MONITORING OF ELDERLY USING SENSORS

Ye Changhe, Lim Li Wei, Prem Chandran

Institute of Systems Science, National University of Singapore, Singapore 119615

ABSTRACT

Singapore's ageing population is a major cause of economic concern given that we are a country with no natural resources. Based on projections from the United Nations (UN), 47 percent of Singapore's total population will be aged 65 years or older in 2050[1] An ageing population brings a unique set of challenges, from reduced economic growth to increased healthcare and social services costs. There is a need to monitor their health in a systematic,cheap,efficient and non-intrusive way so as to respect their need for their privacy. A key indicator of health is the level of activity measured as physical movement via sensor(s).

This report considers whether meaning can be derived from analysing the duration of specific physical movements of the elderly measured via passive sensors (RFID). As currently there is no local dataset to serve as a sample to study, the data set referred to was obtained from UCI repository[2]

Index Terms— Elder, healthcare ,movement, sensor

1. INTRODUCTION

The general health of elderly is of concern to improve the quality of their life. When no longer required to work for a living, an elderly person must have the motivation to get up from lying in bed to engage in meaningful activity. A key indicator of health is the level of activity measured as physical movement via sensor(s) where the diagnostic data analysis can be attributed to the design environment of their rooming which is normally a bed and a chair.This basic forms of physical activity can be narrowed to the duration a person spends in bed, sitting and ambulation especially for post-operative recovery (Fig.1).

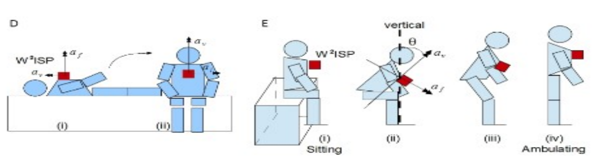

Fig. 1. Forms of Physical Activity.

Analysing duration of such key movements can take on 4 levels of complexity giving rise to increasing value answering key questions as explained below

- **Descriptive level:** What is happening in daily movements? The daily visualisations can allow observation of key time indicators such as duration of sleep/rest/sitting/ambulating times for maximum, minimum and standard deviations for a trend analysis to look for frequency patterns.

- **Diagnostic level:** level:Why is it happening? The cause and effect of the observed time duration can be investigated to uncover underlying reasons by changing variables like the room environment so as to drill down to the root cause and isolate any confounding information.

- **Predictive level:** What's likely to happen based on current trend? Based on historical data such as trends of previous patients can allow preventive measures to be taken just in time, such as if a patient is spending too long in lying on bed or sitting on chair or not showing improvement in postoperative ambulation

- **Prescriptive level:** What needs to be done ? Based on data trends, recommended actions or strategies are adopted to improve the quality of living for the elderly or to help with the postoperative recovery such as redesigning environment through the introduction of physical aids to enable movement.

2. RELATED WORK

For this assignment, we are using a dataset from UCI(University of California, Irvine) (with supporting paper)[2][3]. In this scenario, a passive RFID tag is attached to the subjects (refer to Fig.2).

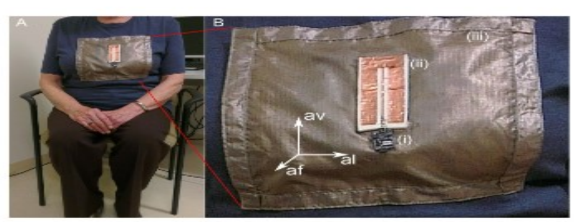

Fig. 2. RFID tagging on subjects.

Multiple antennas (Readers) will be mounted in the rooms (Controlled Space), and will generate data similar to that of an accelerate-meter (AccX,Accy,AccZ). Supporting data which will help improve the classification are the RSSI (Received Signal Strength Indicator), Phase and Frequency of the signal(refer to Fig.3)

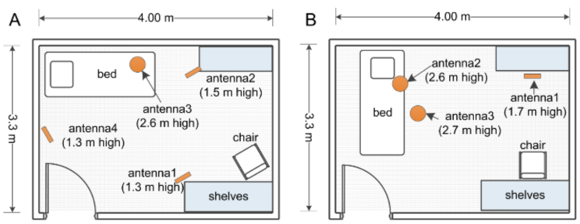

Fig. 3. Antenna Placements.

A total of 87 instances of scripted actions performed by 14 elderly people aged between 66-86 years old, in 2 different rooms with different number of antenna set-ups.

In every instance file (of the 87 instances), there will be varied rows of data sampled from irregular intervals. Every data points will be pre-labelled with a class (by the one recording the action). This factor, alongside with the nature of passive sensor-antenna pair (noise-prone) elevates the difficulty in the accurate interpretation of the data.

The content of the file is as follows:

- **Column 1:** Time in seconds
- **Column 2:**Acceleration reading in G for frontal axis
- **Column 3:** Acceleration reading in G for vertical axis
- **Column 4:** Acceleration reading in G for lateral axis
- **Column 5:** Id of antenna reading sensor
- **Column 6:** Received signal strength indicator (RSSI)
- **Column 7:** Phase
- **Column 8:** Frequency
- **Column 9:** Label of activity

	Time	Acc. Front	Acc. vert	Acc. Lat	id	RSSI	Phase	Freq	Activity Label
0	0.000	0.58862	0.81301	-0.082102	3	-50.5	1.491	920.75	1
1	0.025	0.58862	0.81301	-0.082102	3	-50.5	1.491	920.75	1
2	0.050	0.58862	0.81301	-0.082102	3	-50.5	1.491	920.75	1
3	0.075	0.58862	0.81301	-0.082102	3	-50.5	1.491	920.75	1
4	0.100	0.58862	0.81301	-0.082102	3	-50.5	1.491	920.75	1

Fig. 4. Dataset collected to be analysed.

The objective of this assignment is to classify 4 classes of actions (Sitting on a Chair, Sitting on Bed, Lying in Bed, Ambulating) using the signal produced by the RFID sensor. By classifying the data, after training the model, live data can be fed into it, thereby enabling us to monitor the possible actions the elderly people is taking.

Referring to one Kernel from Kaggle (As a baseline approach)[4], This person's approach is to append every single instance file into one big data set.After combining all the data, he proceeds to classify them using different methods.

```
li=[]
for file in all_files:
    #print(file)
    if file.endswith('.txt'):
        continue
    df=pd.read_csv(file,header=None,index_col=None)
    li.append(df)
df1=pd.concat(li,axis=0,ignore_index=True)
```

Fig. 5. Combining all Data.

Although the accuracy is very high using standard classification models, he is classifying the data as it is: by data point instances. This might not be the right approach as sensor data is streamed over time, and do not occur at one instance of timestamp. It may work to some extent, but may cause misclassification in the middle of an action (if the data points are similar enough, especially during idle times)

Another problem with this dataset is the sparsity of data for some of the classes, especially "Ambulating". Also, ambulating as a single action is too broad to be a class by itself as it consists of multiple actions such as walking, standing, jumping etc.

```
# Going for KNN
knn=KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train,Y_train)
y_pred_knn=knn.predict(X_test)
print("accuracy KNN= ",accuracy_score(Y_test,y_pred_knn))
```

```
accuracy KNN=  0.9123542412925844
```

Fig. 6. KNN Sample Approach.

```
# Going for Random Forest
rforest=RandomForestClassifier()
rforest.fit(X_train,Y_train)
y_pred_rforest=rforest.predict(X_test)
print("accuracy Random Forest= ",accuracy_score(Y_test,y_pred_rforest))
```

```
/opt/conda/lib/python3.6/site-packages/sklearn/ensemble/forest.py:245: F
lt value of n_estimators will change from 10 in version 0.20 to 100 in 0
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)

accuracy Random Forest=  0.9923024159743922
```

Fig. 7. Random Forest Sample Approach.

3. PROPOSED APPROACH

To improve on the classification, we aim to perform the following:

1. Conversion of data points into time series data with a certain window.

2. Augmentation of Sparse data using Auto-Encoder Compression/Decompression.

3. Classification of time series data using:

   (a) Random Forest Ensemble using features extracted by wavelet Coefficient.

   (b) Convolutional Neural Network (CNN)

   (c) Stacked Conv1D with LSTM

First, for our approach, we will enforce a few assumptions: Time is not important(since they are randomly timed), we shift the points to discrete time instance 't'. All antennas will generate a similar type of signal. As such we drop the antenna identifier and combine room 1 and room 2 data.

After combination of data and dropping/separation of labels, the whole signal is chopped whole, into discrete windows. In this case we are using a 40 units of the data as window size and overlap each window by 4 unit, we choose the label of the most common unit as the result for new generated window data (refer to Fig.8).

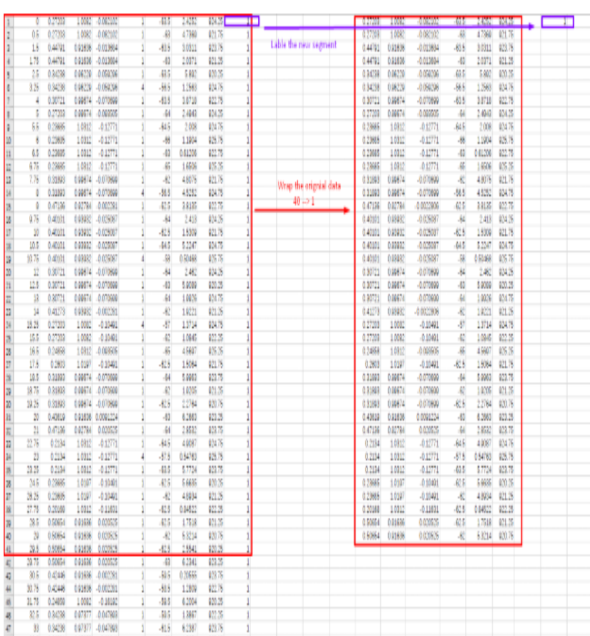

Fig. 8. The neural network classifier structure.

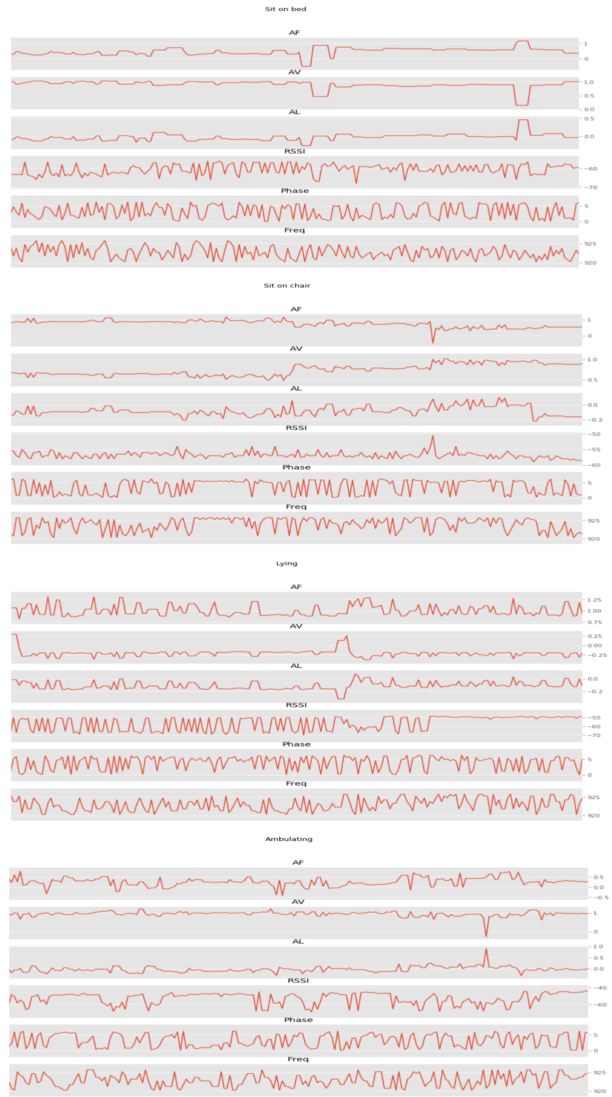

Fig. 9. Signal plot for four classes.

Our first classifier will be one that is similar to the baseline approach: Random Forest. However, instead of using individual data points, we use the windowed time series data for classification. Then we perform wavelet decomposition and generate features by using statistical methods such as mean and standard deviation. These features will then be fed into the random forest classifier, which was used as one of the baseline approach.

After that we will feed the data into CNN and LSTM model according to the result, we can evaluate the performance of each model, from the matrix we can know the classification accuracy of each model by each class and pick up the best one.

Lastly, we tried the data augmentation method by using auto-encoder to generate more noised data and append into our original dataset, then the new data will be feed into the

CNN and LSTM model again and see the if this can enhance the model performance.

For model data partition, we split the data into training data , testing data and validation data (refer to Fig.10).

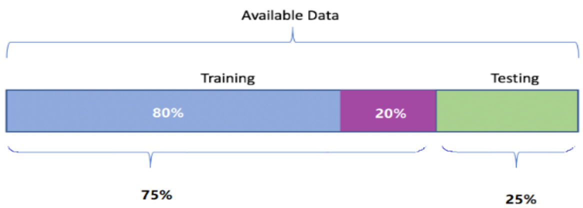

Fig. 10. Data Partition.

4. EXPERIMENTAL RESULTS

For the wavelet decomposition, we are extracting statistical features: namely mean, standard deviation and variance. The classifier (Random Forest) can then classify the data based on the features generated.

```python
# Define the feature extraction method for each subband
def calculate_statistics(list_values):
    mean = np.nanmean(list_values)
    std = np.nanstd(list_values)
    var = np.nanvar(list_values)
    return [mean, std, var]
```

Fig. 11. Statistical Feature Extraction.

Classification Results using Random Forest with (n=100 trees) is as shown. Accuracy is around 95%, which is quite accurate. It is noted that ambulation is misclassified easily due to the broad nature of the range of actions. This is different from the baseline approach as we use time series data instead.

```python
[ ] # Perform classification
    clf = RandomForestClassifier(n_estimators = 100)
    clf.fit(X_train, Y_train)
    test_score = clf.score(X_test, Y_test)

    print('Classification accuracy for test data set: %.4f' % test_score)

    Y_predict = clf.predict(X_test)
    print(pd.DataFrame(confusion_matrix(Y_test.argmax(axis=1), Y_predict.argmax(axis=1)), index=ActivityLabel, columns=ActivityLabel))
```

```
Classification accuracy for test data set: 0.9493
              Sit on bed  Sit on chair  Lying  Ambulating
Sit on bed           965             4     28           2
Sit on chair          78           245     11           5
Lying                 31             0   3238           0
Ambulating            45             1      3          38
```

Fig. 12. Random Forest using Wavelet Coefficients.

Result from classifying statistical features may be accurate enough, but we are exploring even better ways to classify this data. Next, we will do 1-Dimension CNN on the time series data.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 40, 6)]           0
_________________________________________________________________
conv1d (Conv1D)              (None, 40, 32)            992
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 20, 32)            0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 20, 64)            6208
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 10, 64)            0
_________________________________________________________________
flatten (Flatten)            (None, 640)               0
_________________________________________________________________
dense (Dense)                (None, 1024)              656384
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 4100
=================================================================
Total params: 667,684
Trainable params: 667,684
Non-trainable params: 0
_________________________________________________________________
```

Fig. 13. 1-D CNN Model.

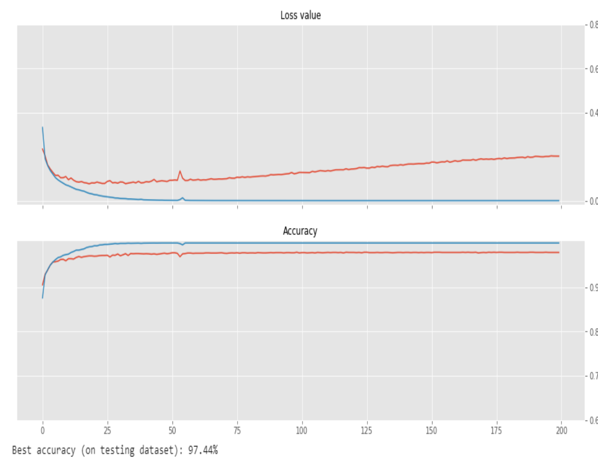

Fig. 14. 1-D CNN Model.

	precision	recall	f1-score	support
Sit on bed	0.9367	0.9678	0.9520	1025
Sit on chair	0.9194	0.9164	0.9179	311
Lying	0.9951	0.9939	0.9945	3268
Ambulating	0.8033	0.5444	0.6490	90
accuracy			0.9744	4694
macro avg	0.9136	0.8556	0.8783	4694
weighted avg	0.9737	0.9744	0.9735	4694

Table 1. Accuracy score for CNN

As we can see by using CNN, we can achieve a better accuracy of 97.44%. Also note that for ambulating process, we get a much better prediction, less misclassification, which improves the performance of the model.

Next we tried using the stacked Conv1D Long-short Term Memory model, which is another useful RNN for classifying time-series data.

These results from using stacked LSTM shows an improvement over using the standard 1-D CNN and hit the 98.30% accuracy, as well as an absence of overfitting.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 40, 6)]           0
_________________________________________________________________
conv1d (Conv1D)              (None, 40, 32)            992
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 20, 32)            0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 20, 48)            7728
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 10, 48)            0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 10, 64)            15424
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 5, 64)             0
_________________________________________________________________
lstm (LSTM)                  (None, 5, 32)             12416
_________________________________________________________________
lstm_1 (LSTM)                (None, 5, 16)             3136
_________________________________________________________________
lstm_2 (LSTM)                (None, 8)                 800
_________________________________________________________________
dense (Dense)                (None, 4)                 36
=================================================================
Total params: 40,532
Trainable params: 40,532
Non-trainable params: 0
_________________________________________________________________
```

Fig. 15. Stacked LSTM Model.

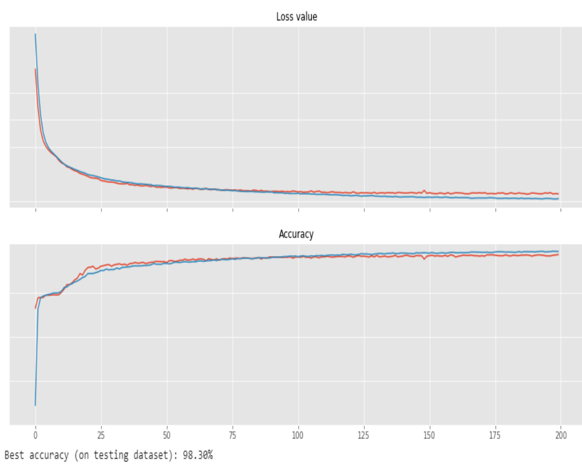

Fig. 16. Stacked LSTM Result.

	precision	recall	f1-score	support
Sit on bed	0.9728	0.9766	0.9747	1025
Sit on chair	0.9075	0.9775	0.9412	311
Lying	0.9951	0.9957	0.9954	3268
Ambulating	0.9167	0.6111	0.7333	90
accuracy			0.9830	4694
macro avg	0.9480	0.8902	0.9112	4694
weighted avg	0.9829	0.9830	0.9823	4694

Table 2. Accuracy score for Stacked LSTM

however, this is not the end, we decide to use auto-encoder to augment the data and add the noisy data to training data pool.And the reconstructed signal as seen below:

After finishing the data augmentation, we train the new larger dataset to our both two CNN and Stacked LSTM model.let see the result below:

Finally, having applied the AE augmentation data, both of the model get slight improvement on classification accuracy on the testing data. This shows that artificially generated data can be used to improve the accuracy of the model. However,

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
layer1_input (InputLayer)    [(None, 240)]             0
_________________________________________________________________
layer1 (Dense)               (None, 120)               28920
=================================================================
Total params: 28,920
Trainable params: 28,920
Non-trainable params: 0
_________________________________________________________________
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
layer1 (Dense)               (None, 120)               28920
_________________________________________________________________
layer2 (Dense)               (None, 240)               29040
=================================================================
Total params: 57,960
Trainable params: 57,960
Non-trainable params: 0
```

Fig. 17. Autoencoder Model.

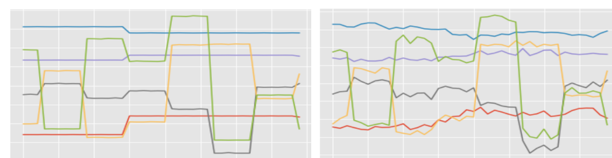

Fig. 18. Reconstruction of Signal for Augmentation

	precision	recall	f1-score	support
Sit on bed	0.9470	0.9650	0.9559	999
Sit on chair	0.9299	0.8997	0.9145	339
Lying	0.9933	0.9954	0.9943	3269
Ambulating	0.8194	0.6782	0.7421	87
accuracy			0.9761	4694
macro avg	0.9224	0.8846	0.9017	4694
weighted avg	0.9756	0.9761	0.9757	4694

Table 3. Accuracy score for 1-D CNN Model with DA.

from the CNN results, we still can see an over-fitting issue.

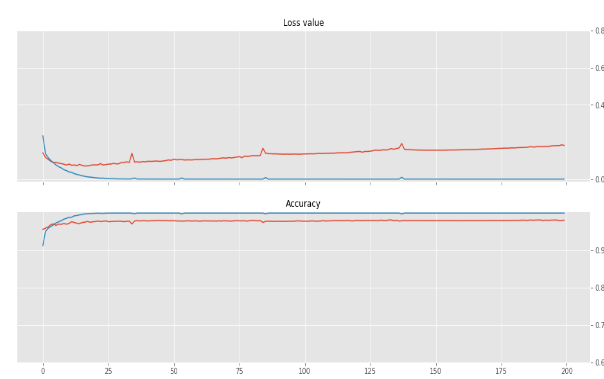

Fig. 19. 1-D CNN Model with data augmentation.

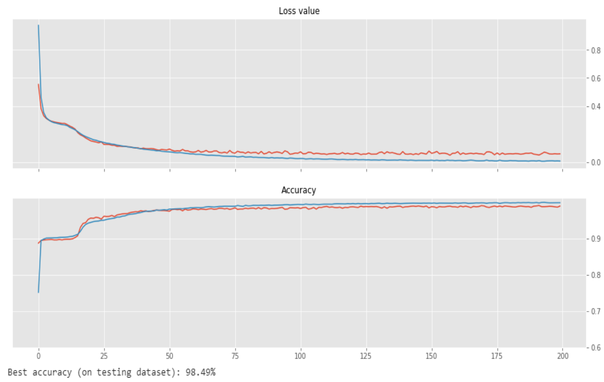

Fig. 20. Stacked LSTM with data augmentation.

	precision	recall	f1-score	support
Sit on bed	0.9665	0.9863	0.9763	1025
Sit on chair	0.9484	0.9453	0.9469	311
Lying	0.9966	0.9939	0.9953	3268
Ambulating	0.8861	0.7778	0.8284	90
accuracy			0.9849	4694
macro avg	0.9494	0.9258	0.9367	4694
weighted avg	0.9847	0.9849	0.9847	4694

Table 4. Accuracy score for Stacked LSTM with DA.

5. CONCLUSION

For RFID Sensors, signal can be noisy. As such, we will need to do some pre-processing and augmentation of data before classification. Using wavelet statistical classification works, but stacked LSTM models seem to work better, even more accurately with data augmentation (Signal Compression). However, these methods serve to improve accuracy with the lack of data. For better quality classification, Ambulation should be separated into multiple classes, and more data is needed.

Attached here is the online reference of the source code[5]

6. REFERENCES

[1] Maegan Liew, "Seniors at work: The new norm for ageing in singapore," .

[2] Ranasinghe D. C. Shi Q. Sample A. P. Shinmoto Torres, R. L., "Activity recognition with healthy older people using a batteryless wearable sensor data set," .

[3] Hoskins S van den Hengel A Ranasinghe DC Torres RL, Visvanathan R, "Effectiveness of a batteryless and wireless wearable sensor system for identifying bed and chair exits in healthy older people," *Sensors (Basel)*, 2016.

[4] Balmukund, "Kernel, classify activity of healthy older people," .

[5] Balmukund, "colab notebook of monitoring of elderly using sensors," .