1. Get accelerometer movement data on internet

```
8435  0.010101,1.461029,0.139923,upst
8436  -0.153107,0.688461,-0.011169,upst
8437  -0.238068,1.117691,-0.013245,upst
8438  0.512146,0.177597,0.049637,downst
8439  0.482162,1.04216,0.117905,downst
8440  0.114914,0.207794,-0.360657,upst
8441  0.17157,1.762344,-0.2565,downst
8442  0.029312,1.163284,-0.183975,upst
8443  -0.018692,0.622955,0.414322,upst
8444  -0.132843,1.506439,-0.180084,upst
8445  0.178772,0.419174,-0.531815,upst
8446  -0.135056,0.838516,0.099365,upst
8447  -0.295853,0.369705,0.031433,downst
8448  -0.603821,-0.012466,-0.321365,downst
8449  0.655624,1.36734,-0.502167,downst
8450  -0.376083,2.180923,-0.816879,upst
8451  0.125992,1.351669,-0.263565,upst
8452  0.002701,1.113678,0.069077,upst
8453  0.304108,1.265457,-0.663284,upst
8454  0.526031,1.770905,0.68512,downst
8455  -0.124268,1.072891,-0.557541,downst
8456  -0.036926,0.957672,0.254761,upst
8457  0.29451,-0.068436,0.292496,downst
8458  0.255508,0.969772,-0.638382,downst
8459  0.006805,0.978363,-0.231689,upst
8460  -0.949234,1.671478,-0.850861,downst
8461
```

2. KNN model in app
3. Run server

onal sensors

Move

-161.5

180

Rotation

163.0

180

Sensor values

Accelerometer (m/s²):	1.72	9.31	-2.55
Gyroscope (rad/s):	0.00	0.00	0.00
Magnetometer (µT):	25.73	12.29	39.55
Rotation:	ROTATION_0		

27.4

180

dditional sensors

Move

21.0

180

Rotation

-10.3

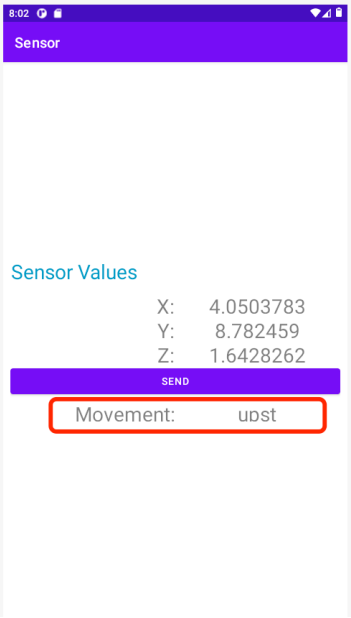
180

Sensor values

Accelerometer (m/s²):	4.05	8.78	1.64
Gyroscope (rad/s):	0.00	0.00	0.00
Magnetometer (µT):	-10.30	19.48	-43.50

Emulator: DPad Device API 30

Emulator: DPad Device API 30



Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp"
    tools:context=".MainActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:orientation="vertical"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="45dp"
                android:layout_margin="1dp"
                android:layout_weight="1"
                android:gravity="left"
                android:text="Sensor Values"
                android:textColor="#268EBD"
                android:textSize="28dp" />
        </LinearLayout>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="1dp"
            android:gravity="center"
            android:orientation="horizontal"
            android:weightSum="2">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="30dp"
                android:layout_margin="1dp"
                android:layout_weight="1"
                android:gravity="right"
                android:text="X:"
                android:textSize="28dp" />

            <TextView
                android:id="@+id/x_value"
                android:layout_width="match_parent"
                android:layout_height="30dp"
```

```
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:textSize="28dp"
        android:gravity="center"
        android:text="0" />
</LinearLayout>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="1dp"
    android:gravity="center"
    android:orientation="horizontal"
    android:weightSum="2">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:gravity="right"
        android:text="Y:"
        android:textSize="28dp" />

    <TextView
        android:id="@+id/y_value"
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:textSize="28dp"
        android:gravity="center"
        android:text="0" />
</LinearLayout>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="1dp"
    android:gravity="center"
    android:orientation="horizontal"
    android:weightSum="2">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:gravity="right"
        android:text="Z:"
        android:textSize="28dp" />

    <TextView
        android:id="@+id/z_value"
        android:layout_width="match_parent"
        android:layout_height="30dp"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:textSize="28dp"
```

```
                android:gravity="center"
                android:text="0" />

        </LinearLayout>


        <Button
            android:id="@+id/send"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Send"
            />

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="1dp"
            android:gravity="center"
            android:orientation="horizontal"
            android:weightSum="2">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="30dp"
                android:layout_margin="1dp"
                android:layout_weight="1"
                android:gravity="right"
                android:text="Movement:"
                android:textSize="28dp" />

            <TextView
                android:id="@+id/movement_value"
                android:layout_width="match_parent"
                android:layout_height="30dp"
                android:layout_margin="1dp"
                android:layout_weight="1"
                android:textSize="28dp"
                android:gravity="center" />
        </LinearLayout>
    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java

```
package com.example.sensor;

import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
```

```
import androidx.appcompat.app.AppCompatActivity;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;

public class MainActivity extends AppCompatActivity {
    // declare X,Y,Z axis object
    private TextView xValue;
    private TextView yValue;
    private TextView zValue;
    private SensorManager sensorManager;
    private SQLiteDB mySQLHelper;
    private List<DataBean> dataList;
    private TextView movementView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        xValue = (TextView) findViewById(R.id.x_value);
        yValue = (TextView) findViewById(R.id.y_value);
        zValue = (TextView) findViewById(R.id.z_value);
        movementView = (TextView) findViewById(R.id.movement_value);
        sensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
        int sensorType = Sensor.TYPE_ACCELEROMETER;
        sensorManager.registerListener(myAccelerometerListener,

sensorManager.getDefaultSensor(sensorType),SensorManager.SENSOR_DELAY_NORMAL)
;
        Button buttonSend = (Button)findViewById(R.id.send);
        buttonSend.setOnClickListener(buttonSendOnClickListener);
    }
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        movementView = findViewById(R.id.movement_TextView);
        initDB();
        getLocation();
        getSensorValue();
        getDB();
        k_value = (int)Math.sqrt(dataList.size());
    }
    final SensorEventListener myAccelerometerListener = new
SensorEventListener(){
        public void onSensorChanged(SensorEvent sensorEvent){
            if(sensorEvent.sensor.getType() == Sensor.TYPE_ACCELEROMETER){
                float X_lateral = sensorEvent.values[0];
                float Y_longitudinal = sensorEvent.values[1];
                float Z_vertical = sensorEvent.values[2];
                xValue.setText(String.valueOf(X_lateral));
                yValue.setText(String.valueOf(Y_longitudinal));
                zValue.setText(String.valueOf(Z_vertical));
            }
```

```
        }
        public void onAccuracyChanged(Sensor sensor , int accuracy){
        }
    };
    public void onPause(){
 /*
 * Even activity is pause, sensor will still work, to save the power of the
phone, will make this onPause
 * But if is using for nursing home, should not include this part, sensor
needs
to work all the time
 * */
        sensorManager.unregisterListener(myAccelerometerListener);
        super.onPause();
    }
    void initDB() {
        mySQLHelper = new SQLiteDB(this);
        //Stetho.initializeWithDefaults(this);
        try {
            InputStreamReader file = null;
            file = new
InputStreamReader(getAssets().open("fallingData.csv"));
            BufferedReader buffer = new BufferedReader(file);
            String line = "";
            while ((line = buffer.readLine()) != null) {
                Log.e("line", line);
                String[] str = line.split(",");
                float x = Float.parseFloat(str[0].toString());
                float y = Float.parseFloat(str[1].toString());
                float z = Float.parseFloat(str[2].toString());
                String c = str[3].toString();
                Log.e("line", x+y+z+c);
                mySQLHelper.insertRecord(x, y, z, c);
                Log.e("Import", "Successfully Updated Database.");
            }
        } catch (IOException e) {
            Log.e("SQLError", e.getMessage().toString());
        }
    }
    void getDB() {
        Log.e("ShowData", "making list");
        dataList = mySQLHelper.queryRecord();
        Log.e("ShowData", "done list");
    }

    void isfalling_KNN(float x2, float y2, float z2) {
        PriorityQueue<DistanceData> heap = new
PriorityQueue<DistanceData>(new

Comparator<DistanceData>() {

public int compare(DistanceData a, DistanceData b) { return

(int)(a.getDistance()-b.getDistance()); }

});
        for (int i = 0; i < dataList.size(); i++) {
```

```
            float x1 = dataList.get(i).getX();
            float y1 = dataList.get(i).getY();
            float z1 = dataList.get(i).getZ();
            float distance_temp = (float) distance(x1,y1,z1,x2,y2,z2);
            heap.offer(new DistanceData(distance_temp,
dataList.get(i).getClass_(),0));
        }
        HashMap<String, Integer> classcount = new HashMap<String, Integer>();
        for (int i = 0; i < k_value; i++) {
            DistanceData tempData = heap.poll();
            if(!classcount.containsKey(tempData.getClass_())){
                classcount.put(tempData.getClass_(), 1);
            } else {
                classcount.put(tempData.getClass_(),
                        classcount.get(tempData.getClass_())+1);
            }
        }
        PriorityQueue<DistanceData> knn_return = new
PriorityQueue<DistanceData>(new
            Comparator<DistanceData>() {
            public int compare(DistanceData a, DistanceData b) { return
                (int)(a.getCount_()-b.getCount_()); }
            });
        Iterator classcountIterator = classcount.entrySet().iterator();
        while (classcountIterator.hasNext()) {
            Map.Entry mapElement
                    = (Map.Entry)classcountIterator.next();
            DistanceData tempData = new DistanceData(0,

String.valueOf(mapElement.getKey()),(int)mapElement.getValue());
            knn_return.offer(tempData);
        }
        movementResult = knn_return.poll().getClass_();
        movementView.setText(movementResult);
        Log.e("Movement", movementResult);
        sentToServer();
    }

    Button.OnClickListener buttonSendOnClickListener
            = new Button.OnClickListener(){

        public void onClick(View arg0) {
            // TODO Auto-generated method stub
            Socket socket = null;
            DataOutputStream dataOutputStream = null;
            DataInputStream dataInputStream = null;

            try {
                socket = new Socket("127.0.0.1", 5000);
                dataOutputStream = new
DataOutputStream(socket.getOutputStream());
                dataInputStream = new
DataInputStream(socket.getInputStream());
                xValue.setText(dataInputStream.readUTF());
                yValue.setText(dataInputStream.readUTF());
                zValue.setText(dataInputStream.readUTF());
            } catch (IOException e) {
```

```
                // TODO Auto-generated catch block
                e.printStackTrace();
            } finally{
                if (socket != null){
                    try {
                        socket.close();
                    } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }

                if (dataOutputStream != null){
                    try {
                        dataOutputStream.close();
                    } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }

                if (dataInputStream != null){
                    try {
                        dataInputStream.close();
                    } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }
            }
        }};
}
```

SQLiteDB.java

```
package com.example.sensor;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.provider.ContactsContract;
import android.provider.SyncStateContract;
import android.util.Log;
import androidx.annotation.Nullable;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
public class SQLiteDB extends SQLiteOpenHelper {
    public static final String CREATE_TABLE = "create table " +
MyConstant.TABLE_NAME +
            " (" + MyConstant.COL_ID + " INTEGER, " + MyConstant.COL_X + "
REAL, " +
            MyConstant.COL_Y +
            " REAL, " + MyConstant.COL_Z + " REAL, " + MyConstant.COL_CLASS +
```

```
"
    varchar(10))";
    private SQLiteDatabase db;
    public SQLiteDB(@Nullable Context context) {
        super(context, MyConstant.DB_NAME, null, 1);
        db = this.getWritableDatabase();
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        Log.d("DatebaseHelper","Create dataset");
        db.execSQL(CREATE_TABLE);
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int i, int i1) {
    }
    public long insertRecord(float x, float y, float z, String class_) {
        ContentValues values = new ContentValues();
        values.put(MyConstant.COL_X, x);
        values.put(MyConstant.COL_Y, y);
        values.put(MyConstant.COL_Z, z);
        values.put(MyConstant.COL_CLASS, class_);
        return db.insert(MyConstant.TABLE_NAME,null, values);
    }
    public List<DataBean> queryRecord() {
        Cursor cursor = db.rawQuery("SELECT * FROM " + MyConstant.TABLE_NAME,
null);
        List<DataBean> recordBeanList = new ArrayList<>();
        if (cursor.moveToFirst()) {
            do {
                // on below line we are adding the data from cursor to our
array list.
                recordBeanList.add(new
DataBean(Float.parseFloat(cursor.getString(1)),
                        Float.parseFloat(cursor.getString(2)),
                        Float.parseFloat(cursor.getString(3)),
                        cursor.getString(4)));
            } while (cursor.moveToNext());
            // moving our cursor to next.
        }
        cursor.close();
        return recordBeanList;
    }
}
```

PoolEchoServer.java

```
import java.net.*;
import java.io.*;
public class PoolEchoServer extends Thread {
   public final static int defaultPort = 5000;
   ServerSocket theServer;
   static int num_threads = 10;
   public static void main(String[] args) {
```

```
    int port = defaultPort;
    try { port = Integer.parseInt(args[0]); }
    catch (Exception e) { }
    if (port <= 0 || port >= 65536) port = defaultPort;
    try {
       ServerSocket ss = new ServerSocket(port);
       System.out.println("Server Socket Start!!");
       for (int i = 0; i < num_threads; i++) {
          System.out.println("Create num_threads "
                             + i + " Port:" + port);
          PoolEchoServer pes = new PoolEchoServer(ss);
          pes.start();
       }
    }
    catch (IOException e) { System.err.println(e); }
 }
 public PoolEchoServer(ServerSocket ss) { theServer = ss; }

 public void run() {
    while (true) {
       try {
          DataOutputStream output;
          DataInputStream input;
          Socket connection = theServer.accept();
          System.out.println("Accept Client!");

          //OutputStream os = s.getOutputStream();
          //InputStream is = s.getInputStream();
          input = new DataInputStream(
          connection.getInputStream() );
          output = new DataOutputStream(
          connection.getOutputStream() );
          //BufferedReader bf = new
          // BufferedReader(new InputStreamReader(is));

          System.out.println("Client Connected and Start get I/O!!");
          while (true) {
             System.out.println("==> Input from Client: "
                                + input.readUTF());
             System.out.println(
               "Output to Client ==> \"Connection successful\"");
             output.writeUTF( "Connection successful" );
             //os.write(n);
             //os.write("Hello Client!!");
             output.flush();
          } // end while
       } // end try
       catch (IOException e) { }
```

```
      } // end while
   } // end run
}
```