**Interim Report - Human Activity Recognition**

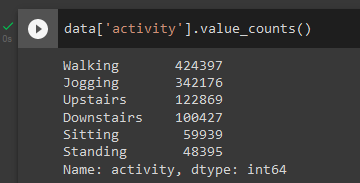
1. Introduction

Human Activity Recognition (HAR) is classifying the activity of a person using responsive sensors that are affected by human movement. Both users and capabilities(sensors) of smartphones increase and users usually carry their smartphones with them. These facts make HAR more important and popular. This work focuses on the recognition of human activity using smartphone sensors using different machine learning classification approaches. Data retrieved from smartphones’ accelerometer sensors are classified to recognize human activity. Results of the approaches used are compared in terms of efficiency and precision.

1. Data exploration

This is the summary of the dataset that I have used. This dataset contains 6 activities having nearly 11 lakhs records.

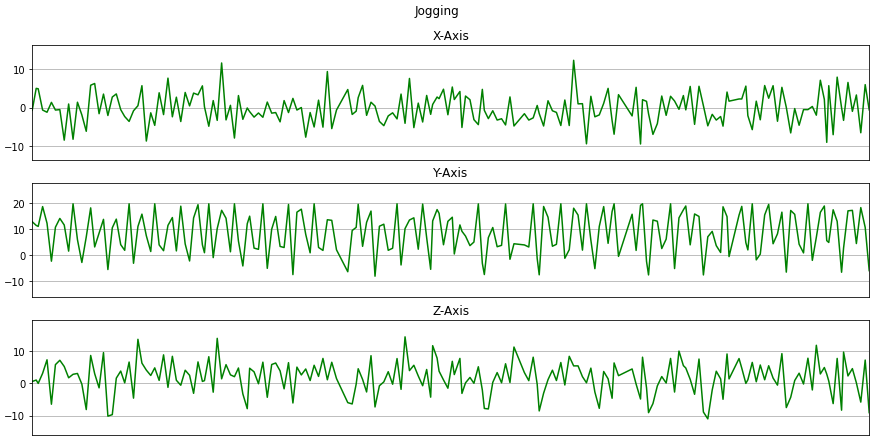


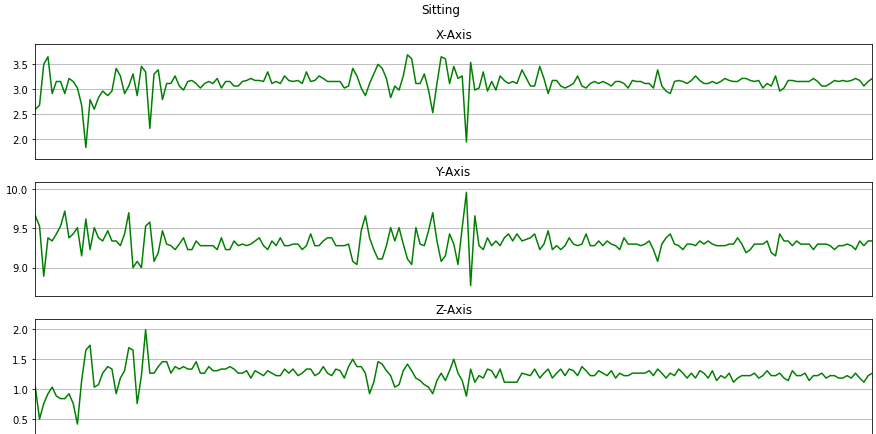


From the above snapshot, we can see that this dataset contains highly unbalanced data. Means here walking and jogging have more no of records i.e. 424397 and 342176 records respectively while standing has 48395 records only. If we use this dataset then it is going to be highly overfitted and skewed towards walking and jogging.So we need to balance the dataset, for that what I did I took only 48395 records from each activity.

After exploring the dataset I tried to plot these accelerometer values for timestamp 10sec so that we can see how the accelerometer data looks visually for each activity. Because each activity follows a specific pattern and by looking at these patterns we can classify which accelerometer values belong to which class.



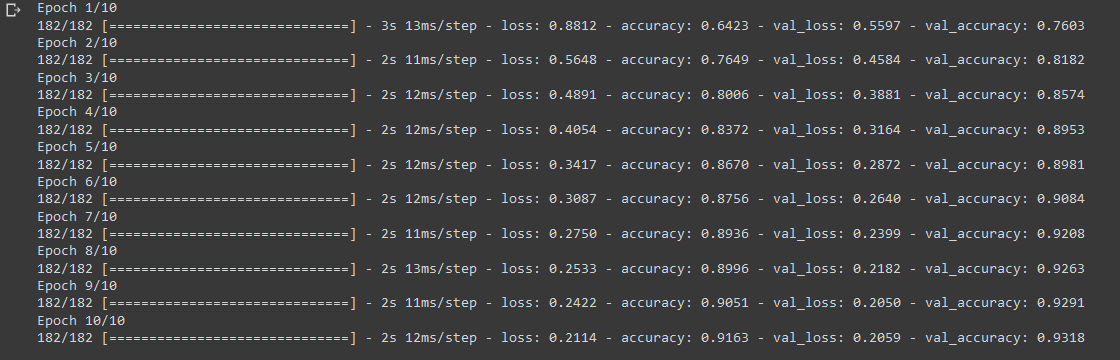




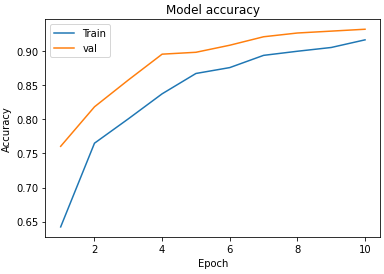
1. Implementation of the CNN-LSTM model

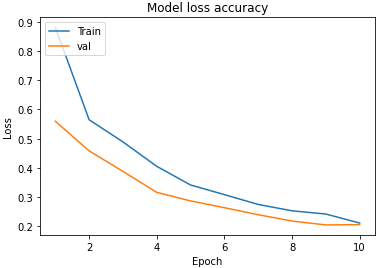
Here I have called sequential layer 1st. In 1st layer, I have added a 2-dimensional convolution layer that is Con2D. Then 16 filters are passed having [2,2] kernel size and activation function relu is used. For 1st layer input shape is x-train. Then I added the dropout layer means randomly 0.1 or 10% of neurons have been dropped. Another layer of CNN having 32 layers and size of [2,2] with activation function relu is added.

In the hidden Convolution layer, we don't need to provide input shapes because it automatically matches the preceding layers. Then 20% dropout will be added. Then add flatten then dense layer having 64 and activation function relu and drop 50% neuron randomly. Then add the final layer. As we need 6 classes and it is a multi-class classification we are taking softmax as an activation function. For compilation Adam optimizer is used and the loss function is sparse categorical cross-entropy. Then the training process will start. I have used several epochs to reach the final accuracy of approx 94%.



The snapshots below represent the learning curves for 10 epochs. 1st graph is plotted between no of epochs and the accuracy of the model. Then the 2nd graph is plotted between no of epochs and loss of a model. Here we got quite a good accuracy. As validation loss is less than training loss we can say that our model is neither overfitting nor underfitting.





The snapshots below represent the learning curves for 50 epochs.

