FALL DETECTION WITH NAÏVE BAYES

Student: Po-Teng Tseng
Advisor: Dr. Anne Hee Hiong Ngu

Outline

- Introduction
- Methodology
- Data Collection
- Data Processing
- Model Generation
- Work Flow
- Experiment (3 types)
- Conclusion

Introduction

- This independent study explores the use of streaming accelerometer data from a commodity based smartwatch device to detect falls.
- The majority of current fall detection applications require specially designed hardware and software, which make them expensive and inaccessible to the general public.
- We collected 270 simulated fall data from 7 different people when they fell on a mattress with different styles.
- We established the baseline accuracy for fall detection that can be achieved by using Naïve Bayes and experimented with different factors that can be used to improve the baseline accuracy.

Methodology

- Collect data with smartphone and smartwatch
  - Android phone and Microsoft Band 2
  - Volunteer fall on a mattress while wearing the watch
- Generate model with collected data
- Classify new data by generated model
- Count consecutive result to decide if there is a fall (heuristic function)
  - A way to predict time series condition while using a point-by-point model

Data Collection

- Implement a mobile app to record data from smartwatch and smartphone
- The mobile app has a button which will mark all incoming data as fall when pressing
  - Less labor intensive
  - The good timing of pressing needs practice to achieve
- 270 falls were collected from 7 different people to increase diversity
- Data frequency is 32ms

Data Processing

- R script to process data.
- Resultant acceleration

$$\sqrt{A_x^2 + A_y^2 + A_z^2}$$

- Smin
  - The minimum acceleration in a period of time.
- Smax
  - The maximum acceleration in a period of time.
- Cvfast
  - The resultant difference between SMax and Smin in three directions

Model Generation

- Implement a Java program using Weka's library.
  - Generate the model
- It can also
  - Run simulation
  - Give static result

Work Flow

- Raw data

```
linear_accelerometer_x, linear_accelerometer_y, linear_accelerometer_z, ms_accelerometer_x, ms_accelerometer_y, ms_accelerometer_z, ms_gyros
-0.15978903, 0.022526503, 0.24414253, -0.06323242, 0.6206055, -0.81591797, null, null, null, null, null, null, null, null, null, null, null,
-0.15978903, 0.022526503, 0.24414253, -0.06982422, 0.63525397, -0.81835943, null, null, null, null, null, null, null, null, null, null, null
-0.06706977, 0.010727406, 0.031069757, -0.06665039, 0.62133795, -0.81152344, null, null, null, null, null, null, null, null, null, null, nul
-0.10940784, 0.007859707, -0.017905235, -0.06665039, 0.62133795, -0.81152344, null, null, null, null, null, null, null, null, null, null, nu
-0.084400356, 0.005696535, 0.03644848, -0.063964844, 0.623291, -0.81518555, -0.063964844, 0.623291, -0.81518555, -9.573171, 47.43903, -51.40
-0.03397715, 0.029271604, 0.11166, -0.06762695, 0.62402344, -0.81909186, -0.06762695, 0.62402344, -0.81909186, 0.39634147, 2.012195, -0.2134
-0.026883366, 0.037628416, 0.08444787, -0.072753906, 0.6169434, -0.8198243, -0.072753906, 0.6169434, -0.8198243, 0.18292683, 2.1036587, -0.9
0.0716272, 0.040759563, -0.016096115, -0.07104492, 0.6245117, -0.8251953, -0.07104492, 0.6245117, -0.8251953, 0.57926834, 2.0731707, -1.0365
0.04347724, 0.023989916, 0.08808613, -0.06567383, 0.6198731, -0.8156738, -0.06567383, 0.6198731, -0.8156738, -0.24390246, 1.4939024, -1.1585
0.020094158, -0.009230137, 0.071772575, -0.06933594, 0.6201172, -0.81176764, -0.06933594, 0.6201172, -0.81176764, 0.27439025, 1.2804878, -1.
0.056479454, 0.017471075, -0.0073165894, -0.07373047, 0.6169434, -0.81176764, -0.07373047, 0.6169434, -0.81176764, -0.21341464, 0.9146341, -
0.09066541, 0.043016195, -0.0956173, -0.07421875, 0.6164551, -0.8120117, -0.07421875, 0.6164551, -0.8120117, -0.15243903, 0.9146341, -1.1585
0.11378306, 0.03278756, -0.077561386, -0.07348633, 0.61157227, -0.8137207, -0.07348633, 0.61157227, -0.8137207, 0.15243903, 0.73170733, -1.2
0.11718017, 0.022707224, -0.06947899, -0.07495117, 0.6098633, -0.82177734, -0.07495117, 0.6098633, -0.82177734, 0.48780492, 0.57926834, -0.8
```

- Processed data (with labeling)

```
"resultant","cvfast","smax","smin","outcome"
1.07897774215205,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
0.954836730418577,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
0.973214327930463,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
0.973214327930463,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
0.978561333521144,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
1.01840589659131,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
1.01840589659131,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
1.01840589659131,0.644385222430007,1.07897774215205,0.94844720398065," notfall"
1.02169065790228,0.182779344275114,1.02884833417595,0.94844720398065," notfall"
1.01354439072987,0.182779344275114,1.02884833417595,0.94844720398065," notfall"
1.01354439072987,0.182779344275114,1.02884833417595,0.94844720398065," notfall"
1.01354439072987,0.182779344275114,1.02884833417595,0.94844720398065," notfall"
1.02884833417595,0.182779344275114,1.02884833417595,0.94844720398065," notfall"
1.01887515402743,0.182779344275114,1.02884833417595,0.94844720398065," notfall"
```

Work Flow (continued)

- The original data
  - Data is collected, processed and then used to train the model
- The new data
  - Data is collected, processed and then classified by model
  - Count consecutive positive results
  - The thresholds which decide the range of a fall is important and may vary with different version of model

```
0.629371,6.416013,3.86924,0.402126,' notfall'
0.982495,6.416013,3.86924,0.402126,' notfall'
0.894114,6.416013,3.86924,0.402126,' notfall'
1.029598,6.416013,3.86924,0.402126,' notfall'
1.403548,6.416013,3.86924,0.402126,' notfall'
1.009324,6.188228,3.86924,0.402126,' notfall'
3.86924,6.188228,3.86924,0.402126,' fall'
2.767565,6.188228,3.86924,0.402126,' fall'
3.816125,6.188228,3.86924,0.402126,' fall'
1.996147,6.188228,3.86924,0.402126,' fall'
1.873868,6.188228,3.86924,0.402126,' fall'
1.873868,6.188228,3.86924,0.402126,' fall'
0.56803,6.188228,3.86924,0.402126,' notfall'
1.134361,1.936517,1.701014,0.402126,' notfall'
1.170538,1.936517,1.701014,0.402126,' notfall'
0.402126,1.936517,1.701014,0.402126,' notfall'
1.486898,1.936517,1.701014,0.402126,' notfall'
```

Heuristic function

```
Int count = 0;
Boolean flagFall = false;
If (result_instance == fall) {
        count++;
}
else if (count is in the range of threshold){
        flagFall = true;
        count = 0;
} else {
        count = 0;
}
```

Experiment

- Run model against test data to get labeled test data, and then count the consecutive positive results to decide if there is a fall.
- Fall and ADL should be tested separately
  - Pure full test
  - Pure ADL test
- 2/3 of collected fall data was used as train data for Naïve Bayes model.
- 1/3 of collected fall data was used as test data for pure full

Experiment (continued)

- 1. Baseline version
  - The first version of model only trained with fall data
- 2. Improved version by adding ADL data into training data
  - Relate with future plan which automatically collects false positive data from the users and re-trains the model
- 3. Improved version by using heuristic function on top of whole system to check phone acceleration
  - If the user falls and the phone is in the user's pocket, there should be acceleration happening

Experiment – baseline

- Train data
  - 2/3 of fall data used as train data for Naïve Bayes model.
- Test data (pure fall)
  - 1/3 of fall data used as test data for pure full
- Test data (ADL)
  - Quick sitting
  - Waving
  - Throwing
  - Jogging

Result (Experiment – baseline)

- Pure fall

Threshold	Detected fall	Accuracy
6~50	75/90	83.33%
5~50	77/90	85.56%
4~50	82/90	91.11%
3~50	85/90	94.44%

Result (Experiment – baseline)

- Pure ADL

Threshold	False positive	Accuracy
6~50	1/90 (0w 1j)	98.89%
5~50	4/90 (0w 4j)	95.56%
4~50	14/90 (2w 12j)	84.44%
3~50	23/90 (7w 16j)	74.44%

Result (Experiment – baseline)

- Overall

Threshold	Overall Accuracy
6~50	91.11%
5~50	90.56%
4~50	87.78%
3~50	84.44%

Result (Experiment – ADL improved)

- Train data
  - 2/3 of fall data
  - Additional ADL data
- Test data (pure fall)
  - 1/3 of fall data used as test data for pure full
- Test data (ADL)
  - Quick sitting
  - Waving
  - Throwing
  - Jogging

Result (Experiment – ADL improved)

- Pure fall

Threshold	Detected fall	Accuracy
6~50	74/90	82.22% ▼
5~50	76/90	84.44% ▼
4~50	80/90	88.89% ▼
3~50	84/90	93.33% ▼

Result (Experiment – ADL improved)

- Pure ADL

Threshold	False positive	Accuracy
6~50	1/90 (0w 1j)	98.89%
5~50	3/90 (0w 3j)	96.69% ▲
4~50	9/90 (0w 9j)	90.00% ▲
3~50	18/90 (2w 16j)	80.00% ▲

Result (Experiment – ADL improved)

- Overall

Threshold	Overall Accuracy
6~50	90.56% ▲
5~50	90.56%
4~50	89.44% ▲
3~50	86.67% ▲

Result (Experiment – phone improved)

- Same train data and test data as ADL improved experiment.
- Add a heuristic function on top of the whole system to double check if a predicted fall is a real fall.
  - Check resultant acceleration from phone
  - Set (resultant acceleration > 5.0) as the condition

Result (Experiment – phone improved)

- Pure fall

Threshold	Detected fall	Accuracy
6~50	74/90	82.22%
5~50	76/90	84.44%
4~50	80/90	88.89%
3~50	84/90	93.33%

Result (Experiment – phone improved)

- Pure ADL

Threshold	False positive	Accuracy
6~50	1/90 (0w 1j)	98.89%
5~50	2/90 (0w 2j)	97.78% ▲
4~50	8/90 (0w 8j)	91.11% ▲
3~50	14/90 (0w 14j)	84.44% ▲

Result (Experiment – phone improved)

- Overall

Threshold	Overall Accuracy
6~50	90.56%
5~50	91.11% ▲
4~50	90.00% ▲
3~50	88.89% ▲

Best combination

- Improved version with phone acceleration

Threshold	Overall Accuracy	Fall Accuracy
4~50	90.00%	88.89%
3~50	88.89%	93.33%

Conclusion

- Adding more ADL data does improve the performance of predicting ADL, but it's a tradeoff which may weaken the ability of predicting fall.
- Considering phone acceleration on top of the prediction can filter out more ADLs, which reduces false positive rate. Waving can be filtered out well.
  - Allowing us to set threshold [3, 50] while still maintaining decent overall accuracy
- The jogging ADL is hard to perfectly handled. If the user can avoid jogging, the app will have a very good accuracy.