A MINI PROJECT REPORT

ON

Human Activity Recognition using Smartphone Dataset

Submitted to Mumbai University

In the partial fulfillment of the requirement for the award of the degree of

Bachelor of Engineering

In

COMPUTER ENGINEERING

By

Miss. Ansari Nooras Fatima M.Hashim (16CO01)

Miss. Shaikh Altamas Shakeel (16CO11)

Miss. Ulde Fahmi Nisar (16CO17)

Under the guidance of

Mrs. Apeksha Gopale

Assistant Professor

Department of Computer Engineering

Anjuman-I-Islam's Kalsekar Technical Campus

Affiliated to Mumbai University

KHANDA GOAN, NEW PANVEL, NAVI MUMBAI, MAHARASHTRA

2016-2017

Department of Computer Engineering

Anjuman-I-Islam's Kalsekar Technical Campus

Affiliated to Mumbai University

KHANDA GOAN, NEW PANVEL, NAVI MUMBAI, MAHARASHTRA

2016-2017

DECLARATION BY THE CANDIDATE

Ansari Nooras,Shaikh Altamas,Ulde Fahmi bearing **Roll number: 16CO01, 16CO11**,**16Co17** hereby declare that the mini project report entitled **"Human Activity Recognition using Smartphone Dataset"**, is a record of bonafide work carried out by me and the results embodied in this project have not been reproduced or copied from any source. The results of this project report have not been submitted to any other University or Institute for the award of any other Degree or Diploma.

Miss.Ansari Nooras Fatima (16CO01)

Miss.Shaikh Altamas (16CO11)

Miss.Ulde Fahmi (16CO17)

Department of Computer Engineering

Anjuman-I-Islam's Kalsekar Technical Campus

Affiliated to Mumbai University

KHANDA GOAN, NEW PANVEL, NAVI MUMBAI, MAHARASHTRA

2016-2017

CERTIFICATE

This is to certify that the project report entitled **"Human Activity Recognition using Smartphone Dataset"**, submitted by **Ansari Nooras,Shaikh Altamas,Ulde Fahmi** bearing **Roll. No.:16CO01, 16CO11,16Co17++** in the partial fulfillment of the requirements for the award of the degree of **Bachelor of Computer Engineering** is a record of bonafide work carried out by her for the course **Mini Project CSP605**.

Mini Project Guide

(Prof. Apeksha Gopale)

Mini Project Coordinator

(Prof. Muhammed Salman Shamsi)

Program Owner

(Prof. Tabrez Khan)

INDEX

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The project “Human activity recognition using smartphone dataset” is an android application developed using android studio. Java language is used primarily in this project. [3]The application helps a user to check his/her activity like sitting, standing, walking, jogging, walking downstairs and walking upstairs. Machine learning model such as CNN (convolution neural networks) model is used to predict which activity is performed most of the time.[2]

After identifying an activity, a user is provided with his/her statistics based on their activities. This project provides a statistical pie chart having percentage of performed activities over a period of time. And then he/she is able to analyze which activity is performed more often. [3]This helps them to know what consequences they can face doing a single activity for most of the time.

So, a user easily evaluates the most performed activity and find what changes he/she should inculcate in their daily life to avoid problems related to most performed activity. [1]

The process of recognition is performed using accelerometer sensor. [6]With the help of x-axis, y-axis and z-axis values calculated by accelerometer; a human activity is recognized. [6]Tensorflow, an open source platform or model for machine learning is used in this project to manage data flow (data regarding list of activities performed by a user) and deployment of this android app. [3]

Some of the applications available today are MI fitbit, Hike step counter and Samsung gear.

1.2 SCOPE

The application recognizes human activity and give proper analysis for it. The application identifies sitting, standing, walking, walking upstairs and downstairs and jogging activities using accelerometer sensor. Machine learning model analyzes the activities and show their performed percentage in pie chart.[3]

The project needs any latest smartphone to run.[3] Using the application, a user is able to identify the most performed activity on the time period basis. The application is limited to only check and verify the activities. It is only a prototype model consisting of basic functions to recognize.

1.3 PROBLEM STATEMENT

Problem statement:

To create a system that will recognize human activity and using machine learning analyze which activity is performed most.[1]

Problem description:

In the world of technology today, there is requirement of interaction between human and technological devices to create effective systems. We know human can interact with computers by giving some input from keyboard, but this is not only the way and this is related to only computers. How can a machine know what humans are performing. whether they are sitting, standing or any movement? [1]

When someone visit doctors for their body pain, they give full information about what they did the whole day but not statistically perfect. [1]Imagine a company having many employees working, how can a boss know which employee is continuously working and which not? Yes, a camera can help but the boss or any other person cannot continuously sit for the whole working day. Even if they sit, this practice is not so effective.

So, there is a need of a system which can recognize human activity. Such a system will serve a great help to more advanced system.

CHAPTER 2

SYSTEM SPECIFICATION

2.1 SYSTEM REQUIREMENT

2.1.1 Hardware Requirements:

1. 4 GB RAM
2. 258.65 MB disk space

2.1.2 Software Requirements:

1. Microsoft Windows XP, Ubuntu 14.0 LTS or Linux
2. Android Studio
3. Spyder to run and check machine learning model in python

2.2 SYSTEM FEATURES

In this project, following list provides the features to be implemented:

- Start detection process
- Detect a human activity
- Display accelerometer values
- Store data into room database
- Use a machine learning model (CNN)
- Display a pie chart
- Stop detection process

Start detection process: Starts the detection process on current timestamp.

Detect a human activity: Detects a human activity by an accelerometer sensor used in this android app.

Display accelerometer values: Displays accelerometer values of all the six activities in a table.

Store data into room database: Store data regarding list of activities performed by user in room database of android.

Use a machine learning model (CNN): Uses convolution neural networks algorithm which is provided with data in room database.

Display a pie chart: Displays a single pie chart of different activities.

Stop detection process: Stops the detection process.

CHAPTER 3

SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE

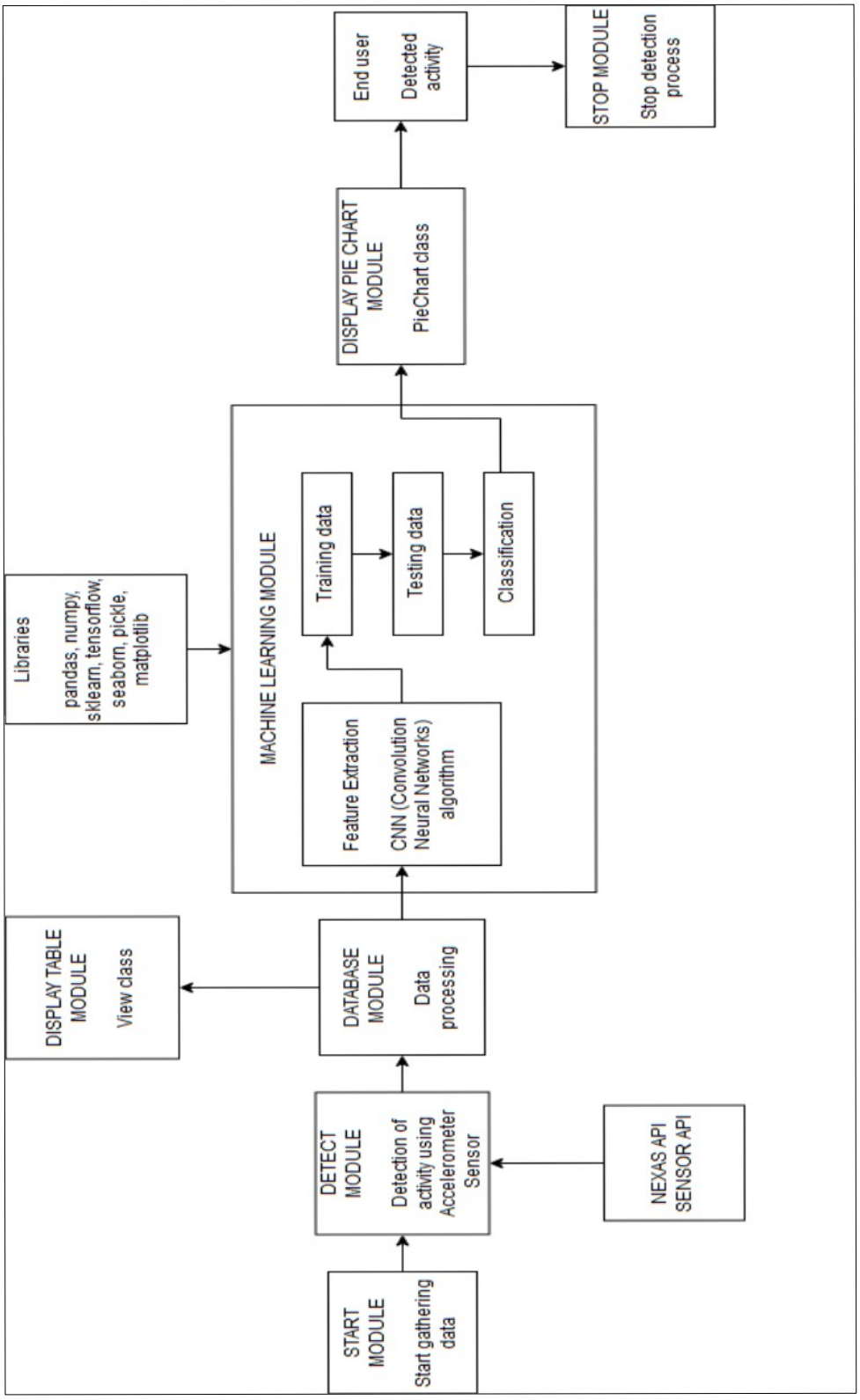

Fig 3.1.1 System Architecture of Human Activity Recognition using smartphone Data set

3.2 MODULES IN THE SYSTEM

Instead of user, app developer has full permission to access the project code. The project has different modules r functions which perform their task an uniting them creates our whole system. This project has following modules:

- Start module
- Detect module
- Display table module
- Database module
- Machine learning module
- Display pie chart module
- Stop module

The names of the modules are given as per the functions they perform.

Start module start the process of activity recognition on current timestamp (even in background).
Detect module simply detect the human activity performed by user. An accelerometer sensor is used in this module.
Display table module start displaying accelerometer values of all six activities which gets changed as orientation of mobile gets changed.
Database module store accelerometer values related to human activity performed at a particular instance. Those values are stored in room database.
Machine learning module use a cnn model to recognize most performed activity of user.
Display pie chart module display a single pie chart consisting of percentages of each activity performed by user thereby notifying user of its most performed activity.
Stop module stop the process of activity recognition that means app stops gathering data from accelerometer.

3.3 USE CASE DIAGRAM

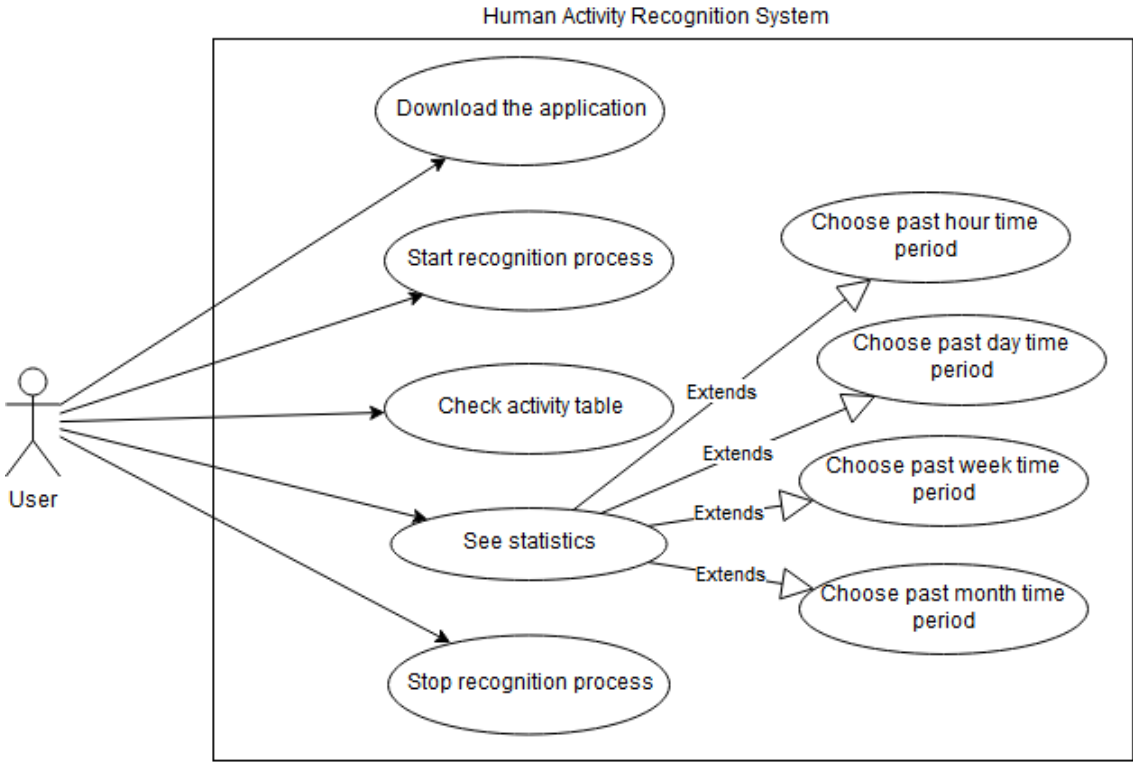

Fig 3.3.1 Use Case Diagram

3.4 ACTIVITY DIAGRAM

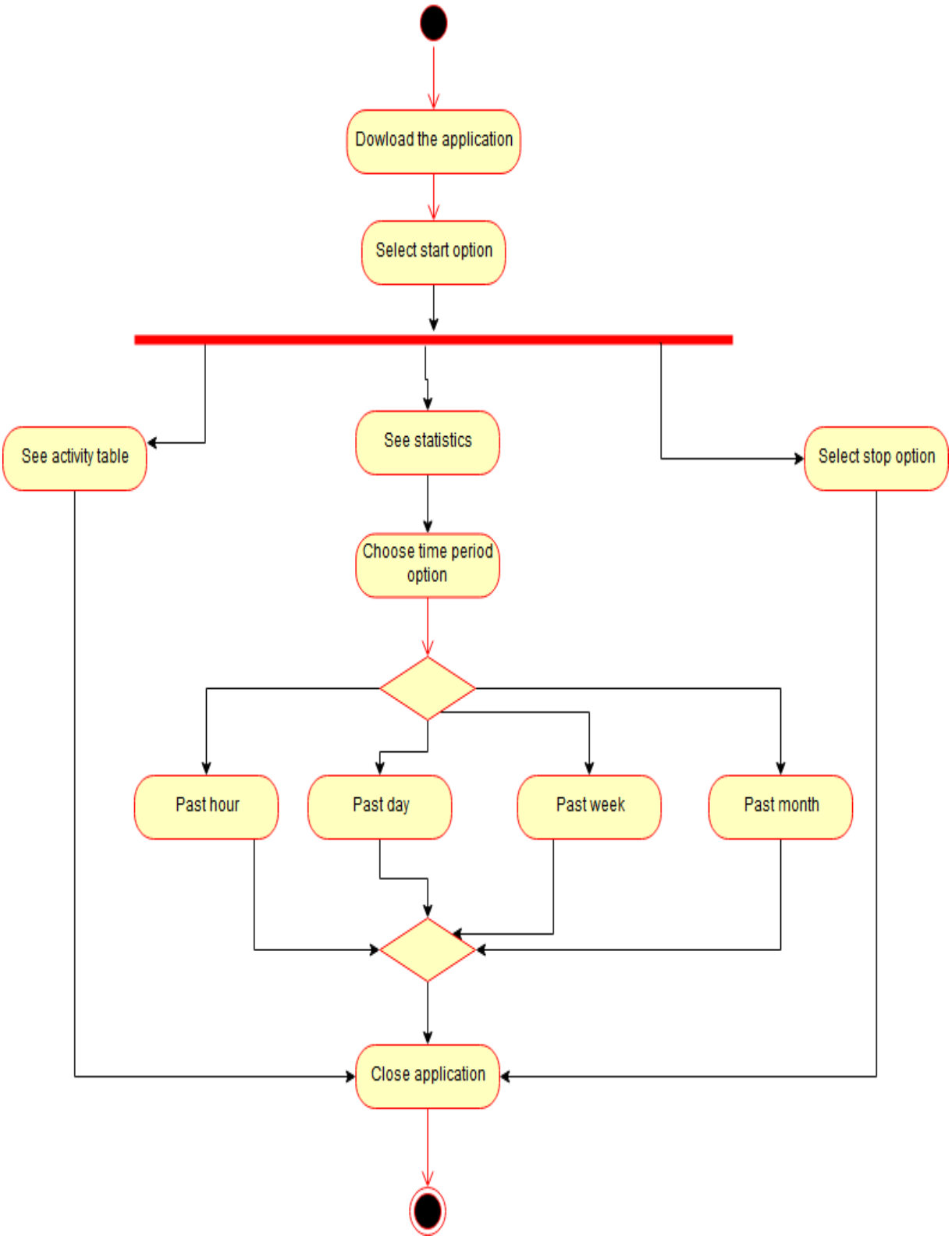

Fig 3.4.1 Activity diagram

CHAPTER 4

IMPLEMENTATION

4.1 CODE SNIPPETS

Activity.java

```
package com.example.har.h;
import android.arch.persistence.room.ColumnInfo;
import android.arch.persistence.room.Entity;
import android.arch.persistence.room.PrimaryKey;
@Entity(tableName = Constants.TABLE_NAME)
public class Activity {
  @PrimaryKey(autoGenerate = true)
  int uid;
  @ColumnInfo(name = Constants.COL1_NAME)
  long startTimestamp;
  @ColumnInfo(name = Constants.COL2_NAME)
  String activityType;
  @ColumnInfo(name = Constants.COL3_NAME)
  int duration;
  public Activity(long startTimestamp, String activityType, int duration) {
    this.startTimestamp = startTimestamp;
    this.activityType = activityType;
    this.duration = duration;
  }
  public int getUid() { return uid; }
  public long getStartTimestamp() { return startTimestamp; }
  public String getActivityType() { return activityType; }
     public int getDuration() { return duration;}
  public void setDuration(int duration) { this.duration = duration;}
}
```

ActivityDao.java

```
package com.example.har.h;
import android.arch.persistence.room.Dao;
import android.arch.persistence.room.Insert;
import android.arch.persistence.room.Query;
import java.util.List;
@Dao
public interface ActivityDao {
  @Query("SELECT min(start_timestamp) as start_timestamp, activity_type, sum(duration)
as duration FROM activity " +
      "where start_timestamp >= :start_timestamp " +
      "GROUP BY activity_type")
  List<Activity> getProportion(long start_timestamp);
  @Query("SELECT COUNT(*) from activity")
  int countEntries();
  @Insert
  void insert(Activity activity);
}
```

AppDatabase.java

```
package com.example.har.h;
import android.arch.persistence.room.Database;
import android.arch.persistence.room.Room;
import android.arch.persistence.room.RoomDatabase;
import android.content.Context;
@Database(entities = {Activity.class}, version = 1)
public abstract class AppDatabase extends RoomDatabase {
  private static AppDatabase INSTANCE;
  public abstract ActivityDao activityDao();
  public static AppDatabase getAppDatabase(Context context) {
    if (INSTANCE == null) {
      INSTANCE =
          Room.databaseBuilder(context.getApplicationContext(), AppDatabase.class,
Constants.DB_NAME)
              .allowMainThreadQueries()
              .build();
    }
    return INSTANCE;
  }
  public static void destroyInstance() { INSTANCE = null;}
}
```

Constants.java

```
package com.example.har.h;
import android.graphics.Color;
public class Constants {
  // MainActivity.java
  public static final int N_SAMPLES = 90;
  public static final float MEAN_X = 0.662868f;
  public static final float MEAN_Y = 7.255639f;
  public static final float MEAN_Z = 0.411062f;
  public static final float SD_X = 6.849058f;
  public static final float SD_Y = 6.746204f;
  public static final float SD_Z = 4.754109f;
  public static final String WALKING = "Walking";
  public static final String SITTING = "Sitting";
  public static final String JOGGING = "Jogging";
  public static final String STANDING = "Standing";
  public static final String UPSTAIRS = "Upstairs";
  public static final String DOWNSTAIRS = "Downstairs";
  // Activity.java
  public static final String TABLE_NAME = "activity";
  public static final String COL1_NAME = "start_timestamp";
  public static final String COL2_NAME = "activity_type";
  public static final String COL3_NAME = "duration";
  // AppDatabase.java
  public static final String DB_NAME = "activity_database";
  // DisplayStatsActivity.java
  public static final String EMPTY = "";
  public static final String OPTION_HOUR = "Past Hour";
  public static final String OPTION_DAY = "Past Day";
```

```
  public static final String OPTION_WEEK = "Past Week";
  public static final String OPTION_MONTH = "Past Month";
  public static final long MILLI_TO_SEC = 1000L;
  public static final int SEC_IN_HOUR = 3600;
  public static final int SEC_IN_DAY = 86400;
  public static final int SEC_IN_WEEK = 604800;
  public static final int SEC_IN_MONTH = 2592000;
  public static final int[] JOYFUL_COLORS = {
      Color.rgb(217, 80, 138), Color.rgb(254, 149, 7), Color.rgb(254, 247, 120),
      Color.rgb(106, 167, 134), Color.rgb(53, 194, 209), Color.rgb(42, 109, 130)};
  public static final int TO_PERCENT = 100;
  public static final int DP = 1;
  public static final float TEXT_SIZE = 10f;
  // RecognitionActivity.java
  public static final String LIB_NAME = "tensorflow_inference";
  public static final String MODEL_FILE = "file:///android_asset/har_classifier.pb";
  public static final String INPUT_NODE = "input";
  public static final String[] OUTPUT_NODES = {"output"};
  public static final String OUTPUT_NODE = "output";
  public static final long[] INPUT_SIZE = {1,1,90,3};
  public static final int OUTPUT_SIZE = 6;
}
```

DisplayStatsActivity.java

```
package com.example.har.h;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Spinner;
import android.widget.Toast;
import com.github.mikephil.charting.charts.PieChart;
import com.github.mikephil.charting.components.Legend;
import com.github.mikephil.charting.data.Entry;
import com.github.mikephil.charting.data.PieData;
import com.github.mikephil.charting.data.PieDataSet;
import com.github.mikephil.charting.formatter.PercentFormatter;
import com.github.mikephil.charting.highlight.Highlight;
import com.github.mikephil.charting.listener.OnChartValueSelectedListener;
import java.lang.ref.WeakReference;
import java.math.BigDecimal;
import java.util.ArrayList;
import java.util.List;
public class DisplayStatsActivity extends AppCompatActivity implements OnChartValueSelectedListener {
  private AppDatabase activityDB;
  private String[] items = new String[] {Constants.OPTION_HOUR, Constants.OPTION_DAY,
Constants.OPTION_WEEK, Constants.OPTION_MONTH};
  private String [] xVal = new String[] {Constants.EMPTY, Constants.EMPTY,
Constants.EMPTY, Constants.EMPTY, Constants.EMPTY, Constants.EMPTY};
```

```
  @Override
  protected void onCreate(Bundle savedInstanceState){
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_stats);
    final Spinner dropdown = findViewById(R.id.spinner1);
    ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
android.R.layout.simple_spinner_dropdown_item, items);
    dropdown.setAdapter(adapter);
    dropdown.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
      @Override
      public void onItemSelected(AdapterView<?> parentView, View selectedItemView, int
position, long id) {
        String choice = dropdown.getSelectedItem().toString();
        long initialTimestamp = getInitialTimestamp(choice);
        getStats(initialTimestamp);
        Log.d("Spinner", choice);
      }
      @Override
      public void onNothingSelected(AdapterView<?> parentView) {
      }
    });
  }
  private long getInitialTimestamp(String option) {
    long currentTime = System.currentTimeMillis() / Constants.MILLI_TO_SEC;
    long initialTime;
    switch (option) {
      case Constants.OPTION_HOUR:
        initialTime = currentTime - Constants.SEC_IN_HOUR;
        break;
      case Constants.OPTION_DAY:
        initialTime = currentTime - Constants.SEC_IN_DAY;
        break;
      case Constants.OPTION_WEEK:
        initialTime = currentTime - Constants.SEC_IN_WEEK;
        break;
      case Constants.OPTION_MONTH:
        initialTime = currentTime - Constants.SEC_IN_MONTH;
        break;
      default:
        initialTime = currentTime - Constants.SEC_IN_DAY;
        break;
    }
    return initialTime;
  }
  private void getStats(long time) {
    activityDB = AppDatabase.getAppDatabase(DisplayStatsActivity.this);
    new retrieveStats(this, time).execute();
  }
  private void drawPieChart(List<Activity> activities) {
    PieChart pieChart = (PieChart) findViewById(R.id.piechart);
```

```
    pieChart.setUsePercentValues(true);
    ArrayList<Entry> yvalues = new ArrayList<Entry>();
    float total = 0.0f;
    for (int i=0; i<activities.size(); i++) {
      total += activities.get(i).duration;
    }
    for (int i=0; i < activities.size(); i++) {
      Activity activity = activities.get(i);
      float percentage = round(activity.duration/total*Constants.TO_PERCENT,
Constants.DP);
      yvalues.add(new Entry(percentage, i));
      xVal[i] = activity.activityType;
    }
    PieDataSet dataSet = new PieDataSet(yvalues, Constants.EMPTY);
    PieData data = new PieData(xVal, dataSet);
    data.setValueTextSize(Constants.TEXT_SIZE);
    data.setValueFormatter(new PercentFormatter());
    dataSet.setColors(Constants.JOYFUL_COLORS);
    pieChart.setDescription(Constants.EMPTY);
    Legend legend = pieChart.getLegend();
    legend.setPosition(Legend.LegendPosition.ABOVE_CHART_CENTER);
    pieChart.setData(data);
    pieChart.invalidate();
    pieChart.setOnChartValueSelectedListener(this);
  }
  public static float round(float d, int decimalPlace) {
    BigDecimal bd = new BigDecimal(Float.toString(d));
    bd = bd.setScale(decimalPlace, BigDecimal.ROUND_HALF_UP);
    return bd.floatValue();
  }
  @Override
  public void onValueSelected(Entry e, int dataSetIndex, Highlight h) {
    if (e == null)
      return;
    Toast.makeText(DisplayStatsActivity.this,
        xVal[e.getXIndex()] + ": " + e.getVal() + "%", Toast.LENGTH_SHORT).show();
    Log.i("VAL SELECTED",
        "Value: " + e.getVal() + ", xIndex: " + e.getXIndex()
            + ", DataSet index: " + dataSetIndex);
  }
  @Override
  public void onNothingSelected() {
    Log.i("PieChart", "nothing selected");
  }
  private class retrieveStats extends AsyncTask<Void,Void,List<Activity>> {
    private WeakReference<DisplayStatsActivity> activityReference;
    long startTime;
    retrieveStats(DisplayStatsActivity context, long time) {
      activityReference = new WeakReference<>(context);
      this.startTime = time;
```

```
    }
    @Override protected List<Activity> doInBackground(Void... voids) {
      return activityReference.get().activityDB.activityDao().getProportion(startTime);
    }
    @Override protected void onPostExecute(List<Activity> activities) {
      Activity first = activities.get(0);
      String time = Integer.toString(first.getDuration());
      String activity_type = first.getActivityType();
      Log.d("Success", "Pulled from DB " + activity_type);
      Log.d("Success", "Pulled from DB " + time);
      drawPieChart(activities);
    }
  }
}
```

MainActivity.java

```
package com.example.har.h;
import android.content.Context;
import android.content.Intent;
import android.hardware.Sensor;   //create instance of a specific type of sensor for accelorem,pressure senso
import android.hardware.SensorEvent;  //provide info about sensor event
import android.hardware.SensorEventListener;  //sensor occur them it go to sensorevent listner
import android.hardware.SensorManager; //create instance of a sensor service
import android.os.AsyncTask;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.TableLayout;
import android.widget.TextView;
import java.lang.ref.WeakReference;
import java.math.BigDecimal;
import java.util.ArrayList;
import java.util.List;
public class MainActivity extends AppCompatActivity implements SensorEventListener {
  private static List<Float> x;
  private static List<Float> y;
  private static List<Float> z;
  private static List<Float> input_signal;
  private SensorManager mSensorManager;  //instance
  private Sensor mAccelerometer;
  private RecognitionActivity activityPrediction;
  private static final String T = "Activity";
  private boolean confidenceView = false;
  private TextView downstairsTextView;
  private TextView joggingTextView;
  private TextView sittingTextView;
  private TextView standingTextView;
  private TextView upstairsTextView;
  private TextView walkingTextView;
  private AppDatabase activityDB;
  private Activity activity;
```

```
  private String current_activity = Constants.JOGGING;
  private int activity_duration = 0;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    x = new ArrayList<Float>();
    y = new ArrayList<Float>();
    z = new ArrayList<Float>();
    input_signal = new ArrayList<Float>();
    mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
//initializing sensor service,permission to use the sensor
    mAccelerometer = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    activityPrediction = new RecognitionActivity(getApplicationContext());
    activityDB = AppDatabase.getAppDatabase(MainActivity.this);
  }
  protected void onPause() {
    super.onPause();
    confidenceView = false;
  }
  protected void onResume() {
    super.onResume();
    confidenceView = false;
  }
  @Override
  public void onSensorChanged(SensorEvent event) {  //getting event i.e values of x, y
and z axiz
    activityPrediction();
    x.add(event.values[0]);
    y.add(event.values[1]);
    z.add(event.values[2]);
    Log.d(T,"X" + event.values[0] + "Y" + event.values[1] + "Z" + event.values[2]);
  }
  @Override
  public void onAccuracyChanged(Sensor sensor, int i) {
  }
  private void activityPrediction()
  {
    if(x.size() == Constants.N_SAMPLES && y.size() == Constants.N_SAMPLES && z.size() ==
Constants.N_SAMPLES) {
      normalizeBySD();
      input_signal.addAll(x); input_signal.addAll(y); input_signal.addAll(z);
      float[] results = activityPrediction.getActivityProb(toFloatArray(input_signal));
      if (confidenceView) {
        walkingTextView.setText(Float.toString(round(results[0], 2)));
        joggingTextView.setText(Float.toString(round(results[1], 2)));
        sittingTextView.setText(Float.toString(round(results[2], 2)));
        standingTextView.setText(Float.toString(round(results[3], 2)));
        upstairsTextView.setText(Float.toString(round(results[4], 2)));
        downstairsTextView.setText(Float.toString(round(results[5], 2)));
```

```
      }
      long unixTime = System.currentTimeMillis() / 1000L;
      String[] activity_results = predictedActivity(results);
      String activity_type = activity_results[1];
      int activity_indx = Integer.parseInt(activity_results[0]);
      if (results[activity_indx] > 0.7) {
        if (confidenceView) {
          TableLayout tableLayout = (TableLayout) findViewById(R.id.TableConfidence);
          int drawableId = getResources().getIdentifier(activity_type.toLowerCase(),
"drawable", getPackageName());
          Log.d("Converted case ", activity_type);
          tableLayout.setBackgroundResource(drawableId);
        }
        activity_duration += 1;
        if (current_activity != activity_type || activity_duration >= 1000) {
          Log.d(current_activity, Integer.toString(activity_duration));
          activity = new Activity(unixTime, current_activity, activity_duration);
          activity_duration = 0;
          current_activity = activity_type;
          new InsertActivity(MainActivity.this, activity).execute();
        }
      }
      x.clear(); y.clear(); z.clear(); input_signal.clear();
    }
  }
  private float[] toFloatArray(List<Float> list)
  {
    int i = 0;
    float[] array = new float[list.size()];
    for (Float f : list) {
      array[i++] = (f != null ? f : Float.NaN);
    }
    return array;
  }
  private void normalizeBySD()
  {
    for(int i = 0; i < Constants.N_SAMPLES; i++)
    {
      x.set(i,((x.get(i) - Constants.MEAN_X)/Constants.SD_X));
      y.set(i,((y.get(i) - Constants.MEAN_Y)/Constants.SD_Y));
      z.set(i,((z.get(i) - Constants.MEAN_Z)/Constants.SD_Z));
    }
  }
  public static float round(float d, int decimalPlace) {
    BigDecimal bd = new BigDecimal(Float.toString(d));
    bd = bd.setScale(decimalPlace, BigDecimal.ROUND_HALF_UP);
    return bd.floatValue();
  }
  public String[] predictedActivity(float[] results) {
    int predictedLabel = 0;
```

```
    String predictedString;
    for (int i = 0; i < 6; i++) {
      if (results[predictedLabel] < results[i]) {
        predictedLabel = i;
      }
    }
    switch (predictedLabel) {
      case 0:
        predictedString = Constants.WALKING;
        break;
      case 1:
        predictedString = Constants.JOGGING;
        break;
      case 2:
        predictedString = Constants.SITTING;
        break;
      case 3:
        predictedString = Constants.STANDING;
        break;
      case 4:
        predictedString = Constants.UPSTAIRS;
        break;
      case 5:
        predictedString = Constants.DOWNSTAIRS;
        break;
      default:
        predictedString = "";
        break;
    }
    String[] result = {Integer.toString(predictedLabel), predictedString};
    return result;
  }
  public void viewTable(View view){
    confidenceView = true;
    Log.d("Boolean Value ", Boolean.toString(confidenceView));
    setContentView(R.layout.view_confidence);
    downstairsTextView = (TextView)findViewById(R.id.downstairs_prob);
    joggingTextView = (TextView)findViewById(R.id.jogging_prob);
    sittingTextView = (TextView)findViewById(R.id.sitting_prob);
    standingTextView = (TextView)findViewById(R.id.standing_prob);
    upstairsTextView = (TextView)findViewById(R.id.upstairs_prob);
    walkingTextView = (TextView)findViewById(R.id.walking_prob);
  }
  public void viewStats(View view) {
    Intent intent = new Intent(this, DisplayStatsActivity.class);
    startActivity(intent);
  }
  public void startAR(View view) {
    mSensorManager.registerListener(this, mAccelerometer,
SensorManager.SENSOR_DELAY_FASTEST); //Registerd accelorometer listner
```

```
    confidenceView = false;
  }
  public void stopAR(View view) {
    super.onPause();
    mSensorManager.unregisterListener(this);
  }
  public void goBack(View view){
    confidenceView = false;
    setContentView(R.layout.activity_main);
  }
  private static class InsertActivity extends AsyncTask<Void,Void,Integer> {
    private WeakReference<MainActivity> activityReference;
    private Activity activity; // only retain a weak reference to the activity
    InsertActivity(MainActivity context, Activity activity) {
      activityReference = new WeakReference<>(context);
      this.activity = activity;
    }
    @Override protected Integer doInBackground(Void... objs) {
      activityReference.get().activityDB.activityDao().insert(activity);
      return activityReference.get().activityDB.activityDao().countEntries();
    }
    @Override protected void onPostExecute(Integer count) {
      String countStr = Integer.toString(count);
      Log.d("Success", "Added to DB " + countStr);
    }
  }
}
```

RecognitionActivity.java

```
package com.example.har.h;
import android.content.Context;
import android.content.res.AssetManager;
import org.tensorflow.contrib.android.TensorFlowInferenceInterface;
public class RecognitionActivity {
  static {
    System.loadLibrary(Constants.LIB_NAME);
  }
  private static RecognitionActivity activityInferenceInstance;
  private TensorFlowInferenceInterface inferenceInterface;
  private static AssetManager assetManager;
  public static RecognitionActivity getInstance(final Context context)
  {
    if (activityInferenceInstance == null)
    {
      activityInferenceInstance = new RecognitionActivity(context);
    }
    return activityInferenceInstance;
  }
  public RecognitionActivity(final Context context) {
    this.assetManager = context.getAssets();
```

```
    inferenceInterface = new TensorFlowInferenceInterface(assetManager,
Constants.MODEL_FILE);
  }
  public float[] getActivityProb(float[] input_signal)
  {
    float[] result = new float[Constants.OUTPUT_SIZE];
    inferenceInterface.feed(Constants.INPUT_NODE,input_signal,Constants.INPUT_SIZE);
    inferenceInterface.run(Constants.OUTPUT_NODES);
    inferenceInterface.fetch(Constants.OUTPUT_NODE,result);
    return result;
  }
  //Downstairs   Jogging      Sitting  Standing   Upstairs   Walking
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="20dp"
  xmlns:android="http://schemas.android.com/apk/res/android"
  tools:context="com.example.har.h.MainActivity"
  android:background="@drawable/runner1"
  android:backgroundTint="#8FFFFFFF"
  android:backgroundTintMode="src_over"
  android:orientation="vertical">
  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/welcome"
      android:textColor="@color/black"/>
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/buttons"
      android:textColor="@color/black"/>
  </LinearLayout>
  <RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:paddingLeft="50dp"
    android:paddingRight="50dp">
    <ImageButton
      android:id="@+id/start"
      android:layout_width="50dp"
      android:layout_height="50dp"
```

```
    android:layout_gravity="center"
    android:src="@drawable/start"
    android:onClick="startAR"
    android:scaleType="fitCenter"
    android:adjustViewBounds="true"
    android:padding="15dp"
    android:layout_alignParentLeft="true"/>
  <ImageButton
    android:id="@+id/stop"
    android:layout_width="50dp"
    android:layout_height="50dp"
    android:layout_gravity="center"
    android:src="@drawable/stop"
    android:onClick="stopAR"
    android:scaleType="fitCenter"
    android:adjustViewBounds="true"
    android:padding="15dp"
    android:layout_alignParentRight="true"/>
</RelativeLayout>
<LinearLayout
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:orientation="vertical">
  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/morebtns"
    android:textColor="@color/black"/>
</LinearLayout>
<RelativeLayout
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:paddingLeft="20dp"
  android:paddingRight="20dp">
  <Button
    android:id="@+id/viewConfidence"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:text="@string/table"
    android:onClick="viewTable"
    android:layout_alignParentLeft="true"/>
  <Button
    android:id="@+id/viewStats"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:text="@string/stats"
    android:onClick="viewStats"
    android:layout_alignParentRight="true"/>
```

```
  </RelativeLayout>
  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/disclaimer"
      android:textSize="8sp"/>
  </LinearLayout>
</LinearLayout>
```

activity_stats.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:id="@+id/activity_select_server"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  tools:context="com.example.har.h.DisplayStatsActivity">
  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/instruction"
    android:gravity="center_horizontal"
    android:textStyle="bold"
    android:padding="5dp"/>
  <Spinner
    android:id="@+id/spinner1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="top"
    android:background="@android:drawable/btn_dropdown"
    android:spinnerMode="dropdown" />
  <com.github.mikephil.charting.charts.PieChart
    android:id="@+id/piechart"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    />
</LinearLayout>
```

view_confidence.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="22dp"
  xmlns:android="http://schemas.android.com/apk/res/android"
```

```
tools:context="com.example.har.h.MainActivity"
android:background="#FFFFFF"
android:backgroundTint="#8FFFFFFF"
android:backgroundTintMode="src_over"
android:id="@+id/TableConfidence">
<TableRow
  android:id="@+id/walking_row"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="22dp">
  <TextView
    android:id="@+id/walking_title"
    android:text="@string/walking"
    android:textAlignment="center"
    android:textSize="20sp"
    android:textStyle="bold"
    android:layout_weight="1"
    android:textColor="@color/black"/>
  <TextView android:id="@+id/walking_prob"
    android:textAlignment="center"
    android:textSize="20sp"
    android:layout_weight="1"
    android:textColor="@color/black"/>
</TableRow>
<TableRow
  android:id="@+id/jogging_row"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="22dp">
  <TextView
    android:id="@+id/jogging_title"
    android:text="@string/jogging"
    android:textAlignment="center"
    android:textSize="20sp"
    android:textStyle="bold"
    android:layout_weight="1"
    android:textColor="@color/black"/>
  <TextView android:id="@+id/jogging_prob"
    android:textAlignment="center"
    android:textSize="20sp"
    android:layout_weight="1"
    android:textColor="@color/black"/>
</TableRow>
<TableRow
  android:id="@+id/sitting_row"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="22dp">
  <TextView
    android:id="@+id/sitting_title"
```

```
    android:text="@string/sitting"
    android:textAlignment="center"
    android:textSize="20sp"
    android:textStyle="bold"
    android:layout_weight="1"
    android:textColor="@color/black"/>
  <TextView android:id="@+id/sitting_prob"
    android:textAlignment="center"
    android:textSize="20sp"
    android:layout_weight="1"
    android:textColor="@color/black"/>
</TableRow>
<TableRow
  android:id="@+id/standing_row"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="22dp">
  <TextView
    android:id="@+id/standing_title"
    android:text="@string/standing"
    android:textAlignment="center"
    android:textSize="20sp"
    android:textStyle="bold"
    android:layout_weight="1"
    android:textColor="@color/black"/>
  <TextView android:id="@+id/standing_prob"
    android:textAlignment="center"
    android:textSize="20sp"
    android:layout_weight="1"
    android:textColor="@color/black"/>
</TableRow>
<TableRow
  android:id="@+id/downstairs_row"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="22dp">
  <TextView
    android:id="@+id/downstairs_title"
    android:layout_weight="1"
    android:text="@string/downstairs"
    android:textAlignment="center"
    android:textSize="20sp"
    android:textStyle="bold"
    android:textColor="@color/black"/>
  <TextView
    android:id="@+id/downstairs_prob"
    android:textAlignment="center"
    android:textSize="20sp"
    android:layout_weight="1"
    android:textColor="@color/black"/>
```

```
  </TableRow>
  <TableRow
    android:id="@+id/upstairs_row"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="22dp">
    <TextView
      android:id="@+id/upstairs_title"
      android:text="@string/upstairs"
      android:textAlignment="center"
      android:textSize="20sp"
      android:textStyle="bold"
      android:layout_weight="1"
      android:textColor="@color/black"/>
    <TextView android:id="@+id/upstairs_prob"
      android:textAlignment="center"
      android:textSize="20sp"
      android:layout_weight="1"
      android:textColor="@color/black"/>
  </TableRow>
  <ImageButton
    android:id="@+id/stop"
    android:layout_width="50dp"
    android:layout_height="50dp"
    android:layout_gravity="center"
    android:src="@drawable/back"
    android:onClick="goBack"
    android:scaleType="fitCenter"
    android:adjustViewBounds="true"
    android:padding="15dp"
    android:layout_alignParentRight="true"/>
</TableLayout>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.har.h">
  <uses-feature android:name="android.hardware.sensor.accelerometer"
android:required="true" />
  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
```

```
    </activity>
    <activity android:name=".DisplayStatsActivity"/>
  </application>
</manifest>
```

build.gradle

```
apply plugin: 'com.android.application'
android {
  compileSdkVersion 26
  defaultConfig {
    applicationId "com.example.har.h"
    minSdkVersion 15
    targetSdkVersion 26
    versionCode 1
    versionName "1.0"
    testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
  }
  buildTypes {
    release {
      minifyEnabled false
      proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
    }
  }
}
dependencies {
  implementation fileTree(dir: 'libs', include: ['*.jar'])
  implementation 'com.android.support:appcompat-v7:26.1.0'
  implementation 'com.android.support.constraint:constraint-layout:1.1.0'
  testImplementation 'junit:junit:4.12'
  androidTestImplementation 'com.android.support.test:runner:1.0.1'
  androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'
  compile 'org.tensorflow:tensorflow-android:+'
  compile 'android.arch.persistence.room:runtime:' + rootProject.archRoomVersion
  compile 'com.github.PhilJay:MPAndroidChart:v2.2.4'
  annotationProcessor 'android.arch.persistence.room:compiler:' +
rootProject.archRoomVersion
}
```

classifier_train.ipynb

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
import tensorflow as tf
from sklearn.model_selection import train_test_split
import pickle
from sklearn import metrics
import seaborn as sns
from tensorflow.python.tools import freeze_graph
%matplotlib inline
```

```
plt.style.use('ggplot')
column_names = ['user-id','activity','timestamp', 'x-axis', 'y-axis', 'z-axis']
x = 'x-axis'
y = 'y-axis'
z = 'z-axis'
def read_data(file_name):
    data = pd.read_csv(file_name, header = None, names = column_names)
    return data
def normalizeBySD(feature):
    mu = np.mean(feature,axis = 0)
    sigma = np.std(feature, axis = 0)
    return (feature – mu)/sigma
features = read_data('data/WISDM_ar_v1.1_raw.txt')
features.dropna(axis=0, how='any', inplace= True)
features[z] = features[z].str.replace(';','')
features[z] = features[z].astype(np.float64)
features.drop_duplicates(column_names, keep= 'first', inplace=True)
features[x] = normalizeBySD(features[x])
features[y] = normalizeBySD(features[y])
features[z] = normalizeBySD(features[z])
activity = ['Walking', 'Jogging',  'Sitting', 'Standing', 'Downstairs', 'Upstairs']
step = 15
num_rows = 90
def segmentDataset(featureList):
    segments = []
    labels = []
    for i in range(0, len(featureList) - num_rows, step):
        xs = featureList[x].values[i: i + num_rows]
        ys = featureList[y].values[i: i + num_rows]
        zs = featureList[z].values[i: i + num_rows]
        try:
            label = stats.mode(featureList["activity"][i: i + num_rows])[0][0]
            if label in activity:
                labels = np.append(labels,label)
                segments.append([xs, ys, zs])
            else:
                print(i, i+num_rows)
        except TypeError:
            print(i, i+num_rows)
    return segments, labels
segments, new_label = segmentDataset(features)
labels = np.asarray(pd.get_dummies(new_label), dtype = np.int8)
segments = np.array(segments)
reshaped_segments = segments.reshape(len(segments),1, 90, 3)
train_x, test_x, train_y, test_y = train_test_split(reshaped_segments, labels, test_size=0.3)
input_height = 1
input_width = 90
num_labels = 6
batch_size = 10
input_name = 'input'
output_name = 'output'
kernel_size = 60
depth = 60
num_channels = 3
kernel_size_pool = 20
```

```
stride_pool = 2
kernel_size2 = 6
depth2 = 6
num_channel2 = depth * num_channels
num_hidden = 256
def initialize_weights(shape):
    initial = tf.truncated_normal(shape, stddev = 0.1)
    return tf.Variable(initial)
def initialize_biases(shape):
    initial = tf.constant(0.0, shape = shape)
    return tf.Variable(initial)
X = tf.placeholder(tf.float32, shape=[None,input_height,input_width,num_channels], name=input_name)
conv1_filters = initialize_weights([1, kernel_size, num_channels, depth])
conv1_biases = initialize_biases([depth * num_channels])
c = tf.nn.relu(tf.add(tf.nn.depthwise_conv2d(X, conv1_filters, [1, 1, 1, 1], padding='VALID'), conv1_biases))
p = tf.nn.max_pool(c, ksize=[1, 1, kernel_size_pool, 1], strides=[1, 1, stride_pool, 1], padding='VALID')
conv2_filters = initialize_weights([1, kernel_size2, num_channel2, depth2])
conv2_biases = initialize_biases([depth2 * num_channel2])
c = tf.nn.relu(tf.add(tf.nn.depthwise_conv2d(p, conv2_filters, [1, 1, 1, 1], padding='VALID'), conv2_biases))
shape = c.get_shape().as_list()
c_flat = tf.reshape(c, [-1, shape[1] * shape[2] * shape[3]])
flat_weights = initialize_weights([shape[1] * shape[2] * depth * num_channels * (depth2), num_hidden])
flat_biases = initialize_biases([num_hidden])
flat = tf.nn.tanh(tf.add(tf.matmul(c_flat, flat_weights),flat_biases))
out_weights = initialize_weights([num_hidden, num_labels])
out_biases = initialize_biases([num_labels])
pred_Y = tf.matmul(flat, out_weights) + out_biases
y_ = tf.nn.softmax(pred_Y, name=output_name)
learning_rate = 0.0001
training_epochs = 2 #50
total_batches = train_x.shape[0] // batch_size
Y = tf.placeholder(tf.float32, shape=[None,num_labels])
loss = -tf.reduce_sum(Y * tf.log(y_))
optimizer = tf.train.GradientDescentOptimizer(learning_rate = learning_rate).minimize(loss)
right_prediction = tf.equal(tf.argmax(y_,1), tf.argmax(Y,1))
accuracy = tf.reduce_mean(tf.cast(right_prediction, tf.float32))
saver = tf.train.Saver()
history = dict(train_loss=[], train_acc=[], test_loss=[], test_acc=[])
with tf.Session() as session:
    tf.global_variables_initializer().run()
    for epoch in range(training_epochs):
        for b in range(total_batches):
            offset = (b * batch_size) % (train_y.shape[0] - batch_size)
            batch_x = train_x[offset:(offset + batch_size), :, :, :]
            batch_y = train_y[offset:(offset + batch_size), :]
            session.run(optimizer,feed_dict={X: batch_x, Y : batch_y})
        _, loss_tr, accuracy_tr = session.run([y_, loss, accuracy],feed_dict={X: train_x, Y : train_y})
        _, loss_te, accuracy_te = session.run([y_, loss, accuracy], feed_dict={X: test_x, Y: test_y})
        history['train_loss'].append(loss_tr)
        history['train_acc'].append(accuracy_tr)
        history['test_loss'].append(loss_te)
        history['test_acc'].append(accuracy_te)
        print ("Epoch: ",epoch," Training Loss: ",loss_tr," Training Accuracy: ", accuracy_tr)
    predictions, acc_final, loss_final = session.run([y_, accuracy, loss], feed_dict={X: test_x, Y: test_y})
    print ("Final loss:", loss_final, "Final Accuracy:", acc_final)
```

```
    pickle.dump(predictions, open("predictions.p", "wb"))
    pickle.dump(history, open("history.p", "wb"))
    tf.train.write_graph(session.graph_def, '.', './checkpoint/har.pbtxt')
    saver.save(session, save_path = "./checkpoint/har.ckpt")
history = pickle.load(open("history.p", "rb"))
predictions = pickle.load(open("predictions.p", "rb"))
fig, ax1 = plt.subplots(figsize=(18, 12))
fig.suptitle("Training session's progress over iterations", fontsize=20)
ax2 = ax1.twinx()
ax2.plot(np.array(history['train_loss']), "r--", label="Train loss")
ax2.plot(np.array(history['test_loss']), "r-", label="Test loss")
ax1.plot(np.array(history['train_acc']), "g--", label="Train accuracy")
ax1.plot(np.array(history['test_acc']), "g-", label="Test accuracy")
ax1.legend(shadow=True, loc='best', fontsize=20)
ax2.legend(shadow=True, loc='right', fontsize=20)
ax2.set_ylabel('Loss values', fontsize=20)
ax1.set_xlabel('Training Epoch', fontsize=20)
ax1.set_ylabel('Accuracy values', fontsize=20)
plt.show()
LABELS = ['Downstairs', 'Jogging', 'Sitting', 'Standing', 'Upstairs', 'Walking']
max_test = np.argmax(test_y, axis=1)
max_predictions = np.argmax(predictions, axis=1)
confusion_matrix = metrics.confusion_matrix(max_test, max_predictions)
plt.figure(figsize=(16, 14))
sns.heatmap(confusion_matrix, xticklabels=LABELS, yticklabels=LABELS, annot=True, fmt="d", cmap="YlGnBu");
plt.title("Confusion matrix")
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.show();
MODEL_NAME = 'har'

input_graph_path = 'checkpoint/' + MODEL_NAME+'.pbtxt'
checkpoint_path = './checkpoint/' +MODEL_NAME+'.ckpt'
restore_op_name = "save/restore_all"
filename_tensor_name = "save/Const:0"
output_frozen_graph_name = 'har_classifier.pb'
freeze_graph.freeze_graph(input_graph_path, input_saver="",
                          input_binary=False, input_checkpoint=checkpoint_path,
                          output_node_names=output_name, restore_op_name="save/restore_all",
                          filename_tensor_name="save/Const:0",
                          output_graph=output_frozen_graph_name, clear_devices=True, initializer_nodes="")
```

4.2 SCREENSHOTS

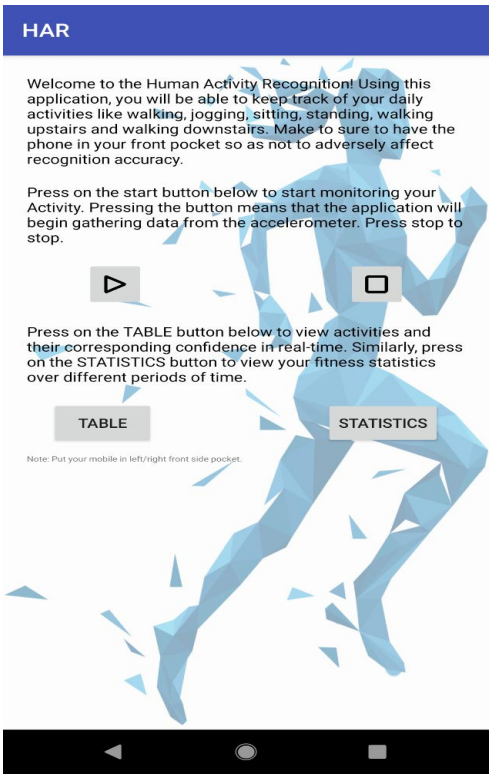

Fig 4.2.1Home Page

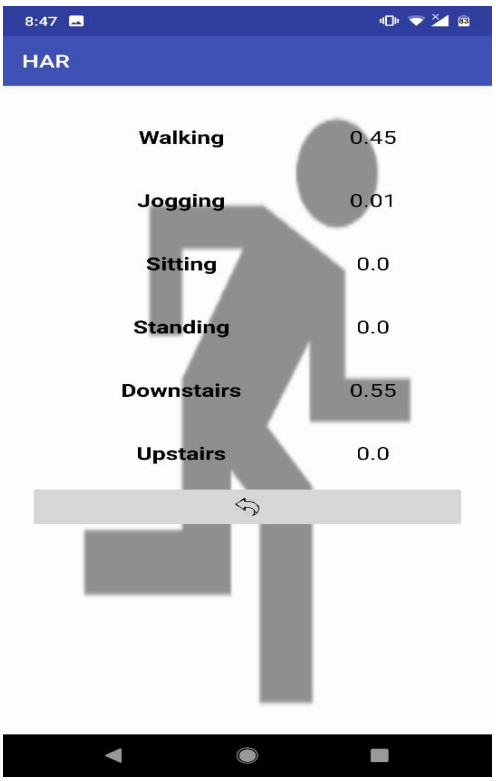

Fig 4.2.1. Jogging

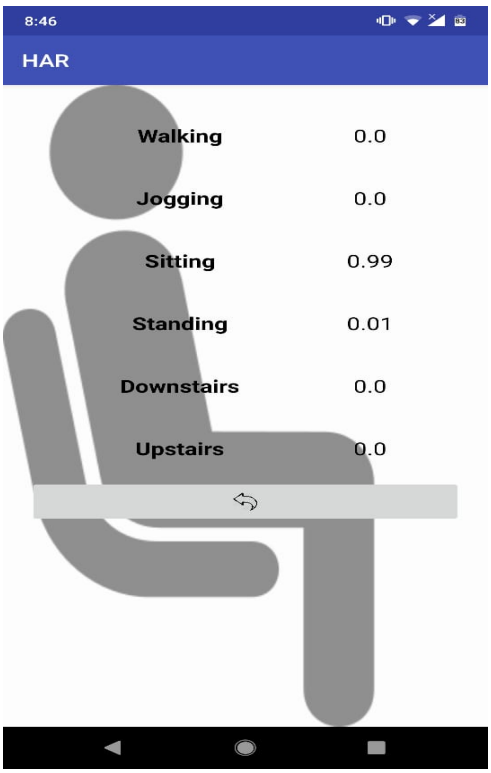

Fig 4.2.3 Sitting

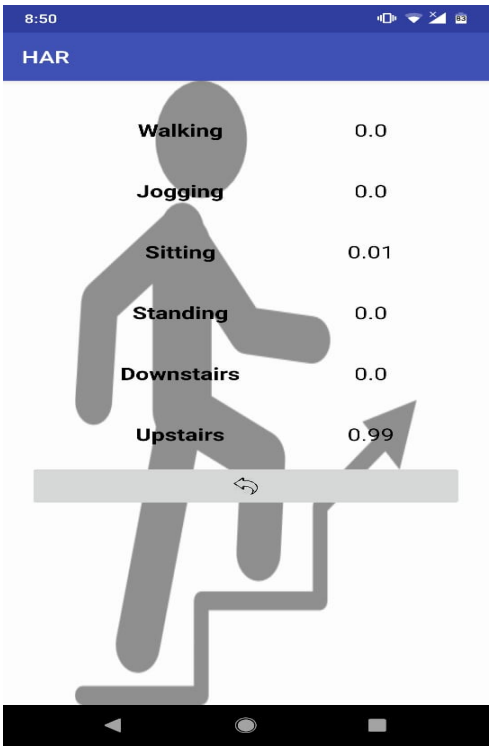

Fig 4.2.3 upstairs

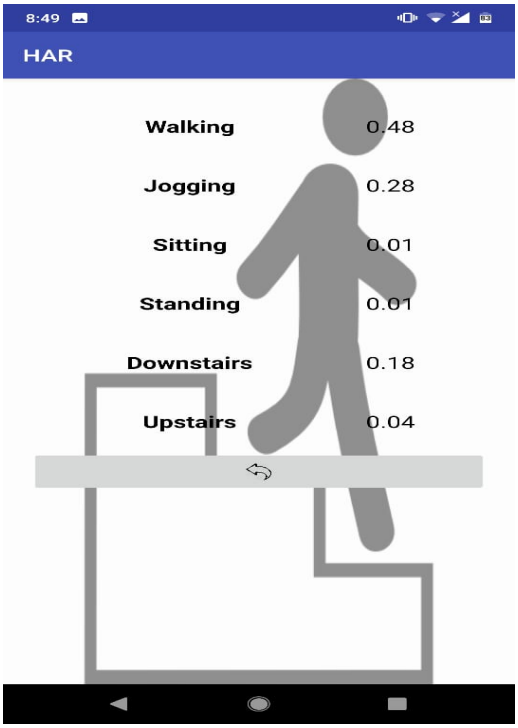

Fig 4.2.3 Downstairs

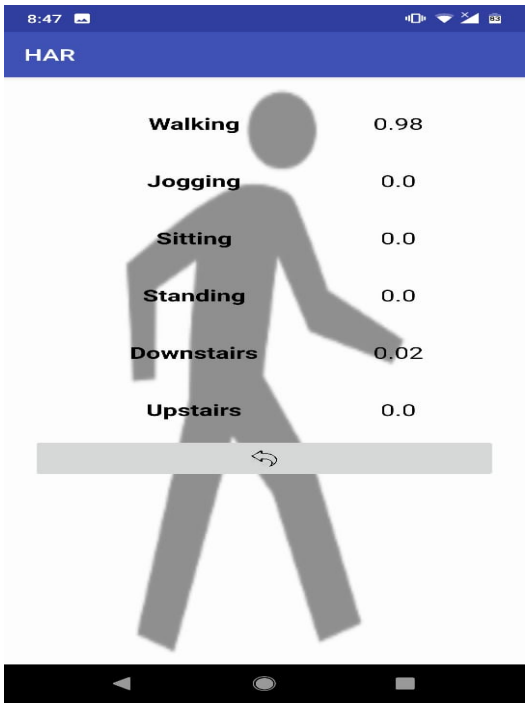

Fig 4.2.4 Walking

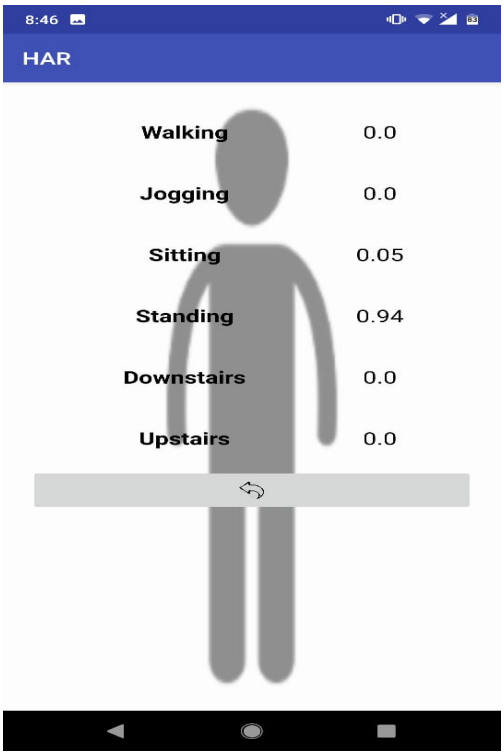

Fig 4.2.5 Standing

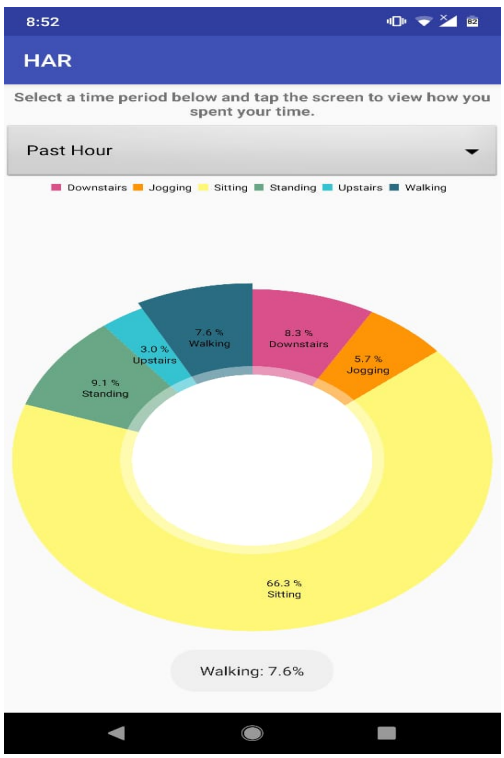

Fig 4.2.6 Pie chart

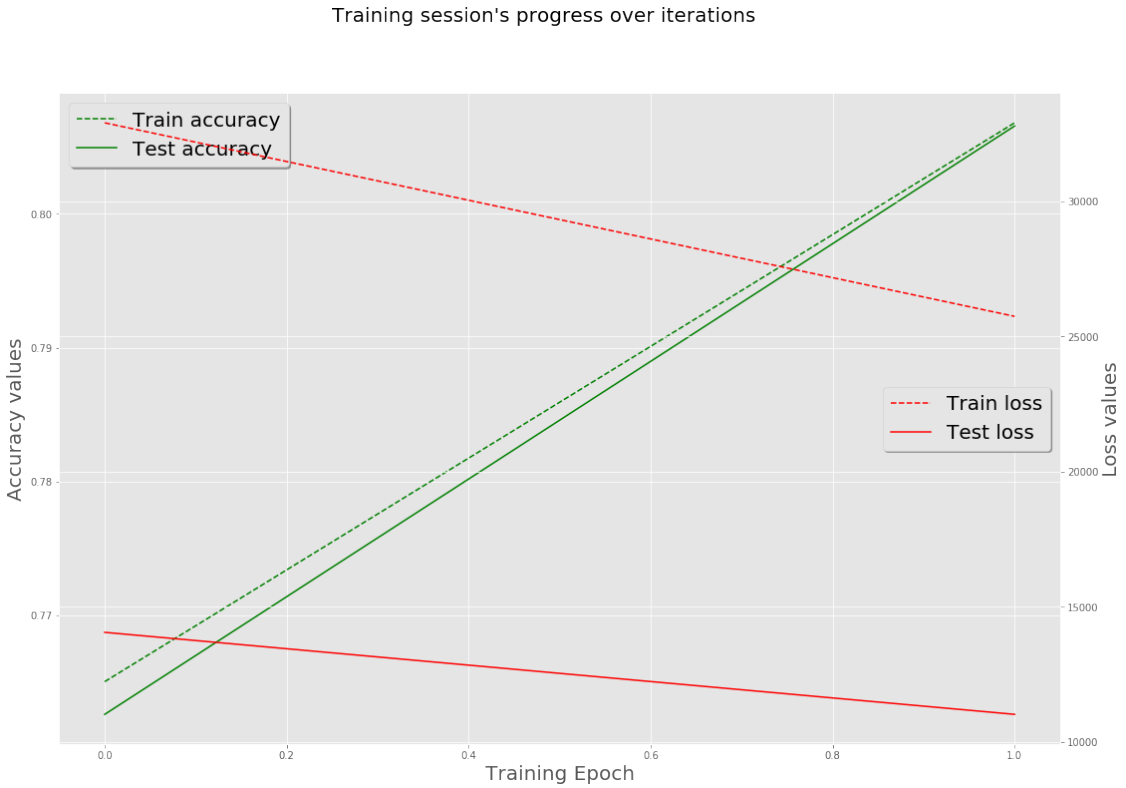

Fig 4.2.7 Training session's progress over iterations

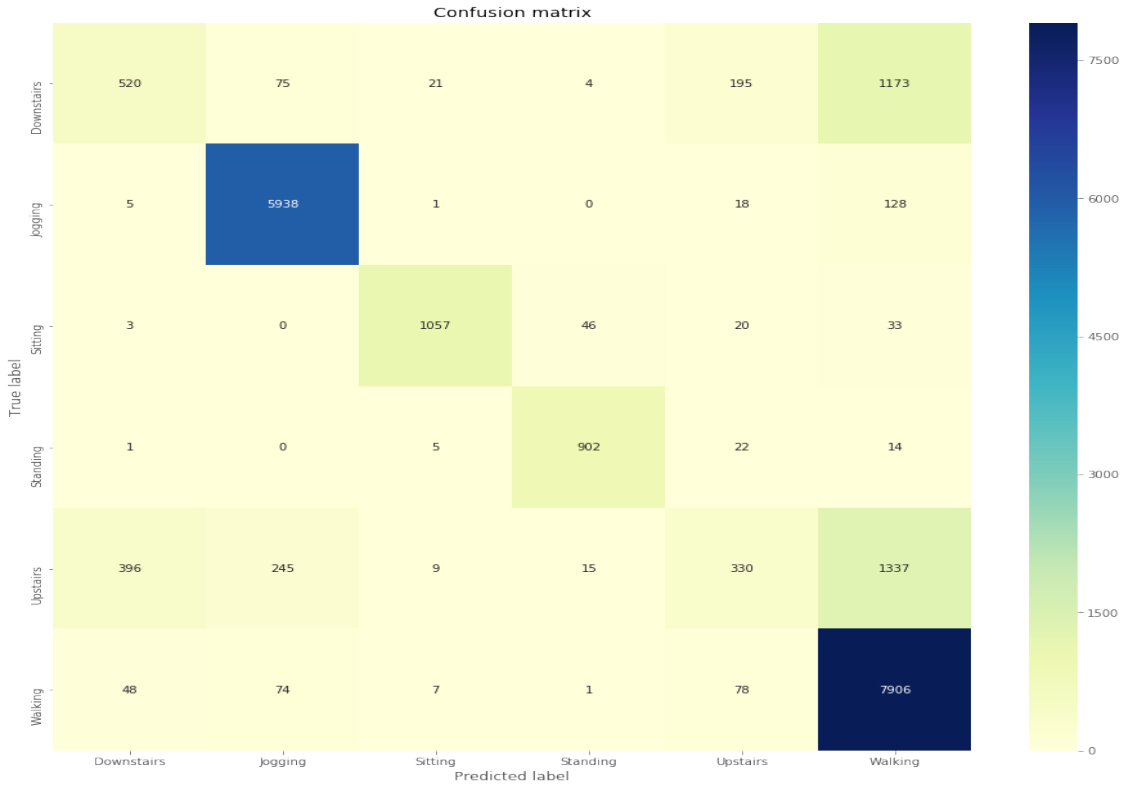

Fig 4.2.8 Confusion Matrix

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

In this world of technology, sensors play an important role to recognize anything from real world. [3]The information sensed by those sensors is very useful and can be used in various applications. Some of the examples are: Gyroscope sensor- Pokemon Go game, Magnetometer- Compass app and Metal detector app, GPS sensor- Uber and Google maps. [6]

Accelerometer sensor with a machine learning model helps in this project to identify most performed activity. [6]So, this project simply helps us to have a good health by not performing a single for longer period. Doing a single activity for hours can lead to body pain problems.[1]

So, from this project, it can be concluded that this problem has thereby reduced to some extent. However, different other modules can also be included for more advanced system. This project is overall a good project to recognize simply human activities and have analysis on it.

It can be concluded that this project provides an answer or resolution to the problem of human activity recognition.

5.2 FUTURE SCOPE

Future scope of human activity recognition may consist:

- New features like reminder to show that particular activity is perform more than the limit can add.
- The new features like daily tracker to track how many calories we burn in how much time can be add.
- A smart home is an environment equipped with sensors that can enhance the safety of residents and monitor their health conditions.
- Healthcare monitoring systems can be designed based on the combination of one or more activity recognition components such as fall detection, human tracking, security alarm.

REFERENCES

[1] Google : www.google.com

[2] Medium-CNN: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

[3] Developers-Basic app: https://developer.android.com/training/basics/firstapp

[4] Medium-Tensorflow: https://medium.com/@elye.project/applying-tensorflow-in-android-in-4-steps-to-recognize-superhero-f224597eb055

[5] Towards data science: https://towardsdatascience.com/android-with-tensorflow-part-1-6897ba617b1e?gi=9c52febbdde2

[6] Developers-Sensor manager: https://developer.android.com/reference/android/hardware/SensorManager

[7] Youtube- Tensorflow: https://www.youtube.com/watch?v=kFWKdLOxykE

[8] Youtube- Accelerometer: https://www.youtube.com/watch?v=pkT7DU1Yo9Q