Nursing Home project

Long Nguyen Hoang

Scenario

- Falls are a growing public health concern and source of injury for older adults.
- Approximately 1 in 5 older adults in United States experience a fall.[1]
- 3 million emergency department visits are related to falls [2] [3]
- Fall death rates among adults age 65 and older has increased more than 3.% from 2007 to 2016[2]
- In 2016, $50 billion was spent on non-fatal falls injuries and $754 million is spent on fatal falls[3]
- Medicare and Medicaid shouldered 75% of these costs[3]

1.Centers for disease control and prevention (CDC). National center for health statistics. (2020). Fall facts. Retrieved from https//www.cdc.gov//fall/adultfalls.html.
2. Burns, E., & Kakara, R. (2018). Deaths from falls among persons Aged ≥65 Years- United States, 2007–2016. Morbidity and Mortality Weekly Report 67(18), 509–514. doi: http://dx.doi.org/10.15585.
3. Florence, C. S., Bergen, G., Atherly, A., Burns, E., Stevens, J. and Drake, C. (2018), Medical costs of fatal and nonfatal falls in older adults. J Am Geriatr Soc, *66*: 693-698. doi: 10.1111/jgs.

Fallalarm approach

Reactive approach:
Incident Reporting (data collection)

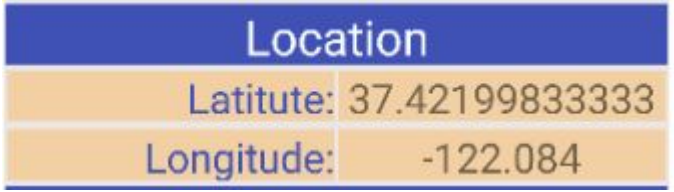
Using wearable devices:
to provide current information
and risk

Accelerometer

```
private SensorManager sensorManager;
    void getSensorValue() {
        sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        sensorManager.registerListener(myAccelerometerListener,
                sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
                SensorManager.SENSOR_DELAY_NORMAL);
        sensorManager.registerListener(myAccelerometerListener,
                sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE),
                SensorManager.SENSOR_DELAY_NORMAL);
    }
    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];
            } else if (sensorEvent.sensor.getType() == Sensor.TYPE_GYROSCOPE) {
                float x2 = sensorEvent.values[0];
                float y2 = sensorEvent.values[1];
                float z2 = sensorEvent.values[2];
            }
        }
        public void onAccuracyChanged(Sensor sensor , int accuracy){
        }
    };
```

Accelerometer data	
X:	7.007348
Y:	-0.5644532
Z:	-6.8421073
Movement:	falling

Location




```
private LocationManager locationManager;
void getLocation() {
        locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
        if (ContextCompat.checkSelfPermission(MainActivity.this,
        Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
                ContextCompat.checkSelfPermission(MainActivity.this,
                Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(MainActivity.this, new String[]
            {Manifest.permission.ACCESS_COARSE_LOCATION,
            Manifest.permission.ACCESS_FINE_LOCATION}, 1);
        }
        locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0, new LocationListener() {
            @Override
            public void onLocationChanged(@NonNull Location location) {
                stringLatitude = String.valueOf(location.getLatitude());
                stringLontitude = String.valueOf(location.getLongitude());
            }
        });
    }
```

Machine learning -KNN

- Most mobile devices are equipped with different kind of sensors
- 3 numbers from Accelerometer sensor
- 3 numbers from Accelerometer sensor
- Split the data into training and testing dataset
- Accuracy test based on the give training and testing dataset
- Input unknown data into the model and do the prediction

```
double distance(float x1, float y1, float z1, float x2, float y2, float z2){
        double distance = 0.0;
        float temp = (x1-x2);
        distance += temp*temp;
        temp = (y1-y2);
        distance += temp*temp;
        temp = (z1-z2);
        distance += temp*temp;
        return Math.sqrt(distance);
    }

    void isfalling_KNN(float x2, float y2, float z2) {
        PriorityQueue<DistanceData> heap = new PriorityQueue<DistanceData>((a, b) -> (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);
        }
```

Server

PoolServer.java

```java
public class PoolServer extends Thread {
  ServerSocket theServer;
  static int num_threads = 10;

  public static void main(String[] args) throws IOException {
    try {
      ServerSocket ss = new ServerSocket(8080);
      System.out.println("Server Socket Start!! on 8000");
      for (int i = 0; i < num_threads; i++) {
        System.out.println("Create num_threads " + i + " Port: 8000.");
        PoolServer myserver = new PoolServer(ss);
        myserver.start();
      }
    } catch (IOException e) {
      System.err.println(e);
    }
  }

  public PoolServer(ServerSocket ss) {
    theServer = ss;
  }
```

```java
  public void run() {
    while (true) {
      try {
        Socket connection = theServer.accept();
        DataOutputStream output = new
        DataOutputStream(connection.getOutputStream());
        DataInputStream input = new DataInputStream(connection.getInputStream());
        String cInput = input.readUTF();
        String[] text = cInput.split(",");
        if(text[1] == "falling"){
            try {
                DateTimeFormatter dtf = DateTimeFormatter
                        .ofPattern("yyyy/MM/dd HH:mm:ss");
                LocalDateTime now = LocalDateTime.now();
                String filename = text[0] + ".log";
                PrintWriter out = new PrintWriter(
                        new BufferedWriter(
                                new FileWriter(filename, true)));
                out.println(text[0] + "," + text[2] + "," + text[3] + "," + now);
                out.close();
            } catch (IOException e) {
                System.out.println("Writing error" + e);
            }
        }
        System.out.println("Client Connected and Start get I/O!!");
        System.out.println("==> Input from Client: " + cInput);
        System.out.println("Output to Client ==> \"Connection successful\"");
        output.writeUTF("Connection successful");
        output.flush();
        input.close();
        connection.close();
      } catch (IOException e) {}
```

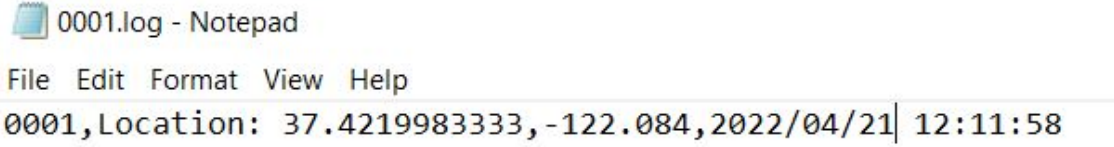
```java
void sentToServer() {
        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
        String P_ID = patientID_editText;
        String msg = "Patient:"+ P_ID + ", " + movementResult +", Location: "+ stringLa + "," +stringLo;
        Socket socket;
        DataOutputStream dataOutputStream;
        DataInputStream dataInputStream;

        try {

            socket = new Socket("192.168.1.32", 8000);
            dataOutputStream = new DataOutputStream(socket.getOutputStream());
            dataOutputStream.writeUTF(msg);
            dataInputStream = new DataInputStream(socket.getInputStream());

            Toast.makeText(MainActivity.this,dataInputStream.readUTF(),
                    Toast.LENGTH_LONG).show();
            dataOutputStream.close();
            dataOutputStream.flush();
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

Server part 2




```
if(text[1] == "falling"){
    try {
        DateTimeFormatter dtf = DateTimeFormatter
                .ofPattern("yyyy/MM/dd HH:mm:ss");
        LocalDateTime now = LocalDateTime.now();
        String filename = text[0] + ".log";
        PrintWriter out = new PrintWriter(
                new BufferedWriter(
                        new FileWriter(filename, true)));
        out.println(text[0] + "," + text[2] + "," + text[3] + "," + now);
        out.close();
    } catch (IOException e) {
        System.out.println("Writing error" + e);
    }
}
```

SMS

```
void sentSMS() {
        String P_ID = patientID_editText;
        String phoneNumber = phone_Edittext;
        String sms_msg = "Patient:"+ P_ID + ", " + movementResult +", Location: "+ stringLa + "," +stringLo;
        try {
            smsManager = SmsManager.getDefault();
            smsManager.sendTextMessage(phoneNumber, null, sms_msg, null, null);
            Log.i("Send SMS", "");
            Toast.makeText(getApplicationContext(), "SMS sent.", Toast.LENGTH_LONG).show();
        } catch (Exception e) {
            Toast.makeText(getApplicationContext(), "SMS faild, please try again.", Toast.LENGTH_LONG).show();
        }
    }
```

Thank you!