Project Documentation

Movement Recognition System

Team ID: 24

21 October 2021

Members

- **Hridita Nur Zaman Tiasha**, 170042041
- **Maksuda Islam Lima**, 170042063
- **Zubair Rahman Tusar**, 170042067

Proposed Project

A wearable device that will identify the abnormal/unhealthy movement of the subject (patient/elderly being) wearing the device. For the remaining short time of this semester, we have constrained ourselves to build a prototype of this project that can do binary prediction (e.g., sitting or not sitting).

Social Impact of the proposed project

Using this proposed embedded system, abnormal movement of patients can be recognized. This will be helpful for caregivers, nurses, and hospitals.

Motivation

Families where a single person is taking care of the elderly or types of patients who might not be able to call out for help every time. Often, the caregivers can be in another room or so, and the person might want to get out of the bed trying to make some movement when they are advised strictly not to move on his own. In such cases, the wearable device on their body will detect if the movement is alarming or not and send the caregiver a Notification.

Recent Work Related to the Proposed Project

A few of the most recent notable works are:

- In the paper, "Accelerometer-based intelligent system for human movement recognition" published in the 5th IEEE International Workshop on Advances in Sensors and Interfaces IWASI, the authors worked on an accelerometer-based approach to recognize movements where the embedded device is situated on the subject's

wrist. In this approach, the data collected from the accelerometer sensor is sent to a server where the recognition is done, and the result is generated. We want to do the recognition on the embedded device rather than on a remote server, an approach to

○ Reduce time delay in sending and receiving the data and results

○ Tackle the possibility of not getting results due to the server being down

Features & Necessary Equipment (Sensors)

Earlier after doing some research on the implementation of ml inference on embedded systems, we decided to work using Arduino Nano 33 BLE board for these reasons:

- Integrated 9 Axis inertial sensor
- Small enough to use in a wearable device
- 256KB of SRAM
- Library support for TinyML
- Available online resources

But, unfortunately, right now, this board is not available in Bangladesh. Even if we order it through Amazon or Ali Express will take a minimum time of 2 weeks. So, we decided to abort this and look out for alternative equipment.

The alternative equipment and sensors needed to implement the features that have been decided for this project till now are as follows:

- To get 9 axis inertial motion data which will be used as the data for recognizing the movement type, we will use the MPU9250 sensor.
- One of our goals is to make the device as small as possible in size to make it comfortable to wear. So we aim to use Raspberry Pi Zero W, which is small in size, to run the model.
- For the connections, we will use Breadboard and Jumper wires
- For debugging purposes, we will use LED lights
- We will need Resistors to maintain the required input voltages for different types of equipment.
- Initially, in our prototype, we want to display the type of activity on a 16x2 LCD Screen.

Description of Used Components

MPU 9250 Sensor:

The MPU-9250 is the world's smallest 9-axis MotionTracking device, has a reduced chip size and power consumption while at the same time improving performance and cost. It leverages an integrated accelerometer, gyroscope, and magnetometer.

Raspberry Pi Zero W:

Raspberry Pi Zero W is a Raspberry Pi Zero board that has been equipped with Wifi and Bluetooth. The Raspberry Pi Zero has a small shape but can still do a lot of things like the other large Raspberry Pi family.

Breadboard & Jumper Wires:

A breadboard is a rectangular board with many mounting holes. They are used for creating electrical connections between electronic components and single-board computers or microcontrollers such as Arduino and Raspberry Pi. The connections aren't permanent and they can be removed and placed again.
A jump wire (also known as jumper, jumper wire, jumper cable, DuPont wire, or cable) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype.

LED lights:

Easily connected on the breadboard. Mostly with the intention of debugging.

Resistors:

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust

signal levels, divide voltages, bias active elements, and terminate transmission lines, among other uses.

LCD Screen:

The LCD (Liquid Crystal Display) is a type of display that uses liquid crystals for its operation.

Workflow (with observation having experimental setup)

Our workflow can be divided into three phases:

1. Model Building
2. Hardware Preparation
3. Deploying the model on Raspberry Pi

Model Building

The steps for Model Building are briefly stated below

- Dataset Collection: The first task to build our model was to collect a dataset on which we will train our machine learning (ml) model.
- Our concerns to collect a dataset were:
    - A dataset consisting of accelerometer data
    - Data collected for Human Activities
    - Data collected from a wearable device which was placed on the wrist of the user
    - Data collected through such sensor/device which is similar to the sensors/devices we are using in terms of frequency, value range, etc
    - Data collected for normal users as, initially, we aim to test our prototype on normal users
- We trained our tree-based model on Google Colab

The difficulties we encountered were:

- The dataset that was hugely available either consisted of multiple sensor data from multiple sensors or the activity types were not convenient for our project like riding bike, car or playing, etc
- In some datasets, the data was collected using advanced sensors

Hardware Preparation

The steps for Hardware Preparation are briefly stated below

- R&D of the necessary hardware components
- Physically getting our hands on the components
- Assembly of sensors and the raspberry pi
- Installation of necessary drivers and OS to run the system
- Establishing connection to the system via local network
- Installation of necessary packages on the board to run our model
- Deployment of the model
- Real-time inference of the sensor collected data

We faced difficulties in almost all of the steps. To mention some are:

- In the R&D step the vast options, availability issues, etc were a bottleneck to our head start
- As mentioned earlier, we had to change our initial choice of hardware due to the availability issue and reorganize our plan
- Being a new learner in this field, to complete the assembly of the sensors and the raspberry we had to study a lot
- One of the most challenging tasks was to install the necessary ml packages. The availability of the packages for raspbian OS was an issue. Also, to install a basic package like pandas the time took was huge (About 3-4 hours)

Deployment

After training our model on Google Colab and preparing the similar environment in our raspberry pi (with all the necessary libraries and packages) we had to run our model on the raspberry pi where we faced some major problems:

- In google colab, the model was trained on a 64-bit environment but raspberry pi runs on 32-bit architecture.

- The raspberry pi itself did not have enough resources (i.e. RAM, CPU, etc) to train the model.
- We had to prepare a 32-bit environment on a separate system
- Then, retrain our model in that environment and finally deploy the model on raspberry pi

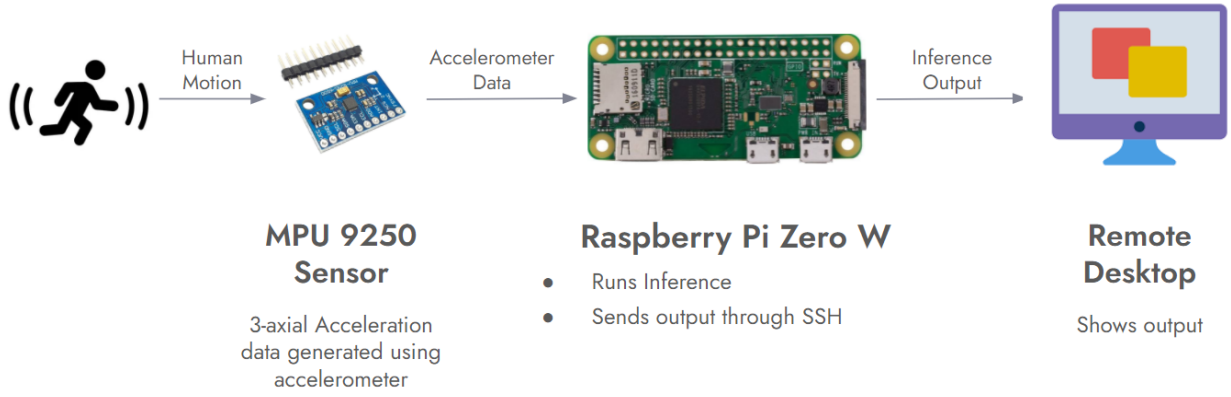

Figure: Workflow

Description of how a user can use the system along with the Video Walkthrough

- Connect the system using a type 2.0 USB cable to any power brick that provides 5 volts. (Any usual smartphone charger will do)
- Connect with the system to a remote desktop through the local network using Putty SSH software
- Run the command "python maindfv4.py" to start the process of inference

Video Link: [Movement Recognition System]
GitHub Link: [movement_recognition_system]