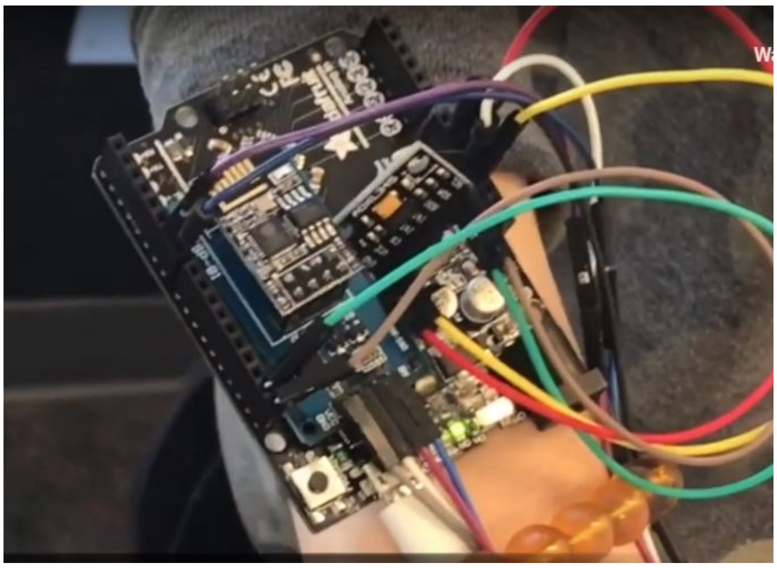
Fall Detection and Classification for People with Dementia

Keep track of the movements of people with dementia via wearable accelerometer, and send corresponding remote alerts to caregivers if falls are detected.

Xinmeng Li - Collect and preprocess data, design and implement classification.
Ziwei Chen - Build circuit, data I/O between driver program and user interface.



Prototype

Abstact

Why are fall detection robots needed for people with dementia? Falling and fall related problems are common with people with dementia due to their "physical weakness, gait changes and poor balance" (Heerma 2019). Moreover, after falling, people with dementia are not able to cognize the falling due to cognitive impairment. Thus, fall detection robots are needed for people with dementia.

We aim to design a wearable robot that detects three types of falling for people with dementia, and once a fall is detected, the robot will send instant alerts to the caregivers so that they could take appropriate actions in time especially when they are not together with the people with dementia.

Our design is to use an accelerometer to detect motions and use a WIFI connector to send alerts to the webpage (a simulation of the caregiver's cellphone). Our working algorithm for this program is to calculate the standard deviation from the training sample (the intentional falling by Ziwei and Xinmeng), then determine the threshold in order to classify three types of falls.

Our results suggest that our classifiers predict the training dataset with a higher accuracy and predict the test dataset with a higher error. We conclude from this outcome that the hardware is not being accurate and lack of enough data points for both training and test data.

In the future, to improve the algorithm, reinforcement learning is a promising approach. Given the fact that individual differences exist extreme motion, and our algorithm on

classifying may have deviations, our ultimate goal is to detect the type of fall accurately for every user. Thus,by applying reinforcement learning, the robot can learn from previous decisions on classifying, so that it can adjust algorithm parameters each time for wrong classified types and make more accurate and precise decisions in the future.

1. Introduction

Falling and fall-related problems are of great concern to people with dementia. The reason is that the risk of falling and sequelae from falling increase as age rises, whereas dementia is a symptom that is more common with elderly people. Also, people with dementia have severe cognitive impairment, so they are not able to cognize that they have fallen down.

After talking to Amy Abrams, the stakeholder of people with dementia, she gave the suggestion that we could design a wearable healthcare robot that is able to detect the falling of people with dementia, to enable the caregivers of people with dementia to receive instant alerts with the falling so that they could take efficient actions even if they are not in the same room.

Focus on this topic, our goal is to design a wearable robot that is able to detect three classes of falling (fall from standing, fall from lying and fall from lying). Once a fall is detected, the robot sends immediate alerts to the caregivers so that the caregivers could take appropriate actions in time. The reason we choose to make the design more precise and accurate on classifying falls rather than detecting falls in general is that we want the caregivers to be well-acknowledged of the incident before rushing to help people with dementia, so that appropriate actions could be taken in time.

The reason that the robot is designed for caregivers to receive instant alerts instead of people with dementia calling for help after falling is that people with dementia have severe cognitive impairment, they would not even realize that they have fallen down, and even if they would realize the fall-down, they may pass out after the falling so that it is too late for them to call for help.

Our design for the fall detection robot contains two major hardware parts, one accelerometer to detect motion of people with dementia and one WIFI connector to send appropriate alerts to the caregivers. As shown in the graph, the falling is an extreme motion in a very short time interval (Jia, et al. "Detecting Human Falls with a 3-Axis Digital Accelerometer"), thus, we decided to use standard deviation to set up different thresholds for different types of falling.

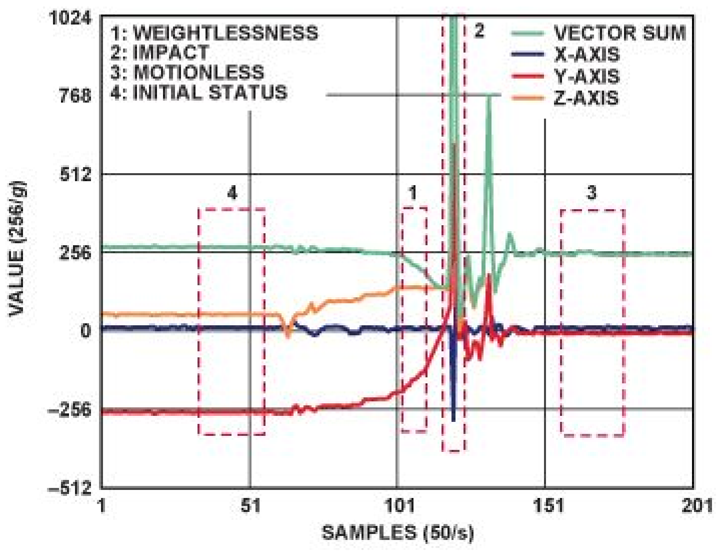


Figure 1. Acceleration change curves during the process of falling.

2. Methodology

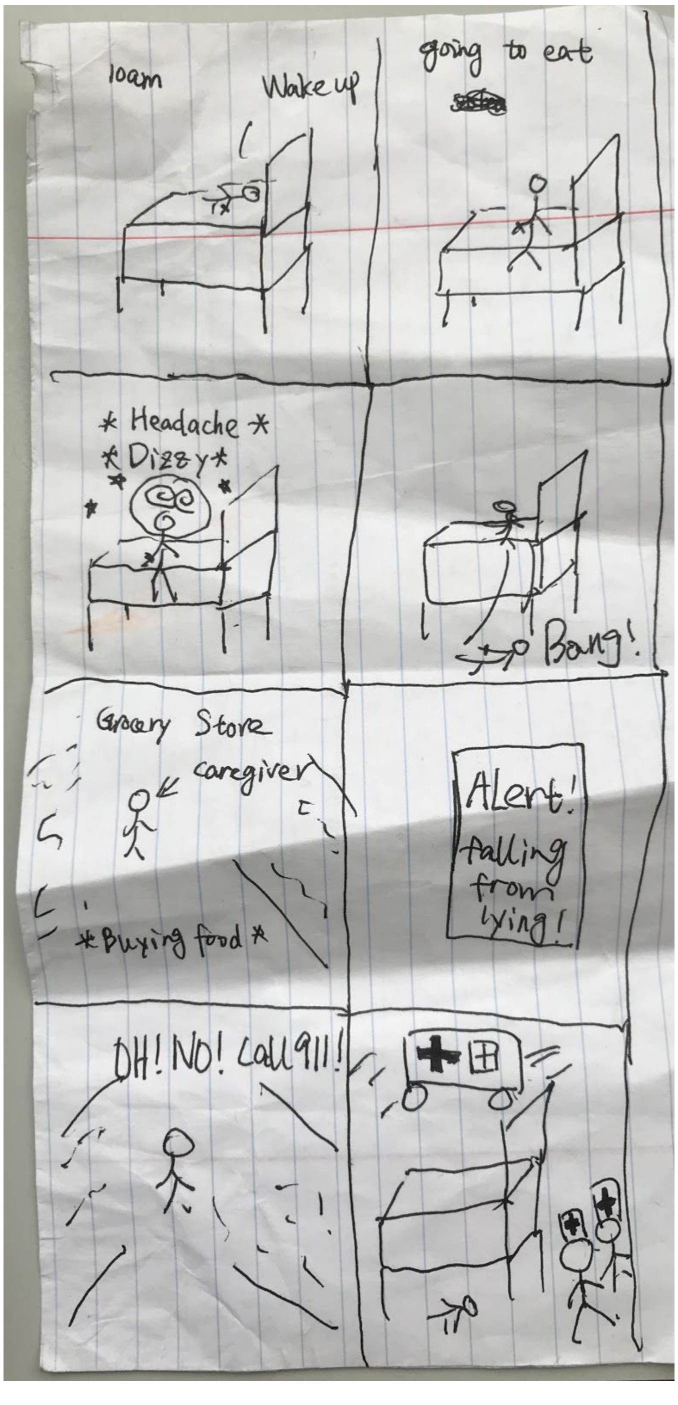
2.1 Project design process.

Our team changed the project idea many times. In the beginning of brainstorming, Ziwei and I have decided to build a device that aims at assisting people with dementia in their daily life. Therefore, we first focused on the loss of orientation and memory of people with dementia, and planned to build a reminder to prompt users to wear GPS tracker to solve the problem of wandering. We then discussed our initial idea with professor and our stakeholder Amy. Professor suggested us to add more functions to our device, and Amy pointed out that people with dementia usually have severe cognitive impairment, which makes it difficult for them to follow all of the instructions of the reminder on their own.

Amy suggested us to transfer our target from people with dementia to their caregivers, who have normal cognitive functions but cannot afford the workload of memorizing all of their tasks. For example, a person with severe dementia is not aware of dehydration, so the caregiver is supposed to prompt him or her to drink every hour. If the caregiver forgets to complete this task, the person with dementia might get dehydrated. Amy said that a reminder containing daily tasks such as going to the restroom or medication customized for caregivers will be meaningful. We then considered building a reminder robot for daily activities and wandering. However, we realized that caregivers have the ability to set up reminders by themselves on a smartphone, which makes it unnecessary for us to design a reminder.

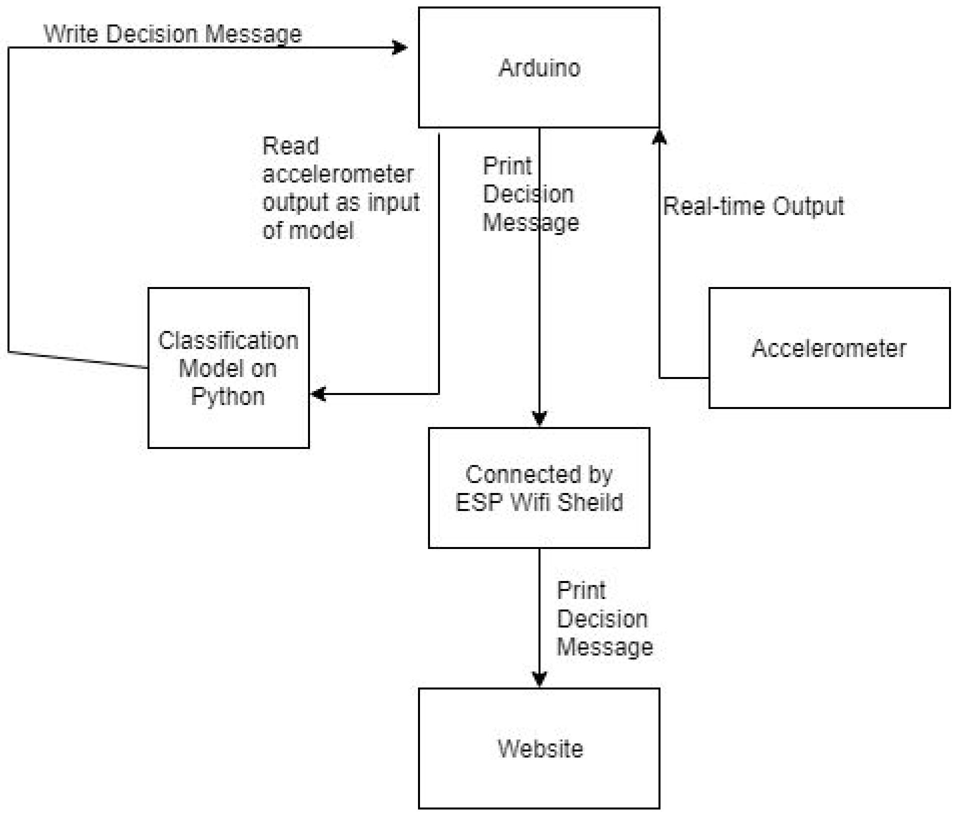
Consequently, we reconsidered the difficulties and challenges that people with dementia always face, and changed our topic to fall detection. Compared with the rest of the products, our project has the following strengths: 1. Most of the products relative to the falling problem of people with dementia in the market target people with dementia and attempt to protect them from getting injured physically, such as non-slippery shoes. Our device is designed for caregivers and attempts to prompt caregivers remotely to assist in people with dementia, who probably lack the capability of seeking help on their own when they fall down. 2. Besides to determine whether the user of our wearable device has fallen down, our model will also classify the falling into various categories. Thus, caregivers will be provided more information about the current situation, which is valuable for them to estimate whether this is an emergent and severe falling and take actions correspondingly. 3. Medical devices are often complained to be too expensive to be afforded by customers. However, the hardware materials of our project are economical. As a result, our device is a financially affordable option for the majority of caregivers of people with dementia.

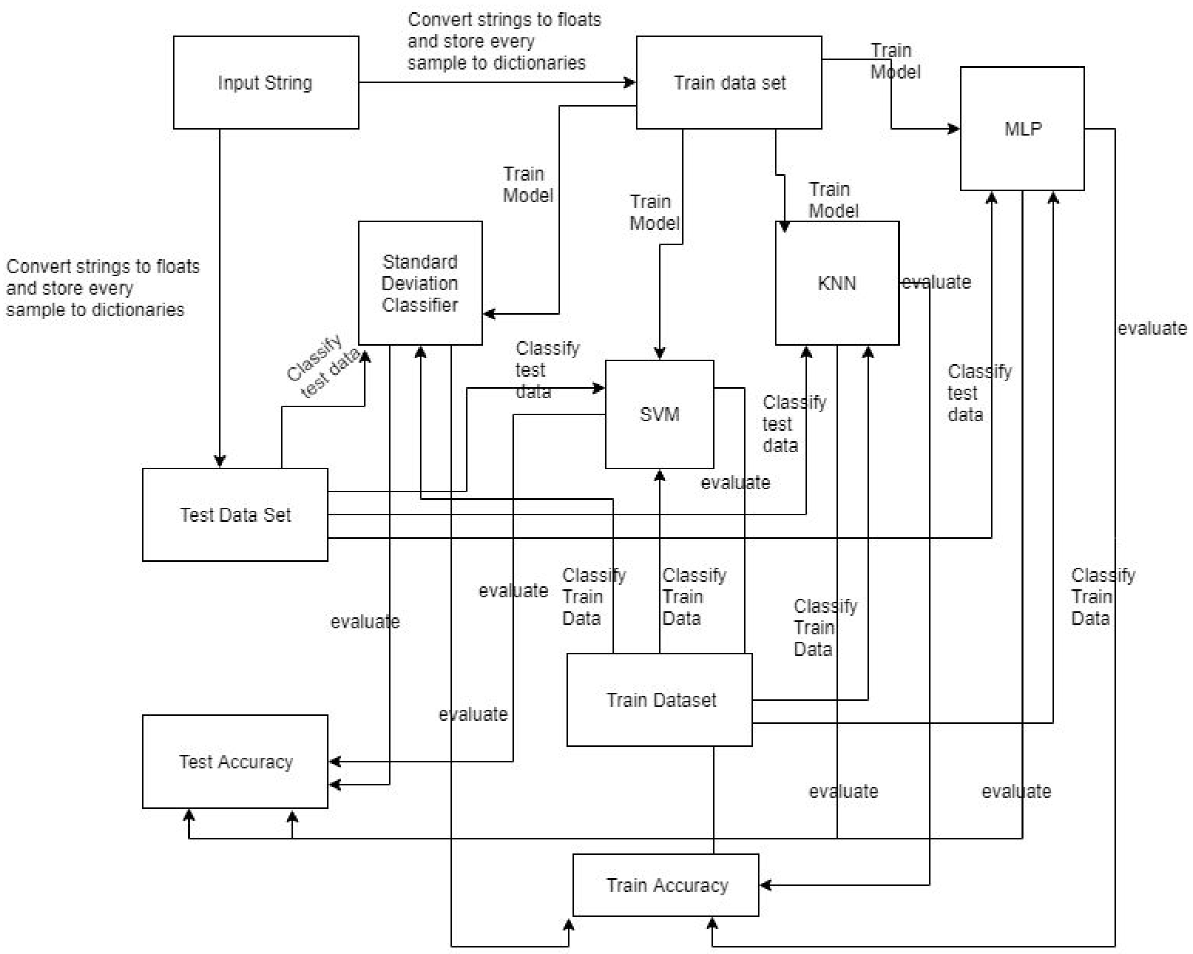
Our storyboard:

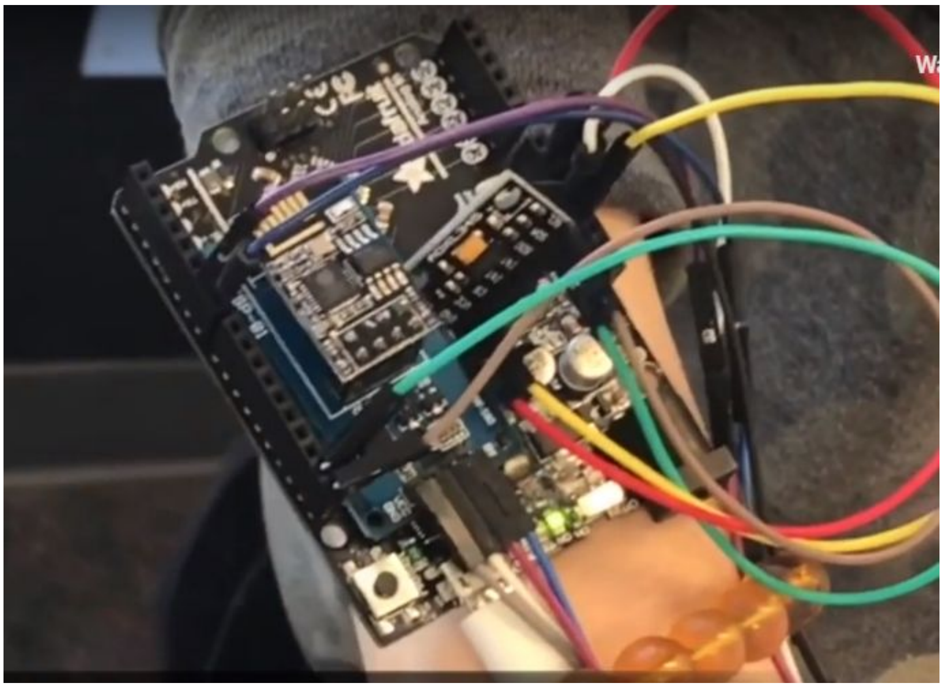


Our initial idea of the process is to read accelerometer output on Arduino, classify the data and then print the decision message on a website by ESP Wifi shield.

2.2 High level hardware and software diagram overview







2.3 Description of the technical graph

Our project completes the fall detection not only via integrating the Wifi shield ESP01, the accelerometer ADXL345 and the arduino board together, but also via doing extensive data analysis and comparing the performances of various models.

2.3.1 From the technical aspects:

First, the Wifi shield build a connection to a certain IP address based on ssid and password of the current network.

Next, the acceleration value on x,y and z axes are measured through the accelerometer and are read every one second. Here we choose the frequency to be 1 read per second, because if the frequency is higher, the system will get memory and power issues. This is one of the hardware limitations of our projects. If we were allowed to shorten the time interval between two reads, we would collect more detailed and valuable acceleration data. After the arduino application reads the acceleration data from the accelerometer, it prints these data to the Serial.

The python code on Jupyter Notebook then reads the output that Arduino application just printed via the PySerial Package. It extracts sample values from the output string, converts them to float, and then stores the data into dictionaries for different classes. After going through the analysis and classification elaborated in 2.3.2, the python model makes a decision of the fall category based on the input data, and writes the corresponding message back to Arduino.

Arduino reads the output from python, and then prints the alert message on the connected IP address. Therefore, caregivers are able to get access to the alert remotely as long as they reload the IP address.

2.3.2 From the experiment design aspect:

Data Collection: We collect data by simulating falling by ourselves, so the number of subjects is two. To reduce the bias, we should simulate various types of movements for each class instead of repeating a single motion. For example, users might walk, eat or brush their teeth when they are not falling down. In these scenarios, the acceleration data will also have fluctuation, but the pattern should be distinguished from patterns of three types of falling. To increase the reliability and stability of our model, it is necessary for us to take the generalization of our model into consideration. Due to time and location limitations, our simulations occur in various locations and time. There are four classes in total: fall from standing, fall from sitting, fall from lying and not fall. For each class, 15 training samples and test samples are collected. We then manually label these samples with their classes, in order to get the accuracy later. Each sample has the acceleration data from 3 axes including x,y,z, and has 20 points in total, which represents 20 seconds. There are four dimensions for each sample: x, y, z, and the combined one xyz (i.e. concatenate x,y,z with each other).

Data Preprocessing: Check whether the data has noise, and whether the fall has an obvious pattern on all of the four dimensions.

Data Analysis: To explore and discover the underlying pattern, we calculate the mean and median of standard deviations and distance between maximum and minimum called d across samples for each class and dimension. Randomly select a dimension and drawing the s and d across samples for each class will also provide us some hints. We then compare these results to select the more indicative one from s and d.

Model Selection:
The first model will depend on the result of data analysis. If we believe that either s or d or both is indicative, we will set a threshold for this feature to classify the input data. The second model is SVM, which is a representative model and the report[4] of Jessica and Binghai also mentions this algorithm. Another model that has been discussed in this report is K nearest model, which is a common non-linear model. The last model is a basic multilayer perceptron.

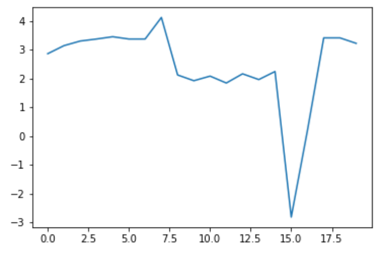
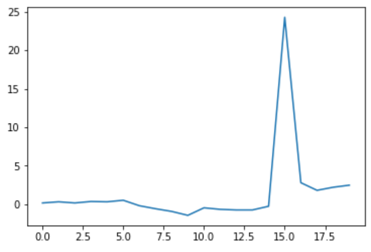
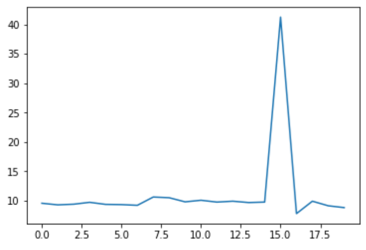
Performance Comparison: Evaluate these four algorithms on both train and test dataset, and select the best one from them.

3.4 *Usability-testing methodology*
Preprocessing before testing: use Xinmeng’s intentional falling data as training data to build the classifier, which classifies three types of falling.

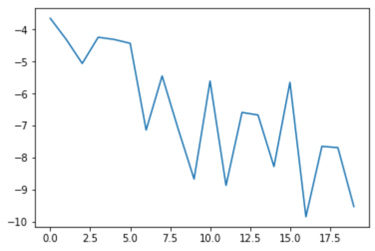
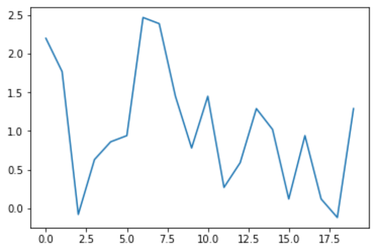
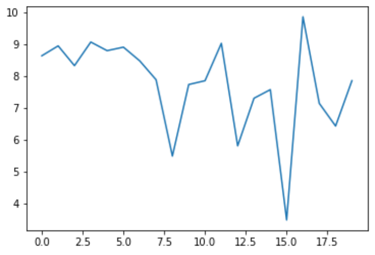
Module	Testing Method	Metrics
Predict correct type of falling on Xinmeng’s (training data)	Run algorithm on each data point and compare the actual falling type with expected falling type.	Accuracy
Predict correct type of falling on Ziwei’s (test data)	Run algorithm on each data point (different than the training data) and compare the actual falling type with expected falling type.	Accuracy

3. Results

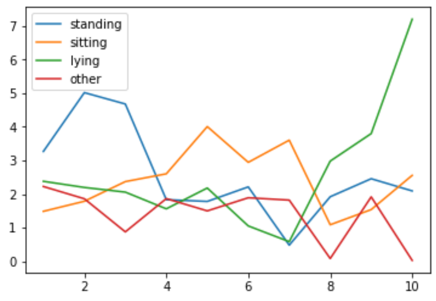
The fall often has an obvious peak across 20 points and does not have much noise. Thus, it is not necessary for us to clean the data.
Below figures show the acceleration values on the x,y,z axis of the first sample of falling from sitting.

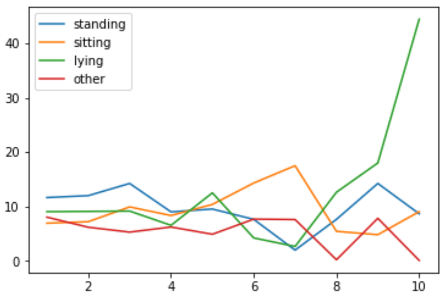




The not fall sample does not have a single obvious peak, and sometimes fluctuates. Below figures show the acceleration values on the x,y,z axis of the second sample that do not fall.

We compare the values of distance between maximum and minimum(right image) and standard deviation(left image) across samples and find out that the standard deviation of different classes are more separate from each other than the distance between maximum and minimum. Below images are the values d and s of the first ten samples on the x axis.





The training accuracies are Std: 0.43636, KNN: 0.69091, SVM: 1, and MLP: 1. The test accuracies are Std: 0.083333, KNN: 0.5, SVM: 0.25, MLP: 0.33333. We observe that although MLP and SVM perform well on training data, they are not good at adjusting models to new test data. It might be because of the deficiency of our training data, which makes it difficult for the model to increase their complexity. Moreover, KNN is very stable at the classification for both training and test dataset. The low training accuracy indicates that KNN is not good at learning from data. The not bad test accuracy shows that KNN is relatively stable, probably because its algorithm utilizes the information of surrounding points and it is impossible for the prediction to be ridiculous. The Std, as we expected, performed poorly, since it is too simple and would be hard to generalize.

The below form shows how many falls have been correctly classified for 3 test cases for each fall type. We observe that our model performs well on detecting falls, but performs badly on classifying them into different groups.

	Standing fall	Sitting fall	Lying fall	Fall	No fall
Std	0	1	0	7	0
KNN	0	1	2	7	3
SVM	0	3	0	9	0
MLP	0	1	0	5	3

4. Discussion

Our algorithm could successfully classify three types of fall: fall from standing, fall from sitting (when attempting to stand up) and fall from lying (rolling down to the ground while sleeping) and send immediate alerts to the caregivers. With a limited dataset, we did expect a high rate of test errors. Since the extreme motion depends on personal differences, the threshold calculated from the standard deviation that is used to classify types of falls varies from person to person. Therefore, the prediction of falling classifiers may not be accurate when switching from user to user. Moreover, our team also meets hardware limitations. For example, the arduino board is always disconnected when we simulate the falls (which might be because the range of our motion is large), so we have to repeat multiple times to get a perfect sample. The arduino can only read data from the accelerometer and write to python once per second. With a short delay time, such as delay(500), the system dysfunctioned. However, the falling always occurs in a really short range of time, and the acceleration change in a couple seconds is usually the most important information for us to make classification. This hardware limitation restricts the information sources for us. Obviously, with only two of us, we can only collect a small amount of data, which makes the impact of outliers really large, and makes it difficult for the models to discover patterns.

Our ultimate goal is to detect the type of fall accurately for every user. To achieve this, reinforcement learning algorithms are the development direction of the future logistics. Each time a robot detects a fall, the robot system will remember the decision it makes and keep track of whether it is a right or wrong decision. The parameters should be justified when making a wrong decision, then the next time the robot makes a decision, it will be a more accurate one than the wrong one before. By repeating the process, this reinforcement learning algorithm enables the fall detection to be more customized to each individual user so that the accuracy increases based on individual motions. It will also be helpful to use other sensors, in the human movements dataset of UCI[2], features such as the angle change are used, which will add more perspectives to the fall detection and classification.

5. References

1. Heerema, Esther. “Understanding Why People With Dementia Fall Can Help Reduce the Problem.” Verywell Health, Verywell Health, 23 Jan. 2019, www.verywellhealth.com/causes-of-falls-in-people-with-dementia-98558.
2. “Human Activity Recognition Using Smartphones Data Set .” *UCI Machine Learning Repository: Human Activity Recognition Using Smartphones Data Set*, archive.ics.uci.edu/ml/datasets/human activity recognition using smartphones.
3. Jia, Ning, et al. “Detecting Human Falls with a 3-Axis Digital Accelerometer.” Detecting Human Falls with a 3-Axis Digital Accelerometer | Analog Devices, www.analog.com/en/analog-dialogue/articles/detecting-falls-3-axis-digital-accelerometer.html.
4. Ling, Binghai, and Jessica Moore. *Human Activity Recognition Using Smartphone Sensors*. 16 Dec. 2016, cs229.stanford.edu/proj2016/report/LingMoore-HumanActivityRecognitionUsingSmartphoneSensors-report.