TRACKING SEQUENCE OF ARM MOVEMENTS USING SMARTWATCHES

B.TECH PROJECT REPORT

Submitted by

HARSHITHA.S

(EE16B071)

for the award of the degree

of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

JUNE 2020

CERTIFICATE

This is to certify that the project report entitled **“TRACKING SEQUENCE OF ARM MOVEMENTS USING SMARTWATCHES”** submitted by **HARSHITHA.S** to the Indian Institute of Technology, Madras for the award of the degree of **Bachelor of Technology** is a bonafide record of project work carried out by her under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Project Guide:

Dr.Pratyush Kumar

Assistant Professor

Department of Computer Science and Engineering

Indian Institute of Technology Madras

Chennai – 600 036.

Place: Chennai

Date: June 2020

Acknowledgement

I would like to thank my project guide Professor Pratyush Kumar for his continuous support and guidance throughout this project. I am very grateful to him for sharing his invaluable time and expertise throughout the span of this project and also for providing me with the required resources.

Last but not the least, I would like to thank my family and friends who were always there to help and support me.

Abstract

In this project, activity performed by the user specifically, the arm movement, is tracked using a smartwatch. Additional wearable sensors or cameras are not used. The solution presented is preferable as smartwatches have become more accessible and relatively less expensive these days. To achieve this objective of activity tracking, data from the sensors, namely the accelerometer and gyroscope, present in the smartwatch was used. This data was mapped to different arm movements using machine learning models. In this project various machine learning models were trained and recurrent neural networks (RNN) gave the best results. The number of repetitions of the activity cycle along with the activity performed at every instant as predicted by the RNN model was displayed on the Android App that was developed and installed on the smartwatch. This project work can be further extended to predict more activities that involve the movement of the entire body. It can then be used for effective monitoring of yoga or exercise cycle with feedback from a professional yoga instructor or trainer.

CONTENTS

LIST OF FIGURES AND TABLES

CHAPTER 1

Introduction

Smartwatches have become increasingly prevalent. They are being used as activity trackers to track the distance walked or run, calorie consumption, and in some cases heartbeat. These devices which are currently being used for general activity tracking, can also be used for specific arm movement monitoring by leveraging the sensors present in it. These devices offer opportunities for activity detection due to their widespread uptake [1].

In this area of activity monitoring, two approaches are usually used. They are done using external sensors/cameras and wearable sensors [1]. The approach that uses external sensors makes use of computer vision and requires the person to be in the prescribed range. These systems may not be feasible and can be expensive too due to excessive amounts of accessories. The approach using wearable sensors is being used for motion capture. This requires the conversion of sensory data to higher-level activity information [1]. The drawback of this approach is that the system causes inconvenience to the users. This is why smartwatches have started gaining popularity in the tracking of activities. They also provide facilities such as multimodal user interfaces and Bluetooth connectivity that are absent in wearable sensors.

To achieve the goal of this project in effectively tracking a sequence of arm movements using a smartwatch, an app is required to be developed to take in the input from the sensors on the smartwatch and also to display the output. This app can be an independent one or can also work along with a smartphone. The focus of this project is also to use minimal devices and thus an independent app for the smartwatch would be ideal.

The task of classification of the sensory data and activity tracking also requires training of machine learning models. Since the output can be determined in the training phase of this data, supervised classification algorithms can be used to tackle this problem. Some of the algorithms that are typically used are Support Vector Machines, decision trees, Naive Bayes and neural

networks. Different variants of neural networks such as Artificial neural networks, Recurrent Neural Networks and Convolutional Neural Networks have gained popularity and are being used for classification as well.

In this project, the possibility of using only the sensors present in a smartwatch to track the arm movement of the user has been explored. The major objectives of this project are to achieve the following:

1. Use minimal sensors to track the movement.
2. Train various machine learning models and find the most suitable model for this project.
3. Explore the different ways to process and store the data for the purpose of prediction.
4. Develop an app for activity tracking and assess the project's potential for further developments.

This report consists of seven chapters. This introduction chapter consists of the objective of this project. Chapter 2 consists of the related work done in the field of activity recognition and tracking. Chapter 3 explains the background information required in understanding the project. Chapter 4 describes the approaches taken to achieve the goal of this project. Chapter 5 lists out the experimentation done to find a suitable machine learning model and the results have been analysed in chapter 6. The final chapter summarizes the work and suggests possible directions for future work.

CHAPTER 2

Literature Review

This chapter is an overview of the related works done in the field of activity tracking and gesture recognition with a closer look on the tracking done using smartwatches.

2.1 Different Techniques Used for Tracking and Recognition

Hand gesture recognition is an important component of today's modern human-computer interaction systems such as virtual reality, computer games and intelligent home appliances [2]. One of the most widely used methods to do this is to use cameras. The images captured by the camera are then processed using computer vision. Feiyu Chen and others [3] used an interesting approach in this domain. They used a wrist-worn camera which improved the recognition of fingers and finger segmentation. High accuracy of 99.83% was achieved in the hand gesture recognition of the numbers from zero to nine. A drawback of this method is that the person needs to be in the field of view of the camera and the computations are intensive and complicated. Also in the case of the wrist-worn camera, having a bulky camera can be uncomfortable.

Another prominent method to track movements is to use wearable sensors. Alessandra Moschetti and others [4] in their work propose to recognise gestures using hand and wrist wearable sensors. IMUs were used to recognise nine different gestures with an accuracy of 89.01%. A sensor was used for the wrist, wearable sensors were used for three fingers and its coordinating module was placed on the backside of the hand. In yet another research work by Alessandra Moschetti and others [5], a comparison between supervised and unsupervised methods to recognise nine gestures was made. To do this, two sensors i.e., one on the wrist and one on the index finger, were used. An accuracy of 94% was achieved using supervised methods and 81% for unsupervised methods, showing the possibility of using unsupervised methods in the future. This method also has a limitation due to the inconvenience caused to the user.

Marco E. Benalcázar and others [6] have proposed another method of hand gesture recognition using muscle activity detection. In this method, a Myo armband was used to measure the surface electromyography or the electrical activity of the muscles of the forearm muscles close to the elbow. An accuracy of 89.5% was achieved in recognising five gestures. The downside of this model is that it is user-specific and the time and frequency features of the EMG change with the thickness and temperature of the skin, the thickness of the fat between the muscle and the skin, velocity of the blood flow and location of the sensor.

Smartwatches are becoming more accessible and are being used for activity tracking. Bobak Mortazavi and others [7] proposed a posture tracking platform based on a smartwatch. In this work they identified the human postures of sitting, standing and lying down using a smartwatch and compared against a smartphone. The smartwatch provided strong results with an F-score of 0.93 indicating its ability to replace smartphones in tracking human postures.

2.2 Usage of Smartwatches in Tracking and the Algorithms Used

Smartwatches are ideally situated for tracking hand-based activities and can form the basis of new biomedical and health applications [8]. These smartphones also have certain limitations such as smaller screen size, limited battery and storage capacity, weaker hardware for sensing and computing [7]. Bobak Mortazavi and others have taken these into account and have trained their data with 30 features using the support vector machine model to obtain an F-score of 0.93.

In another comparative study of smartwatches and smartphones by Gary.M.Weiss and others [8], it was demonstrated that smartwatches could recognise a wide set of activities that included eating-related activities that smartphones could not. Five machine learning algorithms namely Random Forest, decision tree, IB3 instance-based learning algorithm, Naive Bayes and the multi-layer perceptron algorithm, were used to train the data consisting of 43 features which was produced by performing 18 different activities by the test subjects. Since the conventional classification algorithms do not work well on time series data, 43 higher level features had to be produced.

Sheng Shen developed an arm-posture tracking system using the IMU sensors on a smartwatch. The system does not require training similar to Kinect [9]. It leverages the human arm kinematics to reduce the search space and then uses a hidden Markov model to estimate the translational motion of the elbow or wrist location. This system assumes that the person is standing still and requires one to understand the human skeletal structure and its kinematics. Sheng Shen and others used similar methods to trace the geometric motion of the arm [10]. They developed a system that fuses the IMU sensors on the smartwatch and the anatomy of arm joints into a modified hidden Markov model. This system constructs the 3D arm posture and tracks it continuously without any training. But in this project, it is not required to construct the arm posture in 3D and training of machine learning models is preferred.

Yeongju Lee and Minseok Song developed a machine learning system to recognise stereotyped movements in children with developmental disabilities using a smartwatch [1]. This system used a decision tree classifier to classify six different activities using three feature vectors out of 41 feature vectors from the accelerometer and gyroscope. An average recognition accuracy of 91% was achieved. This system requires a complete analysis of the feature vectors to choose the most appropriate ones. It does not use the time series data directly.

Using two IMU sensors present in the Myo armband (one for each arm), Rafael Rego Drumond and others developed a novel Recurrent Neural Network topology based on Long Short-TermMemory cells to classify motions [11]. Although the sensors (same as that in a smartwatch) are present only on the arms, they wanted to infer complete body action and so selected five such activities. A single action was performed continuously for a certain time by the test subject before moving on to the next action. Using 20 features, 96.63% accuracy was achieved in classifying five actions namely standing idle, walking, running, crouching (which includes staying crouched) and swinging a weapon (attacking) that were performed by test subjects. In this project, we intend to use a lesser number of sensors in the smartwatch and a lesser number of features to identify a sequence of arm movements and count the number of repetitions of the same.

CHAPTER 3

Background information

This chapter provides an overview of the system. The hardware consists of the smartwatch and the software component has majorly two parts : Android app for the smartwatch and server-side software. The detailed explanations on each of these components are respectively under the hardware overview section and the software overview section.

3.1 Hardware Overview

This section is an explanation of the hardware component i.e., the smartwatch. For this project, the Fossil Sport smartwatch was used. This smartwatch consists of the following sensors:

- Goldfish 3-axis Accelerometer
- Goldfish 3-axis Gyroscope
- Goldfish 3-axis Magnetic field sensor
- Goldfish Orientation sensor
- Goldfish Ambient Temperature sensor
- Goldfish Proximity sensor
- Goldfish Light sensor
- Goldfish Pressure sensor
- Goldfish Humidity sensor
- Goldfish 3-axis Magnetic field sensor (uncalibrated)
- Game Rotation Vector Sensor
- GeoMag Rotation Vector Sensor
- Gravity Sensor
- Linear Acceleration Sensor
- Rotation Vector Sensor
- Orientation Sensor

To detect the arm movement, the linear acceleration and angular velocity in the x,y and z axes are required. This data is collected from the accelerometer and gyroscope respectively. An accelerometer is a device that measures non-gravitational acceleration and can sense motion and vibration. Gyroscope is a device that measures angular velocity and can sense orientation.

The device's specifications are as follows:
SoC - Snapdragon Wear 3100 chip
Screen size - 1.19-inch OLED
Storage - 4GB
Battery - 350mAh battery
OS - Google Wear OS
Notable sensors - Heart rate, NFC, GPS, Accelerometer, Altimeter, Ambient light Gyroscope

3.2 Software Overview

As mentioned earlier, the software component has two parts - the android app and the server-side software. The Android application was developed using Android Studio. The Android app is to be installed on the smartwatch. This app is responsible for collecting the data from the sensors. The data is collected only when a sensor event occurs i.e., when there is a change in the value of any of the sensors. This data along with the time is sent to the server for further computation.

The server-side software consists of applications to host two servers. The data sent by the smartwatch is stored in one of the servers. The other server is used to predict the arm movements based on the data present in the first server. An API is used to connect to this server which then outputs the predicted arm movement. The prediction is done using the TensorFlow and TensorFlow lite libraries. Both libraries have been used for two different approaches which are explained in the next chapter. The API is called when the stop button is pressed in the Android app.

CHAPTER 4

Problem Formulation and Approach

The objective is to process the data correctly and test it on a pre-trained machine learning model to obtain the predicted arm movement. Different ways to implement this were explored and are described below. Both the approaches involve data collection and preprocessing.

4.1 Data Collection and Preprocessing

As mentioned in the previous chapter, data is collected when a sensor event occurs. Plots of the sensor data for 1000 samples from the accelerometer and gyroscope are shown in figure 1. To enable accurate predictions, it is important to remove unwanted data. It is found that the number of sensor events occurring can vary for 8 to 10 per second. Hence to have uniformity, data is collected at the rate of 7Hz i.e., 7 sensor events are recorded every second. This also ensures that unwanted data is removed. This processed data is then used in two ways to predict the arm movement.

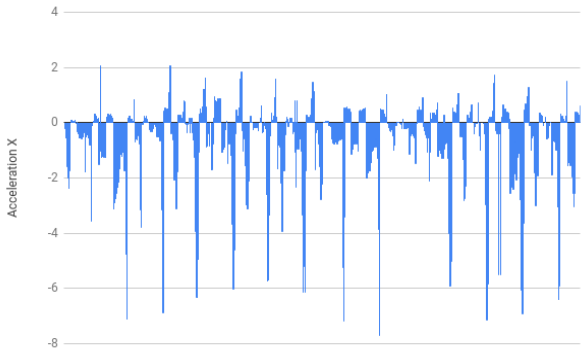

Fig 4.1.a.(i)

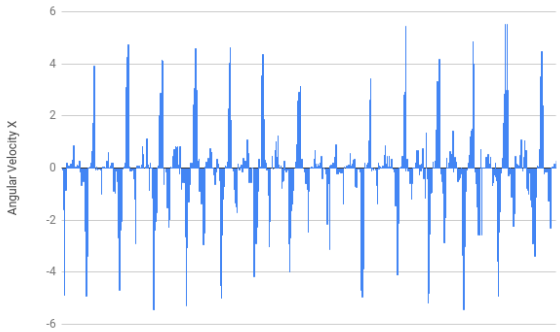

Fig 4.1.a.(ii)

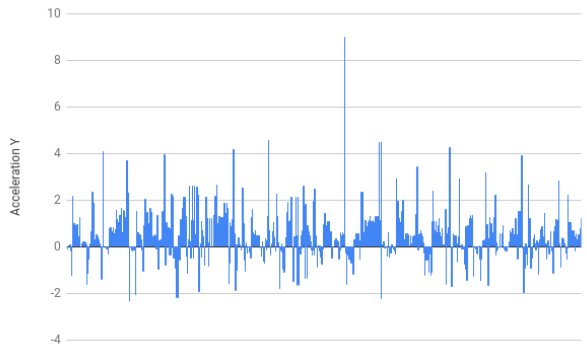

Fig 4.1.b.(i)

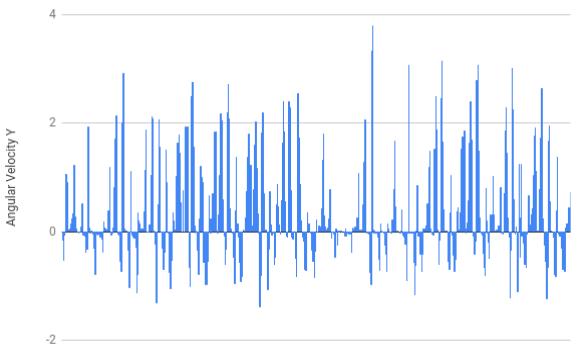

Fig 4.1.b.(ii)

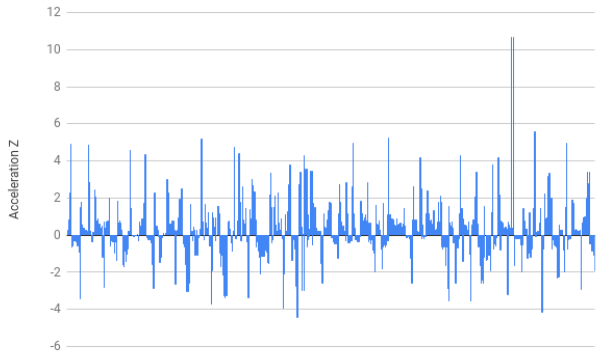

Fig 4.1.c.(i)

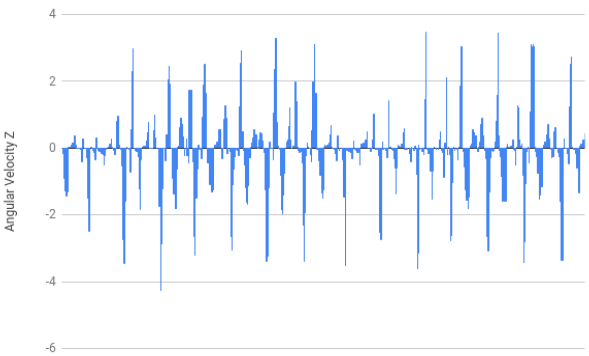

Fig 4.1.c.(ii)

Figure 4.1: Acceleration and angular velocity of 1000 samples of data along the three axes. Fig 4.1.a, Fig 4.1.b, Fig 4.1.c corresponds to X, Y and Z axes respectively. The x-axis in the plots represents sample number and the units for acceleration and angular velocity are m/s^2 and rad/s respectively.

4.2 Approach 1 : Computation done in the Server

In this approach, the sensor data is sent to a database in the server via an HTTP request. On pressing the stop button in the app, an API is called. This API then uses the data present in the database to predict the arm movement for every two seconds or every 14 entries. The predicted arm movement for every two seconds along with the number of activity cycles completed is sent back to the app, which is then displayed. The disadvantage with this approach is that the

predicted arm movements can not be observed instantaneously. The advantage is that the computations performed on the smartwatch are minimised.

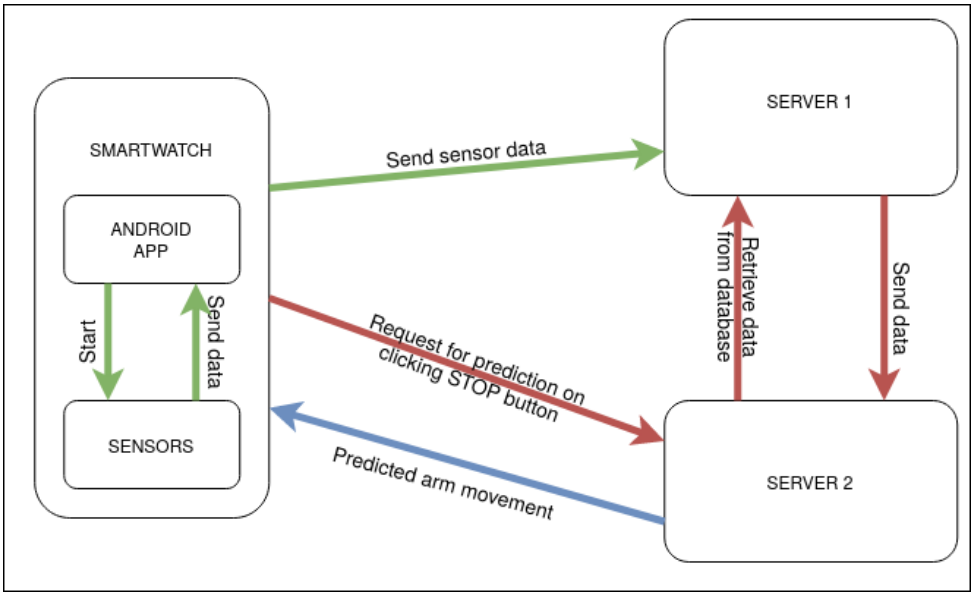

Figure 4.2: Data flow when the computation is done in the server

4.3 Approach 2 : Computation done in the App

In this approach, the sensor data for every two seconds or every 14 entries is stored in the app. Using Tensorflow lite, the data is used to predict the arm movement instantaneously which is displayed in the app. The predictions are stored to calculate and display the total number of activity cycles completed. The advantage of this approach is that the arm movement predictions can be observed continuously.

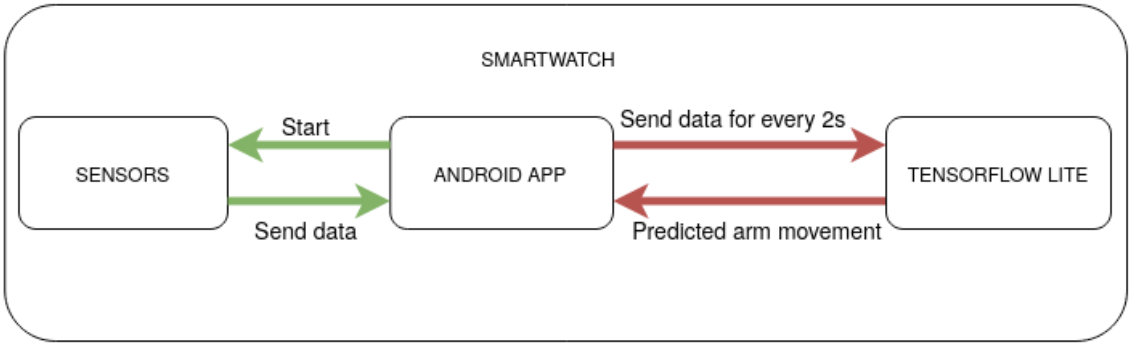

Figure 4.3: Data flow when the computation is done in the app

CHAPTER 5

Experimental setup and results

To implement the approaches described in the previous chapter, a trained machine learning model is required. Various machine learning models were experimented to obtain a suitable one for this project. To do this, tests were performed by one person (the author). For all the tests, the smartwatch was worn on the author's left hand. A set of five arm positions were repeated continuously one after the other. The data thus collected was used to train the machine learning models. Below are the results obtained for the same.

Table 5.1: Experimental results of different machine learning models

Data		**Models**	**Accuracy (in %)**
Class	**No. of items per class**	**(including important parameters)**	
accX, accY, accZ, angularVelX, angularVelY, angularVelZ	25272 X_train shape is 468 X 324 (Note: pca.n_components=175)	Support Vector Machine SVC (C=1.0, kernel='poly')	15.9
		Decision Tree DecisionTreeClassifier (ccp_alpha=0.0, class_weight=None, criterion='entropy')	20.38
		XG Boost XGBClassifier (base_score=0.5, gamma=0, learning_rate=0.300000012, n_estimators=100, num_parallel_tree=1, objective='multi:softprob')	16.56
		Random forest RandomForestClassifier (ccp_alpha=0.0, criterion='entropy', n_estimators=100)	15.29

accX, accY, accZ, angularVelX, angularVelY, angularVelZ	23598 X_train shape is 437 X 324	Neural Network keras.models.Sequential() with tensorflow background loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'] epochs=20, verbose=1	11.17
accX, accY, accZ, angularVelX, angularVelY, angularVelZ	1890 X_train shape is (126, 15, 6)	Recurrent Neural Network keras.models.Sequential() with tensorflow background loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'], verbose, epochs, batch_size = 0, 15, 64 n_timesteps=15, n_features = 6, n_outputs=5 Long Short-Term Memory layer was used.	77.78
	14115 X_train shape is (941, 15, 6)		90.84
time, accX, accY, accZ, angularVelX, angularVelY, angularVelZ	14100 X_train shape is (940, 15, 6)		97.53
time, accX, accY, accZ, angularVelX, angularVelY, angularVelZ	13160 X_train shape is (940, 14, 6)		96.04

In Table 5.1, accX, accY and accZ are the accelerations along x, y and z axes respectively and angularVelX, angularVelY, and angularVelZ are the angular velocities along x, y and z axes

respectively. The confusion matrix corresponding to the last two Recurrent Neural Network Models are shown in figures 5.1.a and 5.1.b. The major difference in the two models is the number of timesteps.

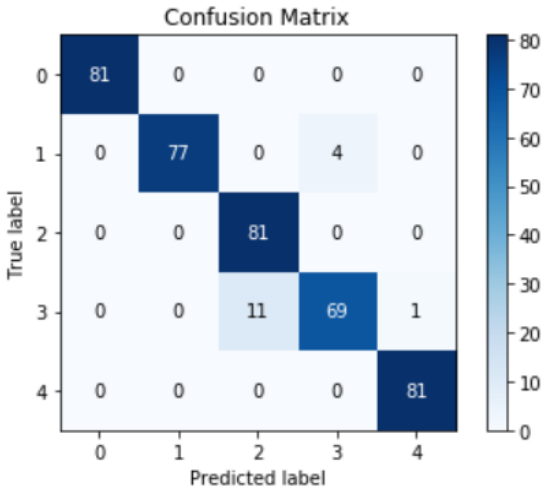

Fig 5.1.a: number of timesteps = 14

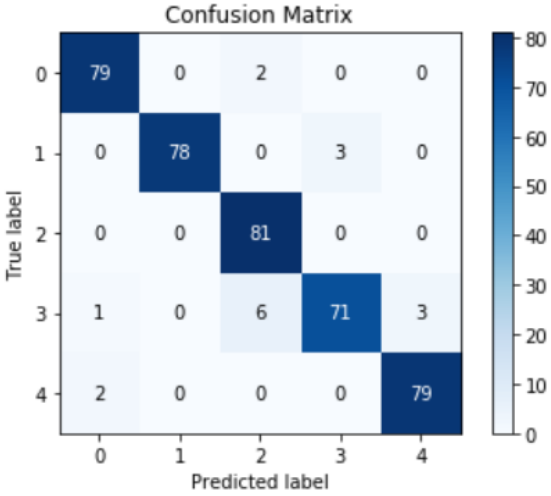

Fig 5.1.b: number of timesteps = 15

Figure 5.1: Confusion matrices obtained for recurrent neural network models

In these matrices, the labels 0,1,2,3 and 4 correspond to the arm positions left, up, front down, and center respectively. These are the five arm movements that were performed.

CHAPTER 6

Result Analysis

The data being collected is a time series data. By training the data with Support Vector Machine, Decision tree classifier, XG boost, random forest and neural network models, poor accuracies were obtained. These models did not work well for the time series sensor data. These multiclass classifiers are generally used for the classification of independent data.

The most suitable machine learning model for time series data is Recurrent Neural Network. Upon using this model for training the data, better accuracies were obtained. As the input data increased, the accuracy also increased. The accuracy was further improved to 97.53% by setting the shuffle parameter to false while sending the input data. This was done because the input data is in a sequential manner and is not to be shuffled. Thus, the best model for this project is RNN.

To improve the results further, more data is to be collected from different participants. This will help in the strengthening of the model. Better methods of filtering and preprocessing can be explored to improve performance. One can also try different combinations of units and nodes in the different layers of the RNN to obtain a better model.

CHAPTER 7

Conclusion

This project demonstrated a new method to track a sequence of arm movements of a person. The proposed method uses the acceleration and angular velocity data provided by the sensors present on the smartwatch as the input. The data can be used in two different ways to predict the movement using a pre-trained RNN model. The performance of the model is affected by the number of input samples, frequency of the input, variety in the data sample, number of movements and the filtering methods.

As smartwatches have become more accessible and relatively less expensive, the proposed method of an activity or exercise (a sequence of arm movements) tracking is convenient. This method can be easily implemented and used by others as it only involves the installation of an Android app on the existing smartwatches.

This project can be further developed to include more arm movements. It can also involve the monitoring of the entire body which will allow the monitoring of yoga or exercise routines. In the future, instantaneous feedback of the yoga or exercise routines from professionals can be incorporated.

References

1. Y. Lee and M. Song, "Using a Smartwatch to Detect Stereotyped Movements in Children With Developmental Disabilities," in *IEEE Access*, vol. 5, pp. 5506-5514, 2017.
2. Shi L., Wang Y., Li J. (2010) “A Real Time Vision-Based Hand Gestures Recognition System”. In: Cai Z., Hu C., Kang Z., Liu Y. (eds) Advances in Computation and Intelligence. ISICA 2010. Lecture Notes in Computer Science, vol 6382. Springer, Berlin, Heidelberg.
3. Chen, F., Deng, J., Pang, Z., Baghaei Nejad, M., Yang, H., Yang, G. “Finger Angle-Based Hand Gesture Recognition for Smart Infrastructure Using Wearable Wrist-Worn Camera” *Appl. Sci.* 2018, *8*, 369.
4. Moschetti, A., Fiorini, L., Esposito, D., Dario, P., Cavallo, F. “Recognition of Daily Gestures with Wearable Inertial Rings and Bracelets”. *Sensors* 2016, *16*, 1341.
5. A. Moschetti, L. Fiorini, D. Esposito, P. Dario and F. Cavallo, "Toward an Unsupervised Approach for Daily Gesture Recognition in Assisted Living Applications," in *IEEE Sensors Journal*, vol. 17, no. 24, pp. 8395-8403, 15 Dec.15, 2017.
6. M. E. Benalcázar, Cristhian Motoche, Jonathan A. Zea, Andrés G.Jaramillo, Carlos E. Anchundia, Patricio Zambrano, Marco Segura, Freddy Benalcázar Palacios, María Pérez, "Real-time hand gesture recognition using the Myo armband and muscle activity detection," *2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM)*, Salinas, 2017, pp. 1-6.
7. Mortazavi, B., Nemati, E., VanderWall, K., Flores-Rodriguez, H.G., Cai, J.Y.J., Lucier, J., Naeim, A., Sarrafzadeh, M., “Can Smartwatches Replace Smartphones for Posture Tracking?”, *Sensors* 2015, *15*, 26783-26800.
8. G. M. Weiss, J. L. Timko, C. M. Gallagher, K. Yoneda and A. J. Schreiber, "Smartwatch-based activity recognition: A machine learning approach," *2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*, Las Vegas, NV, 2016, pp. 426-429.

9. S. Shen, “Arm posture tracking using a smartwatch,” in Proceedings of MobiSys 2016 PhD Forum. ACM, 2016, pp. 9–10.
10. S. Chen, H. Wang and C. Romit Roy, "I am a Smartwatch and I can Track my User's Arm", ACM MobiSys, 2016.
11. R.R. Drumond, B.A.D. Marques, C.N. Vasconcelos, E. Clua “Peek – an LSTM recurrent network for motion classification from sparse data”. Proceedings of the 13th international joint conference on computer vision, imaging and computer graphics theory and applications – volume 1: GRAPP, SciTePress (2018), pp. 215-222.