**Datasets that contain queuing/standing in a line data**

There are papers on HAR that have used *waiting in a line* data but they haven't cited their datasets and they have all used manually generated private datasets. Some of the papers that I came across that use queuing data are:

[QueueSense: Collaborative Recognition of Queuing on Mobile Phones](http://pecs.mines.edu/docs/secon14.pdf)

[Hierarchical algorithm for daily activity recognition via smartphone sensors](https://ieeexplore.ieee.org/document/7389084)

[Qnalyzer: Queuing Recognition Using Accelerometer and Wi-Fi Signals](https://people.cs.nctu.edu.tw/~ldvan/paper_list/Qnalyzer=%20Queuing%20Recognition%20Using%20Accelerometer%20and%20Wi-Fi%20Signals.pdf)

[Divide and Conquer-Based 1D CNN Human Activity Recognition Using Test Data Sharpening †](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5949027/#B35-sensors-18-01055)

[LineKing: Coffee Shop Wait-Time Monitoring Using Smartphones](https://ieeexplore.ieee.org/document/6991601)

[LineKing: Crowdsourced Line Wait-Time Estimation using Smartphones](https://cse.buffalo.edu/~demirbas/publications/lineking.pdf)

**Available Deep Learning Models for HAR**

The following are the deep learning baseline methods for time series classification in literature.

* MLP, FCN, ResNet, Encoder, MCNN, t-LeNet, MCDCNN, Time-CNN, TWIESN

These methods have been discussed in greater detail along with a comparative analysis [here](https://arxiv.org/pdf/1809.04356.pdf). These methods are also compared with the state-of-the-art statistical time series classification methods like [BOSS](https://www2.informatik.hu-berlin.de/~schaefpa/boss.pdf), [COTE](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7069254), DTW. All these comparisons are made on 85 univariate time series in the [UCR and UEA](http://www.timeseriesclassification.com/index.php) archive and 12 multivariate time series. (The repository now contains 128 univariate datasets and 30 multivariate datasets.) [This](https://github.com/hfawaz/dl-4-tsc) GitHub repository gives the code to reproduce the results of the above mentioned deep learning baselines.[This](https://arxiv.org/pdf/1810.07758.pdf) paper provides details on the datasets in the archive and guidelines to use them.

ResNet and FCN are the models which have the highest number of wins on the benchmark datasets.

I have used FCN architecture on the inpocket data.

**Inpocket Data Analysis**

**Results of training FCN on 5 classes**

I had trained the initial model on 5 classes(excluding the In Train class) and I obtained the following confusion matrix on validation.

[[ 420, 23, 0, 55, 116],  
 [ 1, 298, 0, 31, 262],  
 [ 20, 91, 147, 7, 151],  
 [ 2, 18, 236, 872, 80],  
 [ 131, 186, 51, 85, 2101]]

The rows represent the actual values and the columns represent the predictions. An entry *a i,j* (i is the row and j is the column) corresponds to *a i,j* samples belong to class *i* were predicted as *j*.The following is the accuracy ratio for the classes:

0 **0.684** 1 **0.5** 2 **0.35** 3 **0.721** 4 **0.822**

The overall accuracy was **71.28** %.

From the confusion matrix, we can infer that the overall accuracy is a misrepresentation of the model’s performance due to the following reasons:

1. The dataset was highly imbalanced with the walking class being the majority class and the 0.822 accuracy value can be attributed to that.
2. Queue shows an individual accuracy of 0.35 with more than half of Queue samples misclassified as walking.
3. Climbing down also has about half the samples misclassified as Walking.
4. Standing accuracy shows a decent value but that is perhaps because this data didn't include the In Train class, which has time series quite similar to the Standing class.



***Data belonging to different classes looks similar after visualization***

After visualizing the *X\_AXIS, Y\_AXIS, Z\_AXIS accelerometer* time series for a window length of 200 timesteps for all the 6 classes - Climbing Up - 1, Climbing Down - 2, Queue - 3, Standing - 4, Walking - 5, In Train - 6, I have inferred the following:

1. *Standing* and *In Train* have similar time series with accelerometer values close to 0 with a very less standard deviation
2. *Climbing up* and *Climbing down* can be differentiated from the rest of the time series by their high range of accelerometer values. ~~However, these classes are quite similar themselves.~~ What is surprising is that misclassification errors in the confusion matrix between these two classes are quite less. The data was trained on all features and we have visualized only the accelerometer values. Perhaps there was something in the other features to have differentiated between these classes. (Note: The features, however, didn't include barometer values).
3. About **42%** data in *Climbing up* and **28%** data in *Climbing down* class don't possess the characteristic high range value of these classes but are quite similar to other classes like Queue and Walking. These percentages are quite high to be considered as outliers and might hamper the training.
4. *Walking* class have some time series similar to *Queue* class and a few are similar to the *Climbing* classes.

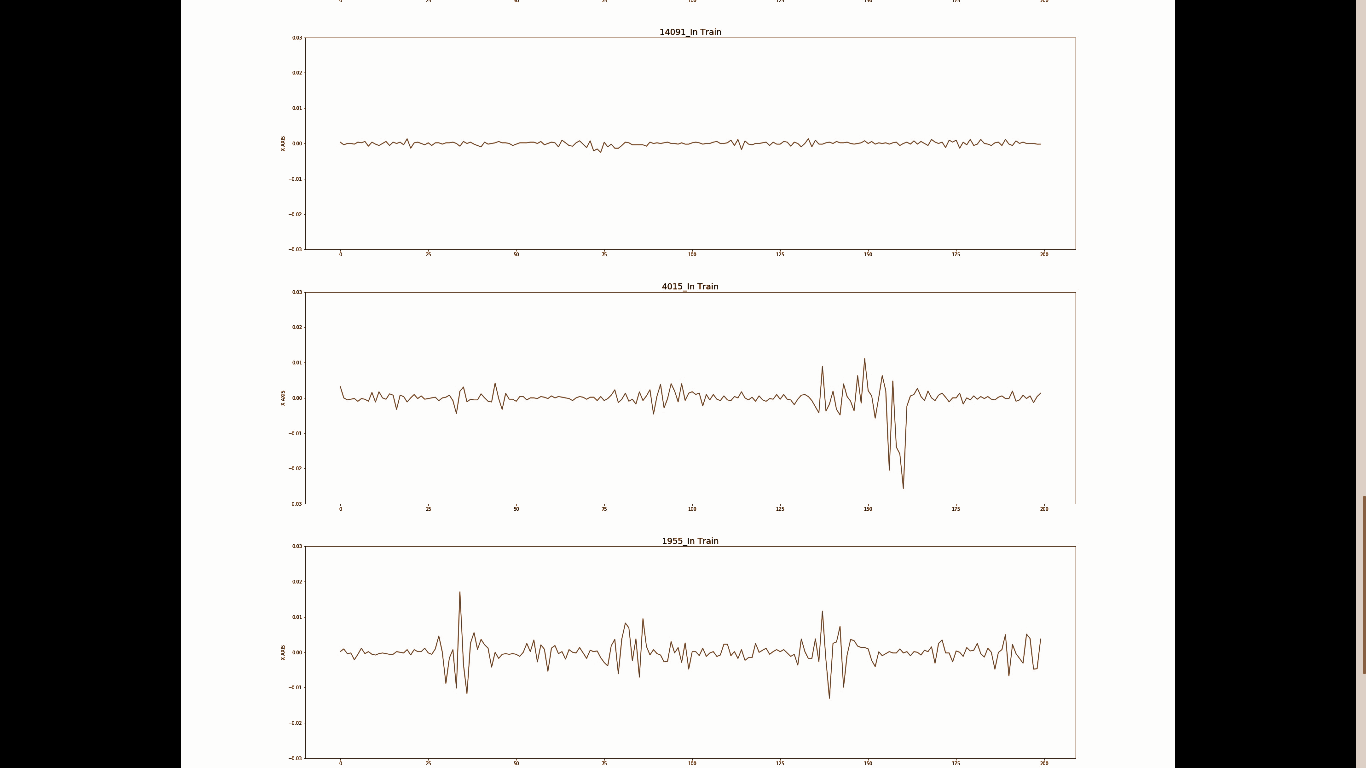
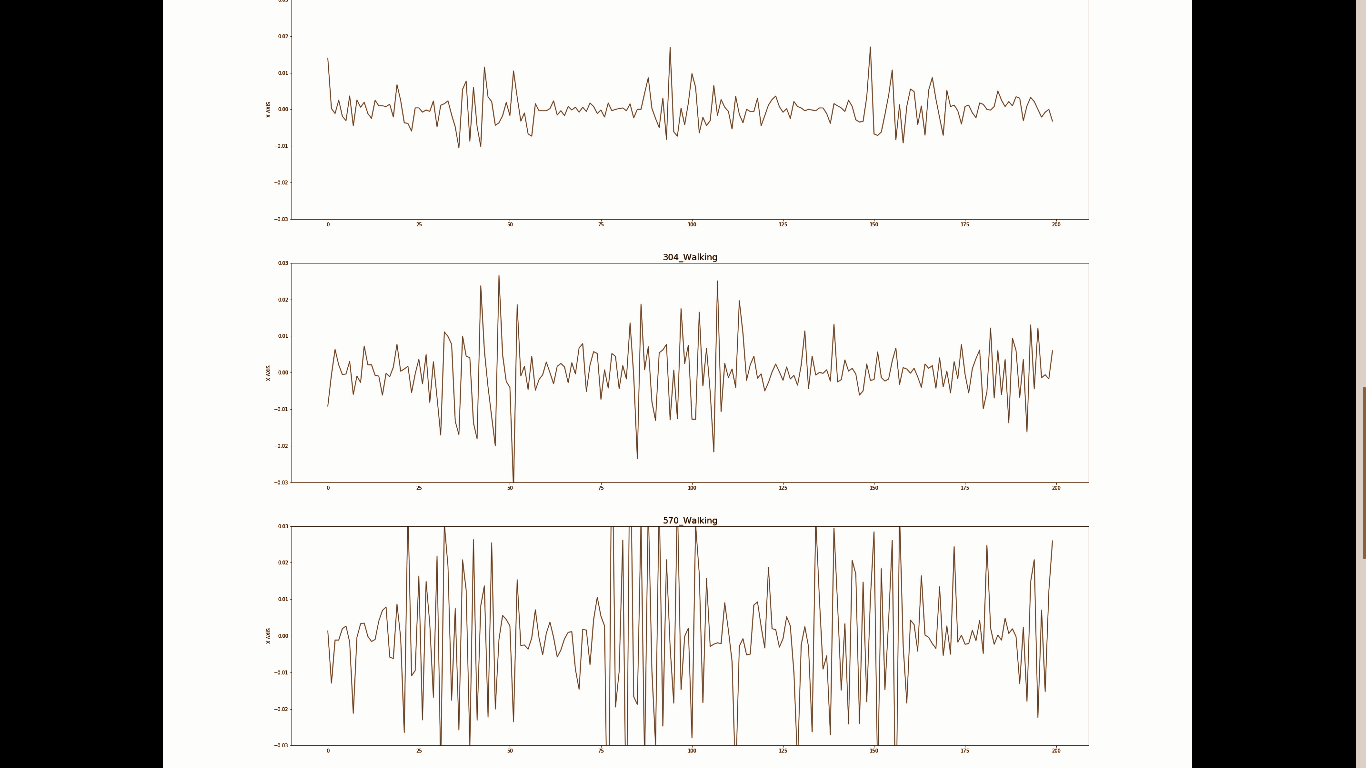
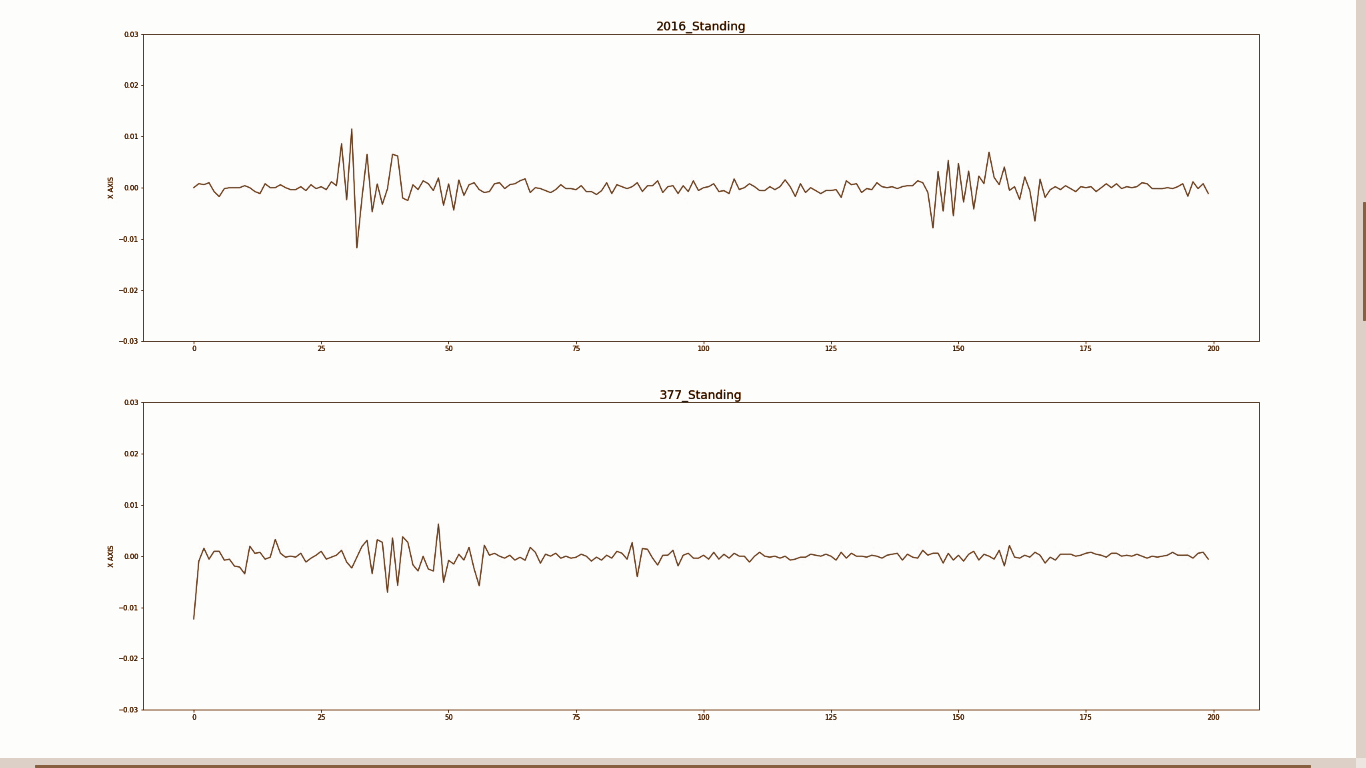
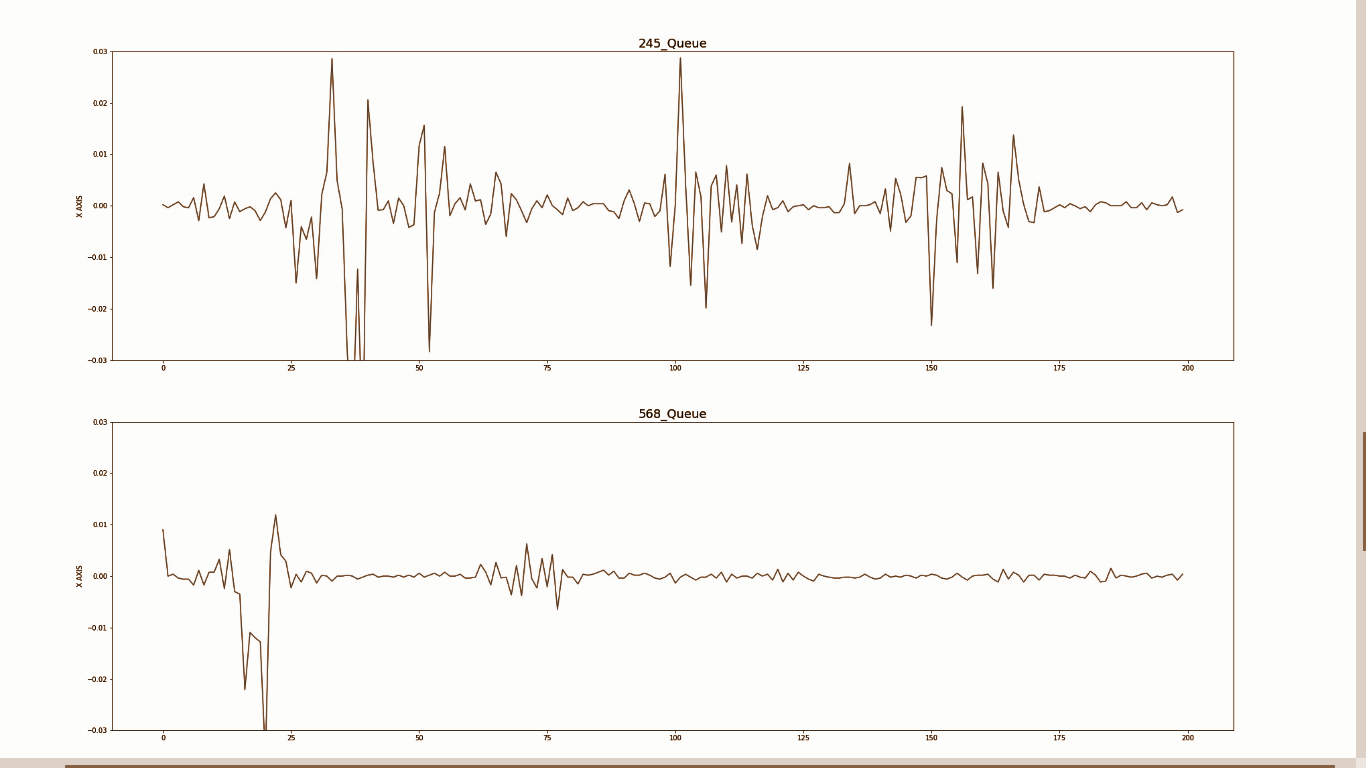
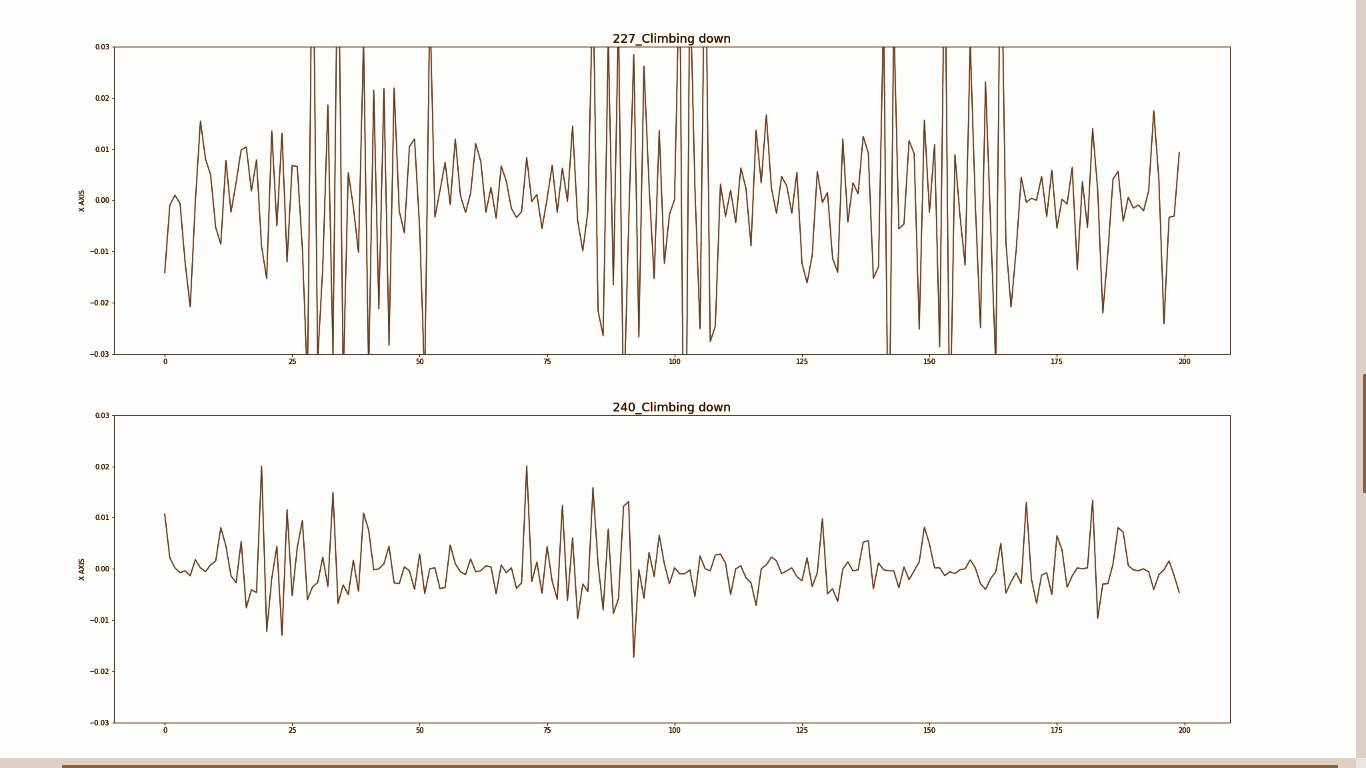
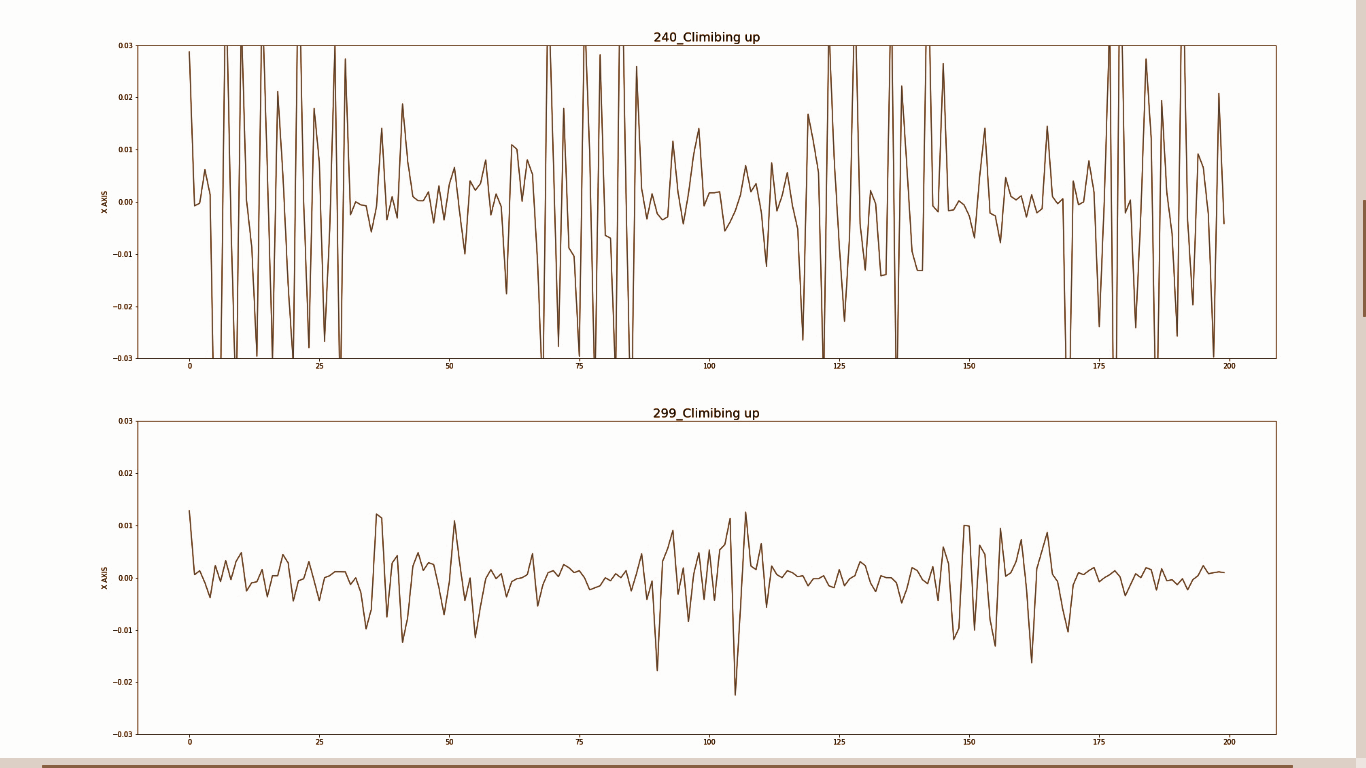
**Length of time series**

The dataset was trained on a time series of length 24 and later it was decided to train it for a length of 35. The pitfalls of using such a length are as follows:

1. *Climbing* classes are characterized by peak and trough. Hence for small lengths, the model will only get to observe the peak and trough separately and is likely to misclassify. For the given data, a peak and a trough span over a length of 50 timesteps.
2. Depending on the waiting time in the queue, it is necessary to have a sufficiently long time series such that the model observes both, the moving and waiting, in the queue. In this case, it is difficult to estimate what can be an appropriate input length for a CNN based network.

**Number of training samples**

The dataset contains an uneven number of training samples per class. However, this can be countered by undersampling to attain class balance. But this will significantly reduce the number of training samples available for training. Also since we are using the windowing technique of generating data samples, there is a tradeoff between the number of samples and length of time series.



**Analysis of the Cascaded Classifier**

The dataset of *standing, walking,* and *queuing* was trained under the following conditions:

1. The first classifier was trained on standing and walking data. The architecture was an FCN + softmax. (89%)
2. The second classifier was trained on the previous data + queue data and had output as ‘queue’ and ‘not queue’. The output of the GAP layer of the first classifier was fed as input to the second and the second classifier was trained with the first classifier frozen. (92%)
3. Together the model gives an accuracy of 86%. However, the accuracy for *queuing* is 69% and *not queuing* is 96% and hence 86% is a misrepresentation of the performance and this model needs to be re-trained and evaluated on balanced class data.

The aim of such a network was that the first classifier will specialize in classifying between Standing and Walking which is an easier problem. And when given Queue input, it won't fire up the cells corresponding to Standing and Walking data and this will make it easier for the second classifier to recognize Queue data.

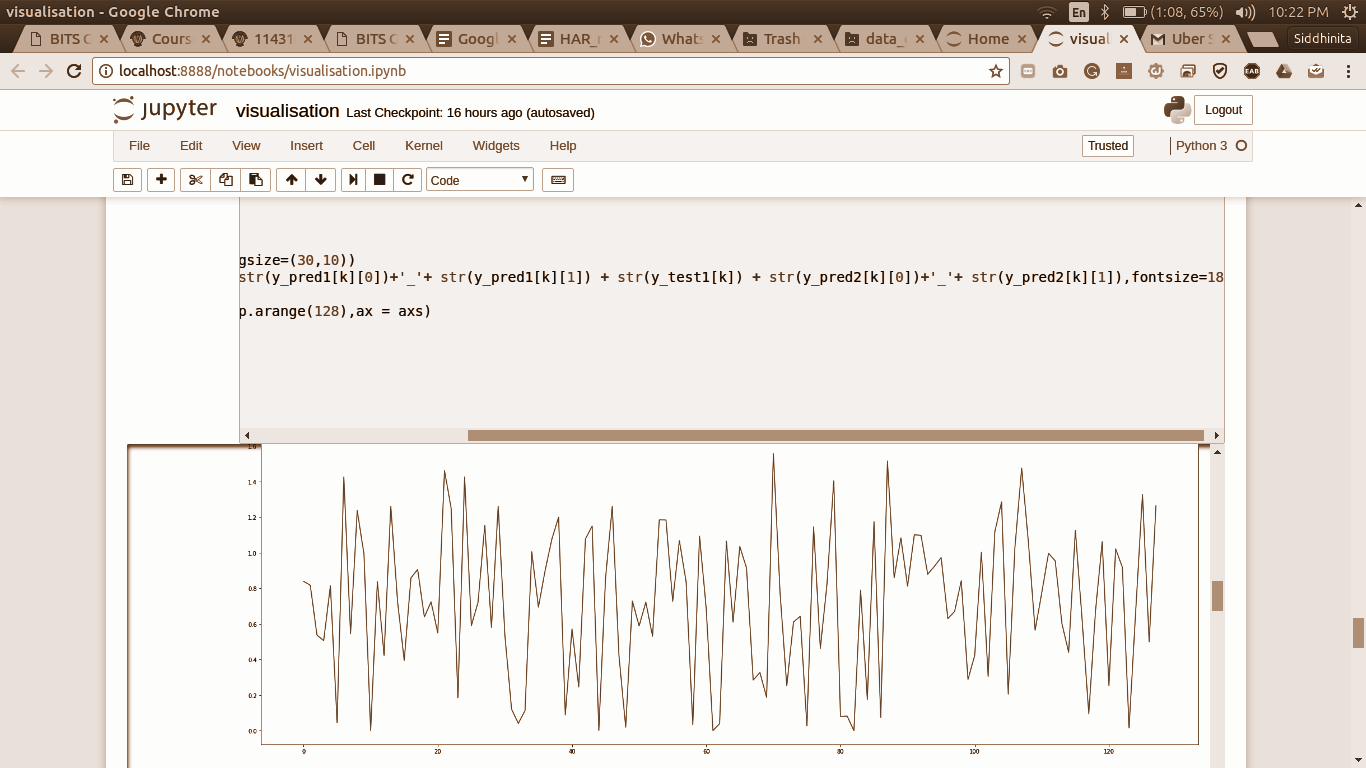
On visualizing the output from GAP layer for all 3 classes, the following nature was observed for the 3 kinds of data: (Queue, Standing, Walking in this order)

Thus, the network encodes them into distinct values and they can be segregated.

The following are the confusion matrix for Standing - Walking and Queue - Not Queue validation data respectively.

[[2362, 374],  
 [ 123, 2012]]

[[ 466, 204],  
 [ 192, 4679]]



**Analysis of Walking Queuing Classifier**

Since *Standing* class is very different from *Walking* and *Queuing*, I decided to focus on the problem of *Walking* and *Queuing*. Since *Walking* data had significantly more samples than *Queue,* I undersampled the data to make the samples equal. I used FCN and also experimented with different architectures, but the model did not converge and hence, stating the accuracy values is pointless. This can be attributed to the following reasons:

1. The similarity between the two classes. As explained in the analysis of data above, about 20% queue data is similar to walking data.
2. The length of the window is small, making it difficult to capture the characteristic nature of any time series.
3. The amount of data is small leading to overfitting.

**Possible future approaches**

Keeping in mind the problems encountered during the course of this classification problem, the following alternatives can be explored:

1. **Less Data Problem**: We can use a pre-trained HAR classifier and finetune it on our dataset.
2. **Increasing Window Length**: The classifier can be evaluated for longer window length.
3. **Dropping out samples that are common between classes**: There was probably some discrepancy in labeling the data. There are long stretches of time series (about 200 and more timesteps) in Queue data that corresponds to Walking data. Even a time series of length 200 produces about 165 samples of length 35 by the windowing technique. Such data needs to be dropped as it hampers the learning process.
4. **Exploring the cascading classifier**: Training the cascading classifier with a balanced class dataset is one way to proceed.
5. **Exploring other datasets**: Trying to solve the problem for this specific dataset can possibly make this problem more complicated than it actually is. So, exploring other datasets could help in moving forward.