Research Project

**Human Activity Recognition**

**Big-Data Systems and Intelligence Analytics**

**CSYE 7245 - SPRING 2018**

Prof. **Nick Bear Brown**

Report By:

**Harshad Pardeshi, Krutika Deshpande, Prithvi Kamath**

**Table of Contents**

[Description of the topic 3](#_Toc510182138)

[Background Research of related work 3](#_Toc510182139)

[Data sources 4](#_Toc510182140)

[What algorithms are being used and code sources 5](#_Toc510182141)

[Importing Data 5](#_Toc510182142)

[Exploratory Data Analysis 6](#_Toc510182143)

[Implementing Machine Learning models 8](#_Toc510182144)

[Using Deep learning models 9](#_Toc510182145)

[Next Steps 11](#_Toc510182146)

[References 11](#_Toc510182147)

# Description of the topic

Human activity recognition plays a significant role in human-to-human interaction and interpersonal relations. Because it provides information about the identity of a person, their personality, and psychological state, it is difficult to extract. The human ability to recognize another person’s activities is one of the main subjects of study of the scientific areas of computer vision and machine learning. As a result of this research, many applications, including video surveillance systems, human-computer interaction, and robotics for human behavior characterization, require a multiple activity recognition system.

As we understand the importance of human activity recognition, have you ever wondered how your smartphone, smartwatch or wristband knows when you’re walking, running or sitting?

Well, your device probably has multiple sensors that give various information. GPS, audio (i.e. microphones), image (i.e. cameras), direction (i.e. compasses) and acceleration sensors are very common nowadays.

We will use data collected from accelerometer sensors which was built from the recordings of 30 study participants performing activities of daily living (ADL) while carrying a waist-mounted smartphone with embedded inertial sensors. The objective is to classify activities into one of the six activities performed viz WALKING, WALKING\_UPSTAIRS, WALKING\_DOWNSTAIRS, SITTING, STANDING, LAYING.

Virtually every modern smartphone has a tri-axial accelerometer that measures acceleration in all three spatial dimensions. Additionally, accelerometers can detect device orientation.

In this part of the series, we will train an LSTM Neural Network (implemented in TensorFlow) for Human Activity Recognition (HAR) from accelerometer data.

# Background Research of related work

In 2001, Paul Viola and Michael Jones invented an efficient algorithm for face detection. Their demo that showed faces being detected in real time on a webcam feed was the most stunning demonstration of computer vision and its potential at the time. Soon, it was implemented in OpenCV and face detection became synonymous with Viola and Jones algorithm.

Every decade or so a new idea comes along that is so effective and powerful that you abandon everything and wholeheartedly embrace it. Deep Learning is that idea of this decade. Deep learning models have crushed other classical models on the task of image classification and they are now state of the art in object detection as well.

One of the first advances in using deep learning for object detection was [OverFeat](https://arxiv.org/abs/1312.6229) from NYU published in 2013. They proposed a multi-scale sliding window algorithm LSTM Neural Network.

We want to apply these learnings about these advances in algorithms for human activity recognition as currently it is a fundamental problem in perception that is receiving increasing attention.

# Data sources

30 volunteers within an age bracket of 19-48 years were asked to perform six activities (WALKING, WALKING\_UPSTAIRS, WALKING\_DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist. Using its embedded accelerometer and gyroscope, 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz was captured. The experiments have been video-recorded to label the data manually. The obtained dataset has been randomly partitioned into two sets, where 70% of the volunteers was selected for generating the training data and 30% the test data.

The sensor signals (accelerometer and gyroscope) were pre-processed by applying noise filters and then sampled in fixed-width sliding windows of 2.56 sec and 50% overlap (128 readings/window). The sensor acceleration signal, which has gravitational and body motion components, was separated using a Butterworth low-pass filter into body acceleration and gravity. The gravitational force is assumed to have only low frequency components, therefore a filter with 0.3 Hz cutoff frequency was used. From each window, a vector of features was obtained by calculating variables from the time and frequency domain.

Each record contains:

* 6 activities for 30 users
* Triaxial acceleration from the accelerometer (total acceleration) and the estimated body acceleration
* Triaxial Angular velocity from the gyroscope
* 128 data points taken within 2.56 seconds with 50% values carry forwarded
* A 561-feature vector with time and frequency domain variables (generated from 128 data points)
* Its activity labels
* An identifier of the subject who carried out the experiment

The dataset includes the following files:

* ‘body\_acc\_x\_train’, ‘body\_acc\_y\_train’, ‘body\_acc\_z\_train’: X, Y, and Z co-ordinates of body accelerometer readings. Has 7352 observations
* ‘body\_gyro\_x\_train’, ‘body\_gyro\_y\_train’, ‘body\_gyro\_z\_train’: X, Y, and Z co-ordinates of gyroscope readings. Has 7352 observations
* ‘total\_acc\_x\_train’, ‘total\_acc\_y\_train’, ‘total\_acc\_z\_train’: X, Y, and Z co-ordinates of total accelerometer readings. Has 7352 observations
* 'features\_info.txt': Shows information about the variables used on the feature vector
* 'features.txt': List of all features
* 'activity\_labels.txt': Links the class labels with their activity name
* 'train/X\_train.txt': Training set has 7352 records
* 'train/y\_train.txt': Training labels for 7352 records
* 'test/X\_test.txt': Test set with 2947 records
* 'test/y\_test.txt': Test labels for 2947 records

# **What algorithms are being used and code sources**

Human activity recognition data is distributed and clean data for analysis, thus we can apply many algorithms to this problem. It’s a classification problem so we apply K nearest neighbors to classify activities. If this algorithm doesn’t give a satisfying solution, we will apply neural networks and LSTM using RNNs to correctly classify into these categories.

# Importing Data

We are programmatically importing data from the below mentioned files along with the labels for their respective activities and generating a 3-D matrix of size (7352 X 128 X 9)

* body\_acc\_x\_train
* body\_acc\_y\_train
* body\_acc\_z\_train
* body\_gyro\_x\_train
* body\_gyro\_y\_train
* body\_gyro\_z\_train
* total\_acc\_x\_train
* total\_acc\_y\_train
* total\_acc\_x\_train

We have a similar set of files to test our algorithms namely:

* body\_acc\_x\_test
* body\_acc\_y\_test
* body\_acc\_z\_test
* body\_gyro\_x\_test
* body\_gyro\_y\_test
* body\_gyro\_z\_test
* total\_acc\_x\_test
* total\_acc\_y\_test
* total\_acc\_x\_test

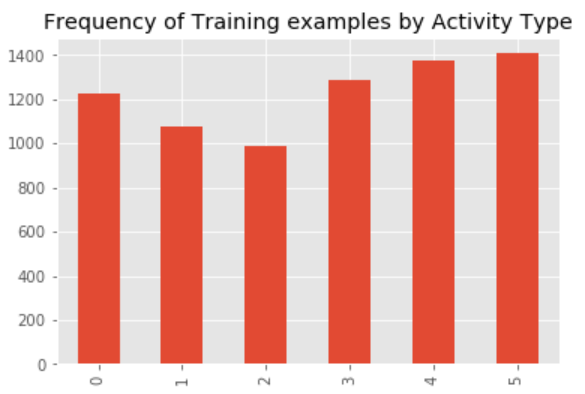
We then convert this 3-D matrix into a dataframe to implement machine learning models.

# 

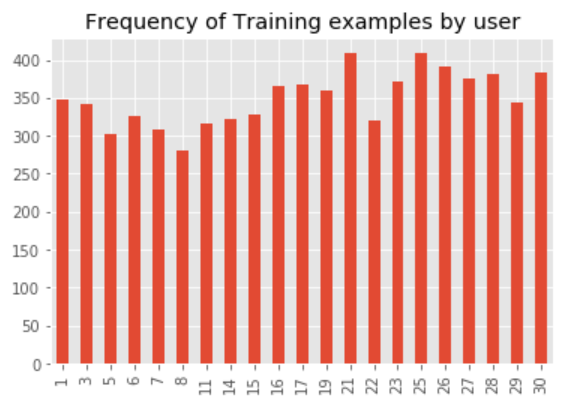
# Exploratory Data Analysis

We perform exploratory data analysis to find the distribution in the dataset

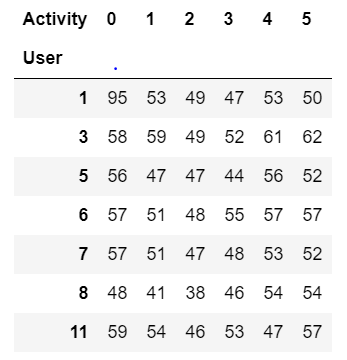
* Frequency of Training examples by Activity Type



* Frequency of Training examples by user



* User-wise Activity frequency

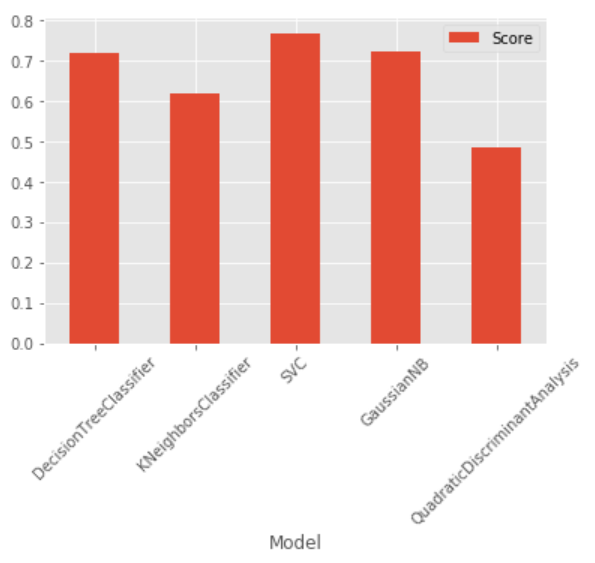


# 

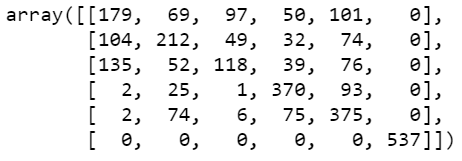
# Implementing Machine Learning models

With exploratory data analysis done, we move on to implementing machine learning models on our dataset. But as we know that our dataset is a 3-D matrix, we’ll have to convert it to a dataframe to implement these models. By converting these models into (7352 X 1152) dataframe we tend to distort the sequence of data. Their respective accuracy is as follows:

|  |  |
| --- | --- |
| **Algorithm Name** | **Accuracy** |
| Decision Tree | 71.97 % |
| K Nearest Neighbors | 61.89 % |
| SVC | 76.95 % |
| Gaussian Naïve Bayes | 72.48 % |
| Quadratic Discriminant Analysis | 48.45 % |



Confusion Matrix for SVC is as follows:



As expected there are lot of prediction errors while implementing these algorithms, hence we implement recurrent neural networks to improve the accuracy.

# Using Deep learning models

As we can see, models that we tried above doesn’t give any satisfactory results. Even if SVC is can achieve an accuracy of 61%, this model will not perform as expected if we change the inputs or move the sensor position. We don’t need to consider 128 readings as different parameters to a model, rather we need to discover patterns from the sequence of readings that we have. To do this we tried to implement Recurrent Neural Network

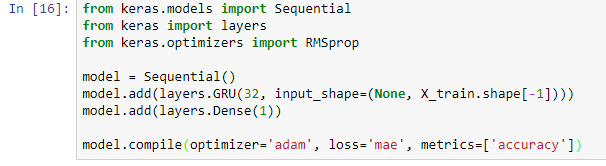
What is RNN?

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs.

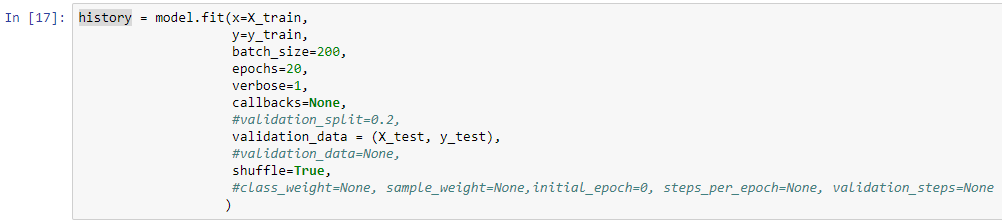
Implementing RNN

We tried implementing a single layer Recurrent Neural Network on our data. To do this we use Keras library. Keras is an open source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or MXNet. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer.

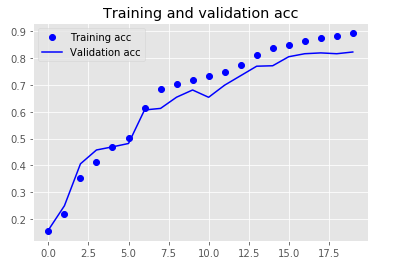
To add a Recurrent layer, we make use of a Gated Recurrent Unit (GRU) layer with 32 units. It has reset gate applied to hidden state before matrix multiplication.



We tried using different optimizers such as SDG, RMSprop and adam. Adam turned out be give best results for our data. After this we train our model for 50 epochs.



We tried splitting training data into training and validation sets. This decreased our accuracy as we had less data to train. Rather we use test data to validate our model. This gives us training accuracy of 94% and a validation accuracy of 88.6%

****

# 

# Next Steps

Further we can add more layers to our models and try using different activation functions. We also plan to implement LSTM and compare which will perform better. Going further, we plan to save our model and export it to an android app which will give input to this model and will predict activity.

# References

1. <https://archive.ics.uci.edu/ml/machine-learning-databases/00240/>
2. Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. A Public Domain Dataset for Human Activity Recognition Using Smartphones. 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2013. Bruges, Belgium 24-26 April 2013
3. <https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones>
4. <https://archive.ics.uci.edu/ml/machine-learning-databases/00240/UCI%20HAR%20Dataset.names>