EAI Endorsed Transactions
on Industrial Networks and Intelligent Systems
Research Article EAI.EU

# Human Activity Recognition System For Moderate Performance Microcontroller Using Accelerometer Data And Random Forest Algorithm

To-Hieu Dao[1,2], Hoang Thi Hai Yen[3], Van-Nhat Hoang[1],Duc-Tan Tran[1,2],Duc-Nghia Tran[4,*]

[1]Faculty of Electrical and Electronic Engineering, Phenikaa University, Yen Nghia, Ha Noi, 12116, Viet Nam
[2]PHENIKAA Research and Technology Institute (PRATI), A&A Green Phoenix Group JSC, No.167 Hoang Ngan, Trung Hoa, Cau Giay, Ha Noi, 11313, Viet Nam
[3]Thai Nguyen University, University of Information and Communication Technology, Thai Nguyen, Viet Nam
[4]Institute of Information Technology, Vietnam Academy of Science and Technology, Ha Noi, Viet Nam

## Abstract

There has been increasing interest in the application of artificial intelligence technologies to improve the quality of support services in healthcare. Some constraints, such as space, infrastructure, and environmental conditions, present challenges with assistive devices for humans. This paper proposed a wearable-based real-time human activity recognition system to monitor daily activities. The classification was done directly on the device, and the results could be checked over the internet. The accelerometer data collection application was developed on the device with a sampling frequency of 20Hz, and the random forest algorithm was embedded in the hardware. To improve the accuracy of the recognition system, a feature vector of 31 dimensions was calculated and used as an input per time window. Besides, the dynamic window method applied by the proposed model allowed us to change the data sampling time (1-3 seconds) and increase the performance of activity classification. The experiment results showed that the proposed system could classify 13 activities with a high accuracy of 99.4%. The rate of correctly classified activities was 96.1%. This work is promising for healthcare because of the convenience and simplicity of wearables.

Received on 19 August 2022; accepted on 14 October2022; published on 09 November 2022

**Keywords:** Classification, recognition, activity, real-time, wearable, microcontroller, moderate performance



doi:10.4108/eetinis.v9i4.2571

## 1. Introduction

In recent years, the digital transformation in healthcare has taken place strongly, and four prominent approaches have emerged: i) Using robots to interact with patients, provide personalized services, and provide advice [1]; ii) Optimizing the use of personal electronic wearable devices to monitor health status and prevent diseases [2, 3]; iii) Using computer vision in healthcare [4]; iv) Actively using artificial intelligence and big data in the treatment of serious diseases [5]. The above approaches require complex data warehouses and digital platforms, with high construction and implementation costs. Thus, they were difficult to apply with real-time activity data and required the establishment of spatial and infrastructure conditions in advance. Wearable sensor technologies aimed to develop into a system that allowed doctors and nurses to monitor patients' health remotely and promptly give them useful advice. Wearable sensors [6] are sensors located directly or indirectly on the body to measure raw signals such as accelerometers, gyroscopes, angles, etc. from human activities. For the monitoring of activities, inertial sensors such as accelerometers and gyroscopes have been extensively employed. Gyroscopes detect changes in orientation, such as rotational displacement, velocity, and acceleration, whereas accelerometer sensors measure changes in velocity and displacement. Consequently, many researchers have been interested in building wearable devices based on

*Corresponding author. Email: nghiatd@ioit.ac.vn.

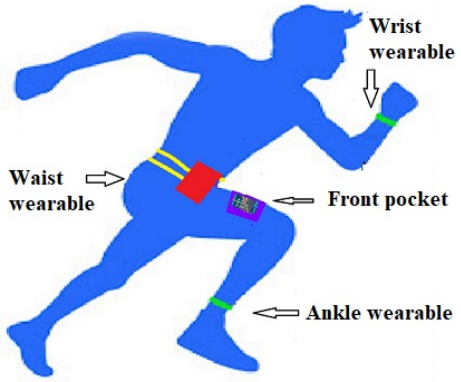


**Fig 1.** Wearable sensor locations on the body.

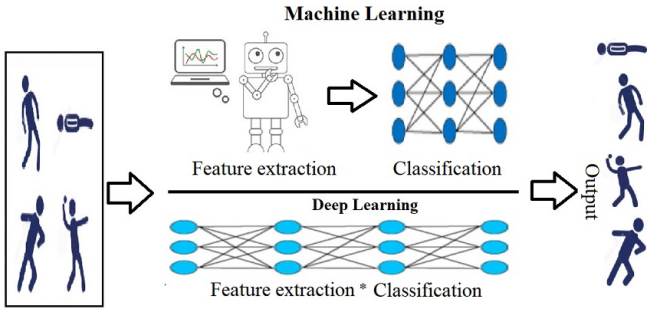


**Fig 2.** Machine learning algorithms were fed a set of relevant features for analysis, while deep learning algorithms worked with raw data and decided on their own the relevant features.

data obtained from inertial sensors to monitor daily activities [7–12].

Internet of things (IoT) applications [13–15] in the field of predicting human activities were growing strongly. Many works on human activity recognition (HAR) [16, 17] ways of increasing interest due to its availability in practical applications such as sports tracking systems [18]; monitoring systems and prevention of sedentary conditions at work in the office [19]. Moreover, sensors integrated with wearable devices have many advantages, such as cost savings, fast response time, cost savings, and low energy consumption, making them suitable for wearable applications [7, 12, 20–22].

Sensor placement (Fig 1) greatly affected the performance of the recognition model [6, 11, 12, 15, 19, 20, 23, 24]. Sensors must be attached at an appropriate location on the body. This was determined by the activities that required monitoring. Smart watches are usually worn on the left wrist. It is often less related to activities that do not use the hand, such as sitting, standing, lying down... Besides, the sensors attached to the waist help with overall body movement monitoring and major body movements. But data collected from waist circumference is often less sensitive to activities involving the use of hands, such as writing, holding, grasping, etc. Therefore, recent works [25–28] have combined many sensors in different locations to increase the amount of information collected about activities. However, carrying many wearable devices on the body could cause discomfort. In addition, wearable devices needed to be compact in size and light in weight to create a comfortable feeling for the users, especially for the elderly or those recovering from an accident. These devices were worn on the body during treatment to monitor the progress of daily activities and detect abnormal activities. An example: a sudden change in posture, such as lying down while walking. Not only that, healthy people could use wearable devices to monitor sports cycles, monitor living habits and determine energy consumption for activities.

The performance of the recognition model improved substantially when machine learning was utilized to identify human activities [3, 6, 7, 10–12, 15, 19, 24, 29]. While the device's integrated processors were low-power microcontrollers, they had memory and processing speed limitations. Besides, ML application recognition models must be optimised and compiled into the C/C++ language before being embedded in the microcontroller. Thus, choosing an embeddable machine learning algorithm on microcontrollers was a big challenge. Embedded devices typically have limited memory, processing power, power capacity, and more. Since embedded systems were designed for specific uses, there were limited resources left for machine learning models. To solve this problem, we built a human activity recognition model based on a machine learning algorithm with low complexity, small size, and embedding ability in microcontrollers.

Machine learning algorithms (Fig 2) based on past experience, observations, and data to predict future corresponding work instead of just following established rules pre-programmed. In general, there were four main approaches in the field of machine learning, including: 1) Supervised learning [30, 31]; 2) Unsupervised learning [32]; 3) Reinforcement learning [33]; 4) Semi-supervised learning [34]. With HAR, supervised learning algorithms would predict the label (activity) of a new descriptive dataset based on the correlation between that label and the previously known as descriptive data. In mathematics, a set of input data had the form $\chi = \{x_0, x_1, x_2, ..., x_n\}$ and a set of labels had the form $\gamma = \{y_0, y_1, x_2, ..., y_n\}$. Where, $x_i$ was a data descriptor vector for the label $y_i$ with $i = 1, 2, ..., N$. Data pairs of the form $(x_i, y_i) \in \chi \times \gamma$ were divided into training and testing datasets. On the training dataset, a correlation function was calculated to map the elements of the set $\chi$ to a corresponding

element of the set $\gamma$. From there, when new data $x$ was available, this function $f$ would predict the label $y$ corresponding to $y = f(x)$. However, this function did not exist in practice, so the function $f$ needed to be built well enough for the best classification performance so that $y \approx f(x)$. Building a good HAR model was influenced by many factors, including the quality of data describing the action, the size of the data window, features used, the machine learning algorithm applied, and the complexity of activities.

In addition, limited memory on wearables made it difficult to process large amounts of data and an increased number of activities. By using the dynamic windowing approach, our study has concentrated on enhancing data quality to capture the timeframe of state change operations. In this study, a device was designed like a belt, which can be worn on the waist in order to classify many activities and cause less discomfort for users. First, we collected accelerometer data for each activity and extracted a set of 31 features (time domain) per data window. Next, sets of features were divided into training and testing sets at the rate of 75/25. The random forest algorithm was applied to classify 13 routine activities. Finally, the appropriate model would be installed on a microcontroller with moderate performance (ESP32). Volunteers supported us in completing research to assess the usefulness of the dynamic window approach. Each device was able to transmit activity data to the data server via wifi or the internet. Through an application called "*Human Activity Classification*" that we provide to the Google Play store, users can track the activities of volunteers. The activity information was logged on the data server and backed up on the integrated memory card of each device. Each volunteer would wear a belt-mounted device. Since then, an internet-based remote activity monitoring system has been developed. With this method, we could evaluate the classification accuracy of wearable devices in a real-time environment. As a result, the experimental result was evaluated and compared with a number of related works on the classification of real-time HAR in the papers [12, 22, 35].

## 2. Related works

Various machine learning algorithms have been applied to build activity recognition models in many recent studies and achieved impressive results. Research by Mannini *et al.* [25] suggested using support vector machine (SVM) classifiers and activity data collected from sensors located at the ankle and wrist. The obtained results showed that the data collected at the ankle was better (10%) than the data collected at the wrist. Unlike them, Bali and co-authors [29] combined information including footsteps, gyroscope acceleration, and heart rate to recognize human activities. The features were extracted using the principal component analysis (PCA) method. Classification algorithms were used in C4.5, random forest (RF), K nearest neighbours algorithm (KNN), and SVM. Both the k-nearest neighbors and Nave Bayes classifiers were compared to the use of accelerometers and gyroscopes separately by Aiguo *et al.* [36]. We know from experiments that using both gyroscopes and accelerometers together improves categorization accuracy. Naive Bayes achieved a better overall accuracy (90.1% and 87.8%) than KNN. A large number of sensors could improve accuracy, but it would be impractical to have to wear them all the time. Adding more sensors would also make the system more expensive. Another study by Biagetti *et al.* [8] proposed a human activity recognition system consisting of wireless sensor network nodes (biological and accelerometer) and transmitted to a computer for data analysis. Results when applying the KNN classifier achieved an overall accuracy of 85.7%. In the study [35], Yang and Zhang propose a wearable operationally categorized system that resembles a wristwatch and is worn on the hand. The time and frequency domain characteristics of the accelerometer data are extracted, followed by the use of the decision tree method. On the STM32L low-power microcontroller, their modelling can be executed in real time. Despite the short number of activities (walking, sitting, jumping, bicycling, and jogging), the accuracy is below 90%. Five biaxial accelerometers were worn in a variety of positions on the study team by Bao *et al.* [37] including the hip, wrist, arm, ankle, and thigh. 20 activities were categorized with an accuracy of 84% by the decision tree classifier. The increased number of sensors, moving data, and accelerometer set to the designated orientation were the restriction, though. Many scientists in the last few years have looked into the possibility of employing a single accelerometer to collect the signal necessary for activity recognition [38]. Piyush Gupta *et al.* [38] used a belt-worn 3-axis accelerometer to construct an activity identification and feature selection system. Using Nave Bayes and KNN, they observed that wrapper-based feature selection was superior than filter-based. Data collection was limited to seven volunteers. All were young (22–28). Thu and co-authors [12] built a real-time recognition system for six activities on low-performance microcontrollers. Their system used two features (mean and standard deviation) in a combined decision tree algorithm. The result achieved above, 92%, was quite good, but this result was lower than the result achieved on their collected data (99%). The difference came from the data itself and the limited number of activities. For example: when switching from a static state (lying) to a dynamic state (sitting up) or from a dynamic state (sitting down) to a stationary state

(sitting), these activities could be confused with the activities in a state of walking or jogging.

## 3. System model

### 3.1. Wearable device

Frequently, the posture and velocity of human movements alter. Different activities produced different 3-axis acceleration values according to where the accelerometer was placed on the body [12, 23, 35]. In this research, accelerometer signals from the MPU6050 sensor were collected. Fig 3 shows the block diagram of the proposed system. The inertial sensor used was the MPU6050, capable of measuring 6 axes, including 3 axes of acceleration and 3 axes of gyroscope. The ESP32 was a power-saving microcontroller circuit that integrated Wi-Fi and Bluetooth. Additionally, the ESP32 was equipped with a 16MB flash memory, which was widely used in IOT applications. Besides, the ESP32 also supported the integration of embedding mini machine learning algorithms through integrated tools such as tensorflow, micropython. Therefore, this was the ideal choice when integrating machine learning algorithms such as random forest, support vector machine, et cetera. In this work, the central processor (ESP32) used I2C communication (inter-integrated circuit) with MPU6050 and DS1307 (time integrated circuit) to collect acceleration data over time at a sampling rate of 20 Hz (20 samples per second).

A human activity usually lasts from 2 seconds to 3 seconds and this time is longer in the elderly or people who have just recovered from an accident. Besides, a sampling frequency of 50Hz or more did not give a better result [36], and the performance of the device depended on the accuracy when combining the classification algorithm with data features. Acceleration values were calculated according to the formula (1). The source of use of the wearable device was a 3.7V-2000 mah lithium battery, so the battery power it could provide was $3.7V \times 2000maH = 7400mWh$.

### 3.2. Energy consumption

The average power consumption of the device in each working hour was about 60mWh. The device could work for up to $7400 : 60 \approx 123.33$ hours, which is equivalent to 5.14 days.

$$A_i = \frac{\frac{Sam_i}{1024} \times R - O_i}{S_i} \quad (1)$$

In that, $A_i$ was the acceleration value in the direction $i$; $Sam_i$ was the sampled value on the $i$-axis; $R$ was the reference suspension resistor; $O_i$ was compensation and $S_i$ was the sensitivity.

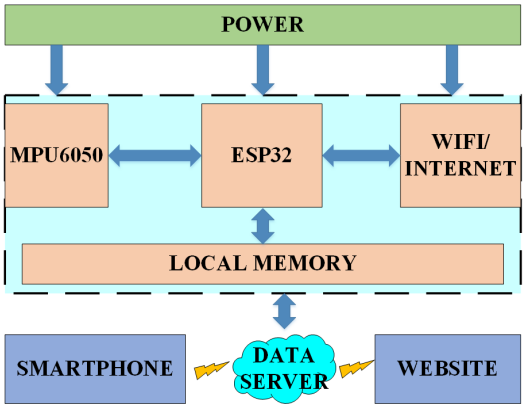


**Fig 3.** Structure of the wirelessly connected wearable device.

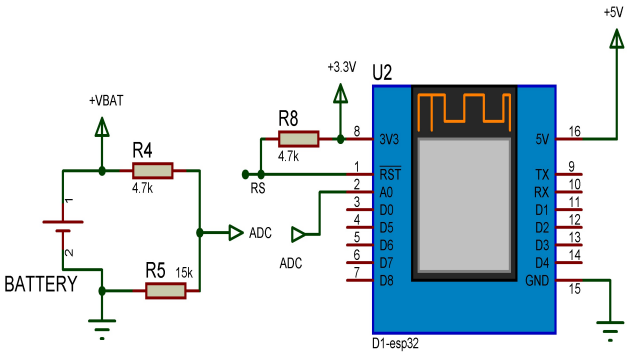


**Fig 4.** A voltage divider circuit diagram for battery power calculation.

In order to limit power depletion during operation, the device needed to be charged after a period of continuous operation. The alarm information was sent to users via the mobile application (Fig 5 left), and the alarm sounds from the device. Because the battery voltage (VBAT) was between 3.7V and 4.2V but the microcontroller's maximum withstand voltage was 3.3V, the battery voltage signal was passed through the voltage divider circuit (Fig 4). Next, the capacity of the battery was calculated by formula (2) as a percentage. As a result, the battery reached 100% when the battery voltage was 4.2V. The device stopped working when the battery voltage dropped to 3.7V, or 0%. On the phone application, the capacity warning icon needed to be charged when the battery capacity was below 10%.

$$Per_{pin} = \left( \frac{adc * V_{ref}}{2^{re} - 1} \times \frac{R_4 + R_5}{R_5} - 3.7 \right) \times \frac{100\%}{0.5} \quad (2)$$

### 3.3. Mobile monitoring application

To collect activity data, the wearable device was configured as a mini server, and users could connect

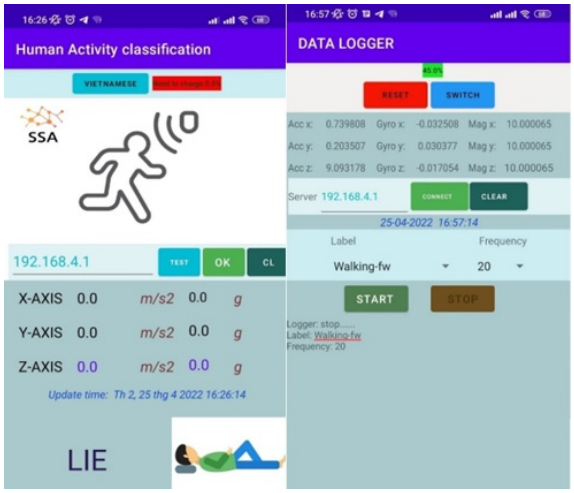


**Fig 5.** Low battery warning on the app (left) and data logger application interface (right).

to the device via wifi thanks to an interactive app called "*Accelerometer Gyrometer Logger*". It is publicly shared on the Google Play store. This data was stored on 2GB of expandable local memory, and this memory communicated with the ESP32 via a standard serial peripheral interface (SPI). Each collecting and monitoring device was a node with its own address code and communicated with the data server by using the HTTP/GET protocol. Activity data was saved on the MySQL database.

The application interface (Fig 5 right) provided users with functions including: setting the sampling frequency; labels corresponding to activity to be collected; 3-axis accelerometer and 3-axis gyroscope; function to delete the collected data when users made a mistake in the operation or the data errors; battery capacity notifications; and the logging function of the collected data was controlled directly via the START/STOP function button. Besides interacting with the software, users could enable or stop the data logging function by using a physical button on the device. In addition, pressing the button for 5 seconds could delete the recorded data. After that, the device would delete the file and reboot.

## 4. Research methods

### 4.1. Sampling and pre-processing

**Public dataset.** The activities of daily living (ADLs) dataset introduced in work [37] described 36 movements, including 20 falls and 16 daily living activities. Data of activities were collected by 17 volunteers (10 males and 7 females). Each volunteer performed each activity in turn, and each action consisted of 5 or 6 tests between 30 seconds and 60 seconds. When performing

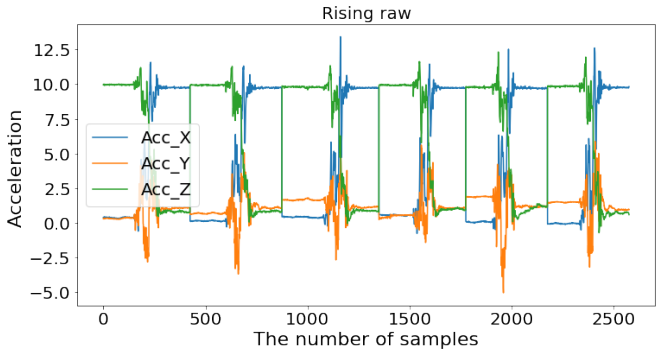


**Fig 6.** Six tests with the activity of rising.

the activities, each volunteer would wear 6 devices at 6 corresponding positions: head, chest, waist, right wrist, right, and right ankle. The collection device included three sensors: an accelerometer, a gyroscope, and a magnetometer with a sampling frequency of 25Hz.

Researchers [23, 38] investigated the fall state based on this dataset and showed that the sensor data collected at the waist position gave the best result. However, the data included a number of similar activities, so it was not necessary to distinguish the details. For example, sit-down on the bed, chair, in the air, or on the sofa. Thus, we would combine these activities into a common action. For example: walking forward and backward into walking; sit-down on different surfaces such as sit-air, sit-bed, sit-sofa, sit-chair into sit-down; tripping, sliding but controlled states into tripover; coughing-sneezing.

The data collected in the public dataset needed to be processed to improve classification accuracy. Fig 6 shows the rising activity after 6 tests over a period of more than 250 seconds, with acc_x, acc_y, and acc_z corresponding to acceleration on the x, y, and z axes. With a sampling frequency of 25Hz, the data of the tests was approximately 430 samples. In the first 150 data samples, the volunteer was horizontal, barely moving (static state), then he sat up for the next 2-3 seconds, and finally sat motionless (static state).

Signal statistics in these tests showed the information of activity was concentrated in signal regions on three x, y or z axes and had a range of greater than $0.5m/s^2$. Therefore, we determined the activity state was static or dynamic on each signal segment of size 1 second (time windows). After reducing the activity data in the static state, the descriptive information of the actions was more concentrated (Fig 7) than before processing.

Besides, non-normal values would be replaced by interpolated values, which were determined by averaging the adjacent signal magnitudes before and after those values. The above process was applied similarly to the remaining activities.

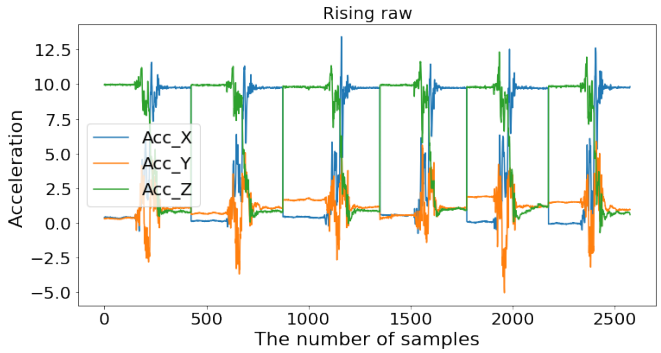


**Fig 7.** Rising after pre-processing.

The public dataset contained the data for activities that follow the same process: state-dynamic-static but including repetitive and non-repeating activities. For example, lying down (lie-down) was a non-repetitive activity, volunteers had to rise and then did it again, similar to activities like sit-down, rising, tripover, and squatting. In contrast, activities such as walking, jogging, and limp could be performed continuously. Therefore, in order to improve the classification performance, we classified three activities in a static state, including sitting, standing, and lying. The data for these activities was the data areas separated from the data processing of non-repeating activities. For example, the descriptive data for two activities sitting and lying were the data areas in the static state before and after the lie-down activity was performed. Similarly, the data of standing was the signal area before the sit-down activity took place. However, the timing of the activities was not exactly the same. For example, the squatting activity had a duration of up to 5 seconds, while the transitions such as lie-down, rising, and sit-down had a duration of 3 seconds. Unlike them, activities that take place when a person moves, such as jogging, walking, tripover, and limp only need 2 seconds to be classified. Therefore, we proposed using different sized windows for each activity in order to provide better activity information. 13 activities were presented in Tab 1.

**Tab 1.** Description activities.

| Activity | Description | Duration time(s) |
|---|---|---|
| Walking | Walking normally, moving slowly, walking forward and backward. | 2 |
| Jogging | Jogging and running. | 2 |
| Squatting | Squatting down then standing up. | 5 |
| Bending | Bending about 90 degrees. | 1 |
| Bendp | Bending to pick up an object on the floor | 4 |
| Limp | Walking with a limp. | 2 |
| Tripover | Tripping or stumbling but then stand up and walking normally | 2 |
| Sit-down | Certain acceleration onto a chair, a sofa, air or bed. | 3 |
| Lie-down | Sitting to lying on the bed. | 3 |
| Rising | From lying to sitting | 4 |
| Lying | Lying horizontally, can turn around horizontally. | 1 |
| Standing | Stand up and standing straight, coughing or sneezing while standing. | 1 |
| Sitting | Sitting motionless, back against the chair and body slightly tilted back. | 1 |


**Fig 8.** The wearable device (red) was attached to the waist and secured by an elasticated strap. The x-axis of the accelerometer was parallel to the earth's gravity.

**Private dataset.** The private dataset was built based on a group of volunteers wearing waist data collection devices (Fig 8) and performing the following activities: walking, jogging, squatting, bending, bend-p, limp, tripover, sit-down, lie-down, rising, lying, standing, and sitting. This group consisted of 20 students, including 12 males and 8 females. The recorded data had the following format: activity, timestamp, sensor, values of x, y, and z, frequency. Activity logger files with a type of TXT were named with the activity and the time when it started to create the file. When performing data collection, activities such as sit-down, lie-down, bendp, squatting, and rising were performed as an unwritten rule: nothing before starting in the short time, doing activity, nothing after finishing. For example, rising

included lying-rising-standing; lie-down included from sitting-lying down to bed-lying.

The process of data collection activities was applied at a different collection time. We set the survey time to 3 seconds for non-repetitive activities (squatting, bendp, tripover, lie-down, sit-down, rising). Each activity took place for a period of 3 seconds and was repeated in subsequent discrete intervals. This would help our data be centralised and avoid loss when activity data is divided into time windows. The remaining activities, such as bending, limp, sitting, standing, lying, walking, jogging were surveyed at continuous intervals (30 seconds to 50 seconds). Collected data would be segmented into 3 seconds fixed size windows because this dataset has been sampled for limited periods of time.

## 4.2. Data transformation

Data segmentation was one of the stages of the receiving process. This phase allowed us to understand the impact of signals by dividing activity data after preprocessing. The data window had two approaches, including dynamic and static windows.

**Static window size.** With this windowing approach, the data of activities were segmented into equally sized data windows as shown in Fig 9. The size of windows was an important parameter for gathering a lot of information in this approach. As the size of the window changed, the amount of information changed accordingly. If the window size was too large, the processor would take too long to process the information, and there was a chance that information from other activities might appear. In addition, choosing a window with a short size caught information loss or misinformation with transitions occurring in a short time. Therefore, this approach was suitable for the classification of activities in a stationary state (sitting, standing, lying, bending) and repetitive activities for a long time (jogging, walking, limp).

Datasets used after preprocessing, removing noise, would be segmented and labelled. The data segmentation was done on the computer, so the static windowing method was applied at this stage. However, the size of the window would depend on how long each activity takes place. The window size applied to each activity in the public dataset was shown in Tab 1. For example, lie-down and sit-down were 3 seconds, squatting was 5 seconds, and rising was 4 seconds. Slightly different from public data, activity data in private data was segmented into fixed-sized windows of 3 seconds. The number of observations after data segmentation was presented in Tab 2.

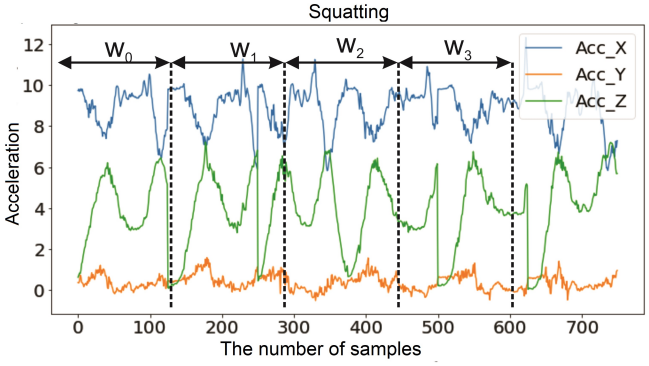


**Fig 9.** The static window method would divide an activity's data into sliding windows of the same size over time.

**Tab 2.** Statistics of data windows for each activity on the dataset.

| Activity | The number of windows | |
|---|---|---|
| | Public dataset | Private dataset |
| Walking | 1010 | 132 |
| Jogging | 321 | 76 |
| Squatting | 82 | 101 |
| Bending | 538 | 58 |
| Bendp | 91 | 106 |
| Limp | 439 | 121 |
| Tripover | 193 | 138 |
| Sit-down | 370 | 129 |
| Lie-down | 125 | 74 |
| Rising | 114 | 89 |
| Lying | 971 | 115 |
| Standing | 882 | 113 |
| Sitting | 1046 | 118 |

**Dynamic window size.** With this approach, the window size was not constant with activity. In Fig 10, the data of squatting (squat down then stand up) was split into different sized windows to detect when this activity took place. In practice, activities took place at different time intervals, and applying dynamic windowing was necessary to improve activity predictability. Therefore, we applied the dynamic window method to the proposed model in conducting experiments.

When conducting experiments, we used windows of size 1 second to determine the static state or the dynamic state. Static and dynamic windows were determined by calculating range ($H$) by formula (9) and average resultant acceleration ($ARA$) by formula (12). Where ARA was the average of the square root of the sum of the squares of the signal strength on three axes. Research in [39] collected breathing data when a person was in a stationary state. This state was determined by X. Sun by calculating the average resultant acceleration for the 3-axis acceleration measured on the smartwatch. Accordingly, when a person was not moving, the calculation results were

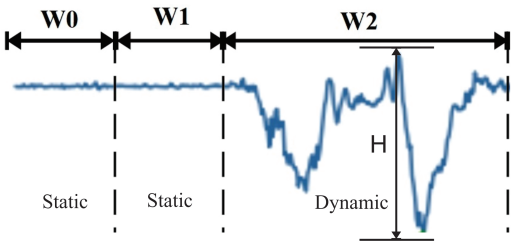


**Fig 10.** Data correlation window.

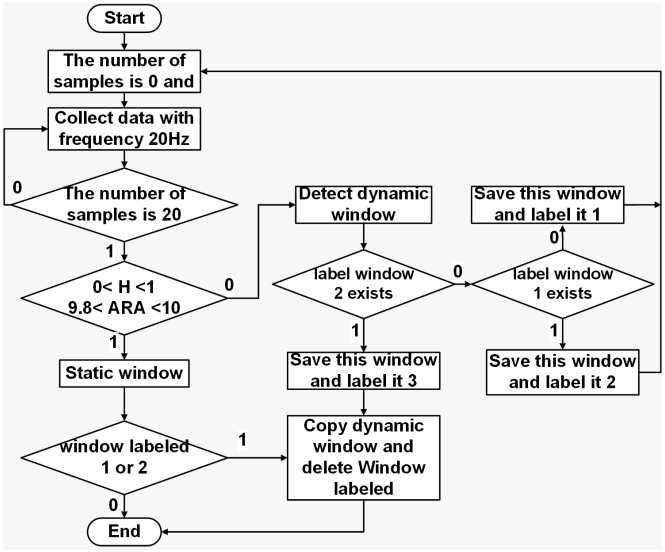


**Fig 11.** After analyzing every second of data for signal changes, the algorithm identified dynamic windows after 1–3 seconds.

approximately $10m/s^2$. Therefore, a time window was defined as a static state if two conditions were satisfied: $H$ runs from 0 to $1m/s^2$ and $ARA$ runs from 9.6 to $10m/s^2$.

The window size changed from 1-3 seconds had a great influence on the classification results shown in the study [40]. Therefore, in this study, we limited the maximum number of seconds of data to 3 seconds to identify an activity. This was understood that the data window size would be automatically changed for each activity until the next activity is in a static state or the time between two consecutive activities should not exceed 3 seconds. This algorithm is depicted in Fig 11.

## 4.3. Feature engineering

After converting to time windows, the data was still difficult to apply directly to the machine learning algorithm and not the entire sample of activity data needed for the classification process. Therefore, the data of activities was transformed into a new dataset consisting of selected features in the time domain compatible with the classification algorithms. These features reflected the correlation between the original data and each activity and helped to improve the accuracy of the classification model. This work considered 12 statistical features including: mean ($\mu$), standard deviation ($\sigma$), median ($Med$), average absolute difference ($AAD$), maximum value ($Max$), minimum value ($Min$), range, root mean square ($RMS$), correlation coefficient ($Corr$), average resultant acceleration ($AAD$), correlation between the average of the original signal series ($\mu_{start}$) and the average of the final signal series ($\mu_{stop}$) over a time window ($CAOSS$). These features were extracted in three axes (x, y, z) based on the respective formulas (3)-(13).

$$\mu_{X_j} = \frac{1}{N_W - 1} \sum_{i=0}^{N_W - 1} x_i \tag{3}$$

$$\sigma_{X_j} = \sqrt{\frac{1}{N_W - 1} \sum_{i=0}^{N_W - 1} (x_i - \mu_{X_j})^2} \tag{4}$$

$$Med_{X_j} = x_{\frac{N_W}{2}} \quad \text{with } X_j \text{ was sorted} \tag{5}$$

$$AAD_{X_j} = \frac{1}{N_W - 1} \sum_{i=0}^{N_W - 1} |x_i - \mu_{X_j}| \tag{6}$$

$$Min_{X_j} = Minimum(x[i]) \tag{7}$$

$$Max_{X_j} = Maximum(x[i]) \tag{8}$$

$$Range_{X_j} = Max_{X_j} - Min_{X_j} \tag{9}$$

$$RMS_{X_j} = \frac{\sum_{i=0}^{N_W - 1} |x_i|^2}{N_W - 1} \tag{10}$$

$$Corr(X_j, Y_j) = \frac{\frac{1}{N_W - 1} \sum_{i=0}^{N_W - 1} (x_i - \mu_{X_j})(y_i - \mu_{Y_j})}{\sigma_{X_j} \sigma_{Y_j}} \tag{11}$$

$$ARA_{X_j} = \frac{1}{N_W - 1} \sqrt{\sum_{i=0}^{N_W} ({x_i}^2 + {y_i}^2 + {z_i}^2)} \tag{12}$$

$$CAOSS = \mu_{stop} - \mu_{start} \tag{13}$$

In that, $X_j$ was jth data segment $j$; $x_i$ , $y_i$, $z_i$ were $i^{th}$ acceleration values on the 3 x, y, and z axes on the $X_j$; $N_w$ was the number of samples on the window $X_j$. Correlation coefficient ($Corr(A_j, B_j)$) was a function that calculated correlation between pairs of axes: $(x, y)$, $(y, z)$ và $(x, z)$. In $CAOSS$, $\mu_{start}$ and $\mu_{stop}$ were the average values of signals on each x, y, and z axis into the first and last second, respectively. The T-distributed Stochastic Neighbour Embedding (t-SNE) [3, 7, 12] tool made it simple to see how these features affected the

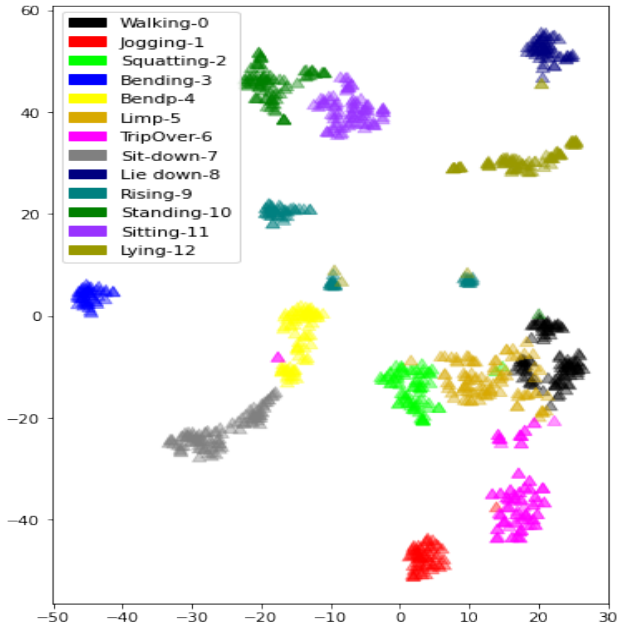


**Fig 12.** Activities might be easily recognized by the use of features. There was some confusion between the various activities in the moving state, though.

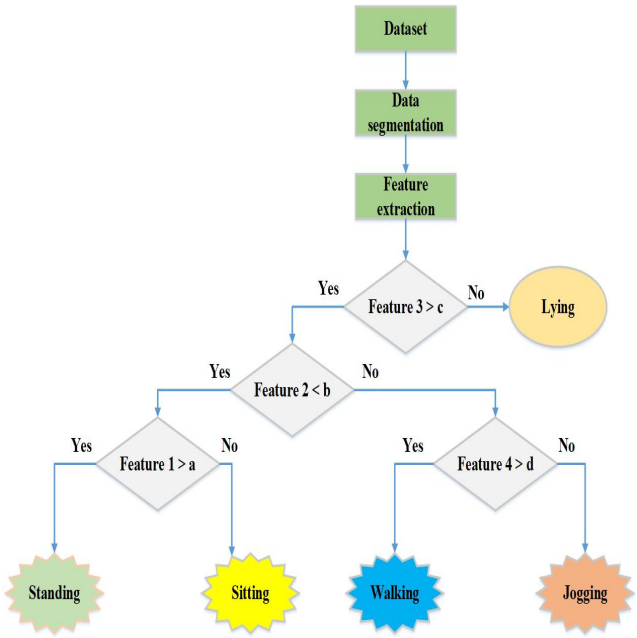


**Fig 13.** The decision tree would build relationships between features and activities.

system (Fig 12). To visualize high-dimensional data, there was a technique called t-SNE. By converting similarity between data points into joint probabilities, it attempted to reduce the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data. Although there were up to 13 activities that needed to be classified, the number of activities that were confused was minimal. The activities in the moving state were where the confusion was most prevalent.

A feature vector with 31 dimensions was calculated from 13 measurements on the x, y, and z axes of the MPU6050 sensor (acceleration). These features were extracted from each data segment (time window) of each activity. Then, the features were merged and created into a feature vector representing the corresponding data segment. These feature vectors would be divided into training and testing sets at the rate of 75/25.

## 4.4. Recognition model

**Random forest.** While most data processing workloads with machine learning algorithms run in the cloud, there is another trend towards on-device machine learning algorithms (on-device-ML). On-device-ML means deploying machine learning models on embedded devices for direct inference at the real-time data source. Because the prediction and classification of activities take place directly, these devices did not need an external network and ensured that sensitive data was not revealed on the public internet. Cloud-based ML requires embedded devices to send data to the cloud for inference. As a result, the cloud-based computational model is virtually unlimited, but causes latency. There is always a delay when transferring data to and from embedded devices. Embedded systems are often deployed in locations where connectivity is limited, so it is not practical to operate ML for human activity recognition systems as well as the reliability of data in wireless transmission. Not all machine learning algorithms could be applied to microcontrollers; they needed to be miniaturized and optimized to run on low-power devices without too much loss of accuracy. Machine learning algorithms of suitable complexity and size could be embedded on microcontrollers, including decision tree (DT) and random forest (RF). Decision tree [41] was an algorithm that could imitate human thinking. This algorithm was based on the correlation between data to understand the logic between input and output data (Fig 13). Each tree node (rhombic) represented a data feature threshold, each branch represented a rule, and each leaf (ellipse) represented an activity or prediction label.

Decision tree models often suffer from the overfitting problem that leads to false predictions. To fit the data, it kept creating new nodes, and eventually the tree became too complex. Hence, it worked greatly on the training data but started making a lot of errors on the testing data. These algorithms are often

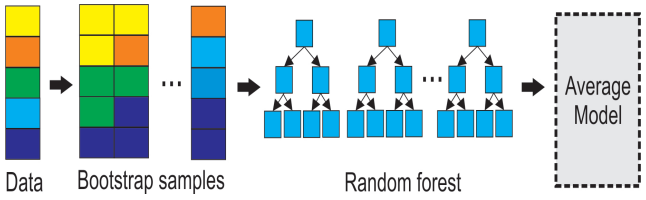


**Fig 14.** Many single decision trees are built that will work with random data from a dataset. Thus, the results of object prediction were aggregated based on the majority rule from these trees.

unstable when adding a new layer of data and are not suitable for large datasets. Thus, it was difficult to apply to microcontrollers when the number of output predictions was large. Furthermore, not all of the chosen features are appropriate for activity. This feature might be effective for one activity but not for another.

Multiple decision trees were gathered using the random forest technique to create a more robust model (Fig 14). This algorithm uses the bootstrap technique [42] to randomly select data from the dataset (random sampling with replacement). As a result, each new dataset might be duplicated, and a new decision tree would be built on this dataset. The process of building for each decision tree had a random element, so the decision trees in the RF algorithm might be different. The output prediction results that were aggregated from many independent decision trees will give optimal results.

Features are involved in and have an impact on the outcome of activity prediction with RF. In this research, the recognition model using the random forest algorithm was built on the training data. The testing data yielded the best recognition model, which would then be converted into C/C++ and integrated into the ESP32.

**Human activity recognition using a random forest algorithm for wearable devices.** The feature set obtained after feature extraction would be randomly divided into a training set and a test set at a ratio of 75/25. The recognition model was trained on a labeled training dataset corresponding to each feature vector. Then, the classification result based on the test data would help to evaluate the built model. For the random forest algorithm, we used the publicly shared sklearn library to build the model. Besides, we used a parameter n_estimators to set the number of single decision trees, and each branch of a decision tree would work with 1 feature. In this work, our parameter set was 50. Fig 15 shows the process of building a classification algorithm on a wearable device.

To build the best recognition model and achieve high accuracy, the model was trained on the computer, and the best model was transformed to be embedded in

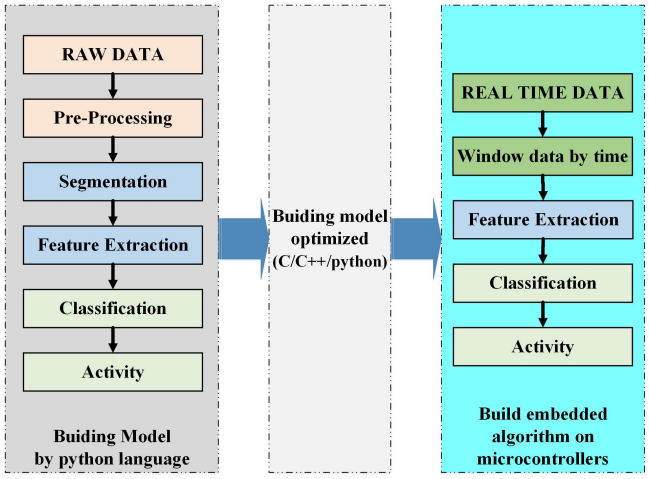


**Fig 15.** Construction diagram of the recognition system.

the ESP32 microcontroller. The computer that had been used to train the model and operate the algorithm had the following configuration: Dell G515, 16GB ram, Intel (R) Core I7-9750H processor, 256GB SSD, 1TB HDD. The random forest model applied to the ESP32 was illustrated in Fig 16. Activity 1 or activity 2 represented 1 of 13 daily activities. The final prediction result for an activity was based on the prediction results of a majority of individual decision trees. The recognition model that gave the best results on the test dataset was compiled into a library named "RF_acc.h". This was included in the embedded executable on the ESP32 processor. Accordingly, the ESP32 integrated into the wearable would read the acceleration signal from the MPU6050. After collecting enough data samples, ESP32 would extract a vector of 31 features from each window and perform classification based on this feature vector. This process was performed every 1 second, corresponding to 20 data samples. If 1 second windows were static, the system would classify the activity immediately and continue to classify every 1 second, followed by the same rule. Conversely, if the current window was in a dynamic state, the processor would save 1 second of previous data and continue to consider data windows in the next 1 data second. The human activity recognition system would result when a data window size reaches a maximum interval of 3s or a next 1 second window in a static state. Thanks to the soft change of the data sample size (dynamic window size), activities were detected faster with less delay. At the same time, our system could understand the start time and end time of a forward activity (lie-down, sit-down, rising, bendp, tripover).

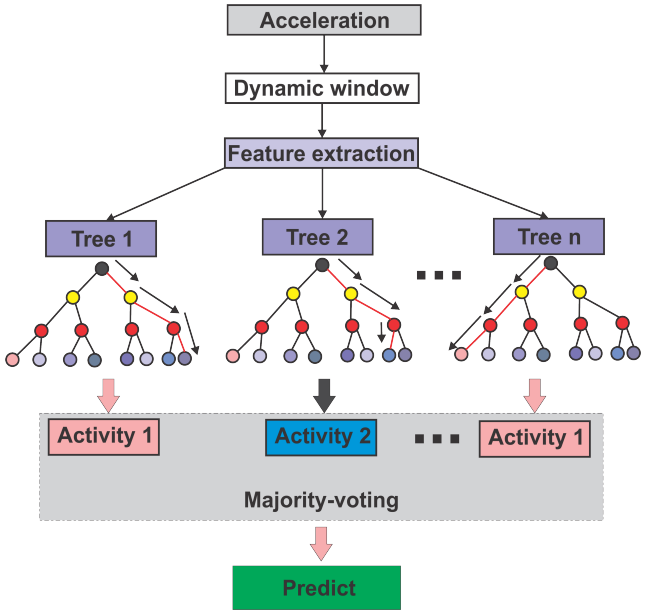


**Fig 16.** Random forest was implemented on the wearable.

## 4.5. Evaluation indicators

**Confusion matrix.**

$$ACC = \frac{TP + TN}{TP + FP + FN + TN} \tag{14}$$

$$SEN = \frac{TP}{TP + FN} \tag{15}$$

$$PPV = \frac{TP}{TP + FP} \tag{16}$$

$$NPV = \frac{TN}{TN + FN} \tag{17}$$

The recognition models in this study were evaluated based on the following indicators: accuracy ($ACC$), sensitivity ($SEN$), negative predictive value ($NPV$), and positive predictive value ($PPV$). These indexes were calculated based on the parameters including true positive ($TP$), true negative ($TN$), false positive ($FP$), and false negative ($FN$) according to the corresponding formula (14)-(17).

Considering an activity of sitting, $TP$ was the number of these activities that were correctly classified as compared to the actual label, $FN$ was the number of these activities that were misclassified, $FP$ was the number of other activities that were mistakenly classified as sitting, and $TN$ was the number of activities other than sitting that were correctly classified. A specific example of the definition of parameters $TP$, $TN$, $FP$, and $FN$ is presented in Tab 3. In this example, Sitting has $TP = 100$, $FN = 3$, $FP = 7$, and $TN = 120$; Standing has $TP = 50$, $FN = 6$, $FP = 2$, and $TN = 170$.

**Tab 3.** Example about evaluation parameters with sitting, standing, and bending.

| | | Human activity | | |
|---|---|---|---|---|
| | | Sitting | Standing | Bending |
| **Predicted** | Sitting | 100 | 2 | 1 |
| | Standing | 4 | 50 | 2 |
| | Bending | 3 | 0 | 70 |

**Cross validation.** The input data must be large to achieve a good type of recognition model. However, this was unlikely because it could not determine how much data was needed. The K-fold cross validation was applied to provide a lot of data to train the client to validate the model with multiple tests simultaneously. This method would apply to proposed models and other machine learning models such as decision tree (DT), support vector machine (SVM), K nearest neighbours (KNN), gradient boosted decision tree (GBDT). With this method, the data was divided into $k$ equal parts. In which, training data was used $(k-1)$ parts, the rest was testing data. The model output was evaluated based on the results, including the mean $(\mu)$ and standard deviation $(\sigma)$ of $k$ times of data divisions. This research applied cross-validation with $k = 5$ in the evaluation process.

# 5. Results

## 5.1. Model performance

**Performance evaluation on the public dataset.** The classification result of the proposed model was presented in the form of a confusion matrix (Fig 17). The result of applying the best recognition model on the test data (public dataset) reached 96.5%. Activities were properly classified with maximum ratios like bending (134/134), bendp (22/22), lie-down (31/31), standing (220/220), sitting (261/261), lying (242/242). Slightly lower result were the activities of limp (91/109), tripover (38/47), rising (27/28), sit-down (88/91), walking (244/252), jogging (77/80), and the lowest was squatting (13/20).

Activities were misclassified often due to too fast or too long execution time or similarity to other activities. For example, when a person was walking on crutches with an injured leg, the movement speed was so slow that it could be mistaken for standing. Squatting was a combination of sitting down, sitting, standing up, and standing for a period of time. This usually occurs when a person exercises (gym). However, if this took a long time, squatting could be mistaken for sit-down. With the public dataset, sit-down activity (6/20) was mistaken for squatting. In addition, the activities of limp and walking were confused with each other due

**Tab 4.** Evaluation of applying the RF algorithm on the public dataset.

| Activity | ACC(%) | SPE(%) | PPV(%) | NPV(%) |
|---|---|---|---|---|
| Walking | 98.1 | 96.8 | 92.1 | 99.4 |
| Jogging | 99.4 | 96.2 | 92.8 | 99.8 |
| Squatting | 99.5 | 65 | 100 | 99.5 |
| Bending | 100 | 100 | 100 | 100 |
| Bendp | 100 | 100 | 100 | 100 |
| Limp | 98.2 | 83.5 | 90.1 | 98.7 |
| Tripover | 99.1 | 80.9 | 90.5 | 99.4 |
| Sit-down | 99.4 | 96.7 | 93.6 | 99.8 |
| Lie-down | 100 | 100 | 100 | 100 |
| Rising | 99.9 | 96.4 | 100 | 99.9 |
| Standing | 99.9 | 100 | 99.1 | 100 |
| Sitting | 100 | 100 | 100 | 100 |
| Lying | 100 | 100 | 100 | 100 |
| **All** | **99.5** | **93.5** | **96.8** | **99.7** |

to the similarity in posture or in people with minor injuries in the leg, less hindering movement. Some other activities were mistakenly classified as walking, but the number of these activities was insignificant. For example, 2/80 jogging, 3/47 tripover, 1/91 sit-down, and 1/28 rising were mistaken for walking.

Details of the performance evaluation of the proposed recognition model on the public data set are described in Tab 4. The two indexes, $ACC$ and $NPV$, of all activities, were 98%. In particular, activities such as sitting, bending, bendp, lie-down had the indexes of $ACC$, $SPE$, $PPV$, and $NPV$ were 100%. This was followed by the figure for squatting, limp and tripover activities, with $ACC$, $PPV$, and $NPV$ over 90%. However, the difference lies in the $SPE$ index. The activities of limp and tripover had a $SPE$ index of over 80%, while squatting was only 65%. This was consistent with our analysis of how long these activities take. Overall, the proposed model had the indexes of $ACC$, $SPE$, $PPV$, and $NPV$ indexes over 93%, which was good.

The result of the classification of activities when applying 5-folds cross validation on the public dataset was good (Tab 5). The proposed model had the best result, stood at $\mu = 96\%$. This was greater than the results for algorithms of GBDT, KNN and SVM, making up 95.3%, 94.9%, and 94.2%, respectively. Yet, the result for DT was the lowest, standing at 92.6%. Besides, the recognition model using GBDT algorithm gives the lowest standard deviation $\sigma = 0.4\%$. The classification results had a low standard deviation of 0.4%-0.7%. This showed that the data from activities was less scattered and had high reliability.

**Tab 5.** Evaluation of recognition models with 5-fold cross-validation.

| | | RF (%) | DT (%) | GBDT (%) | SVM (%) | KNN (%) |
|---|---|---|---|---|---|---|
| Public dataset | $\mu$ | 96.0 | 92.6 | 95.3 | 94.2 | 94.9 |
| | $\sigma$ | 0.5 | 0.7 | 0.4 | 0.6 | 0.6 |
| Private dataset | $\mu$ | 99.1 | 97.0 | 98.0 | 98.4 | 97.4 |
| | $\sigma$ | 0.6 | 0.8 | 0.8 | 0.5 | 0.8 |

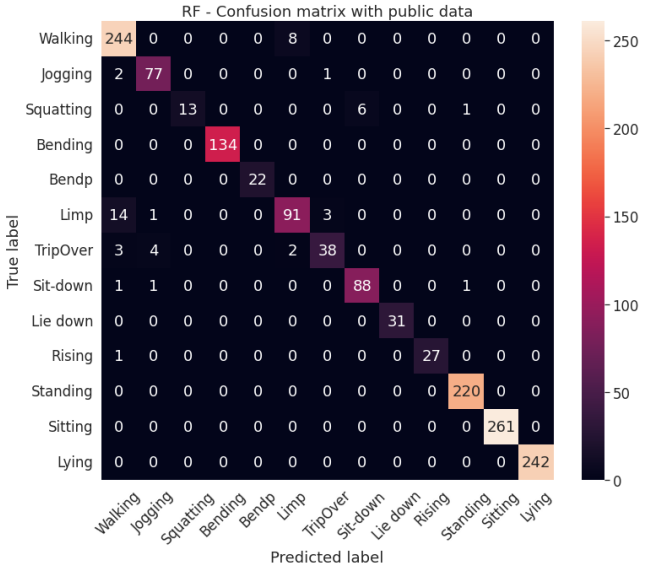


**Fig 17.** Classification result for RF algorithm on public data.

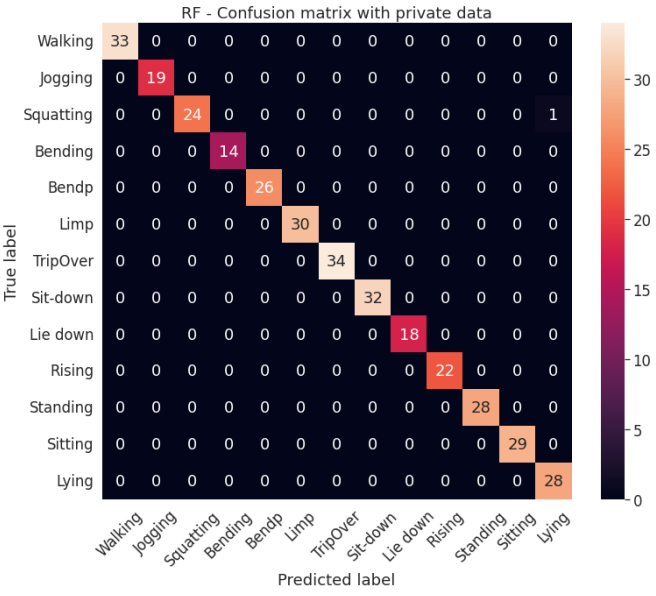


**Fig 18.** Classification result for RF algorithm on the private dataset.

**Tab 6.** Result of applying the RF on the private dataset.

| Activity | ACC(%) | SPE(%) | PPV(%) | NPV(%) |
|---|---|---|---|---|
| Walking | 100 | 100 | 100 | 100 |
| Jogging | 100 | 100 | 100 | 100 |
| Squatting | 99.7 | 96.0 | 100 | 99.7 |
| Bending | 100 | 100 | 100 | 100 |
| Bendp | 100 | 100 | 100 | 100 |
| Limp | 100 | 100 | 100 | 100 |
| Tripover | 100 | 100 | 100 | 100 |
| Sit-down | 100 | 100 | 100 | 100 |
| Lie-down | 100 | 100 | 100 | 100 |
| Rising | 100 | 100 | 100 | 100 |
| Standing | 100 | 100 | 100 | 100 |
| Sitting | 100 | 100 | 100 | 100 |
| Lying | 99.7 | 100 | 96.6 | 100 |
| **All** | **100** | **99.7** | **99.7** | **100** |

**Tab 7.** Evaluation of experimental result.

| Activity | ACC(%) | SPE(%) | PPV(%) | NPV(%) |
|---|---|---|---|---|
| Walking | 98.8 | 98.4 | 89.6 | 99.8 |
| Jogging | 99.3 | 95.1 | 96.7 | 99.5 |
| Squatting | 99.4 | 93.4 | 100 | 99.4 |
| Bending | 99.6 | 100 | 94.3 | 100 |
| Bendp | 99.3 | 93.0 | 98.1 | 99.4 |
| Limp | 99.4 | 96.6 | 96.6 | 99.7 |
| Tripover | 99.0 | 90.6 | 96.0 | 99.2 |
| Sit-down | 99.8 | 100 | 98.0 | 100 |
| Lie-down | 99.7 | 96.2 | 100 | 99.7 |
| Rising | 99.9 | 100 | 97.9 | 100 |
| Standing | 99.3 | 94.5 | 96.3 | 99.5 |
| Sitting | 99.1 | 96.5 | 93.2 | 99.7 |
| Lying | 99.6 | 100 | 94.4 | 100 |
| **All** | **99.4** | **96.5** | **96.2** | **99.7** |

**Performance evaluation on the private dataset.** The private dataset gave a result of 99.7%. True to our previous analysis, the limited time (3 seconds) method of collecting activity data has helped our data to be highly reliable. The negative impacts on classification results were minimised. These negative impacts were caused for three reasons: 1) One other activity interfered when an activity took place over a long period of time; 2) The time difference of an activity with each volunteer; 3) The data sampling process had not been well controlled leading to excess or lack of activity data. These catches especially affect transitions such as: rising, lie-down, sit-down, tripover, bending pick up (bendp), squatting. If the timing of these activities could not be unknown, the data describing them threatened to degrade the performance of recognition models. Fig 18 shows most of the actions distinguished with a 100% accuracy rate, except for squatting (24/25). However, the classification results obtained were impressive. The problems encountered when classifying on the public dataset had almost been solved.

The classification results on the private dataset showed the strong performance of the proposed model. The evaluation indexes of $ACC$, $SPE$, $PPV$, and $NPV$ all reached over 96% (Tab 6). Except for squatting and lying, all activities had indicators reaching 100%. This was greater than the result for squatting with $ACC$ = 99.7%, $SPE$ = 96%, $PPV$ = 100%, $NPV$ = 99.7%, and lying with $ACC$ = 99.7%, $SPE$ = 100%, $PPV$ = 96.6%, $NPV$ = 100%. However, the results of the recognition model evaluation on this dataset was impressive with $ACC$ = 100% and $NPV$ = 100%.

The results when applying 5-folds cross validation on this dataset were also gradually different when they were applied on the public dataset. The proposed model got the best results with 99.1%, but the standard deviation increased to 0.6%. Other recognition models such as GBDT, SVM, KNN, and DT give results of 98%, 98.4%, 97.4%, and 97%, respectively. Similar to the proposed model, the recognition models applying algorithms such as DT, GBDT, and KNN had their standard deviation increase by 0.8%. Unlike that, the model applying the SVM algorithm had the standard deviation reduced to 0.5%. However, the resulting difference from 0.5%-0.8% was not significant. Overall, the results of the proposed model evaluation on both datasets were good.

## 5.2. Experimental evaluation

The experimental process was conducted on volunteers for a period of 30 seconds to 60 seconds, and the sampling frequency was 20Hz. Volunteers wore wearables suggested and performed all 13 activities according to a predefined scenario. Transitions such as sit-down, lie-down, rising, tripover, and bendp were limited to a maximum execution time of 3 seconds. The remaining activities took place in 10-15 seconds. A portion of the experimental procedure for finding mixed activity sequences was shown in Fig 19. Since we concentrated on recognizing when activities were started in a dynamic state, activities could be discriminated against more quickly. When tested with the real sequence of activities, the device was still able to detect the activity with high accuracy even though the sampling method for the activities was uniform and no additional activities were present.

Overall, the proposed model reached 96.1%. The results of classification versus actual observation were presented as a confusion matrix as shown in Fig 20. Besides, transition activities were mistaken for each other when the speed of them was so slow. For example, 2/52 lie-down was mistaken for lying, and

1/49 rising was mistaken for lying or sitting. This was similar to squatting, where 1/61 squatting was mistaken for rising or lie-down. However, the rate of occurrence of these errors was not significant. In general, the proposed model gave good classification results. The experimental result helped to evaluate the proposed model's performance when the input was real-time data (Tab 7). The classification accuracy with activities ($ACC$) and the correct prediction rate of non-occurrence actions ($NPV$) were both above 99%. The lowest correct prediction rate for actions ($PPV$) was 89.6% for walking activity, while this reached 100% for squatting and lie-down activities. A sensitivity ($SPE$) of over 90% showed that it was feasible to apply dynamic windowing methods to detect activities, especially with state transition activities. For example, bendp and lie-down activities had corresponding sensitivity indexes of $SPE = 93\%$ and $SPE = 96.2\%$. Mover, rising and sit-down activities were both $SPE = 100\%$. In general, the proposed model had good evaluation indicators with $ACC = 99.4\%$, $SPE = 96.5\%$, $PPV = 96.2\%$, $NPV = 99.7\%$ in reality. These overall evaluation indicators had a negligible difference with those calculated when evaluating our model on public and private datasets. In addition, the proposed model, when conducting experiments, has achieved relatively uniform indexes of over 90%.

The classification performance of our model has significantly improved between the public dataset and real-time data. For example, squatting and tripover increased from 65% and 80.9% to 93.4% and 90.6%, respectively. This result was possible thanks to the sampling process having been improved to increase the quality of the activities and applying the dynamic window method to understand the activity process.

## 6. Discussion

This work attempted to develop a recognition model for a real-time application that would recognize human activity. The dynamic window technique was merged and optimized with the applicable algorithm (random forest). As a result, the human activity recognition system is much more accurate. The result showed that the ability to classify real-time activities on wearables was good at 96.1%, although it was slightly lower than the evaluation results on public and private data. The first reason was the dispersion among feature vectors in real-time data. Meanwhile, with public and private datasets, activities were closely monitored and feature quality was enhanced. Additionally, because activities occur in sequential order in daily life, the acquired data may be more homogeneous than real-time data. In reality, training and testing datasets derived from real-world scenarios may differ significantly. The existence of so many emergent scenarios made it hard to gather an exhaustive set of training data from all types of activity. As a result, the samples of test data would be different from those of training data.

The recognition model was unable to perform well if the training dataset was insufficient. Therefore, we tried to collect activity data for a limited time and surveyed many volunteers. In addition, it was a significant task to classify 13 activities. Previous studies investigated some repetitive activities such as sitting, lying, standing, walking, walking upstairs and walking downstairs as in [43]; walking, jogging, upstairs, downstairs, sitting, standing in [20, 44]. With these activities, the application of static windows gave good results [41, 45, 46]. However, with state transition activities, the static window had a major drawback: it was not able to determine the activity time because the time of these activities was not the same.

Many related works in the field of HAR have been interested in real-time classification capabilities and the application of different classification algorithms (Tab 8).

Thu *et al.* [12] applied 2 features (mean and standard deviation) to a 3-axis accelerometer on each time window of size 6 seconds and combined it with a decision tree algorithm. The result when applied experimentally was 92%, and the accuracy was 95.2%. Their device classified 6 activities, including sitting, standing, lying, walking, and jogging, in real-time. A three-level decision tree algorithm (DT) was built as a recognition model suitable for low-performance microcontrollers, but the classified activities were repetitive activities over time and they had low complexity. Besides, the time of 6 seconds for each classifier applied was too long, leading to a large delay if changing activities. As the number of activities increased and more complex ones were added, it was difficult for their model to achieve high accuracy because of memory limitations and usage features. Similar to Yang's study [35], the stm32 microcontroller was used in his study to embed a real-time recognition model using decision tree algorithm (C4.5). In their study, they used up to 16 features from 6 measures, including mean, magnitude of the acceleration of the three axes, variance, cumulative, skewness, and coefficients. The classification accuracy for five activities, including sitting, walking, jumping, jogging, and cycling, was 90% on average. This established that the DT algorithm could not provide perfect accuracy, as their idea was that data acquired from several people would be trained and analyzed jointly. Consequently, it is possible that their algorithm is not optimal for the individual.

Embedding on a recognition model on low-performance microcontrollers required optimization of machine learning algorithms, the number of features and time window size, so the results were not good [12, 35]. In particular, the static window method in

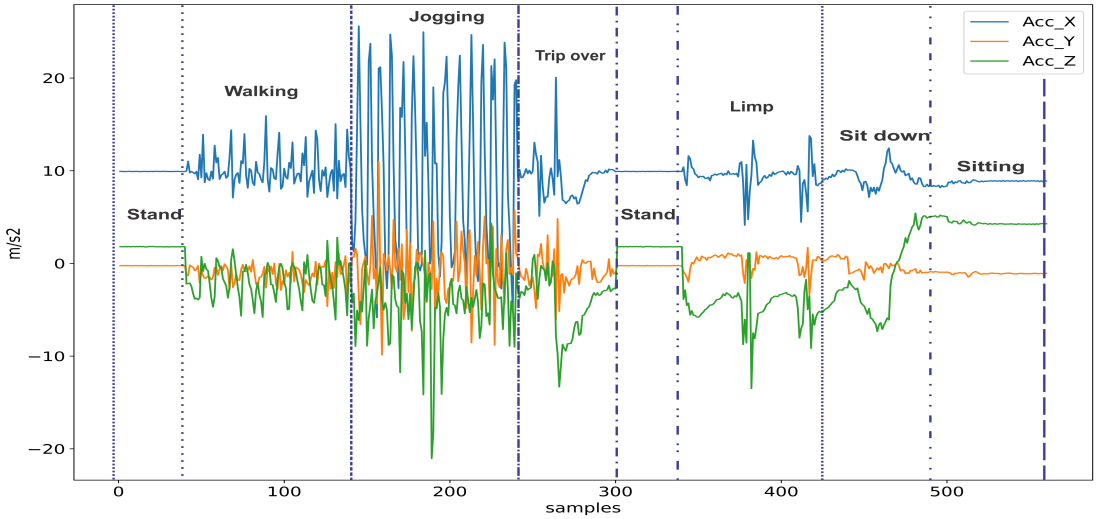


**Fig 19.** The actual sequencing of activities in an experiment.

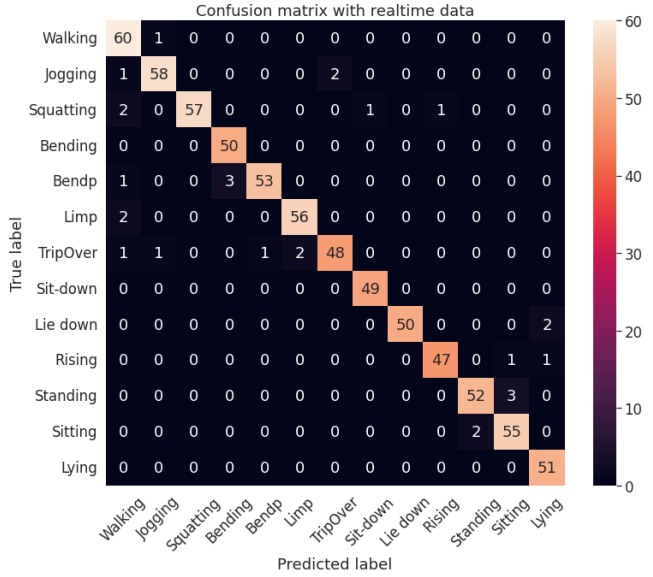


**Fig 20.** The confusion matrix synthesized the experimental results with the proposed recognition model embedded in a wearable device.

those works had shown limitations on the quality of features as well as limitations on the number of activities that could be classified.

Another direction of real-time human activity recognition was presented in the paper [22]. Suto *et al*. tested the classification results between offline and online with 15 features extracted from accelerometer and gyroscope sensors. A wearable used was a phone that ran the Android operating system. The device was attached to the right ankle and was performed by 3 volunteers (2 men and 1 woman). The study investigated 3 methods, including artificial neural networks (ANN), convolutional neural networks (CNN), and 1NN (similar to KNN). Their results showed that the ANN model was highly accurate. CNN was not a good choice for the real-time classification problem because the training time was too large. Applying a time window of 3.88 seconds with 50% overlap had achieved 88.8% when performing real-time classification with 7 activities, including sitting, standing, walking, jogging, running, and lying. The result for accuracy was low for three activities of jogging (8.1%), cycling (13.5%), and sitting (16.2%) when the first volunteer performed the test. This showed data collected from the ankle position was not the best choice. Results improved in the next 2 volunteers reaching over 80% with these actions. However, the recognition model using the ANN algorithm was difficult to apply to the real-time classifier directly on the microcontroller platform because it had a large training time and high complexity. The usability of this model on phones was high, but the large phone size would cause discomfort when placed at the ankle. Besides, the battery power on the phone was not suitable for applications that would operate continuously for a long time.

The application of real-time human activity recognition on wearable devices was a problem with many challenges in terms of microcontroller memory usage, algorithm complexity, sampling rate, and time to survey an activity. We have used up to 31 features in the time domain and used a highly complex random forest algorithm to solve the classification problem of 13 routine actions. The results achieved 96.1% with 99.4% accuracy, showing high applicability in practice. In addition, our model was aimed at complex problems

**Tab 8.** Compare several related studies.

| Comparison | Accuracy (%) | Result (%) | Model) | Features |
|---|---|---|---|---|
| Thu [12] | 95.2 | 92 | DT | 6 |
| Yang [35] | 90.1 | 90.0 | DT | 16 |
| Suto [22] | - | 88.8 | ANN | 15 |
| Ours | 99.4 | 96.1 | RF | 31 |

such as analyzing the movements and activities of many different subjects, such as firefighters, searchers and rescuers, and newcomers recovering from accidents. The state of activity was closely related to their health status. Especially for firefighters, their environment often has influencing factors such as smoke, fire, and high temperatures. Activities are often intense and constantly changing. Monitoring the activities of firefighters will help the commander to provide flexible and timely support. In addition, the fire environment is often unpredictable, so wearable-based monitoring is a viable option.

## 7. Conclusion

This work presented a real-time activity recognition system integrated onto wearable devices. The dynamic window method and the random forest algorithm were combined to create the system using a novel methodology. We created a private dataset comprising 13 activities in addition to the public dataset, including walking, jogging, squatting, bending, bendp, limp, tripover, sit-down, lie-down, rising, standing, and sitting. With the assistance of volunteers, this data was gathered via the ESP32 integrated wearable and the MPU6050 sensor (accelerometer). The suggested recognition model was evaluated on these two datasets using both the confusion matrix and 5-fold cross-validation.

The recognition model building process consists of two steps: First, we extracted feature vectors with 31 dimensions per time window of changed size. These vectors were labeled and combined with the random forest algorithm to build a recognition model for 13 activities. Next, this model was embedded on the ESP32 (a high-speed microcontroller) and was tested with real-time data. Experimental results showed that the features were highly correlated with the activities to be classified by applying the dynamic window method. Static activities (sitting, standing, bending, lying) were quickly detectable after just 1 second of data collection. Transition activities (rising, sit-down, lie-down, squatting, tripover) were classified with over 99% accuracy. The classification results with real-time data reached 96.1%, and the accuracy of 99.4% showed that the proposed model had high performance. Research results were shared at *"https://github.com/daohieuictu/HAR-realtime-random-forest"*. To increase the recognition model's accuracy, we will combine different algorithms when developing it in the future. Besides, we tend to study the support system for firefighters with complex behaviors (rolling, crawling) and survival states (falling, unconscious).

## 8. Ethical Approval

Research volunteers are students and lecturers from our university. All agreed to participate in the experiment, and their information was kept confidential.

## 9. Conflict of interest

On behalf of all authors, the corresponding author states that there is no conflict of interest.

**Acknowledgement.** This research was the product of the scientific project [T2022-07-13] titled "*Research and design a real-time activity recognition system to support human health monitoring*", sponsored by the Thai Nguyen University, University of Information and Communication Technology (ICTU).

## References

[1] F. Lanza, V. Seidita, and A. Chella, "Agents and robots for collaborating and supporting physicians in healthcare scenarios," *Journal of Biomedical Informatics*, vol. 108, no. January, p. 103483, 2020. [Online]. Available: https://doi.org/10.1016/j.jbi.2020.103483

[2] M. M. Rodgers, V. M. Pai, and R. S. Conroy, "Recent advances in wearable sensors for health monitoring," *IEEE Sensors Journal*, vol. 15, no. 6, pp. 3119–3126, 2015.

[3] N. C. Minh, T. H. Dao, N. Q. Huy, D. N. Tran, N. T. Thu, and D. T. Tran, "Evaluation of Smartphone and Smartwatch Accelerometer Data in Activity Classification," in *2021 8th NAFOSTED Conference on Information and Computer Science*. IEEE, 2021, pp. 33–38.

[4] L. Mo, F. Li, Y. Zhu, and A. Huang, "Human physical activity recognition based on computer vision with deep learning model," *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, vol. 2016-July, 2016.

[5] N. Zhu, J. Cao, K. Shen, X. Chen, and S. Zhu, "A decision support system with intelligent recommendation for multi-disciplinary medical treatment," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 16, no. 1s, pp. 1–23, 2020.

[6] S. Chandra Mukhopadhyay, "Wearable Sensors for Human Activity Monitoring: A Review," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1321–1330, 2015.

[7] T. H. Dao, V. C. Ngo, Q. H. Nguyen, D. N. Tran, and D. T. Tran, "Building Human Activity Recognition System using Accelerometers and Machine Learning Methods on

Low Performance Microcontrollers," *Research and Development on Information and Communication Technology*, vol. 12/2021, no. 2, pp. 69–76, 2021.

[8] G. Biagetti, P. Crippa, L. Falaschetti, S. Orcioni, and C. Turchetti, "Human activity monitoring system based on wearable sEMG and accelerometer wireless sensor nodes," *BioMedical Engineering Online*, vol. 17, no. S1, pp. 1–18, 2018. [Online]. Available: https://doi.org/10.1186/s12938-018-0567-4

[9] S. Chung, J. Lim, K. J. Noh, G. Kim, and H. Jeong, "Sensor data acquisition and multimodal sensor fusion for human activity recognition using deep learning," in *Sensors (Switzerland)*, vol. 19, no. 7, 2019.

[10] G. Şengül, M. Karakaya, S. Misra, O. O. Abayomi-Alli, and R. Damaševičius, "Deep learning based fall detection using smartwatches for healthcare applications," *Biomedical Signal Processing and Control*, vol. 71, no. October 2021, p. 103242, 2022.

[11] Y. Zhao, R. Yang, G. Chevalier, X. Xu, and Z. Zhang, "Deep Residual Bidir-LSTM for Human Activity Recognition Using Wearable Sensors," *Mathematical Problems in Engineering*, vol. 2018, 2018.

[12] N. T. Thu, T.-h. Dao, B. Q. Bao, D.-n. Tran, P. V. Thanh, and D.-T. Tran, "Real-Time Wearable-Device Based Activity recognition Using Machine Learning Methods," *International Journal of Computing and Digital Systems*, vol. 12, no. 1, pp. 321–333, 2022. [Online]. Available: https://dx.doi.org/10.12785/ijcds/120126

[13] D. N. Tran, T. N. Nguyen, P. C. P. Khanh, and D. T. Trana, "An IoT-based Design Using Accelerometers in Animal Behavior Recognition Systems," *IEEE Sensors Journal*, vol. 12, no. 18, pp. 17 515–17 528, 2021.

[14] P. C. P. Khanh, D.-T. Tran, V. T. Duong, N. H. Thinh, and D.-N. Tran, "The new design of cows' behavior classifier based on acceleration data and proposed feature set," *Mathematical Biosciences and Engineering*, vol. 17, no. 4, pp. 2760–2780, 2020. [Online]. Available: https://www.aimspress.com/article/doi/10.3934/mbe.2020151

[15] V. Bianchi, M. Bassoli, G. Lombardo, P. Fornacciari, M. Mordonini, and I. De Munari, "IoT Wearable Sensor and Deep Learning: An Integrated Approach for Personalized Human Activity Recognition in a Smart Home Environment," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8553–8562, 2019.

[16] N. Damodaran, E. Haruni, M. Kokhkharova, and J. Schäfer, "Device free human activity and fall recognition using WiFi channel state information (CSI)," *CCF Transactions on Pervasive Computing and Interaction*, vol. 2, no. 1, pp. 1–17, 2020. [Online]. Available: https://doi.org/10.1007/s42486-020-00027-1

[17] P. Kumar and S. Chauhan, "RETRACTED ARTICLE: Human activity recognition with deep learning: overview, challenges and possibilities," *CCF Transactions on Pervasive Computing and Interaction*, vol. 3, no. 3, p. 339, 2021. [Online]. Available: https://doi.org/10.1007/s42486-021-00063-5

[18] J. Qi, P. Yang, M. Hanneghan, S. Tang, and B. Zhou, "A hybrid hierarchical framework for gym physical activity recognition and measurement using wearable sensors," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1384–1393, 2019.

[19] P. Casale, O. Pujol, and P. Radeva, "Human activity recognition from accelerometer data using a wearable device," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6669 LNCS, 2011, pp. 289–296.

[20] M. Milenkoski, K. Trivodaliev, S. Kalajdziski, M. Jovanov, and B. R. Stojkoska, "Real time human activity recognition on smartphones using LSTM networks," *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2018 - Proceedings*, pp. 1126–1131, 2018.

[21] P. Van Thanh, D. T. Tran, D. C. Nguyen, N. Duc Anh, D. Nhu Dinh, S. El-Rabaie, and K. Sandrasegaran, "Development of a Real-Time, Simple and High-Accuracy Fall Detection System for Elderly Using 3-DOF Accelerometers," *Arabian Journal for Science and Engineering*, vol. 44, no. 4, pp. 3329–3342, 2019. [Online]. Available: https://doi.org/10.1007/s13369-018-3496-4

[22] J. Suto, S. Oniga, C. Lung, and I. Orha, "Comparison of offline and real-time human activity recognition results using machine learning techniques," *Neural Computing and Applications*, vol. 32, no. 20, pp. 15 673–15 686, 2020. [Online]. Available: https://doi.org/10.1007/s00521-018-3437-x

[23] A. T. Özdemir and B. Barshan, "Detecting Falls with Wearable SensorsUsing Machine Learning Techniques," *Sensors*, vol. 14, pp. 10 691–10 708, 2014.

[24] T. H. Dao, M. H. Le, D. N. Tran, and D. T. Tran, "Xay dung mang giam sat hanh vi trong toa nha su dung cong nghe wifi," in *REV-ECIT2021*. 978-604-80-5958-3, 2021, pp. 48–53.

[25] A. Mannini, S. S. Intille, M. Rosenberger, A. M. Sabatini, and W. Haskell, "Activity recognition using a single accelerometer placed at the wrist or ankle," *Medicine and Science in Sports and Exercise*, vol. 45, no. 11, pp. 2193–2203, 2013.

[26] C. Torres-Huitzil and M. Nuno-Maganda, "Robust smartphone-based human activity recognition using a tri-axial accelerometer," in *2015 IEEE 6th Latin American Symposium on Circuits and Systems, LASCAS 2015 - Conference Proceedings*, 2015, pp. 2–5.

[27] D. Rodriguez-Martin, A. Samà, C. Perez-Lopez, A. Català, J. Cabestany, and A. Rodriguez-Molinero, "SVM-based posture identification with a single waist-located triaxial accelerometer," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7203–7211, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2013.07.028

[28] D. Naranjo-Hernández, L. M. Roa, J. Reina-Tosina, and M. Á. Estudillo-Valderrama, "SoM: A smart sensor for human activity monitoring and assisted healthy ageing," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 12 PART2, pp. 3177–3184, 2012.

[29] S. Balli, E. A. Sağbaş, and M. Peker, "Human activity recognition from smart watch sensor data using a hybrid of principal component analysis and random forest algorithm," *Measurement and Control (United Kingdom)*, vol. 52, no. 1-2, pp. 37–45, 2019.

[30] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *ACM International Conference Proceeding Series*, vol. 148, 2006, pp. 161–168.

[31] A. Mannini and A. M. Sabatini, "Machine learning methods for classifying human physical activity from on-body accelerometers," *Sensors*, vol. 10, no. 2, pp. 1154–1175, 2010.

[32] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2013, pp. 8595–8598.

[33] J. Bassen, B. Balaji, M. Schaarschmidt, C. Thille, J. Painter, D. Zimmaro, A. Games, E. Fast, and J. C. Mitchell, "Reinforcement Learning for the Adaptive Scheduling of Educational Activities," in *Conference on Human Factors in Computing Systems - Proceedings*, 2020, pp. 1–12.

[34] D. Guan, W. Yuan, Y. K. Lee, A. Gavrilov, and S. Lee, "Activity recognition based on semi-supervised learning," in *Proceedings - 13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA 2007*, no. 1, 2007, pp. 469–475.

[35] F. Yang and L. Zhang, "Real-time human activity classification by accelerometer embedded wearable devices," in *2017 4th International Conference on Systems and Informatics, ICSAI 2017*, vol. 2018-Janua, no. Icsai, 2017, pp. 469–473.

[36] A. Wang, G. Chen, J. Yang, S. Zhao, and C.-Y. Chang, "A Comparative Study on Human Activity Recognition Using Inertial Sensors in a Smartphone," *IEEE Sensors Journal*, vol. 16, no. 11, pp. 4566–4578, 2016.

[37] L. Bao and S. S. Intille, "Activity Recognition from User-Annotated Acceleration Data," in *Pervasive Computing*, A. Ferscha and F. Mattern, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1–17.

[38] W. Xiao and Y. Lu, "Daily Human Physical Activity Recognition Based on Kernel Discriminant Analysis and Extreme Learning Machine," *Mathematical Problems in Engineering*, vol. 2015, p. 790412, 2015. [Online]. Available: https://doi.org/10.1155/2015/790412

[39] R. Igual, C. Medrano, and I. Plaza, "A comparison of public datasets for acceleration-based fall detection," *Medical Engineering and Physics*, vol. 37, no. 9, pp. 870–878, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.medengphy.2015.06.009

[40] S. Abbate, M. Avvenuti, P. Corsini, J. Light, and A. Vecchio, "Monitoring of Human Movements for Fall Detection and Activities Recognition in Elderly Care Using Wireless Sensor Network: a Survey," *Wireless Sensor Networks: Application-Centric Design*, pp. 1–22, 2010.

[41] A. T. Özdemir, "An analysis on sensor locations of the human body for wearable fall detection devices: Principles and practice," *Sensors (Switzerland)*, vol. 16, no. 8, 2016.

[42] X. Sun, L. Qiu, Y. Wu, Y. Tang, and G. Cao, "Sleepmonitor: monitoring respiratory rate and body position during sleep using smartwatch," in *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 3, 2017, pp. 1–22.

[43] B. Fida, I. Bernabucci, D. Bibbo, S. Conforto, and M. Schmid, "Varying behavior of different window sizes on the classification of static and dynamic physical activities from a single accelerometer," *Medical Engineering and Physics*, vol. 37, no. 7, pp. 705–711, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.medengphy.2015.04.005

[44] K. Maswadi, N. A. Ghani, S. Hamid, and M. B. Rasheed, "Human activity classification using Decision Tree and Naïve Bayes classifiers," *Multimedia Tools and Applications*, vol. 80, no. 14, pp. 21 709–21 726, 2021.

[45] T. H. Lee, A. Ullah, and R. Wang, "Bootstrap Aggregating and Random Forest," *Advanced Studies in Theoretical and Applied Econometrics*, vol. 52, pp. 389–429, 2020.

[46] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A Public Domain Dataset for Human Activity Recognition Using Smartphones," in *Proceedings of the 21th international European symposium on artificial neural networks, computational intelligence and machine learning*, 2013, pp. 437–442.