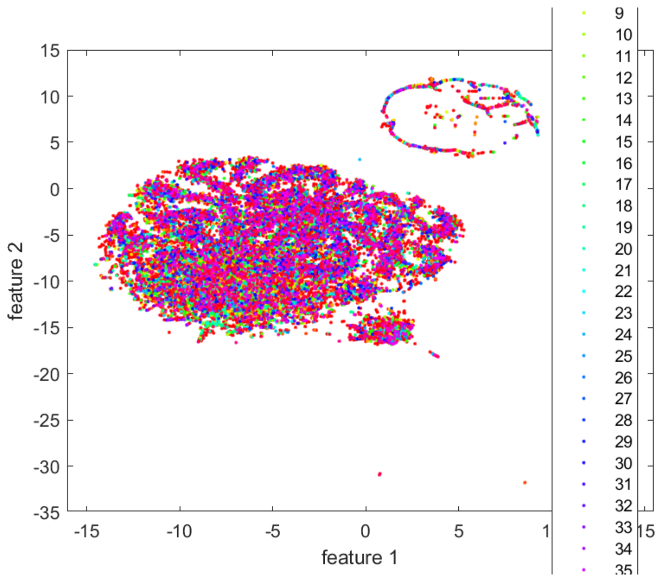


Figure 8: Validation TSNE visualisation

3.2 Results for Updated (velocity and Acceleration) Dataset

1. Training Accuracy : around 70 percent
2. Testing Accuracy : around 65 percent
3. Validation Accuracy : around 64 percent

Time taken for Training Network: 10 mins on gpu

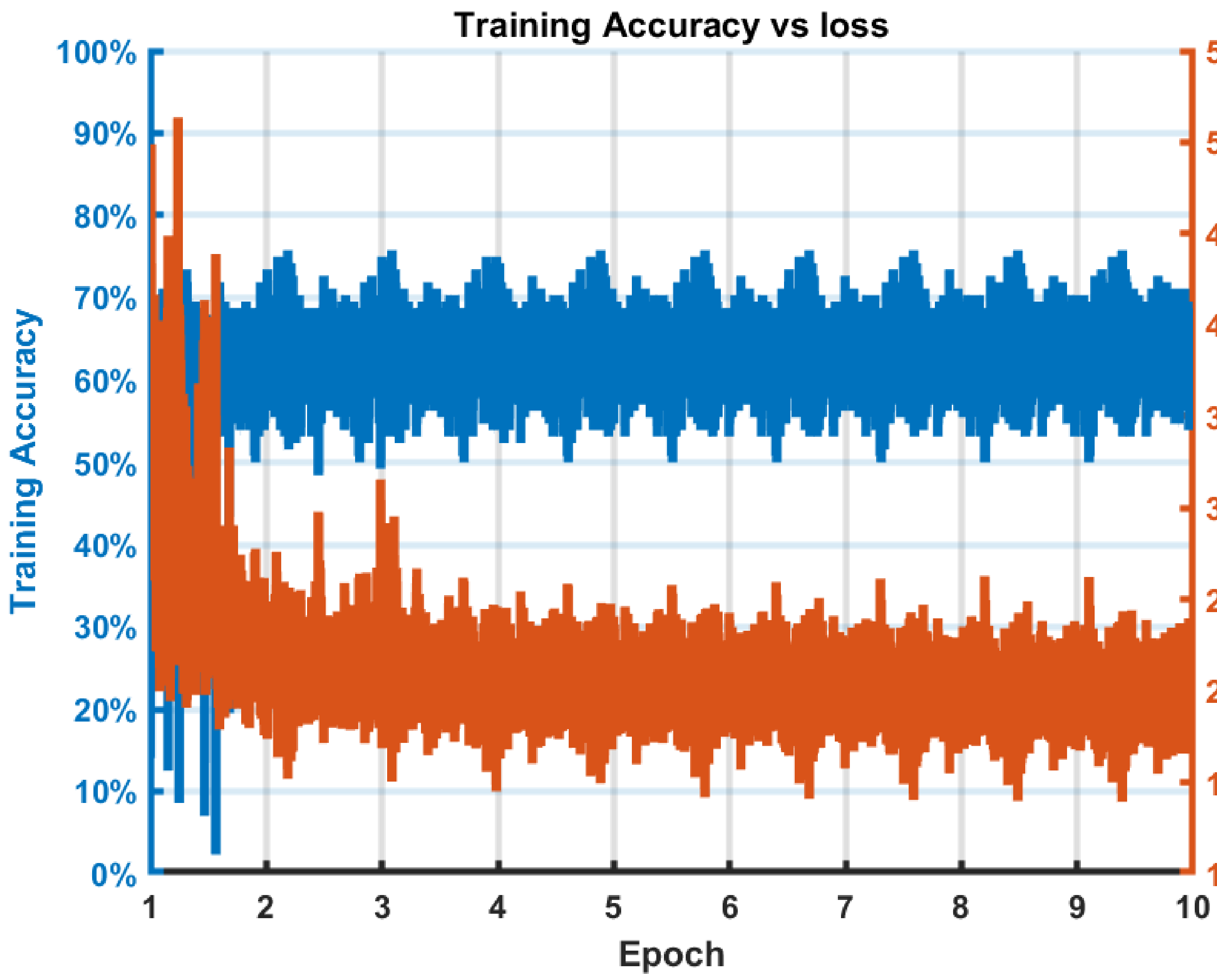


Figure 9: Accuracy vs Loss

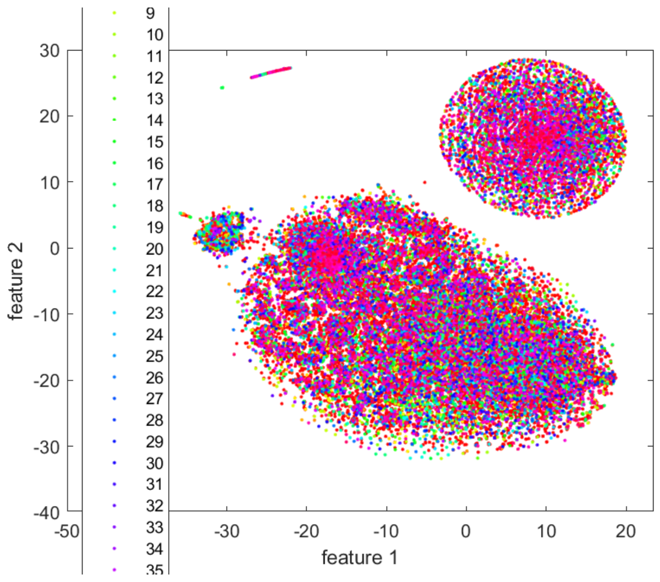


Figure 10: Training TSNE visualisation

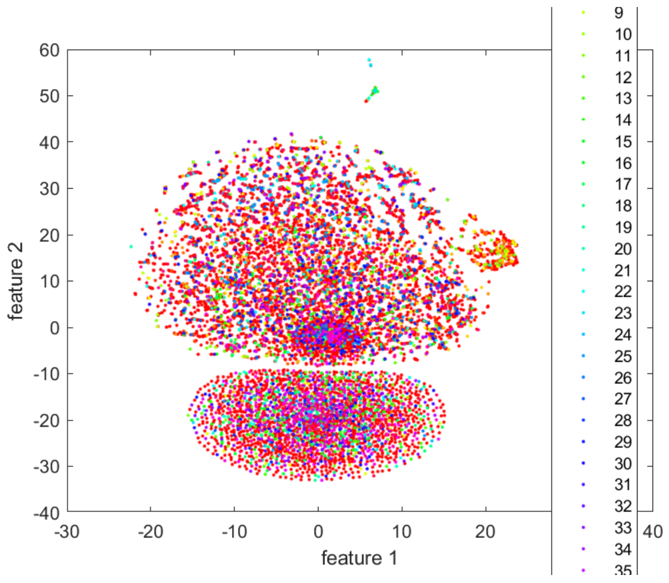


Figure 11: Testing TSNE visualisation

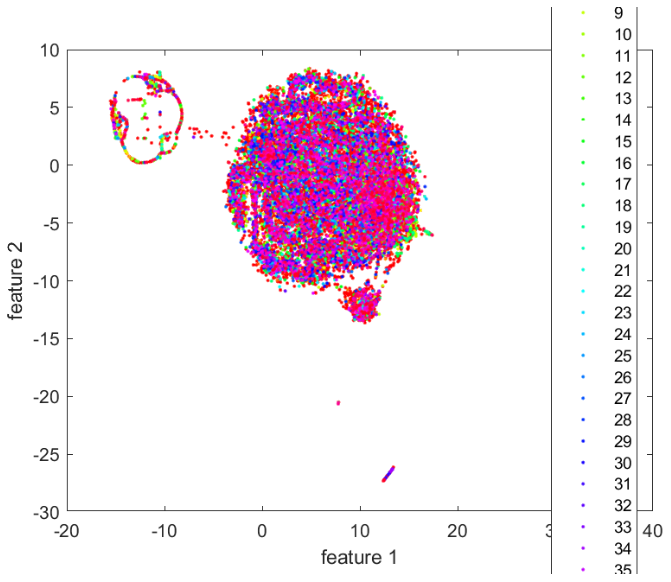


Figure 12: Validation TSNE visualisation

3.3 Results for Reduced Dataset

1. Training Accuracy : around 70 percent

2. Testing Accuracy : around 65 percent

3. Validation Accuracy : around 64 percent

Time taken for Training Network: 5 mins on gpu

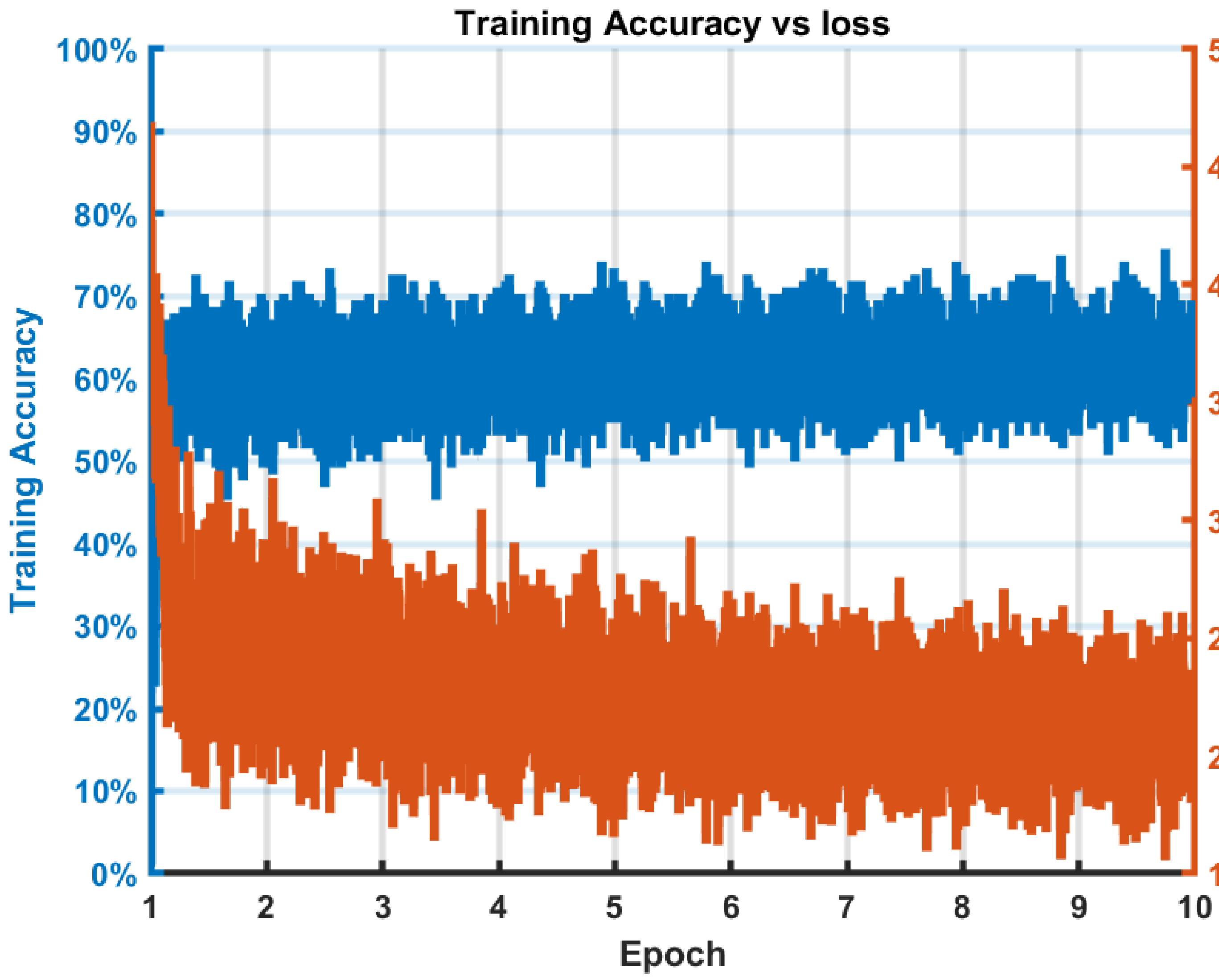


Figure 13: Accuracy vs Loss

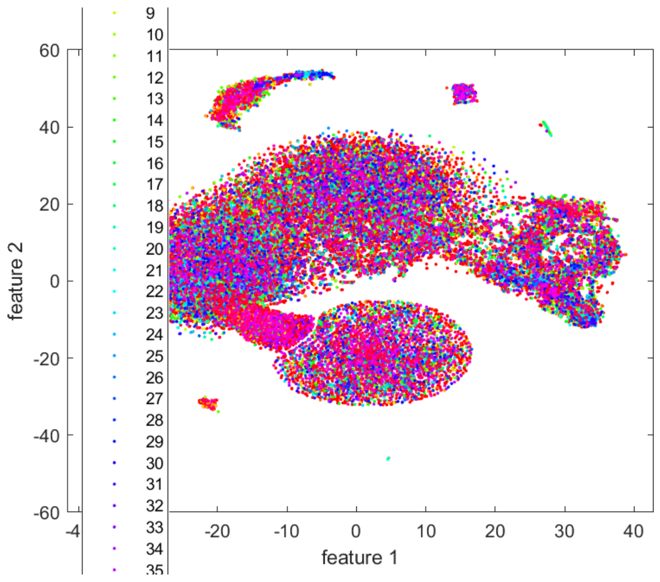


Figure 14: Training TSNE visualisation

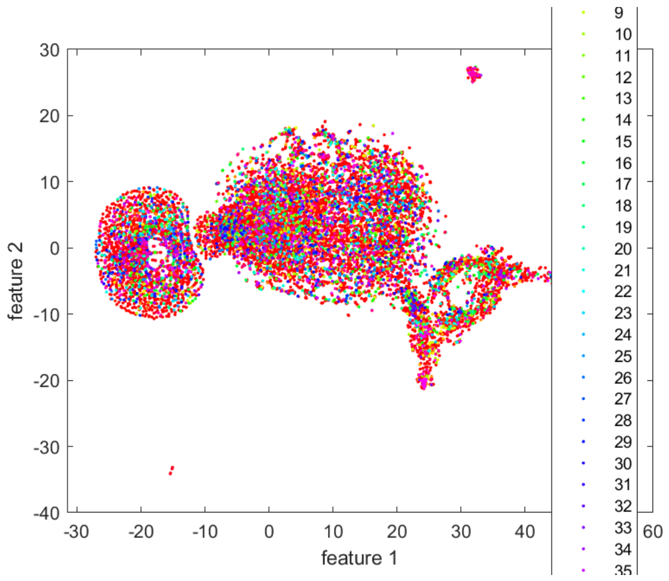


Figure 15: Training TSNE visualisation

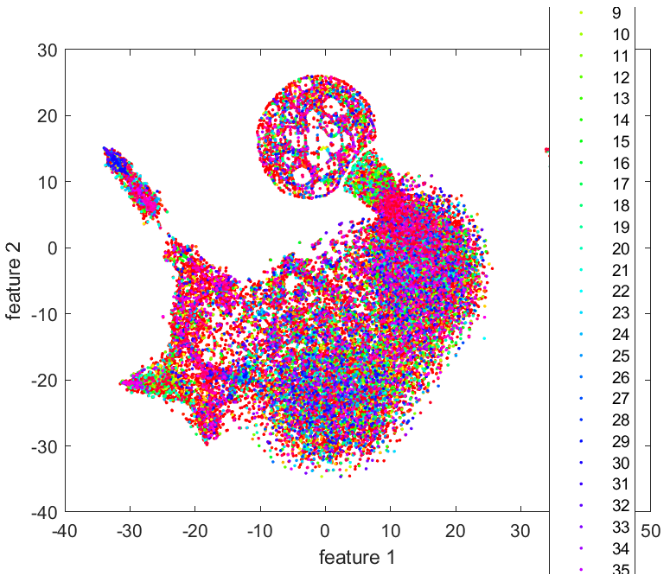


Figure 16: Training TSNE visualisation

4 Follow-up Questions

1.Loss Function: I used Conv inbuilt function in matlab, which uses Cross entropy Loss as a loss function.

2. Confusion and Classification matrix: Values were too big to be displayed in heatmap, I just got the empty matrices shown below. And as i was using someone else's GPU I couldnt ask ask again to use his system.

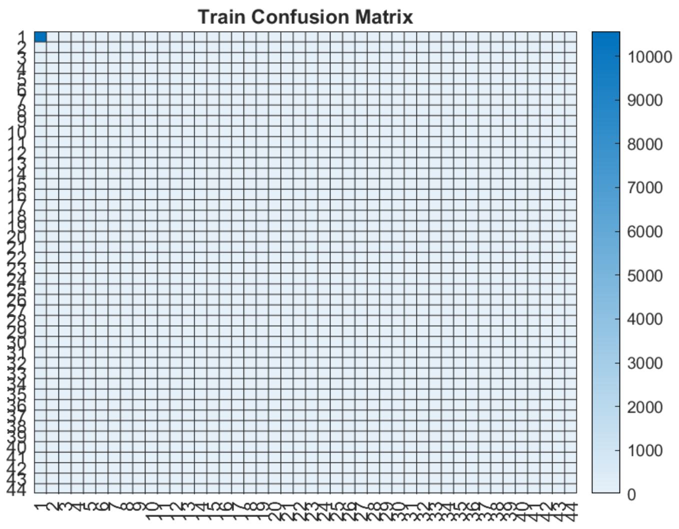


Figure 17: Training Confusion Matrix

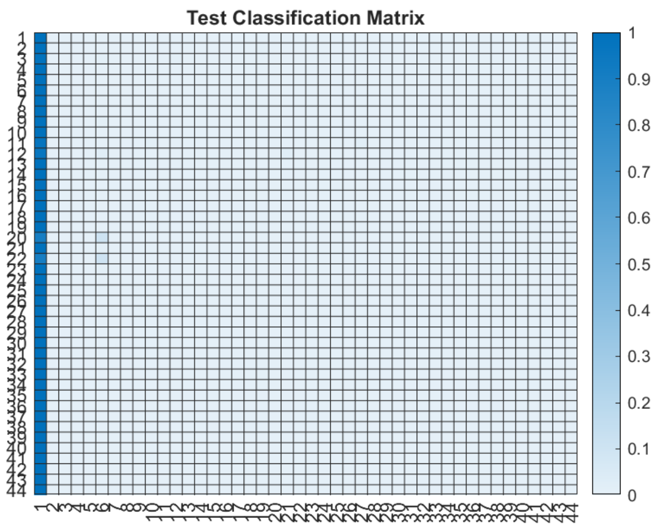


Figure 18: Test classification matrix

3. Data size Issue: This is the fixed data I got, and I think it was not enough for the project. No of parameters for my network was more than 20,000 and according to the rule of 10, I wanted at-least 10*20,000, I got 130019.

I can justify it because when i tried to add another layer in my network it gave me input size issue, because my data was down scaled so much already it could not perform convolution on it.

4. Data augmentation was not required, my data includes raw position, velocity and acceleration data, so I attempted feature engineering instated. Results of the selected features from VR shown above.

5 Conclusion

I did classification on 3 different modified Taiji dataset, but couldn't see any substantial difference in accuracies. I got similar accuracies for all 3 datasets, Training: 68 percent, Testing 65, Validation 64.

I was getting upto 73 percent accuracy before splitting the data in train/val but afterwards I got hardly 70 percent. Maybe its because the training data reduced further.

Learning: Learnt to built my own deep learning network.

Problems: I faced a huge gpu issue and had to ask for help from a friend. My confusion as well as classification matrix were too large that it displayed blank matrix(heatmap function) with 43 classes. So I did not show it.

Future work: if more time is provided: Maybe more data can be added to the dataset to see the difference in results in future.

To conclude: Adding velocity and acceleration feature did not improve the accuracies much and also when I reduced the pressure pixel less than 20 pixels, accuracies were not much affected.

6 References

1. Pattern recognition and Machine learning author: Christopher M. Bishop