Human Activity Recognition

Project Report

Pradnya Rajendra Nimbekar (11693010), Tayyab Munir (11716089)

School Of Electrical Engineering and Computer Science

Washington State University

Pullman, WA. USA

I. Abstract

Human Activity Recognition (HAR) is one of the booming scientific topics in the last few decades. It helps scientists to collect data through multiple platforms(sensors) or we can say other multimodal activity recognition methods, to perform different scientific experiments to determine different human aspects like postural transitions and other important health variables.

II. INTRODUCTION

Human Activity Recognition is nowadays an active research field that aims to understand human behavior through the interpretation of sensory information gathered from people and the environment they live in [1]. These days, one of the most active ways of recording human activities are handheld or wearable smart devices which are equipped with multiple sensors to record the combination of different Human activities. Sensors have a big role in making smartphones or wearables more functional and aware of the environment. This makes it possible to collect vast amounts of information about the user's daily life and activities [2]. Moreover, one of the benefits of today's smartphones or other devices developments is that they combine inertial sensors such as accelerometers, gyroscopes, magnetometers and so on that can be used for work on detection of human activities [3]. The underline motivation behind the project is to contribute in the areas of healthcare, gaming and exergaming such as full-body motion-based games [4]. Also, HAR is used widely for monitoring the activities of elderly people staying in rehabilitation centers for chronic disease management and disease prevention [5]. The main challenge was to predict the activity given a snapshot of sensor data, typically data from one or a small number of sensor types. Generally, this problem is framed as a univariate or multivariate time series classification task, and as there are no obvious or direct ways to relate the recorded sensor data to specific human activities and each subject may perform an activity with significant variation, resulting in variations in the recorded sensor data. The intent is to record sensor data and corresponding activities for specific subjects, fit a model from this data, and generalize the model to classify the activity of new unseen subjects from their sensor data. Our solution approach is based on classification models and convolutional neural network. Traditionally, strategies from the field of signal processing were utilized to analyze and is still the collected sensor data. Such strategies were for feature engineering, making domain-specific, sensor-specific, or signal processing-specific features and sees of the initial information. A restriction of this approach is the signal processing and domain skill required to analyze the crude information and build the highlights required to fit a demonstrate. This skill would be required for each new dataset or sensor methodology. In substance, it is costly and not scalable. In a perfect world, learning strategies may well be used that naturally learn the highlights required to form exact expectations from the crude information specifically. This would permit new issues, new datasets, and new sensor modalities to be embraced rapidly and cheaply

III. PROBLEM DEFINITION

Human activity recognition is the problem of human body gestures or motion via sensors and determine human activity or action. This project uses different machine learning models to discover multiple human activity patterns and specifically detecting which activity a person is doing, through signals received by sensors, by analyzing the patterns in the results through a predictive model, and design and generate individualized output result to further predict the accuracy of the results.

IV. Solution Approach:

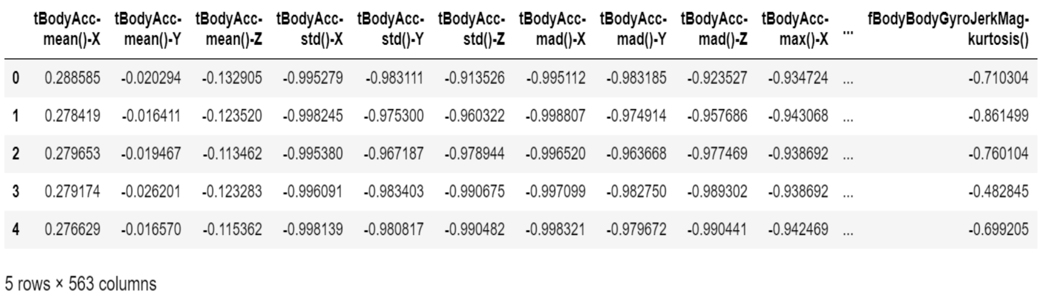
The retrieved dataset collected the data for six different activities of 30 volunteers. The underlying assumption is that the activities are clearly defined without any error in defining it. Looking at the numerical dataset and categorical target variable, first we choose to perform exploratory data analysis to understand the variable which can classify the target variables. Second, we employed different classification algorithms and found that support vector machine, convolutional neural network with Keras sequential model and logistic regression performed better than other algorithms. After digging deeper into results, we found that Support vector machine worked better than logistic regression by 0.01% on two count i.e. seating and standing.

V. Experiments and Results

Data Input:

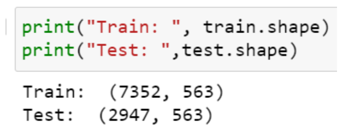
The human activity recognition database is built by recording 30 subjects performing the activities of daily living. The values are captured by a waist-mounted smartphone with embedded inertial sensors. The 30 volunteers for the experiments are of the age bracket 19-48 years. Each volunteer performed six activities. (Walking, Walking_upstairs, Walking_downstairs, Sitting, Standing, Laying). The dataset is labeled manually with help of video recording. The obtained dataset has been randomly partitioned into 70% volunteers for training data and 30% for test data.
Dataset Information: https://www.kaggle.com/uciml/human-activity-recognition-with-smartphones
The dataset carries 10299 number of instances and 563 attributes with no missing values. The attribute information for each record contains

- Triaxial acceleration from the accelerometer and the estimated body acceleration
- Triaxial Angular velocity from the gyroscope
- A 563-feature vector with time and frequency domain variables
- An identifier of the subject who experimented.



	tBodyAcc-mean()-X	tBodyAcc-mean()-Y	tBodyAcc-mean()-Z	tBodyAcc-std()-X	tBodyAcc-std()-Y	tBodyAcc-std()-Z	tBodyAcc-mad()-X	tBodyAcc-mad()-Y	tBodyAcc-mad()-Z	tBodyAcc-max()-X	...	fBodyBodyGyroJerkMag-kurtosis()
0	0.288585	-0.020294	-0.132905	-0.995279	-0.983111	-0.913526	-0.995112	-0.983185	-0.923527	-0.934724	...	-0.710304
1	0.278419	-0.016411	-0.123520	-0.998245	-0.975300	-0.960322	-0.998807	-0.974914	-0.957686	-0.943068	...	-0.861499
2	0.279653	-0.019467	-0.113462	-0.995380	-0.967187	-0.978944	-0.996520	-0.963668	-0.977469	-0.938692	...	-0.760104
3	0.279174	-0.026201	-0.123283	-0.996091	-0.983403	-0.990675	-0.997099	-0.982750	-0.989302	-0.938692	...	-0.482845
4	0.276629	-0.016570	-0.115362	-0.998139	-0.980817	-0.990482	-0.998321	-0.979672	-0.990441	-0.942469	...	-0.699205

5 rows × 563 columns



```
print("Train: ", train.shape)
print("Test: ",test.shape)
```

```
Train:  (7352, 563)
Test:  (2947, 563)
```

Exploratory Data Analysis:

In EDA, we found that there is no missing values or duplicate rows in the dataset. Different activities were visualize using bar chart and pie chart for count and it is found that distribution of classes is fairly balanced. Further, we plotted graph for sensor data collected by accelerometer and Gyrometer and observed that maximum data is collected with accelerometer.

```
['STANDING' 'SITTING' 'LAYING' 'WALKING' 'WALKING_DOWNSTAIRS'
 'WALKING_UPSTAIRS']
**************************************
LAYING                1407
STANDING              1374
SITTING               1286
WALKING               1226
WALKING_UPSTAIRS      1073
WALKING_DOWNSTAIRS     986
Name: Activity, dtype: int64
```

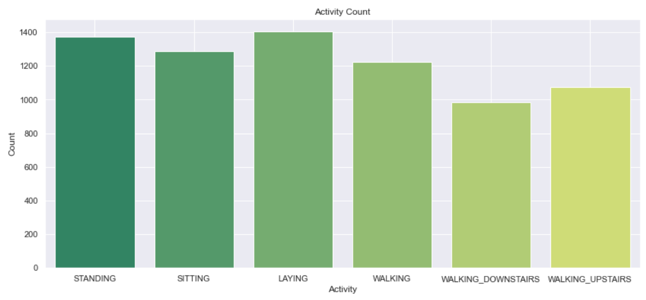


Fig: Count Based on Activity

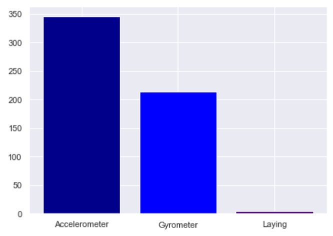


Fig: Data based on sensors

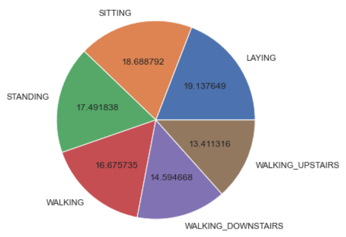
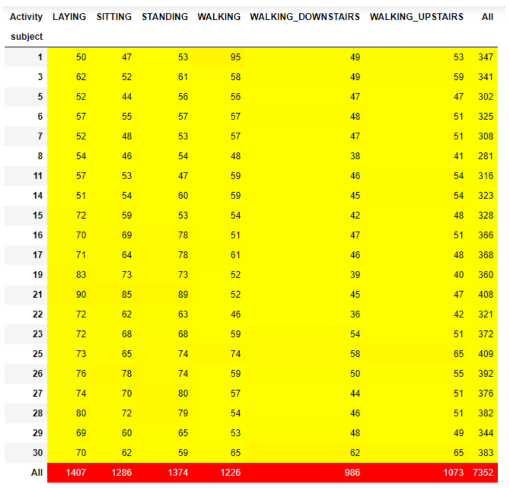


Fig: Activity Percentage



Activity subject	LAYING	SITTING	STANDING	WALKING	WALKING_DOWNSTAIRS	WALKING_UPSTAIRS	All
1	50	47	53	95	49	53	347
3	62	52	61	58	49	59	341
5	52	44	56	56	47	47	302
6	57	55	57	57	48	51	325
7	52	48	53	57	47	51	308
8	54	46	54	48	38	41	281
11	57	53	47	59	46	54	316
14	51	54	60	59	45	54	323
15	72	59	53	54	42	48	328
16	70	69	78	51	47	51	366
17	71	64	78	61	46	48	368
19	83	73	73	52	39	40	360
21	90	85	89	52	45	47	408
22	72	62	63	46	36	42	321
23	72	68	68	59	54	51	372
25	73	65	74	74	58	65	409
26	76	78	74	59	50	55	392
27	74	70	80	57	44	51	376
28	80	72	79	54	46	51	382
29	69	60	65	53	48	49	344
30	70	62	59	65	62	65	383
All	1407	1286	1374	1226	986	1073	7352

Fig: Coss Tabulation

Evaluation:

Model evaluation is the integral component of a project and it targets to estimate the generalization accuracy of a model on future data. Model evaluation metrics are required to quantify the model performance. In this project, we focus on supervised classification learning models: K-Nearest Classifier, Logistic Regression, Decision Tree, Random Forest, Support Vector Machine and also using neural network. For evaluation, we have used classification accuracy, confusion matrix and precision call. Accuracy for five classification models and neural network are as given below:

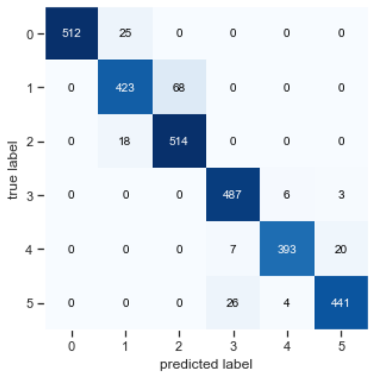
Models	Accuracy
K-Nearest Classifier	0.89
Logistic Regression	0.96
Decision Tree	0.82
Random Forest	0.87
Support Vector machine	0.96
Neural network	~ 0.94

Activity	Logistic Regression	Support Vector machine
Laying	1	1
Sitting	0.96	0.97
Standing	0.9	0.9
Walking	0.95	0.95
Walking_Downstairs	0.99	0.99
Walking_Upstairs	0.96	0.97

Models	Accuracy
Logistic Regression	0.96
Support Vector machine	0.96

From above data, it is clear that accuracy for Logistic Regression, Support Vector Machine and Neural Network are good. Next, if we consider precision for Logistic Regression and

Support Vector Machine as these models are giving accuracy of 96%. then we see below that Support Vector machine is the best if we consider all 6 activities:
For Neural network, below is the confusion matrix:



And the accuracy for the neural network model is approximately 94%.

Baseline Approach:

As the target variable is categorical in nature hence, we choose basic classification algorithms- K-Nearest Classifier, Logistic Regression, Decision Tree, Random Forest, Support Vector Machine. Later, we have implemented neural network with Keras Sequential model which is based on linear stack of layers. We choose this model in addition with other classification models because, recently, deep neural network models have started delivering on their promises of feature learning. Deep Learning is becoming a very popular subset of machine learning due to its high level of performance across many types of data. They are capable of performing automatic feature learning from the raw sensor data and out-perform models fit on hand-crafted domain-specific features.

Modelling:

After exploratory data analysis, we used different algorithms to understand the performance of each model. As the target variable is categorical in nature hence, we choose below classification algorithms: -

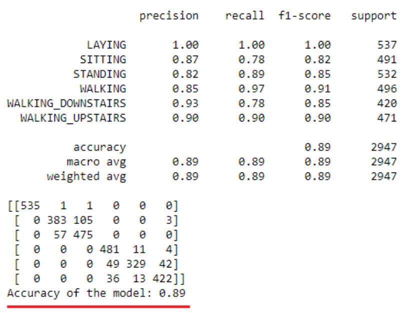
- K-Nearest Classifier
- Logistic Regression
- Decision Tree
- Random Forest
- Support Vector Machine

Followed by above algorithms, we employed convolutional neural network with Keras sequential model. The Keras library in Python makes it pretty simple to build a CNN. Also, complex models can be quickly built by writing code with Keras. Even though this model takes longer time to train the model than TensorFlow, we choose Keras because it is designed for small datasets. We have dataset which has only 30 participants. Also, Keras has high level API and runs on top of tensorflow, it is easy to use and facilitates faster development. The model type that we would be using is Sequential. Sequential is the easiest way to build a model in Keras. It allows you to build a model layer by layer. We use the ‘add()’ function to add layers to our model. These are convolution layers that will deal with our input data. Activation is the activation function for the layer. The activation function we will be using for the layers is the ReLU, or Rectified Linear Activation. This activation function has been proven to work well in neural networks.

Results:

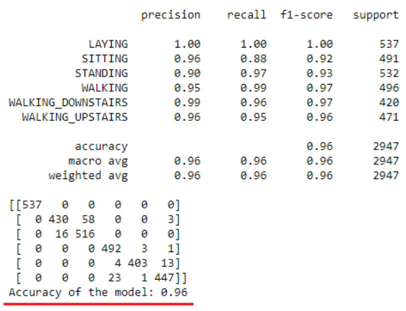
Classification Models Used And Its Results

Algorithm1: K-Nearest Classifier



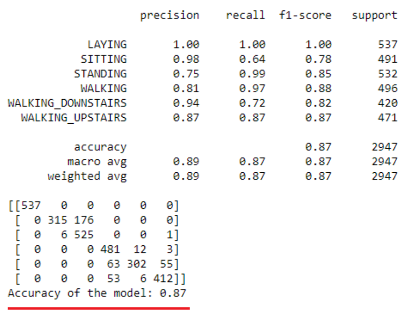
```
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       537
           SITTING       0.87      0.78      0.82       491
          STANDING       0.82      0.89      0.85       532
           WALKING       0.85      0.97      0.91       496
WALKING_DOWNSTAIRS       0.93      0.78      0.85       420
  WALKING_UPSTAIRS       0.90      0.90      0.90       471

          accuracy                           0.89      2947
         macro avg       0.89      0.89      0.89      2947
      weighted avg       0.89      0.89      0.89      2947

[[535   1   1   0   0   0]
 [  0 383 105   0   0   3]
 [  0  57 475   0   0   0]
 [  0   0   0 481  11   4]
 [  0   0   0  49 329  42]
 [  0   0   0  36  13 422]]
Accuracy of the model: 0.89
```

Algorithm2 : Logistic Regression



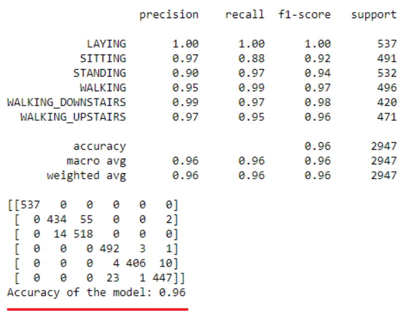
```
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       537
           SITTING       0.96      0.88      0.92       491
          STANDING       0.90      0.97      0.93       532
           WALKING       0.95      0.99      0.97       496
WALKING_DOWNSTAIRS       0.99      0.96      0.97       420
  WALKING_UPSTAIRS       0.96      0.95      0.96       471

          accuracy                           0.96      2947
         macro avg       0.96      0.96      0.96      2947
      weighted avg       0.96      0.96      0.96      2947

[[537   0   0   0   0   0]
 [  0 430  58   0   0   3]
 [  0  16 516   0   0   0]
 [  0   0   0 492   3   1]
 [  0   0   0   4 403  13]
 [  0   0   0  23   1 447]]
Accuracy of the model: 0.96
```

Algorithm3: Random Forest Classifier



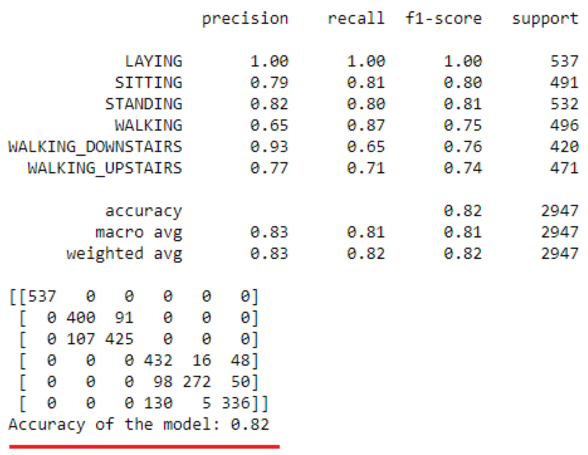
```
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       537
           SITTING       0.98      0.64      0.78       491
          STANDING       0.75      0.99      0.85       532
           WALKING       0.81      0.97      0.88       496
WALKING_DOWNSTAIRS       0.94      0.72      0.82       420
  WALKING_UPSTAIRS       0.87      0.87      0.87       471

          accuracy                           0.87      2947
         macro avg       0.89      0.87      0.87      2947
      weighted avg       0.89      0.87      0.87      2947

[[537   0   0   0   0   0]
 [  0 315 176   0   0   0]
 [  0   6 525   0   0   1]
 [  0   0   0 481  12   3]
 [  0   0   0  63 302  55]
 [  0   0   0  53   6 412]]
Accuracy of the model: 0.87
```

Algorithm4: Support Vector Machine



```
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       537
           SITTING       0.97      0.88      0.92       491
          STANDING       0.90      0.97      0.94       532
           WALKING       0.95      0.99      0.97       496
WALKING_DOWNSTAIRS       0.99      0.97      0.98       420
  WALKING_UPSTAIRS       0.97      0.95      0.96       471

          accuracy                           0.96      2947
         macro avg       0.96      0.96      0.96      2947
      weighted avg       0.96      0.96      0.96      2947

[[537   0   0   0   0   0]
 [  0 434  55   0   0   2]
 [  0  14 518   0   0   0]
 [  0   0   0 492   3   1]
 [  0   0   0   4 406  10]
 [  0   0   0  23   1 447]]
Accuracy of the model: 0.96
```

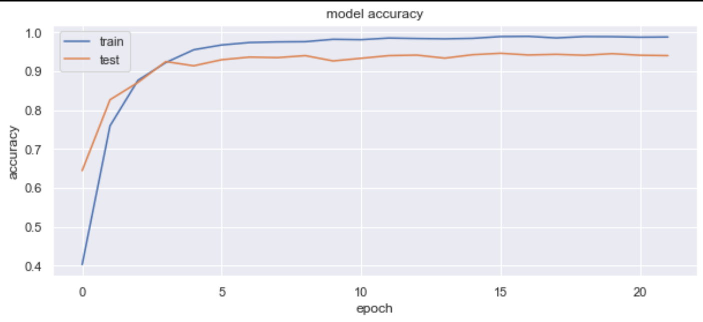
Algorithm5: Decision Tree Classifier

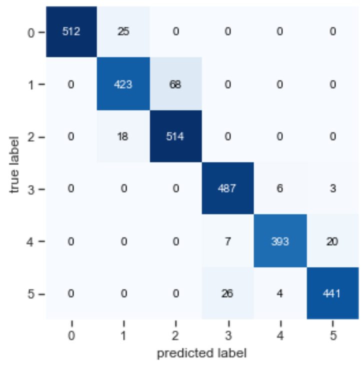


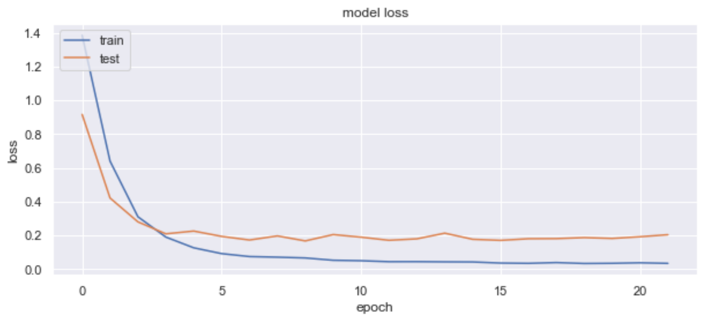
```
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       537
           SITTING       0.79      0.81      0.80       491
          STANDING       0.82      0.80      0.81       532
           WALKING       0.65      0.87      0.75       496
WALKING_DOWNSTAIRS       0.93      0.65      0.76       420
  WALKING_UPSTAIRS       0.77      0.71      0.74       471

          accuracy                           0.82      2947
         macro avg       0.83      0.81      0.81      2947
      weighted avg       0.83      0.82      0.82      2947

[[537   0   0   0   0   0]
 [  0 400  91   0   0   0]
 [  0 107 425   0   0   0]
 [  0   0   0 432  16  48]
 [  0   0   0  98 272  50]
 [  0   0   0 130   5 336]]
Accuracy of the model: 0.82
```

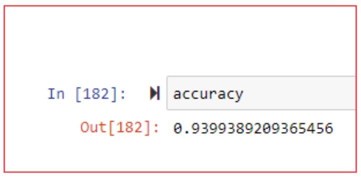
The above figure shows the different classification models with their accuracy, precision, recall and f1-score. We can see that Logistic regression and Support Vector Machine have the good accuracy score, that is, 96%. If we compare these two algorithms we can see that for activities Sitting and walking upstairs precision for SVM is better than Logistic regression. Therefore, we say the best result can be obtained by SVM.

Result for convolution neural network with Keras sequential model:









```
In [182]:  accuracy
Out[182]: 0.9399389209365456
```

Then we built a deep neural network(CNN) with sequential model which is the easiest way to build a model in Keras. It allows you to build a model layer by layer. We use the ‘add()’ function to add layers to our model. These are convolution layers that will deal with our input data. Activation is the activation function for the layer. The activation function we will be using for our first 2 layers is the ReLU, or Rectified Linear Activation. This activation function has been proven to work well in neural networks. Softmax makes the output sum up to 1 so the output can be interpreted as probabilities. The model then makes its prediction based on which option has the highest probability. We use the ‘accuracy’ metric to see the accuracy score on the validation set. Accuracy of the model is 93. 99 which is approximately 94%. It is not as good as SVM in terms of accuracy but neural network model has ability to learn by themselves and produce the output that is not limited to the input provided to them. These networks can learn from examples and apply them when a similar event arises, making them able to work through real-time events. Even if a neuron is not responding or a piece of information is missing, the network can detect the fault and still produce the output and they can perform multiple tasks in parallel without affecting the system performance

VI. CONCLUSIONS AND FUTURE WORK

The human activity recognition database is built by recording 30 subjects performing the activities of daily living. The values are captured by a waist-mounted smartphone with embedded inertial sensors. These results not only allow these models to be efficient but in future they will give enhanced results in terms on Human Activity recognition, Human Computer interaction and other platforms where Machine learning is necessary. However, there is always room for improvement and we believe our best result can be made more better as we progress in technology and data. We believe that adding more features like time when the activity performed, Height, weight, gender and age, we can draw more valuable insights in addition to predicting the activities.

REFERENCES

[1] Reyes-Ortiz JL., Oneto L., Ghio A., Samá A., Anguita D., Parra X. (2014) Human Activity Recognition on Smartphones with Awareness of Basic Activities and Postural Transitions. In: Wermter S. et al. (eds) Artificial Neural Networks and Machine Learning – ICANN 2014. ICANN 2014. Lecture Notes in Computer Science, vol 8681. Springer, Cham.

[2] E. Bulbul, A. Cetin and I. A. Dogru, "Human Activity Recognition Using Smartphones," 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, 2018, pp. 1-6, doi: 10.1109/ISMSIT.2018.8567275.

[3] Md Atiqur Rahman Ahad, Anindya Das Antar and Masud Ahmed Series: Intelligent Systems Reference Library, Year: 2021, Volume 173, Page 7.

[4] O. C. Ann and L. B. Theng, "Human activity recognition: A review," 2014 IEEE International Conference on Control System, Computing and Engineering (ICCSCE 2014), Batu Ferringhi, 2014, pp. 389-393, doi: 10.1109/ICCSCE.2014.7072750.

[5] Yoshimitsu, K., Muragaki, Y., Maruyama, T., Yamato, M., & Iseki, H. (2014). Development and Initial Clinical Testing of “OPECT.” Operative Neurosurgery, 10(1), 46–50. doi:10.1227/neu.0000000000000214