Chapter 1

Abstract

Human brain is the Central Processing unit of our Body. Interpreting the Brain Signals has been the Subject of interest for Many decades. The Main problem problem We face in achieving this task is the Complexity of the equipments and also the cost. One of the most common equipments a person is carrying is a Smartphone, which is capable of doing many powerful things if it's full potential is utilized. Also another thing is our body's movements. They are a reflection of our brain's state. So assuming this Hypothesis We can predict the Brain waves by recording the body movements. Hence Here we present a solution that addresses the above problem in a very ergonomic and Cost effective way. Our solution is to get the body's movement Data using Smartphone sensors like Accelerometer for a particular part of the body. By recording and analysis of Accelerometers Data along with corresponding Brain waves of a particular Activity, we are creating a mapping from sensor reading domain to Brain Waves Domain. So We first we tried to prove that "Activity 's Sensors reading are actually related to Brain Waves". To achieve this we recorded some common Activities like Walking, jogging and running 's Data for Acceleration of body parts and Brain Waves (EEG) signals. We also gathered data for different body and tried getting the best possible position to keep the accelerometer sensors. Then We trained a Random Forest classifier for classifying the the activities based on Only Accelerometer Readings, then we classified the activity based only on Brain Waves signals and then We classified the activity Using both the reading. We got good classification results showing that Body's accelerometer Data is indeed related to Brain Waves.

Chapter 2

Objective

Our Objective is to make a system that uses Smartphone's sensors readings and predict the Brain waves corresponding to various activities, We can also use Smartphone’s Other sensors like Gyroscope and Magnetometer for more accurate results but not every smartphone contain those sensors, So we limited our sensor span to Accelerometer when predicting while we did use other sensor while activity classification. Apart from predicting the brain waves from accelerometer we also tried to get the best possible body part for placing the sensor and getting the sensor values. For prediction we used various machine learning algorithms like Random forest classifier, Linear regression and random forest regression.

Chapter 3

System Architecture

The system Architecture consist of following:

3.1 Android Application:

Details: The android application is the a very crucial part of system. It is responsible for interacting with android OS. It takes required permissions from user for gathering the sensors data from system. Android app gathers the available sensors data records them locally in a file, is responsible for making websocket connection to server for real time data transfer and also acts as relay node between Mind wave and server. It gathers the raw data from the Mindwave parses it to different variables containing various waves like alpha, beta , delta and theta. It then modifies the data in a suitable manner so as the transmission and further processing of the data can be carried out on server side in easy manner. We are sending the data by appending it in a String (java.lang.String) data structure. Then this String is sent as Raw Message string by android app. For making connections and other networking related task we are using a java library jar file provided by autobahn for websocket. It can be included by importing the jar as a library which contains all the functions necessary for implementing the networking task such as connecting to a given IP address and port (9000 in our case), sending message String, receiving data, and maintaining persistent connection between server and client.

Working: The Android app developed identifies the number of sensors available in the smartphone using Sensor Manager and registers a Sensor Event Listener to Sensor

Event, whenever an event happens, it triggers the callback listener which then get the current sensor value from the system's Sensor manager class and convert it into String format and store it in a local variable and pass it as argument to websocket Connection instance's sent message function. The WebSocket Connection object handles the task of connecting, sending, maintaining and closing the connection. Network related operations are done on a different Thread other than Main/GUI thread, as the network related operations could take unknown time delays, if they were implemented on Main thread, it could leave the system unresponsive ultimately OS will shut down the app unexpectedly. Hence all network related operations are carried out on different threads. The Connection object is made a Static variable so that it remains and be persistent in complete lifecycle of the android application. Making it static also give us another advantage like it can called from any other different class or Activity. However making any object or variable Static is against the Object Oriented Design Pattern and is usually discouraged in computer science world as it has global scope and it is very hard to predict the state of global variable as anything could be modifying it. But In our case the application is comparably small and maintenance of code is not an issue, hence we are slacking the design of application a bit by using Static variable as it helps us making the application easier to build. Other part of android application looks after the communication between MindWave and android. Here also we need a different thread. The MindWave also connects to android by Bluetooth. We used Bluetooth Adapter class of Android for above purpose.

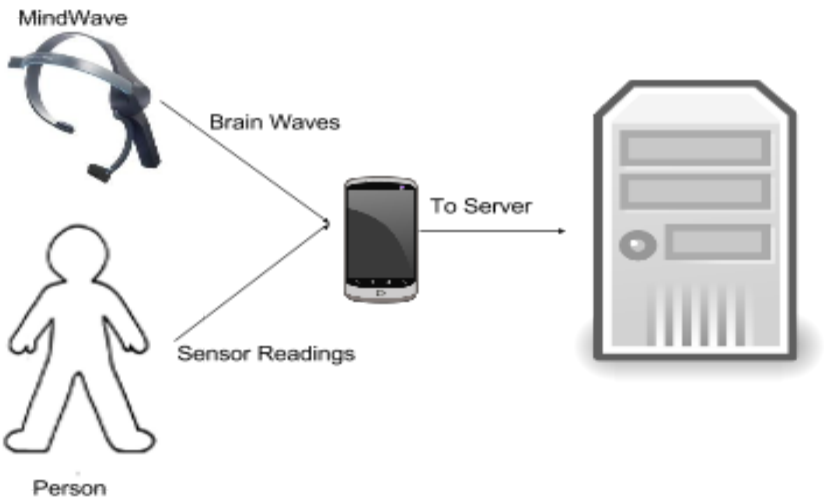

Figure 1

(System Architecture)

3.2 Websocket Server

The Server used is a Websocket based server. A websocket is different from usually used HTTP and HTTPS type of server. Like HTTP and HTTPS , the Websocket is also based on TCP/IP, hence it can also be used in place where HTTP can be implemented without much effort. The websocket protocol is completely backward compatible. Websocket initiates its connection similarly to HTTP connection, as it sends HTTP headers but with some extra information like upgrade header etc. The Server identifies the headers and upgrades the connection to websocket from HTTP by breaking the normal HTTP connection and reconnecting on the same TCP/IP . This process is known as Websocket handshake. Although the websockets uses the default ports like 80

and 443 same as HTTP and HTTPS but anyone can choose a different one if he wishes to. We are using port 9000 in our project in order to avoid collision with any other system services.

Request Headers from Clients:

```
GET ws://echo.websocket.org/?encoding=text HTTP/1.1
Origin: http://websocket.org
Connection: Upgrade
Host: echo.websocket.org
Sec-WebSocket-Key: uRovscZjNol/umbTt5uKmw==
Upgrade: websocket
Sec-WebSocket-Version: 13
```

Reply from server:

```
HTTP/1.1 101 WebSocket Protocol Handshake
Date: Fri, 10 Feb 2012 17:38:18 GMT
Connection: Upgrade
Upgrade: WebSocket
Access-Control-Allow-Origin: http://websocket.org
Access-Control-Allow-Credentials: true
Sec-WebSocket-Accept: rLHCkw/SKsO9GAH/ZSFhBATDKrU=
Access-Control-Allow-Headers: content-type
```

One of the many advantages of using websocket is that it consumes very less bandwidth compared to same data in HTTP. Websocket reduces the kilobytes of HTTP data to some bytes. It also reduces the latency of the transfer of data from 150 milliseconds to 50 milliseconds. Websockets are most useful when you want to get the real time update on any website. Suppose you want to get latest stock data then one solution is to refresh the pages continuously or other method is to poll the server again and again. Both of the

above has their own downsides like refreshing the webpage is not too much user friendly approach, while polling the server is not good either due to latency. Hence in such scenarios websockets are the best options. Once the handshake is done and connection is upgraded to websocket from HTTP the server can push data as soon it changes without client intervention forever.

With polling the client makes requests at regular interval to server and get the response. The problem with this strategy is that the real world is not predictable as it will only work if the changes are periodic. Also if there is no information available there will be many unnecessary opening and closing connection which will add load on the network. A more advance version of polling is long polling. In long polling the server keeps alive the connection for a long time and if there is any change within that interval it will notify the client or ese will terminate the connection. It is observed that when the messages volume is very high long polling does not provide any substantial performance gain over simple polling.

So using the Simple HTTP either by polling or long polling included many unnecessary headers which also introduced latency in the system along with data overhead. As we know HTTP is Half Duplex so as to simulate the full duplex connectivity generally two connections are used. One for upstream and another for downstream. This extra connection increases all the overhead to twice and also introduces extra overheads for mutual syncing of information for both the connection over burdening the server CPU and adding a lot of complexity. The graph below shows how the websocket compares to HTTP connection for full duplex , real time information exchange. As we can see

websockets performs much better than HTTP. hence we will be using websockets in our application.

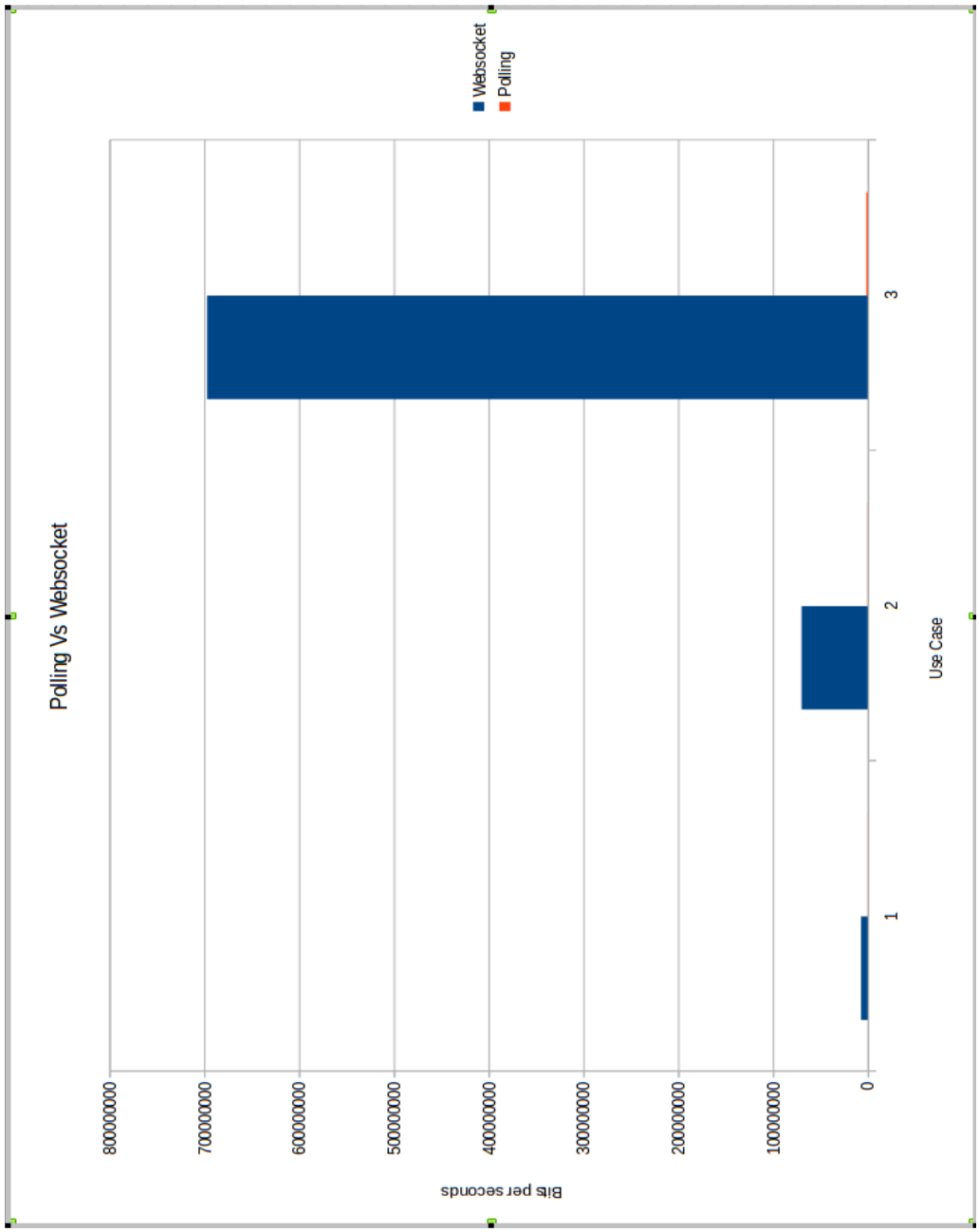

Figure 3

Working: The android application connect to this WS server and gives the data in well structured manner. The data fetched is stored in a file in the system. Then a server script written in python fetches the data from the file and applies in the machine learning algorithm used for classification or regression and gives the result. The result could be shown either on server machine itself or could be send to some other place or can be sent to the source also.

3.3 MindWave:

Mind Wave is a EEG measuring tools manufactured by NeuroSky. It has Bluetooth capabilities in it and can pair up with other devices like android and Laptop or any other device having bluetooth. For using Mind Wave we wear it on our head in such a way that sensor is in the front on forehead. The neurons in our brain when fires emits electrical signals. The patterns and frequency of can be measured by placing a sensor on the scalp. The mind Tools line of product contains a Neurosky's ThinkGear technology which measures the analog electrical signals commonly referred to as brain waves and then processed to give the digital signals. The ThinkGear technology then makes it available for further use. The table given below shows some general well known frequency that reflects the state of a person.

Brain Wave	Frequency Range	Mental state and condition
Delta	0.1 to 3 Hz	Deep, Dreamless sleep, unconscious
Theta	4 to 7 Hz	Intuitive, creative,recall, imagination
Alpha	8 to 12 Hz	Relaxed(Not Drowsy),conscious
Low Beta	12 to 15 Hz	Formerly SMR, relaxed yet focussed
Mid Range Beta	16 to 20 Hz	Thinking, Aware of surrounding
HIgh Beta	21 to 30 Hz	Alertness, Agitation

Table 1

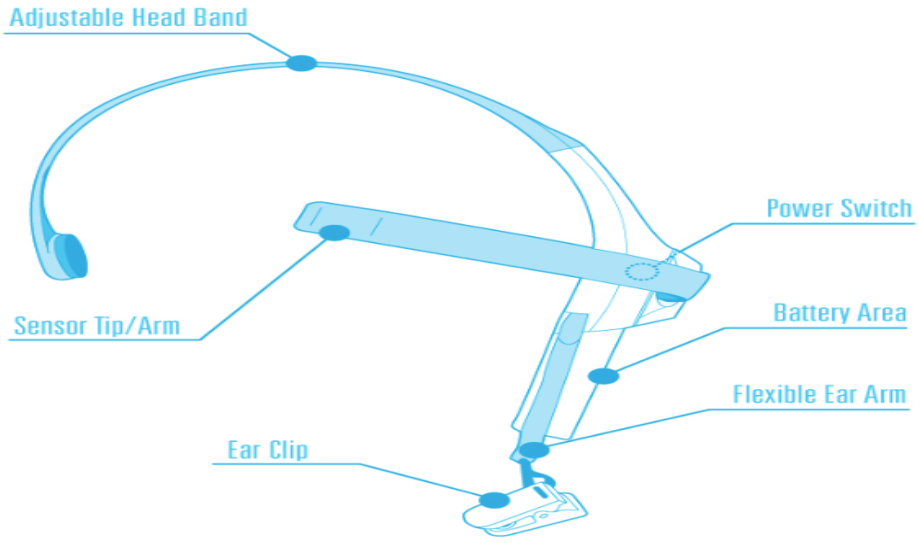

Figure 4

(MIndWave Headset)

Chapter 4

Data Gathering

We are collecting mainly two types of Data namely EEG Waves and second is Accelerometer Data. For accelerometer X,Y,Z components we made our app which collects the Sensor Data and send to our server if connected to internet or else will save to a local file system. The second Data we are using is EEG Waves, for which we are using NeuroSky's MindWave which gives alpha, beta, gamma and theta waves. The MindWave connects to our Android app by Bluetooth and logs the EEG data along with corresponding acceleration reading and both things are being maintained by a single Device there is a very less chance of desynchronization. The Data is then send to server as a string where it gets stored and further processing is done in cloud. We are using Autobahn ‘s Websocket server written in Python for communication. We also tried other techniques like using IMU sensors to collect the data but due to ease of access to android smartphone sensors and considering their availability. We are not using usual HTTP GET/POST methods for uploading the data to server rather we used Websockets. The good point about websocket is, it creates a bidirectional full duplex connection based on **TCP.** Currently we are not using bidirectional communication feature and could have used just the basic http methods but keeping in mind the future architecture about the server being able to push information asynchronously to clients in case of some event may come in handy. So we propose using websockets instead of HTTP.

4.1 Data Set:

After the Android sends the data packet to server which contains the information about the accelerometer reading as well as Brain waves readings from Mind Wave.The data gets stored locally on the server machine in CSV format from where it can be fetched later by operating system while we carried out the analysis. As we are using two completely different instruments for gathering data for different variables like android smartphone for the accelerometer readings and Mind wave for the brain reading giving alpha,beta theta and delta waves. Due to difference The sampling rate of both the waves is different with accelerometer rate be higher(approx 17 times) than mind wave module. Hence to account for this gap we tried several techniques like filling missing values with zeroes, Imputation and buffering the previous reading till a new reading comes in. We also tried reducing the accelerometer data to match the brain waves data by getting the median of variable length sliding window. The window length is decided by the next incoming brain wave reading. All four methods described above were tried at different places as per the need. The training and testing data were then extracted from this data set collected by us by using a randomly splitting function of python which splits the data in two parts in a given ratio.

Chapter 5

Workflow

In order to predict the various brain waves corresponding to activities by means of acceleration values. We first tried to find if there is actually a relation between accelerometer values and the activity we are doing. Then similarly following the above reasoning, we also carried out the experimentation to find the the relation between Brain waves and the activity we are doing. A way to identify this is by trying to classifying the given activity based on input features like brain waves or accelerometer values. The classifier we used is Random Forest classifier for above purposes.

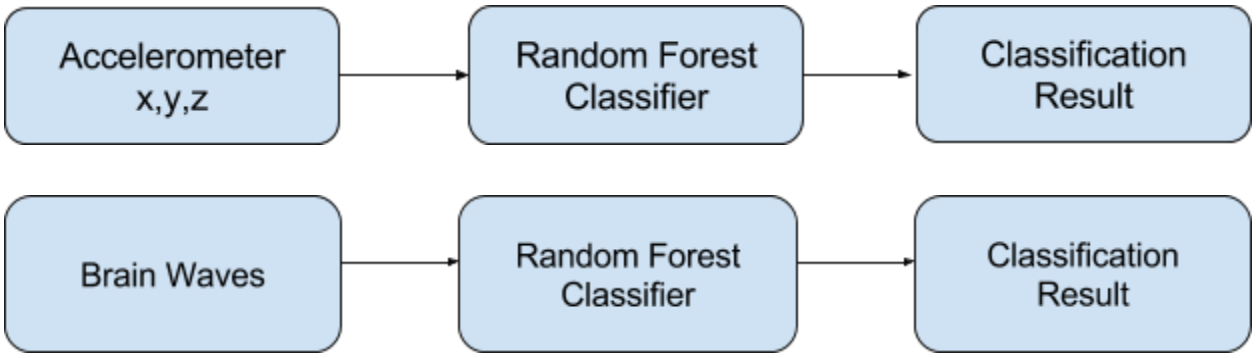

Figure 5

Classifier Flow Diagram For Activities

After prediction from both of the above methods we analysed the error between the actual activity and the activity that is being predicted by classification method. The results are summarised ahead. After the we were convinced about that there is a relation about accelerometer and Brain waves with activity, We tried to predict the brain waves from the

accelerometer readings. For this prediction we have tried various Regression techniques like Linear Regression, Random Forest Regression.

Chapter 6

Machine Learning Algorithms

6.1 Linear Regression:

In Linear Regression we try to model a linear relationship between variables. One variable(also known as explanatory variable) is predicted on the basis of other variables (also known as dependent variables). A linear regression line has an equation of the form $Y=a+bX$, where X is the explanatory variable and Y is the dependent variable. The slope of the line is b, and a is the intercept (the value of y when $x = 0$). The Most common method to fit a line is to minimise sum of least square errors for individual observation points (vertical deviations from each data point to the line). Linear Regression consists of 3 stages – (1) Correlations and Directional analysis of Data, (2) Making of the model, (3) evaluating the Model.

6.2 Random Forest Regression and Classification:

Random forest is a way of combining the results of many decision trees so as to eliminate the the overfitting on a single tree. Each of these trees generates a classification for a given set of attributes. In other words, random forests are an ensemble learning method for classification and regression that operate by constructing a lot of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. However Random forest comes at

the expense of a some loss of interpretability, but generally greatly boosts the performance of the final model.

Chapter 7

Deciding Best position of Sensor

For deciding the best position of sensor while keeping in mind the ergonomics of user using the system. We decided to use the UCI machine learning repository (Center for Machine Learning and Intelligent Systems) ‘s Realistic sensor displacement benchmark dataset.Dataset consists of various sensor data like Magnetometer, Accelerometer and Gyroscope for various positions on body like Left Upper Arm, Right Upper Arm, Left Thigh, Right Thigh, Left Lower Arm, Right Lower arm, left Calf and Right Calf and Back. Each sensor measures the X, Y and Z component of their measurement unit Vectors. All these data set is available for activities like walk, jog, run, jump and cycle. We used this data set as training data set for our Random Forest tree. for activities mentioned above. From this data set we randomly splitted the dataset into two parts. One for training and other for testing purposes. After splitting we applied Random Forest Classifier for classification. The position which gives best classification can be said as Best position for gathering accelerometer and other sensor’s data. The results of this experiments are attached at the end of report in Appendix.*

7.1 Conclusion for Best Position:

If we take averages of diagonal indices, then compare for each body part then we get

For Sensor: Accelerometer, Magnetometer and Gyroscope

Body Part→ Average of diagonal(correctly classified Instances)

Thigh → 3207

Lower Arm→ 3243

Upper Arm→ 3208

Calf→ 3217

For Sensor: Accelerometer, Magnetometer

Body Part→ Average of diagonal(correctly classified Instances).

Thigh→ 3039

Lower Arm→ 3065

Upper Arm→ 2999

Calf→ 3217

As we can see while using only 2 sensors Calf is the most optimum position and when we use 3 sensors Lower Arm is good place. Still in case of 3 sensors keeping the smartphone near calf is not practical so Lower Arm is preferred in this case too. Ergonomically placing sensor on calf is not so suitable. Hence we preferred the Arm position.

Chapter 8

Classification of Activities

8.1 Classification of Activities based on Acceleration values:

Here we are considering only 3 basic activities i.e Rest, Walk, Jog to be classified. We used the full data set and did not do any data reduction. As we can see from the confusion matrix mentioned below the classifier is very good at detecting the activity between Rest and jog. But detection between walk and jog is not very clear. It may be due to the fact that walking's reading pattern resembles a little the pattern of jogging hence there are some false detection as 8 and 17 between jogging and walking. But this difference is not very big as can be seen and can be neglected for the our context.

Rest	437	0	0
Walk	1	697	8
Jog	0	17	401
	Rest	Walk	Jog

Confusion Matrix Corresponding to Random forest Classifier

Some Definitions:

1. **precision** is the fraction of retrieved instances that are relevant.
2. **Recall** is the is the fraction of relevant instances that are retrieved.
3. **f1-score** is a measure of a test's accuracy. It considers both the precision *p* and the recall *r* of the test to compute the score: *p* is the number of correct positive results divided by the number of all positive results, and *r* is the number of correct positive results divided by the number of positive results that should have been returned.The F_1 score can be interpreted as a weighted average of the precision and recall, where an F_1 score reaches its best value at 1 and worst at 0.

Hence looking at the Classification report given below, the precision for all the 3 classes is very close to 1.00 also the final score resulting from all 3 is approximately is 0.98 which is quite good.

	precision	recall	f1-score	support
Rest	1.00	1.00	1.00	437
Walk	0.98	0.99	0.98	706
Jog	0.98	0.96	0.97	418
avg / total	0.98	0.98	0.98	1561

Classification report

Table 2

8.2 Classification of Activities based on EEG waves:

We wanted to see if there is any relation between activity and brain wave So we did activity classification. We did imputation of brain wave data by buffering till the next value arrives from the android. Similar to earlier case here also we have considered the 3 activities namely Rest,Walk and Jog. The results are shown below.

Rest	459	0	0
Walk	0	722	0
Jog	0	0	380
	Rest	Walk	Jog

Confusion Matrix Corresponding to Random forest Classifier

	precision	recall	f1-score	support
Rest	1.00	1.00	1.00	459
Walk	1.00	1.00	1.00	722
Jog	1.00	1.00	1.00	380
avg / total	1.00	1.00	1.00	1561

Classification report

Table 3

8.3 Classification of Activity based on Combined Data:

We also classified the activities using combined data from both the sources and the results are given below.

Rest	453	0	0
Walk	0	711	2
Jog	0	7	388
	Rest	Walk	Jog

Confusion Matrix Corresponding to Random forest Classifier

	precision	recall	f1-score	support
Rest	1.00	1.00	1.00	453
Walk	0.99	1.00	0.99	713
Jog	0.99	0.98	0.99	395
avg / total	0.99	0.99	0.99	1561

Classification report

Table 4

Chapter 9

Results

9.1 Prediction of Brain waves by Linear Regressor:

For predicting the Brain waves we first tried the Linear regression. We kept the x,y,z values of accelerometers as input and individual brain waves as output. The results are given below are using the buffering technique for imputation till the next value comes in.

Wave Number	Regression coefficients	Mean square error	variance score
wave 1	[-0.01, 0.02, -0.06]	0.07	0.02
wave 2	[-0.06, 0.07, -0.11]	0.07	-0.02
wave 3	[0.04, -0.02, -0.06]	0.07	0.00
wave 4	[-0.05, 0.05, -0.1]	0.07	0.01
wave 5	[-0.12, 0.02, 0.36]	0.16	-1.36
wave 6	[-0.06, 0.06, -0.08]	0.08	-0.17
wave 7	[0.27, -0.2 , -0.03]	0.15	-1.12

Regression Report

Table 5

Result: As we can see from the above table. The prediction from the Linear regression model is not very good and we should try some other alternative.

9.2 Prediction of EEG waves by Random Forest Regressor :

wave Number	R^2	MSE	RMSE	Corr
Wave 1	-0.1080809579	0.0820995269	0.2703238628	54.63270979
Wave 2	-0.0405429489	0.0531214443	0.230480898	40.88675342
Wave 3	-0.1073935296	0.0899489192	0.2999148533	71.00478523
Wave 4	-0.1078965079	0.0731731202	0.2705053054	55.27558647
Wave 5	0.4994636663	0.0719213795	0.2681816167	210.25449792
Wave 6	-0.0640396226	0.0323558215	0.1798772402	11.75331098
Wave 7	0.0221350613	0.170097772	0.4124291114	350.60898166

Regression Report

Table 6

The above table shows the result of random forest regression without using any type of imputation and rather putting ‘0’ at missing values in brain wave readings. We can see that although MSE and RMSE are relatively small but the value of correlation is also not very good and also r-squared value is also on the negative side. Hence the regression remained non-conclusive.

After this we tried taking the median value of variable length sliding window of accelerometer value readings. The length was decided based on the next value of brain wave arrival which was approximately 17 reading of accelerometer. The results for this is shown below.

wave Number	R^2	MSE	RMSE	Corr
Wave 1	-0.3139607507	0.08818639843	0.29696194	3.65456603
Wave 2	-0.30251828	0.07235039	0.268980294	2.53115003
Wave 3	-0.0693646310	0.0908554425	0.30142236	6.06563749
Wave 4	-0.173653864	0.0787386935	0.280604158	3.76880589
Wave 5	-0.506385239	0.201368730	0.448741273	6.94554264
Wave 6	-0.288882682	0.031474446	0.1774103907	0.40124889
Wave 7	-0.361796469	0.2414196079	0.49134469	21.88141117

Regression Report(Sliding Window)

Table 7

We can see that MSE and RMSE are relatively small but R-squared is also very less and correlation is also not very good. Hence the results are non-conclusive.

Chapter 10

Appendix

Confusion Matrix for determining the Best Position

Sensor: Accelerometer,Magnetometer

Position: Left Calf

walk	4604	51	32	8	30
jog	30	3346	294	18	24
run	27	398	3633	10	6
jump	51	41	19	690	0
cycle	23	38	12	0	3157
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer

Position: Right Calf

walk	4577	54	5	1	58
jog	63	3443	327	33	50
run	22	417	3563	21	5
jump	40	86	35	662	3

cycle	49	20	2	1	3005
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer

Position: Left Lower Arm

walk	4524	53	22	8	12
jog	132	3348	335	25	9
run	14	452	3649	9	9
jump	47	57	41	660	11
cycle	4	25	7	2	3087
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer

Position: Right Lower Arm

walk	4454	107	27	3	38
jog	196	3364	210	4	44
run	37	192	3859	7	28
jump	21	56	40	703	31
cycle	55	40	16	0	3010
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer

Position: Left Thigh

walk	4623	78	2	0	1
jog	214	3291	312	20	13
run	9	330	3679	29	5
jump	20	130	105	612	4
cycle	9	1	0	2	3053
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer

Position: Right Thigh

walk	4473	51	13	2	54
jog	141	3313	309	31	19
run	10	469	3596	18	9
jump	25	104	37	660	8
cycle	69	32	6	0	3093
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer

Position: Left Upper Arm

walk	4484	79	8	3	17
jog	183	3310	304	37	53
run	19	296	3618	19	65
jump	23	113	119	547	33
cycle	57	26	32	10	3087
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer

Position: Right Upper Arm

walk	4507	98	17	1	19
jog	186	3278	297	48	71
run	22	285	3615	40	58
jump	18	128	128	531	34
cycle	40	59	27	7	3028
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Left Calf

walk	4579	10	1	1	0
jog	12	3758	156	3	1
run	4	208	3800	1	0
jump	19	0	2	809	0
cycle	0	0	3	0	3175
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Right Calf

walk	4725	15	0	0	1
jog	38	3651	179	2	2
run	10	217	3682	0	1
jump	8	4	8	815	2
cycle	7	0	1	0	3174
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Left Thigh

walk	4631	51	0	0	0
jog	81	3626	171	0	0
run	7	260	3763	0	0
jump	1	1	5	833	0
cycle	0	0	0	0	3112
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Right Thigh

walk	4547	11	1	0	0
jog	46	3698	156	2	0
run	1	204	3873	0	0
jump	0	3	2	832	0
cycle	3	2	0	0	3161
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Right Lower Arm

walk	4566	17	4	0	0
jog	56	3704	156	4	1
run	2	157	3849	3	1
jump	5	1	0	816	0
cycle	0	9	2	0	3189
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Right Lower Arm

walk	4588	18	2	0	0
jog	46	3768	72	1	0
run	1	80	3900	3	0
jump	1	1	7	812	0
cycle	0	0	0	0	3242
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Left Upper Arm

walk	4671	39	13	1	0
jog	85	3548	196	3	0
run	14	126	3921	0	2
jump	0	0	2	822	0
cycle	0	1	0	0	3098
	walk	jog	run	jump	cycle

Sensor: Accelerometer,Magnetometer, Gyroscope

Position: Right Upper Arm

walk	4569	64	3	0	0
jog	125	3652	146	4	3
run	17	145	3819	2	0
jump	2	8	1	786	0
cycle	0	1	0	0	3195
	walk	jog	run	jump	cycle

Chapter 11

Future Work To be Done

We have tried to predict brain waves based on accelerometer whereas above said work can also include more sensors like Gyroscope and Magnetometer. More Complex emotional brain waves can also be predicted given enough time and infrastructure.