ICCCN-2022

Manchester Metropolitan University | Centre for Advanced Computational Science

Universal Innovators

AIP Conference Proceedings

Indexed in

Scopus WEB OF SCIENCE

INTERNATIONAL CONFERENCE ON COMPUTING AND COMMUNICATION NETWORKS (ICCCN-2022)

Certificate

This is to certify that **Prof. /Dr./ Mr./ Ms. Stephen Azeez** is a presenter /co-author of the paper titled **Human Activity Recognition using Ensemble Machine Learning Classifiers** presented at the **International Conference on Computing and Communication Networks (ICCCN-2022)** jointly organized by Manchester Metropolitan University, Manchester, United Kingdom & UNIVERSAL INOVATORS on **19th-20th November 2022**.

Prof. Omer Rana
General Chair
Cardiff University, UK

Dr. Ali Kashif Bashir
Conference Chair
Manchester Metropolitan University, UK

Human Activity Recognition using Ensemble Machine Learning Classifiers

Shagufta Henna,[a] David Aboga,[b] Muhammad Bilal,[c] and Stephen Azeez[d]
Atlantic Technological University,
Ireland.

Abstract. Activity recognition offers a wide range of applications in various industrial processes and healthcare. This work proposes an approach to collect data from a spherical coordinate system using smartphones, then extract the highly efficient features using advanced preprocessing. Paper also proposes an algorithm to recognize activity using various ensemble machine learning approaches based on extracted features. These approaches are evaluated under various combinations of features to analyze the accuracy, sensitivity, specificity, and training time. Experimental results reveal that weighted KNN performs best among all models by achieving 96.2% accuracy with 12 features. On the other hand, Bagged tree ensemble classifiers perform better than subspace KNN ensemble classifiers with an accuracy of 95.3% using 12 features.

INTRODUCTION

Activity recognition is used to recognize various daily human activities using different observations in an environment [1]. Recently, human activity recognition (HAR) is desirable in industries using artificial intelligence/ machine learning assisted with internet of things (IoT) devices. HAR consists of different modules of data acquisition, segmentation, feature extraction, and classification [2]. Recently, human activity prediction is on the rise due to advancements in wearable technologies. Smart-phone sensor data has been successfully used to avoid harmful effects of a sedentary lifestyle in the health area, such as preventing obesity and cancer [3]. Sensors such as accelerometers and gyroscope available on smartphones and smart-watches are used for raw data collection that is later used for HAR using machine learning algorithms [4]. Healthcare sensors, e.g., accelerometers and angular velocity are used for various applications, ranging from medical diagnosis to fall detection [5]. In [6], data from wearable sensor devices is collected from ten different activities using volunteers. This work uses a non-parametric weighted attribute extraction algorithm for the classification with an accuracy of 90%. In [7], stationary and mobile activities are performed by subjects using an inertial sensor such as an accelerometer and gyroscope. In this work, the Gaussian SVM classifier records an accuracy result of 78.0%. In [8], a Gaussian SVM model demonstrated an accuracy of 92.6% to predict stationary activities using sensor data. In [9], decision trees are used as an ensemble to classify various human activities of aged people. Algorithm correctly recognizes 19 activities with an accuracy of 93.44%. One of the recent works in [10, 11] recognizes various body gestures and movements while using the gaming console. Another work in [12] proposed HAR for predicting the transportation routines of humans. Another work in Attal et al. in [13] classifies various human activities in an industry setting using wearable devices.

Due to diverse applications of HAR, persuasive computing is investing significant efforts to address various challenges using sensors. In that context, two methods for data acquisition for human activity recognition using sensors are proposed in [14]. The first method uses a wearable sensor acquisition, whereas the second method relies on custom hardware or smartphones with sensors such as accelerometers and gyroscopes. In another work, authors use sensor data from 30 subjects to build a dataset for activity recognition using a multiclass support vector machine. Another work in [15], captures rectangular sensor-based data from a smartphone. It extracts 31 features for human activity classification. Furthermore, it also introduced a dimensionality reduction to select features that best explain the data. A work in [16] uses UCI for human activity recognition. In their research, authors have considered the use of support vector machines fusion and K-nearest neighbor with satisfactory results.

In this paper, we propose human activity recognition using a sensor data acquisition process. This work differs from previous works to perform advanced pre-processing on sensor data to train various single and ensemble machine learning classifiers. This pre-processing involves time synchronization, feature engineering, data smoothing, and

[a]Corresponding author: shagufta.henna@atu.ie
[b]Electronic mail: L00150803@student.lyit.ie
[c]Electronic mail: Muhammbilal@gmail.com
[d]Electronic mail: L00162428@atu.ie

labeling. Further, data is segmented using a cross-validation approach before training the selected classifiers. In contrast to existing works, this work considers the use of a spherical coordinate system in addition to rectangular coordinates for sensor-based training data for activity recognition. It also utilizes the sum vector magnitude method to compute the radius of the spherical coordinates.

MACHINE LEARNING-BASED ACTIVITY CLASSIFICATION: MATERIALS AND METHODS

The proposed methodology consists of three layers: data acquisition, data cleaning and conversion, and classifier as discussed below.

1. Data Acquisition Layer Data required for this research is collected using MATLAB mobile application. Specifically, this stage creates 3-axial raw sample data from smartphones. The recorded activity includes values from the accelerometer, angular velocity, orientation, and magnetometer sensors with a sampling rate of 50hz. Raw data samples are generated by conducting six different activities, i.e., walking, sitting, standing, lying on a flat surface, climbing up the stairs, and climbing down the stairs separately. The age of each volunteer ranges from 25 to 30 with an activity that lasted for 10 seconds.

2. Data Cleaning and Conversion Layer The data cleaning and transformation process used is explained below.

a). Time Synchronization: This stage implements time synchronization, an essential phase on the time-series sensor-based data, using VLOOKUP before extracting features from the sensor data. Due to the possibility of difference in time duration or intervals from each sensor or node, time synchronization is an essential step in processing sensor data.

b).Feature Extraction: Feature extraction from raw data is another crucial step in activity recognition before applying machine learning algorithms. A common approach for extracting time-frequency domain acceleration features is mean, standard deviation, energy, and spectral entropy. We perform feature extraction on our X, Y, and Z coordinates using a mathematical approach called the spherical coordinate system. Equation 1, 2, and 3 illustrates the spherical coordinate system conversion formula. The spherical coordinate system is a three-dimensional space system that calculates the radial distance from a fixed origin point, the zenith direction, which is the polar angle measured from a fixed direction θ, and the angle projection ϕ.

$$r = \sqrt{x^2 + y^2 + z^2} \tag{1}$$

$$\phi = \arctan \frac{y}{x} \tag{2}$$

$$\theta = \arccos \frac{z}{r} \tag{3}$$

Using the spherical coordinate in Equation 1, 3-dimensional sensor data is normalized that is later used for feature extraction as listed in Table I.

c). Data Smoothing and Labelling: This stage has computed moving average using a low pass filter to smooth the sensors data. D Often raw sensor data may be insufficient for extracting meaningful information. Noise and other interference from the environment may make sensor data noisy. The moving average computation uses a fixed window with three-point series based on an interval of 3, and the output is the average as new data points. A smaller moving average interval helps to retain the information of the original dataset. However, if a window is too small, the sensor data may end up being noisy. Furthermore, if the window size is too large, it can result in loss of crucial information.

d).Feature Selection: The processed data consists of 4380 rows and 13 columns including the activity class and the rest 12 columns as features columns. In experiment settings have used a combination of different features to observe accuracy, sensitivity, specificity, the area under the curve (AUC), and Mathew's correlation coefficient (MCC).

3. Classifier Layer This section discusses various single and ensemble ML classifiers for activity recognition.

a). Weighted K Nearest Neighbor (KNN) Weighted KNN uses nearest k points that are assigned with higher weight compared to the k points that are far from the query point. Model is tuned for various hyperparameters, i.e., the number of nearest neighbors, weight, and square inverse. The model selects k neighbors closer to our test data by calculating the distance. It also calculates the weight of k neighbors and computes the weighted sum for each category. The algorithm given in Algorithm 1 uses Euclidean distance as the distance metrics where all points are in n space and distance from p to q is calculated using the Equation 4.

TABLE I. Features names and description.

SVM_acc	Sum of Vector Magnitude for Accelerometer
Arctan_acc	Arctangent for Accelerometer
Arccos_acc	Cosine Inverse for Accelerometer
SVM_ang	Sum of Vector Magnitude for Angular Velocity
Arctan_ang	Arctangent for Angular Velocity
Arccos_ang	Cosine Inverse for Angular Velocity
SVM_orien	Sum of Vector Magnitude for Orientation
Arctan_orien	Arctangent for Orientation
Arccos_orien	Cosine Inverse for Orientation
SVM_mag	Sum of Vector Magnitude for Magnetometer
Arctan_mag	Arctangent for Magnetometer
Arccos_mag	Cosine Inverse for Magnetometer

$$\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{4}$$

We calculate weight using a square inverse kernel where an observation is closer to the new observation thereby giving higher weight to contribute to classification.

$$\hat{y} = Max_r \left(\sum_{i=1}^{k} w_{(i)}, y_{(i)} = r \right) \tag{5}$$

In Equation 5, r denotes total number of classes and indicator variable the sum obtained for each class as 1. Finally, the weight $w(i)$ is calculated through the square inverse function $\frac{1}{|d|}$.

Input Data: Training set and classes, i.e., $M = \{x_i, y_i\}$
1: **function** ***SequreInv*** (x) ▷ Call square-inverse function
2: $d(p,q) \Leftarrow \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$ ▷ For each new observation or test data
3: **return** $d(p,q)$
4: **loop**$x \in x_i, x$
5: $predicted_{class} \Leftarrow SequreInv(x)$ ▷ compute the weight using square inverse

Algorithm 1: Weighted KNN algorithm.

b).Fine Gaussian SVM Further to weighted KNN, we also considers SVM kernel function called the gaussian kernel function. The major advantage of using Gaussian SVM is its ability to detect complicated non-linear classification tasks. The non-linear function is introduced to replace the dot functions to enable the model to fit its maximum margin hyperplane in a function space. With kernel tricks such as Gaussian kernel, SVM can efficiently perform nonlinear classification.

$$k(x_1, x_2) = exp\left(\frac{-||x1 - x2||^2}{2\sigma^2} \right) \tag{6}$$

Using the calculated Euclidean distance of X_1 and X_2 from Equation 6, Algorithm 2 computes dot product of angular distance of X_1 and X_2 . Firstly we applied a manual scaling of 0.87 to divide each element of the predictor matrix using a predefined value to perform the necessary kernel to compute a Gram matrix. Furthermore, a one-vs-one multiclass method is utilized for the classification problem. A one-vs-one approach trains one learner for each class with a motive of distinguishing the classes. In the Algorithm, the parameter for box constraint is denoted as C.

c).Gaussian Naïve Bayes (GNB) Gaussian Naïve Bayes operates as a probabilistic classifier and is useful for multi-classification based on the Bayes theorem. Naïve Bayes is a scalable classifier that normally requires a good number of parameters to learn the feature during training. The GNB algorithm is presented in Algorithm 3. GNB

Input: Data:Hyper-parameters, i.e., $M = \{x_i, y_i\}$
1: **function *kernel*(x_1, x_2)**
2: $k(x_1, x_2) = exp\left(\frac{-||x1-x2||^2}{2\sigma^2}\right)$ ▷ Set kernel function to gaussian
3: return $k(x_1, x_2)$
4: **loop**$x_1 \in X_1$ and $x_2 \in X_2$ ▷ X_1 is training set and X_2 is the new observation
5: $predicted_{class} \Leftarrow kernel(x_1, x_2)$ ▷ Predict class of x_2 using the new hyperplane

Algorithm 2: Fine Gaussian SVM.

computes both the mean and variance from training data. In Algorithm 3, the Gaussian function computes the mean and variance in our training data.

$$\hat{P}(y = k|x_{1,\ldots,}x_P) = \frac{\pi(y=k)\prod_{j=1}^{p} p\left(x_j|y=k\right)}{\sum_{k=1}^{k}\pi(y=k)\prod_{j=1}^{p} p\left(x_j|y=k\right)} \tag{7}$$

The prior probability for the class is denoted as $\pi(Y = k)$ based on each predictor $X_1 \ldots X_p$ as given in Equation7. The Gaussian Naïve Bayes algorithm estimates a separate Gaussian distribution for each class.

Input:X, i.e, training dataset, $Z = (z_1 \ldots z_n$
1: **function *Pdf*($x, mean, sd$)**
2: $pdf(x, mean, sd) = \left(\frac{1}{(\sqrt{(}2\times PI)\times sd)}\right) \times exp\left(-((x-mean^2)/(2^s d^2))\right)$ ▷ Gaussian function for each class
3: return $k(x_1, x_2)$
4: **loop**$x \in X$ ▷ for each class compute mean and standard deviation
5: $mean(x) = 1/n \times sum(x)$
6: $sd(x) = \sqrt{(\frac{1}{n})}$
7: $\times sum(xi - mean(x)^2)$

Algorithm 3: Gaussian Naïve Bayes.

d).Bagged Trees The paper also has evaluated bagged trees as an ensemble method to improve results. The bagged tree technique utilizes a bootstrap aggregating method to avoid overfitting, thereby reducing bias and variance error. Bagged trees use a subset of training data for training, and after training the models, a vote of their output is taken. We ensemble 30 decision trees as learners with a learning rate of 0.1 to improve the accuracy of single ML models. The bagged trees algorithm is presented in Algorithm 4.

Input:M be set of bootstrap samples, $Y = 30$ number of learners, X,testdata
1: **loop**$l = 1$ to Y
2: **loop**$b = 1$ to M
3: $Train(l, b)$
4: **loop**$l = 1$ to $Trained_Y$
5: **loop**$x = 1$ to X
6: $vote(x, l) \leftarrow l.prdict(x)$

Algorithm 4: Bagged trees.

e.Subspace K-Nearest Neighbor Another machine learning classifier we have used for the activity recognition is subspace KNN given in Algorithm 5. The algorithm creates an ensemble of KNN with several dimensions, learning rate, number of learners, and splits. The ensemble of KNN does not improve classification accuracy until there are variations in the dataset. Subspace KNN addresses these problems using necessary discriminant information as given in Algorithm 5. In this work, we use various subspaces for each KNN classifier to avoid training without discriminant information.

Input: d features, B number of bootstrap samples, m learners

loop $l = 1$ to m
    select random sample r from d features
    construct kNN_i using the bootstrap sample B_i
    Evaluate accuracy of kNN **if** $accuracy\ (kNN_i) > Th$ **then**
    $model \leftarrow (kNN_i)$
    end
loop $i = 1$ to $model$
    $Bscore_{i+1} \leftarrow ensemble(model_{i+1})$ **if** $Bscore_{i+1} < Bscore_i$ **then**
    $model_{selected} \leftarrow model_{i+1}$
    end

Algorithm 5: Subspace K-Nearest neighbor.

Algorithm6 summarizes all the steps required for activity recognition based on various machine learning classifiers. Line 1-6 lists the data acquisition. Line number 5-3 is essential pre-processing including feature engineering, data smoothing, labeling, and cross-validation.

Input: Dataset $Dataset_{activity}$ using x sensors

loop $Dataset_{activity} \neq \emptyset$
    synchronize timestamps for all x sensor 's data ▷ Pre-process time-series data
    VLOOKUP $(A2, F:I, 2, TRUE)$
    Convert all x sensor data to spherical coordinates for feature extraction
    $\varphi = arctan\frac{y}{x}$
    $\theta = arccos\frac{z}{\sqrt{x^2+y^2+z^2}} = arccos\frac{z}{r} = arctan\frac{\sqrt{x^2+y^2}}{z}$
    MovingAvg $\leftarrow \frac{A_1+A_2...+A_n}{n}$ ▷ Compute moving average for spherical coordinate sensor data
Features $\leftarrow$ ExtractFeatures(MovingAvg)
Model=Train(WeightedKNN,Fine Gaussian SVM,Gaussian Naïve Bayes,Bagged trees,Subspace K Nearest Neighbor)
return $(bestModel \leftarrow Validate(model))$

Algorithm 6: ML-based human activity classification.

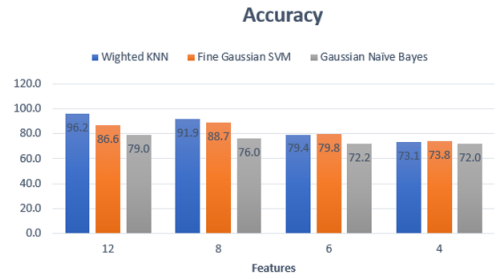
PERFORMANCE EVALUATIONS

We have generated the data using the MATLAB mobile application which after pre-processing is used to train the ML classifiers.To validate the results, we apply the 5-fold-cross validation. We analyze the performance of single and ensemble ML classifiers for the activity recognition in terms of time complexity and confusion matrix-based measures such as accuracy (ACC), sensitivity, specificity, MCC, and area under the curve (AUC).

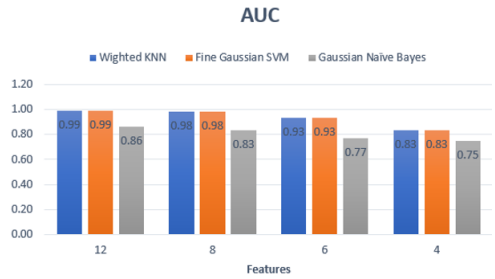
Performance Analysis of Single and Ensemble ML Classifiers

All single and ensemble models demonstrate inter-cluster differences using stationary and mobile activities. Table II summarizes the performance comparison of single and ensemble ML classifiers to predict a different number of activities with a varying number of features.

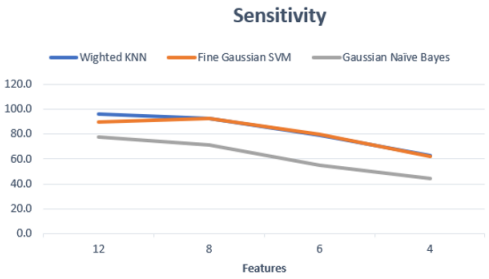
Accuracy: As depicted in Figure 1a and Figure 2a, Gaussian SVM achieves an accuracy of 86.6% as compared to the 96.2% observed by the weighted KNN model. The results reveal that with 12 features, weighted KNN performs better than all other classifiers. Bagged trees demonstrate highest accuracy of 95.3%. Results reveal that accuracy, sensitivity, and specificity degrade with a combination of 8,6, and 4 features. Bagged trees classifier, however, achieves less time to train for 6 and 8 features.



(a) Accuracy of single ML classifiers.

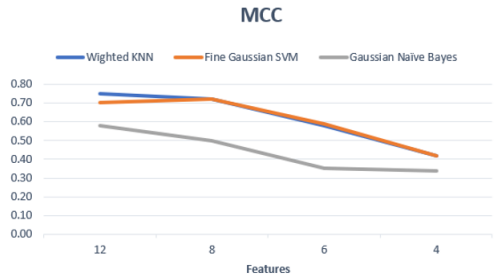


(b) AUC of single ML classifiers.



(c) Sensitivity of single ML classifiers.

(d) Specificity of single ML classifiers.



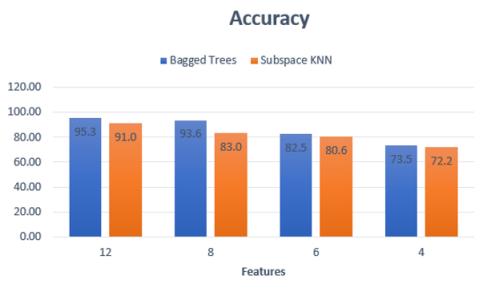
(e) MCC of single ML classifiers.

FIGURE 1. Performance of single ML classifiers.

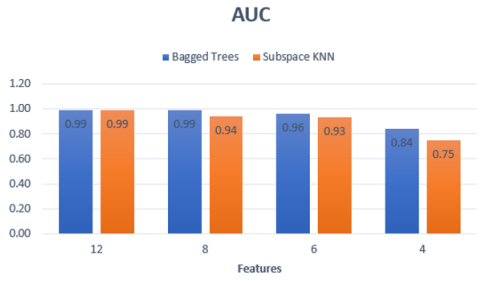
Sensitivity: As can be observed from Figure 1c and Figure 2c, Bagged trees achieve a high percentage of sensitivity. Weighted KNN achieves good results with the help of a weighted function. Another model with good performance is subspace KNN that benefits from the voting approach. Gaussian SVM also performs better than the Naïve Bayes with a sensitivity of 77.9. Bagged tree ensemble classifiers outperform all other models, including the second ensemble classifier, when implemented with all classes of features. Weighted KNN outperforms other single ML models with higher sensitivity with 12 features. However, it underperforms the Gaussian SVM when trained with 8 and 6 features. **Specificity**: In Figure 1d and Figure 2d, specificity measures the rate of true negatives predicted by the single and ensemble ML classifiers. It is observed that all the models perform better in predicting the numbers of observations that are not part of an activity. It is observed from the Table II that the specificity of all the models is between 79.6% to 80.3% for all the features.

Mathew's correlation coefficient (MCC): MCC as depicted in Figure 1e and Figure 2e considers all false positive and negative values for efficient statistical evaluation. A perfect score for MCC is a value close to 1, 1, or plus 1. We observe that all the models record MCC values below 0 or -1. Weighted KNN shows the highest MCC of 0.75 for single ML models. Bagged tree ensemble classifiers have the overall highest MCC value of 0.77 that is explicitly better than the subspace KNN value of 0.72. Weighted KNN performs best in predicting the rate of correct prediction when implemented with 12 features. It outperforms all single ML classifiers used in this work as evident from Table II and the graphs. In terms of accuracy, bagged tree ensemble classifiers perform better than subspace KNN ensemble classifiers that use weak learners.

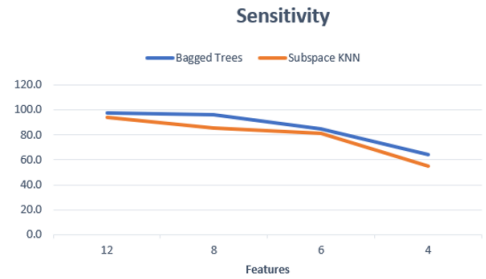
AUC: ROC curve tells how good an ML classifier is to distinguish different classes. Figure 1b and Figure 2b plot



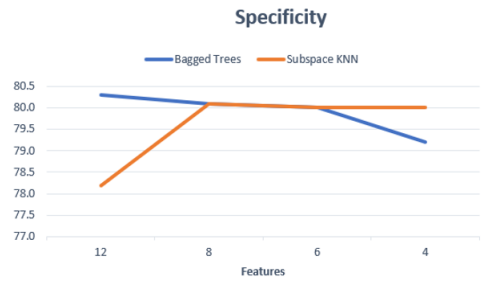
(a) Accuracy of ensemble ML classifiers.



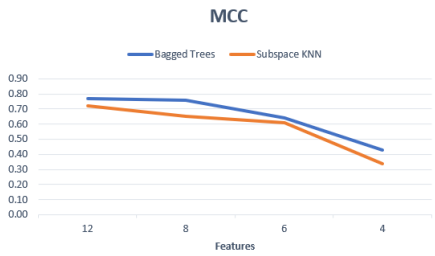
(b) AUC of ensemble ML classifiers.



(c) Sensitivity of ensemble ML classifiers.



(d) Specificity of ensemble ML classifiers.



(e) MCC of ensemble ML classifiers.

FIGURE 2. Performance of ensemble ML classifiers.

the area of a curve using true positive rates and false-positive rates. Climbing down the stairs is considered as one of the positive class and other activities as negative classes.Different thresholds ranging from 0 to 1 are used for the evaluations. Experimental results reveal that all the models demonstrate an AUC of above 0.75 using a different set of features with the same positive and negative classes. Table II shows the training time of single and ensemble ML classifiers for activity prediction. It shows that the training time for the fastest with a value of 9.55 secs. Bagged ensemble classifier also train faster with a value of 9.57secs using 12 features. In conclusion, bagged trees and weighted KNN models should be deployed in the production using different features. These models record the least false positive and false negative with higher sensitivity, specificity, AUC, accuracy, and MCC.

CONCLUSIONS

Activity recognition aims to recognize different human activities from observations in a given environment using sensor data. Data used in this research is generated using smartphones by four subjects. We have performed feature engineering steps to transfer and extract features from the sensors data. In the experiments, we have used the extracted features to evaluate the performance of single and ensemble classifiers. Weighted KNN performs best among the single machine learning classifiers with an overall accuracy of 96.2% when implemented with 12, 8, and 6 features. The second-best single machine learning classifier is fine Gaussian SVM, which records 86.6% accuracy, followed by Gaussian naive Bayes with 79.0%. Results show that the ensemble classifiers yield satisfactory results, whereas the

TABLE II. Performance comparison of single and ensemble classifiers.

Model.	Featurs	Accuracy	AUC	Sensitivity	Specificity	MCC	Training time
Weighted-KNN	12	96.2	0.99	95.9	80.1	0.75	9.5543 sec
Fine Gaussian SVM	12	86.6	0.99	90.0	80.0	0.70	15.732 sec
GNB	12	79.0	0.86	77.9	80.0	0.58	12.214 sec
Bagged Trees	12	95.3	0.99	97.3	80.3	0.77	9.5704 sec
Subspace KNN	12	91.0	0.99	94.3	78.2	0.72	13.432 sec
Weighted-KNN	8	91.9	0.98	92.2	80	0.72	7.6662 sec
Fine Gaussian SVM	8	88.7	0.98	92.2	80.0	0.72	12.413 sec
GNB	8	76.0	0.83	71.2	79.6	0.50	1.3236 sec
Bagged Trees	8	93.6	0.99	96.2	80.1	0.76	9.0679 sec
Subspace KNN	8	83.0	0.94	85.2	80.1	0.65	9.4162 sec
Weighted-KNN	6	79.4	0.93	78.8	80.0	0.58	1.6083 sec
Fine Gaussian SVM	6	79.8	0.93	79.6	80.0	0.59	11.753 sec
GNB	6	72.2	0.77	54.7	80.0	0.35	1.2377 sec
Bagged Trees	6	82.5	0.96	84.5	80	0.64	8.8587 sec
Subspace KNN	6	80.6	0.93	81.0	80	0.61	6.6096 sec
Weighted-KNN	4	73.1	0.83	62.5	80.0	0.42	1.4799 sec
Fine Gaussian SVM	4	73.8	0.83	62.0	80.0	0.42	10.582 sec
GNB	4	72.0	0.75	44.3	80.0	0.34	1.2277 sec
Bagged Trees	4	73.5	0.84	64.2	79.2	0.43	9.5158 sec
Subspace KNN	4	72.2	0.75	54.7	80	0.34	6.6515 sec

bagged tree shows an accuracy of 95.3% compared to subspace KNN with an accuracy of 91.0%. Other performance measures used for analysis, i.e., sensitivity, AUC, specificity, MCC, and training time show that the weighted KNN ML classifiers and ensemble learning classifiers perform best for activity recognition.

REFERENCES

1. Y. Chen, L. Yu, K. Ota, and M. Dong, "Robust activity recognition for aging society," Journal of Biomedical and Health Informatics **6**, 1754–1764 (2018).
2. H. Braganca, J. Colonna, W. Lima, and E. Souto, "A smartphone lightweight method for human activity recognition based on information theory," Sensors **7**, 1856 (2020).
3. N. Lathia, G. Sandstrom, C. Mascolo, and P. Rentfrow, "Happier people live more active lives: Using smartphones to link happiness and physical activit," PLOS ONE **1** (2017).
4. N. Yan and T. Chen, J.and Yu, "A feature set for the similar activity recognition using smartphone," in *In Proc. International Conference on Wireless Communications and Signal Processing (WCSP)* (2018) pp. 1–6.
5. D. Yang, J. Huang, X. Tu, G. Ding, T. Shen, and X. Xiao, "A wearable activity recognition device using air-pressure and imu sensors," IEEE Access **7**, 6611–6621 (2018).
6. H. Y.-L. and L. S., "Application of nonparametric weighted feature extraction for an inertial-signal-based human activity recognition system," in *In Proc. of the Conference on Applied System Innovation (ICASI)* (2017) p. 1718–1720.
7. D. Phan and D. Tran, "Human activities recognition in android smartphone using support vector machine," in *In Proc. International Conference on Intelligent Systems, Modelling and Simulation (ISMS)* (2016) p. 64–68.
8. I. Joudeh and A.-M. Cretu, "Wifi channel state information-based recognition of sitting-down and standing-up activities," in *In Proc. International Conference on Iymposium on Medical Measurements and Applications (MeMeA)* (2019) p. 1–6.
9. Z. Feng, L. Mo, and M. Li, "A random forest-based ensemble method for activity recognition," in *In Proc. of the International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (2015) p. 5074–5077.
10. E. Oliveira, "Activity recognition in a physical interactive robogame," in *In Proc. of the IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)* (2017) p. 92–97.
11. K. Sowmya, "Construction workers activity detection using bof," in *In Proc. of the International Conference on Recent Advances in Electronics and Communication Technology(ICRAECT)* (2017) p. 159–163.
12. C. Wang and Z. Peng, "Deep learning model for human activity recognition and prediction in smart homes," in *In Proc. of the International Conference on Intelligent Transportation, Big Data Smart City (ICITBS)* (2020) p. 741–744.
13. F. Attal and S. Mohammed, "Physical human activity recognition using wearable sensors," Sensors **15**, 31314–31338 (2015).
14. J.-L. Reyes-Ortiz and L. Oneto, "Transition-aware human activity recognition using smartphones," Neurocomputing **171**, 754–767 (2016).
15. L. F., S. Y., and C. W., "Up and down buses activity recognition using smartphone accelerometer," in *In Proc. of the Information Technology, Networking, Electronic and Automation Control Conference* (2016) p. 761–765.
16. J. A. and K. V., "Human activity classification in smartphones using accelerometer and gyroscope sensors," IEEE Sens. J. , 1169–1177 (2017).