Fall Detection using Wearable Sensor Data

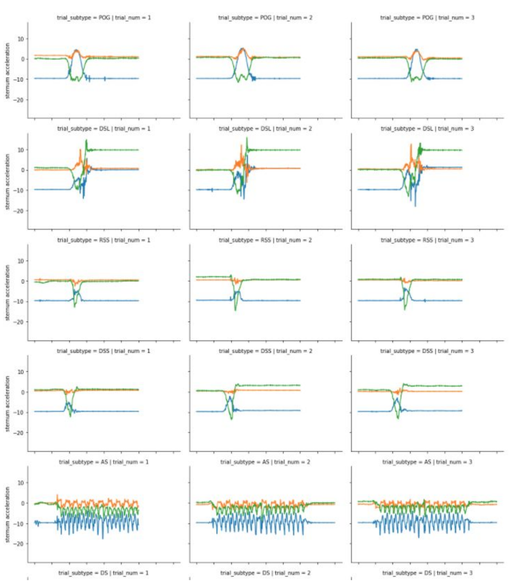
Gustavo Felhberg, Jorge Marcano and Muhammad Muhaimin
Professional Master's Program in Big Data - Simon Fraser University

The Problem

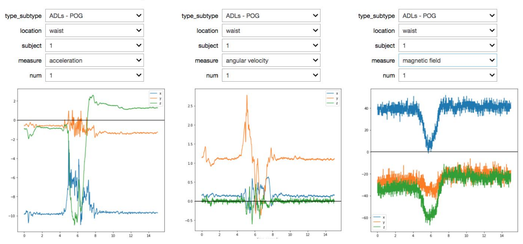
- Falls represent the second leading cause of accidental or unintentional injury deaths among older adults worldwide
- SFU's Research: "A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials"
- Simulations: 7 types of falls, 5 near falls and 8 activities of daily life. 10 participants simulating 3 trials for each category, resulting in 600 trials
- Our goal:
  - Analyze the data produced by the sensors for interesting information
  - Detect Falls using Machine Learning Algorithms trained with combinations of other sensors
  - Simulate the use of real time data

Data Analysis

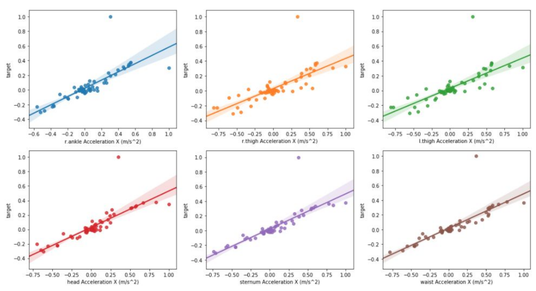
Trials by Type



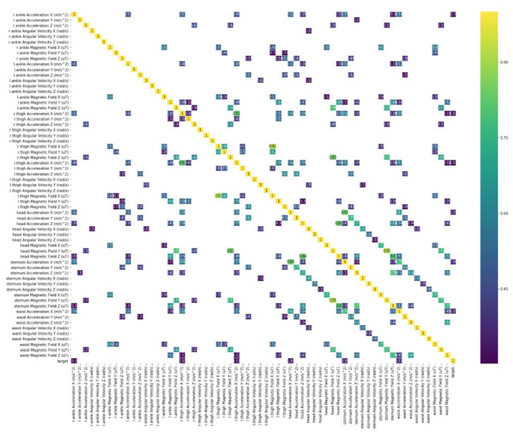
Trials Comparison

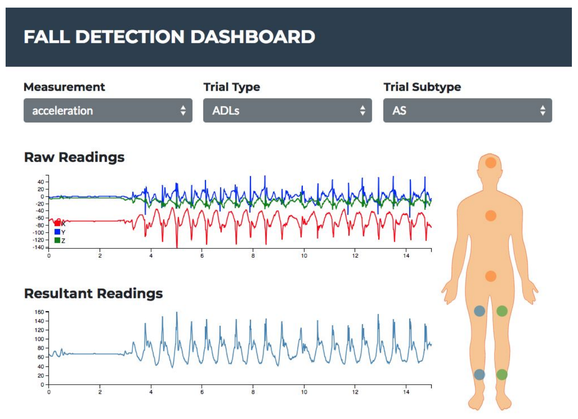


Most Correlated Features with Target



Pearson Correlation

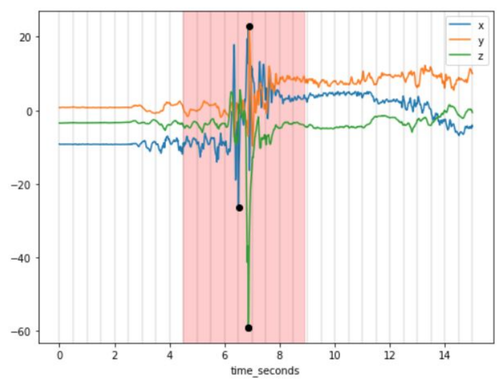




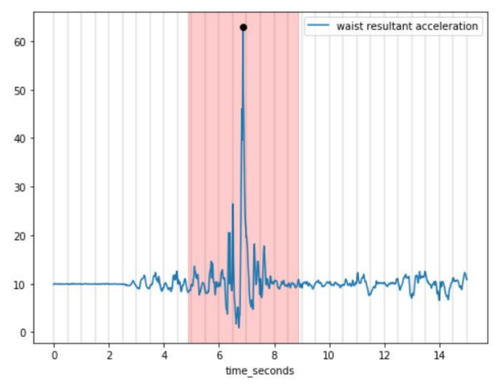
Machine Learning

- There isn't a definition of a fall agreed upon by different researchers
- We know it has an increase in acceleration when it begins and a "landing" phase where the acceleration flattens
- Based on this, different researchers have tried different methods to preprocess data
- We try 4 different preprocessing methods and show results for each of them
- 3 of the methods use SVM models and the other one uses Convolutional Neural Networks
- We used techniques like adding class weight, oversampling, undersampling etc to address imbalance in Fall vs Non Fall (1:10 ratio.) in training data
- Each method uses certain body parts and metrics to train with as well as a specific body part to use when finding the acceleration peaks
- SVM Parameters : RBF kernel, kernel coef 0.01, degree 3 for the RBF
- CNN Parameters : 50 filters, filter size 3, ReLU activation in the Cov layer, 1 Maxpooling, 0.01 LR in the Dense layer with a sigmoid activation function
- Using subjects 6-10 for training and 1-5 for testing

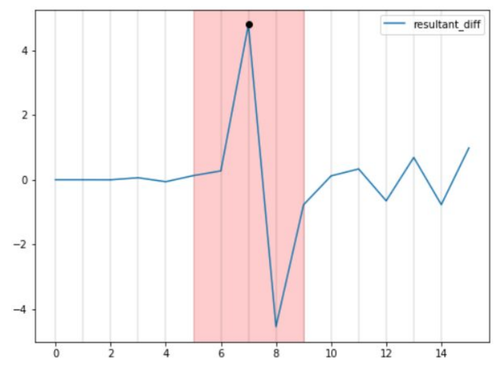
1. Overlapping window:
1.1. Drop resultant columns
1.2. Group the data in 0.5 second windows
1.3. Find the absolute peak acceleration of each axis
1.4. Create the window starting from the smallest absolute axis to the maximum absolute axis to consider a fall and the rest as non-falls



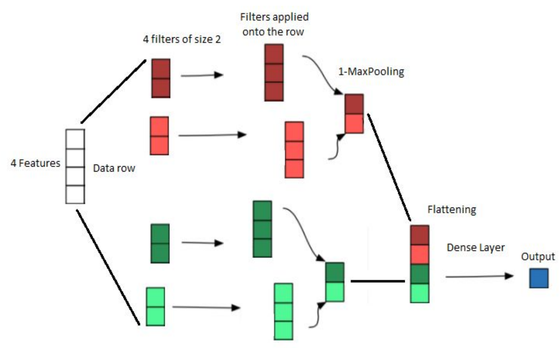
2. Resultant acceleration peak window:
2.1. Group data in 0.5 second windows
2.2. Find the peak resultant acceleration and consider the peak plus the window of 4 seconds around it as falls, the rest as non-falls



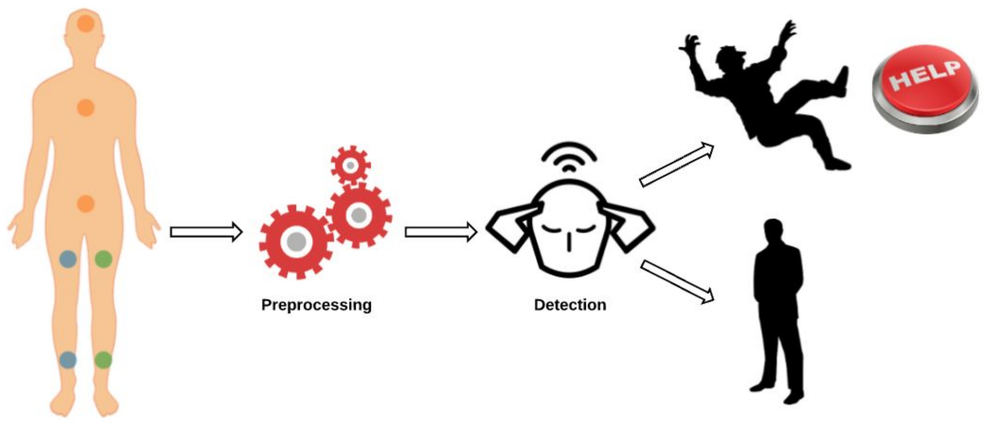
3. Resultant difference windows:
3.1. Group data in 1.0 second windows
3.2. Calculate the resultant acceleration mean difference between each pair of consecutive rows
3.3. Find the peak resultant difference and consider that plus the 4 second window around it as fall, the rest as non-falls



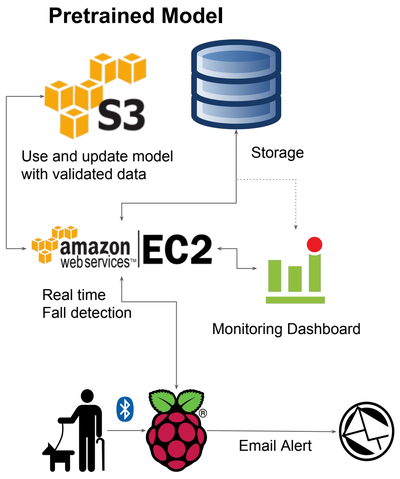
4. Convolutional Neural Networks:
4.1. Find the peak resultant acceleration and label the peak and the window of 4 seconds around it as falls and the rest as non-falls
4.2. Transform data into numpy arrays and reshape it to fit the CNNs



Real Time Fall Detection



Pretrained Model

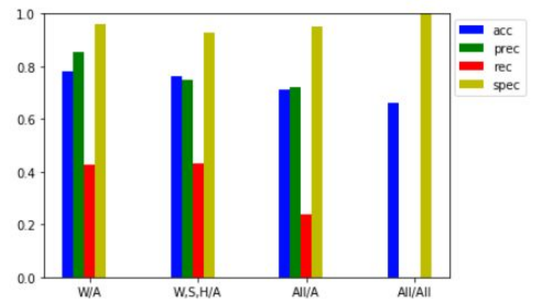


Real life Use Cases

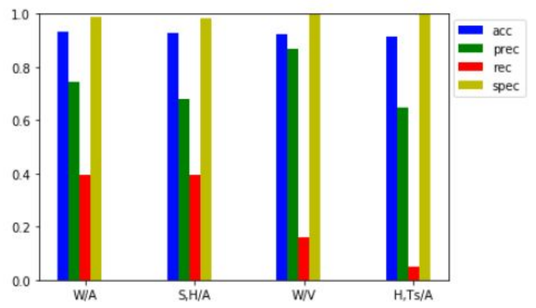
- For real time monitoring we have uploaded our pre trained model to AWS S3 which connected to our Python based API hosted in AWS EC2
- This API can provide real time prediction using the model and also update it using validated data
- A small raspberry pi based server or mobile app can be used to acquire the data from the sensor, process it and get real time prediction using our API. It can also alert care provider about relevant event
- Our python api can keep track of the prediction logs for all patient using a persistent storage
- With our api we can also access this data and visualize relevant metrics to care provider and machine learning engineer

Results

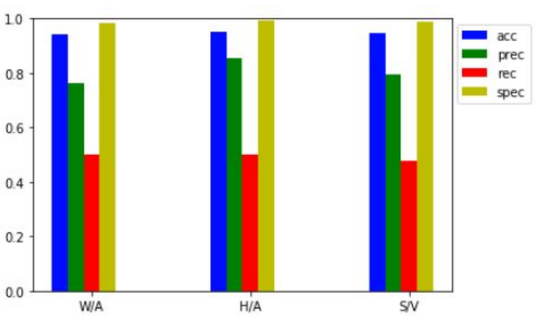
1. Overlapping window:



2. Resultant acceleration peak window:



3. Resultant difference windows:



4. Convolutional Neural Networks: